

```

1 /*
2 <span id="gsh" data-title="GShell" data-author="sato@its-more.jp">
3 <meta charset="UTF-8">
4 <meta name="viewport" content="width=device-width, initial-scale=1.0">
5 <link rel="icon" id="GshFavIconURL" href=""><!-- place holder -->
6 <span id="GshVersion" hidden="">gsh-0.8.5--2020-11-22--SatoxITS</span>
7 <title id="GshTitle">GShell-0.8.5 by SatoxITS</title>
8
9 <div id="GshHeading">
10 <div id="GshTopbar" class="MetaWindow"></div>
11 <div id="GshPerfMon"></div>
12 <div id="GshSidebar" class="SbSideBar" style=""><div id="GshIndexer" style=""></div></div>
13 </div>
14 <div id="GshMain">
15 <div id="GshBanner" height="100px" onclick="shiftBS();">
16 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.8.5 // 2020-11-22 // SatoxITS</note></div>
17 </div>
18 <span id="FeaturesView"></span>
19 */
20
21
22
23 <!-- ===== Work { ===== -->
24 <span id="Counter_WorkCodeSpan">
25 /*
26 <details open="true"><summary>Counter Watcher</summary>
27 <!-- ----- Counter // 2020-1120 SatoxITS { --->
28 <h2>Counter Watcher</h2>
29
30 <div id="CountWatcher_1"><div class="NFrame"><div class="Number">ElementId: <input id="CounterId" class="CounterText_Number" type="text" value="adswc_countertext">
31 Until: <input id="untilCount" class="Number" type="text" value="80000"> <input id="untilCountLocalStorage" type="checkbox">LocalStorage
32 Remain: <span id="remainCount" class="Number" type="text" value="00000"></span>
33 Current: <span id="lastCount" class="Number" type="text" value="00000"></span>
34 Time: <span id="lastTime" class="Number" type="text" value="00:00"></span>
35 Interval: <input id="Elapsed" class="Number2_Number" type="text" value="00" style="text-align:right;" /> <input id="Interval" class="Number2_Number" type="text" value="60">
36 <div id="ProgressBar" class="ProgressBar"><div id="Progress" class="Progress"></div></div> WebSurkit: <input id="CountWebSurkit" type="checkbox" checked=""></input></div>
37 </div>
38 </div>
39 <span id="DummyCounter" contenteditable="true">7777</span><br>
40
41
42 <style>
43 @keyframes Progress100 {
44   from { width:100%; }
45   to { width:0%; }
46 }
47 #ProgressBar {
48   margin:4px;
49   margin-left:8px;
50   width:300px;
51   width:80%;
52   height:14px;
53   border:1px solid #228;
54   background-color:#228;
55 }
56 #Progress {
57   xxx-animation-name:Progress100;
58   xxx-animation-duration:60s;
59   margin:0px;
60   padding:0px;
61   width:100%;
62   width:0%;
63   height:100%;
64   border:0px;
65   background-color:#ffffff80;
66   font-size:12px;
67   line-height:1.0;
68 }
69 #CountWatcher_1 {
70   xpadding:10px !important;
71   width:100% !important;
72   white-space:nowrap !important;
73   overflow:scroll !important;
74   xoverflow:hidden;
75 }
76 .NFrame {
77   xmargin:2px !important;
78   padding:10px !important;
79   padding-left:8px !important;
80   xborder:1px dashed #228 !important;
81   display:block !important;
82   width:100% !important;
83   background-color:#404080ff !important;
84   font-family:Courier New !important;
85   white-space:pre !important;
86 }
87 .Number {
88   margin:2px !important;
89   padding:2px !important;
90   font-family:Courier New !important;
91   font-size:12pt !important;
92   display:inline !important;
93   width:40pt !important;
94   height:16pt !important;
95   line-height:1.1 !important;
96   vertical-align:middle !important;
97   color:#fff !important;
98   border:1px !important;
99   background-color:#404080ff !important;
100   xfont-weight:bold !important;
101 }
102 .Number2 {
103   width:20pt !important;
104   overflow:scroll !important;
105 }
106 .CounterText {
107   width:180px !important;
108 }
109 </style>
110 <script>
111 function reload(){
112   location.reload();
113 }
114 function to2d(num){
115   if( num < 10 ){
116     return '0' + num;
117   }else{
118     return num;
119   }
120 }
121 function timeHMS(){
122   h = d.getHours();
123   m = d.getMinutes();
124   s = d.getSeconds();
125   return to2d(h) + ':' + to2d(m) + ':' + to2d(s);
126 }
127 function CWatch_Init(){
128   until = localStorage.getItem('UntilCount');
129   if( 0 < until && until <= 100000 ){
130     untilCount.value = until;
131     untilCountLocalStorage.checked = true;
132     console.log('++ Set until:'+untilCount.value);
133   }
134 }
135 CWatch_Init();
136 function watchCounter(){
137   d = new Date();
138   m = d.getTime();
139   now = m / 1000;
140   lastTime.innerHTML = timeHMS() + ' (' + now + ')';
141
142   counterId = CounterId.value;
143   cnt = document.getElementById(counterId);
144   if( cnt == null ){
145     cnt = DummyCounter;
146   }
147   count = parseInt(cnt.innerHTML);
148   until = parseInt(untilCount.value);
149   lastCount.innerHTML = count;
150   remain = until - count;
151   remainCount.innerHTML = remain;
152 }
153 var cwatch_gllinked = false;
154 var sentCount = 0;
155 function sendCount(){
156   counterId = CounterId.value;
157   cnt = document.getElementById(counterId);
158   if( cnt == null ){
159     cnt = DummyCounter;
160   }
161   count = parseInt(cnt.innerHTML);
162   console.log('++ sendCount:'+count);
163   CW_sendCounterValue(count);
164   console.log('++ sentCount:'+count);
165 }
166 function CW_sendCounterValue(count){
167   if( cwatch_gllinked == false ){
168     GJ_Join();
169     cwatch_gllinked = true;
170   }
171   msg = 'COUNTER '+count;
172   GJ_BcastMessage(msg);
173   //console.log('SentCount:'+count);
174 }
175 function CWatch_Setup(){
176   if( cwatch_gllinked == false ){

```

```

177     GJ_Join();
178     cwatch_gjlinked = true;
179 }
180 function setUntilCount(){
181     nuntil = untilCount.value;
182     if ( 0 < nuntil && nuntil <= 100000 ){
183         localStorage.setItem('UntilCount',nuntil);
184     }else{
185         nuntil = 80000;
186         untilCount.value = nuntil;
187     }
188     suntil = localStorage.getItem('UntilCount');
189     if ( suntil == null ){
190         suntil = 'undef';
191     }
192     alert('setUntilCount: '+nuntil+' <- '+suntil);
193 }
194 untilCount.addEventListener('change',setUntilCount);
195 setNextWatch();
196 }
197
198 var intervalMs = parseInt(Interval.value) * 1000;
199 var numProgress = 0;
200 var StartWatch = 0;
201 var NextWatch = 0;
202 var showProgressTimer = null;
203 function setNextWatch(){
204     d = new Date();
205     var now0 = d.getTime();
206     StartWatch = now0;
207     NextWatch = now0 + intervalMs;
208     if( showProgressTimer != null ){
209         clearTimeout(showProgressTimer);
210     }
211     showProgressTimer = window.setTimeout(showProgress,1000);
212 }
213 function showProgress(){
214     d = new Date();
215     //now = d.getTime(); // now is overwritten in sendCount() !!!
216     now1 = d.getTime();
217     //console.log(++1 now'+now1);
218     if( sentCount < 1 ){
219         sentCount += 1;
220         sendCount();
221     }
222     //console.log(++1 now'+now1);
223 }
224 remt = NextWatch - now1;
225 perc = ((remt / intervalMs) * 100).toFixed(2);
226
227 //console.log('Progress = '+remt+'ms, '+perc+'%');
228 numProgress += 1;
229 Progress.innerHTML = perc + ' ('+numProgress+')';
230 Progress.style.width = perc + '%';
231 Elapsed.value = (remt / 1000).toFixed(0);
232
233 watchCounter();
234 remC = parseInt(remainCount.innerHTML);
235 if( remC <= 0 ){
236     console.log(++1 Finished remainCount:'remC);
237     remainCount.innerHTML += ' (Reached)';
238     return;
239 }
240
241 showProgressTimer = window.setTimeout(showProgress,1000);
242 if( remt < 0 ){
243     reload();
244 }
245 }
246
247 fi = document.getElementById('FeaturesView');
248 if( fi != null ){
249     fi.appendChild(CountWatcher_1);
250 }
251 window.addEventListener('load',setNextWatch);
252 untilCount.addEventListener('change',setNextWatch);
253 Interval.addEventListener('change',setNextWatch);
254 </script>
255
256 <input id="Counter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
257 <input id="Counter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
258 <input id="Counter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
259 <span id="Counter_WorkCodeView"></span>
260 <script id="Counter_WorkScript">
261 function Counter_openWorkCodeView(){
262     function Counter_showWorkCode(){
263         showHtmlCode(Counter_WorkCodeView,Counter_WorkCodeSpan);
264     }
265     Counter_WorkCodeViewOpen.addEventListener('click',Counter_showWorkCode);
266 }
267 Counter_openWorkCodeView(); // should be invoked by an event
268 </script>
269 </details>
270 <!-- Counter_WorkCodeSpan -->
271 *//</span>
272 //<!-- ===== Work } ===== -->
273
274
275
276
277 //<!-- ===== Work { ===== -->
278 //<span id="BinB_WorkCodeSpan">
279 /*
280 <details><summary>Browser in Browser</summary>
281 <!-- ===== BinB // 2020-1119 SatoxITS { -->
282 <h2>Browser in Browser</h2>
283
284 <style>
285 </style>
286 <script>
287 var xhr = new XMLHttpRequest();
288 xhr.open('GET', '/gshell/gsh.go.html');
289 xhr.onreadystatechange = function () {
290     var DONE = 4; // readyState 4 means the request is done.
291     var OK = 200; // status 200 is a successful return.
292     if (xhr.readyState == DONE) {
293         if (xhr.status == OK) {
294             console.log(xhr.responseText);
295             // 'This is the output.'
296         }else{
297             console.log('Error: ' + xhr.status);
298             // An error occurred during the request.
299         }
300     }
301 }; // Send the request to xxx.php
302 //xhr.send(null);
303 </script>
304
305 <input id="BinB_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
306 <input id="BinB_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
307 <input id="BinB_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
308 <span id="BinB_WorkCodeView"></span>
309 <script id="BinB_WorkScript">
310 function BinB_openWorkCodeView(){
311     function BinB_showWorkCode(){
312         showHtmlCode(BinB_WorkCodeView,BinB_WorkCodeSpan);
313     }
314     BinB_WorkCodeViewOpen.addEventListener('click',BinB_showWorkCode);
315 }
316 BinB_openWorkCodeView(); // should be invoked by an event
317 </script>
318 </details>
319 <!-- BinB_WorkCodeSpan -->
320 *//</span>
321 //<!-- ===== Work } ===== -->
322
323
324
325
326 //<!-- ===== Work { ===== -->
327 //<span id="Topbar_WorkCodeSpan">
328 /*
329 <details><summary>Topbar</summary>
330 <!-- ===== Topbar // 2020-1008 SatoxITS { -->
331 <h2>Topbar</h2>
332 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
333 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
334 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
335 <span id="Topbar_WorkCodeView"></span>
336 </details>
337
338 <style>
339 #GshHeading {
340     display:inline;
341     overflow:visible;
342 }
343 .ConfigIcon {
344     position:absolute;
345     top:-6px;
346     left:92%;
347     width:32px;
348     height:32px;
349 }
350 .MetaWindow {
351     z-index:1000;
352     position:relative;
353     display:block;

```

```

354 overflow:visible !important;
355 width:99.9%;
356 height:22px;
357 top:22px;
358 border:1px solid #22a;
359 margin:0px;
360 left:0.0%;
361 line-height:1.0;
362 font-family:Georgia;
363 color:#fff;
364 font-size:12pt;
365 text-align:center;
366 vertical-align:middle;
367 padding:4px;
368 xxbackground-color:rgba(0,8,170,0.8);
369 background-color:#3a4861;xxx-PBlue;
370 vertical-align:middle;
371
372 .MetaWindow:hover {
373   color:#000;
374   border:1px solid #22a;
375   background-color:rgba(255,255,255,1.0);
376 }
377
378 #GshBanner {
379   overflow:visible;
380   display:block;
381   width:100%;
382   height:100px;
383   left:inherit !important;
384 }
385 </style>
386 <script>
387 function Topbar_openWorkCodeView(){
388   function Topbar_showWorkCode(){
389     showHtmlCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
390   }
391   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
392 }
393
394 function ConfigClick(){
395   if( 0 <= AffView.style.zIndex ){
396     AffView.style.saved_zIndex = AffView.style.zIndex;
397     AffView.style.zIndex = -1000;
398     GshSidebar.style.zIndex = -1;
399     GshPerfMon.style.zIndex = -1;
400   }else{
401     //AffView.style.zIndex = AffView.style.saved_zIndex;
402     AffView.style.zIndex = 1;
403     GshSidebar.style.zIndex = 1;
404     GshPerfMon.style.zIndex = 1;
405     GMenu.style.zIndex = 10000000;
406   }
407   console.log('AffZidex='+AffView.style.zIndex);
408 }
409
410 function Gshell_initTopbar(){
411   GshTopbar.innerHTML = GshTitle.inerHTML;
412   //<img id="ConfigIcon" class="ConfigIcon">
413   if( true ){
414     cfgi = document.createElement('img');
415     cfgi.id = 'ConfigIcon';
416     cfgi.setAttribute('class','ConfigIcon');
417     GshTopbar.appendChild(cfgi);
418     cfgi.src = ConfigICON_DATA;
419     //cfgi.style.zIndex = 1000000000;
420     //cfgi.addEventListener('click',ConfigClick);
421     GshTopbar.addEventListener('click',ConfigClick);
422   }
423 }
424 </script>
425 <!-- Topbar_WorkCodeSpan -->
426 *! //</span>
427 <!-- Work } ----- -->
428
429 <!-- Work { ----- -->
430 <span id="Indexer_WorkCodeSpan">
431 /*
432 <details><summary>Indexer</summary>
433 <!-- ----- Indexer // 2020-1007 SatoxITS { -->
434 <h2>Indexer</h2>
435 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
436 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
437 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
438 <span id="Indexer_WorkCodeView"></span>
439 </details>
440 <style id="SidebarIndex">
441 #gsh {
442   display:block;
443   xxoverflow:scroll !important;
444 }
445 #GshMain {
446   z-index:1;
447   position:relative;
448   display:block;
449   width:50% !important;
450   left:19.5% !important;
451 }
452 #GshSidebar {
453   z-index:0;
454   position:relative !important;
455   overflow:auto;
456   resize:both !important;
457   xxoverflow-y:hidden !important;
458   xxxheight:100px !important;
459   xxxdisplay:inline !important;
460   left:0px;
461   top:0px;
462   width:19.5%;
463   min-width:80px;
464   xxxheight:100% !important;
465   height:0px;
466   color:#F00;
467   xxbackground-color:rgba(64,64,64,0.5);
468   xxbackground-color:#DFE3EB;xxx-PBlue;
469   background-color:#eeeeee;xxx-PBlue;
470 }
471 #GshPerfMon {
472   position:relative;
473   display:block;
474   overflow:visible;
475   z-index:0 !important;
476   xxheight:12pt;
477   font-family:monospace, Courier New !important;
478   font-size:9pt !important;
479   color:#F84;
480   top:-20px;
481 }
482 #GshPerfMon:hover {
483   z-index:3 !important;
484 }
485 #GshSidebar:hover {
486   z-index:2;
487   overflow-x:visible !important;
488   background-color:rgba(255,255,255,0.7);
489   width:50%;
490 }
491 #GshIndexer {
492   z-index:0;
493   position:relative;
494   resize:both !important;
495   height:100%;
496   left:0px;
497   top:0px;
498   scroll-behavior: overflow !important;
499   padding-left:4pt;
500   font-size:0.5em;
501   white-space:nowrap;
502   xxx-background-color:rgba(64,160,64,0.6) !important;
503   color:#7794c6;xxx-PBlue;
504   xxbackground-color:#DFE3EB;xxx-PBlue;
505   background-color:#eeeeee;xxx-PBlue;
506 }
507 #GshIndexer:hover {
508   z-index:10000000;
509   overflow-x:visible !important;
510   color:#000000 !important;xxx-PBlue;
511   xxxbackground-color:#FFFFFF;xxx-PBlue;
512   background-color:rgba(255,255,255,0.7);
513   padding-right:0px;
514   width:80%;
515 }
516 #GshIndexer:select {
517   color:#000000 !important;xxx-PBlue;
518   background-color:#FFFFFF;xxx-PBlue;
519 }
520 .IndexLine {
521   font-size:8pt !important;
522   font-family:Georgia;
523   display:block;
524   xxx-color:#fff;
525   xxx-color:#efff5;xxx-PBlue;
526   xxx-color:#41516d;xxx-PBlue;
527   xxx-color:#7794c6;xxx-PBlue;
528   padding-right:4pt;
529 }
530 .IndexLine:hover {

```

```

531 font-size:10pt!important;
532 xxx-color:#228;
533 xxx-background-color:#fff;
534 xxxcolor:#fff;xxx-PBlue;
535 color:#516487;xxx-PBlue;
536 background-color:rgba(220,220,255,1.0);xxx-PBlue;
537 xxxtext-shadow:1px 1px #e3;
538 text-shadow:1px 1px #eee;
539 xxxbackground-color:#516487;xxx-PBlue;
540 xxxtext-decoration:underline !important;
541 }
542 </style>
543
544 <script id="Indexer_WorkScript">
545 function Indexer_openWorkCodeView(){
546 function Indexer_showWorkCode(){
547 showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
548 }
549 Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
550 }
551 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
552 Indexer_openWorkCodeView();
553
554 var startPerfDate = new Date();
555 var prevPerfDate = startPerfDate;
556 function ShowResourceUsage(){
557 d = new Date();
558 perf = '';
559 perf += '<'+font color="gray">UA: ' + window.navigator.userAgent + '<'+font<br>\n';
560 perf += DateShort0(startPerfDate) + '<br>\n';
561 perf += DateShort0() + '<br>\n';
562 elps = d.getTime() - startPerfDate.getTime();
563 itvl = d.getTime() - prevPerfDate.getTime();
564 perf += 'Elapsed: ' + elps/1000 + ' s<br>\n';
565 perf += 'Skew: ' + (itvl-1000) + ' ms<br>\n';
566 prevPerfDate = d;
567
568 if( performance.memory != undefined ){
569 m0 = performance.memory;
570 mu0 = (m0.usedJSHeapSize / 1000000.0); //toFixed(6);
571 perf += 'Memory: '+mu0+' MB<br>\n';
572 }
573 perf += '<br>\n';
574
575 //GshSidebar.innerHTML = perf;
576 GshPerfMon.innerHTML = perf;
577 //GshIndexer.innerHTML = 'Memory: '+mu0+' MB';
578 //console.log('-- PerfMon heap: '+mu0+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
579 if( true ){
580 GshSidebar.style.zIndex = 1000;
581 GshIndexer.style.zIndex = 0;
582 GshPerfMon.style.zIndex = 1;
583 //GshSidebar.appendChild(GshPerfMon);
584 if( document.getElementById('primary') == null ){ // not in WordPress
585 // GshPerfMon.style.position = 'absolute';
586 }
587 GshPerfMon.style.display = 'block';
588 GshPerfMon.style.marginLeft = '4px';
589 //GshPerfMon.style.top = '45px';
590 GshPerfMon.style.position = 'relative';
591 //GshPerfMon.style.position = 'absolute';
592 //topy = GshTopbar.getBoundingClientRect().top;
593 //topy = parseInt(topy) + 40;
594 //GshPerfMon.style.top = topy + 'px';
595 GshPerfMon.style.left = '0px';
596
597 GshMain.style.top = -GshPerfMon.getBoundingClientRect().height;
598 }
599 }
600 function ResetPerfMon(){
601 GshPerfMon.removeAttribute('style');
602 GshSidebar.removeAttribute('style');
603 }
604
605 var iserno = 0;
606 var GeneratedId = 0;
607 function generateIndex(ni,e,chv,nch,ht){
608 // https://developer.mozilla.org/en-US/docs/Web/API/Element
609 c = '';
610 if( e.classList != null ){
611 c = e.classList.value;
612 }
613 //console.log('-- <'+e.nodeName+'> #' + e.id + ' .'+c+' '+e.attributes);
614 if( e.nodeName == '#text' ){ return ''; }
615 if( e.nodeName == '#comment' ){ return ''; }
616 if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
617 id = e.innerHTML;
618 GeneratedId += 1;
619 eid = 'GeneratedId-'+GeneratedId;
620 e.id = eid;
621 }else
622 if( e.nodeName == 'SUMMARY' ){
623 id = e.innerHTML;
624 GeneratedId += 1;
625 eid = 'GeneratedId-'+GeneratedId;
626 e.id = eid;
627 }else
628 if( and(e.nodeName == 'DIV',c=='xxxxxxxxxxxxxxxxentry-content') ){
629 console.log('-- DIV entry-content begin');
630 id = e.innerHTML;
631 GeneratedId += 1;
632 eid = 'GeneratedId-'+GeneratedId;
633 e.id = eid;
634 console.log('-- DIV entry-content end hash-child=',e.hasChildNodes());
635 }else
636 if( e.id == '' || e.id == 'undefined' ){
637 return '';
638 }else{
639 id = '#' + e.id;
640 e.id = e.id;
641 }
642 iserno += 1;
643 ht = '<' + 'div id="GeneratedEref'+iserno+' class="IndexLine" href="'+eid+'"' +
644 + isernos + 'ni'+e.nodeName + ' ' + id;
645 if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+/div>'; }
646 if( !e.hasChildNodes() ){ return ht + '<'+/div>'; }
647 chv = e.childNodes;
648 nch = e.childNodes.length;
649 if( chv != null ){ nch = chv.length; }
650 ht += ('+chv+' + '<'+/div>';
651 for( let i = 0; i < chv.length; i++){
652 sec = ni+'.'+i;
653 if( ni == '' ){ sec = i; }
654 ht += generateIndex(sec,chv[i],null,0);
655 }
656 return ht;
657 }
658 function onClickIndex(e){
659 tid = e.target.id;
660 tge = document.getElementById(tid);
661 eid = tge.getAttribute('href');
662 rx = tge.getBoundingClientRect().left.toFixed(0)
663 ry = tge.getBoundingClientRect().top.toFixed(0)
664 if( false ){
665 alert('index clicked mouse(x='+e.x+', y='+e.y+')'
666 + '\ntid='+tid + ' rx='+rx + ',zy'+ry
667 + '\neid=' + eid + '\n'
668 + '\nhtml='+ tge.outerHTML);
669 }
670 ee = document.getElementById(eid);
671 sx = 'NaN';
672 sy = ee.getBoundingClientRect().top;
673 console.log('sx'+sx+',sy'+sy);
674 ee.scrollIntoView()
675 window.scrollTo(sx,sy)
676 //window.scrollTo({left:'Non',top:sy,behavior:'smooth'});
677 }
678 function Indexer_afterLoaded(){
679 sideindex = document.getElementById('GshIndexer');
680 ht = '<' + 'h3>G-Indexer<'+/h3>';
681 ht += generateIndex("",document.getElementById('gsh'),null,0,'');
682 if( (pri = document.getElementById('primary')) != null ){
683 ht += generateIndex("",pri,null,0,'');
684 }
685 ht += '<'+br>';
686 ht += '<'+br>';
687 ht += '<'+br>';
688 ht += '<'+br>';
689 sideindex.innerHTML = ht;
690 sideindex.addEventListener('click',onClickIndex);
691
692 if( (pri = document.getElementById('primary')) != null ){
693 console.log('-- Seems in WordPress');
694 pri.style.zIndex = 2000;
695
696 GshSidebar.style.setProperty('position','relative','important');
697 GshSidebar.style.top = '-140px';
698 //GshSidebar.style.setProperty('position','absolute','important');
699 //GshSidebar.style.top = '0px';
700
701 GshSidebar.style.setProperty('width','200px','important');
702 GshSidebar.style.setProperty('overflow','scroll','important');
703 GshSidebar.style.resize = 'both';
704
705 GshSidebar.style.left = '-100px';
706 GshIndexer.style.left = '100px';
707 GshIndexer.style.height = '1400px';

```

```

708 gsh.appendChild(GshSidebar); // change parent
709 }else{
710     console.log('-- Seems not in WordPress');
711     GshSidebar.style.setProperty('position','fixed','important');
712 }
713 }
714 //document.addEventListener('load',Indexer_afterLoaded);
715
716 DestroyIndexBar = function(){
717     sideindex = document.getElementById('GshIndexer');
718     sideindex.innerHTML = "";
719     sideindex.style = "";
720 }
721 </script>
722
723 <!-- Indexer_WorkCodeSpan -->
724 *//</span>
725 //<!-- Work -->
726
727
728 /*
729 <h2>GShell // a General purpose Shell built on the top of Golang</h2>
730 <p>
731 <note>
732 It is a shell for myself, by myself, of myself. --SatoxITS(^_^)
733 <a href="gsh-0.6.2.go.html">prev.</a>
734 </note>
735 <p>
736 <div id="GJFactory_x"></div>
737
738 <div>
739 <span id="GshMenu" class="GshMenu">
740 <span class="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
741 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
742 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
743 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
744 <span id="gsh-Winid" onclick="win_jump(0.1);">0</span>
745 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
746 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
747 <span class="GshMenu" id="GshMenuStop" onclick="html_stop(this,true);">Stop</span>
748 <span class="GshMenu" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
749 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
750 <span class="GshMenu" id="GshMenuSign" onclick="html_sign(this);" style="background-color:#f0f0f0;">Source</span>
751 </span>
752 </div>
753 </div>
754
755 *//
756
757 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
758 <h3>Fun to create a shell</h3>
759 <p>For a programmer, it must be far easy and fun to create his own simple shell
760 rightly fitting to his favor and necessities, than learning existing shells with
761 complex full features that he never use.
762 I, as one of programmers, am writing this tiny shell for my own real needs,
763 totally from scratch, with fun.
764 </p><p>
765 For a programmer, it is fun to learn new computer languages. For long years before
766 writing this software, I had been specialized to C and early HTML(1-1).
767 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
768 on demand as a novice of these, with fun.
769 </p><p>
770 This single file "gsh.go", that is executable by Go, contains all of the code written
771 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
772 HTML file that works as the viewer of the code of itself, and as the "home page" of
773 this software.
774 </p><p>
775 Because this HTML file is a Go program, you may run it as a real shell program
776 on your computer.
777 But you must be aware that this program is written under situation like above.
778 Needless to say, there is no warranty for this program in any means.
779 </p>
780 <address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
781 </details>
782
783 *//
784
785 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
786 <h3>Cross-browser communication</h3>
787 <p>
788 ... to be written ...
789 </p>
790 <h3>Vi compatible command line editor</h3>
791 <p>
792 The command line of GShell can be edited with commands compatible with
793 <a href="https://www.washington.edu/computing/unix/vi.html">vi</a> or <a href="https://www.washington.edu/computing/unix/vi.html">vim</a>.
794 As in vi, you can enter <b>:!</b> to execute a command mode</b> by <b>ESC</b> key,
795 then move around in the history by <b>:k</b> / <b>:p</b> or <b>:code</b>,
796 or within the current line by <b>:lh f w b o $ %</b> or so.
797 </p>
798 </details>
799
800 *//
801
802 <details id="gsh-gindex">
803 <summary>index</summary><div class="gsh-src">
804 Documents
805 <span class="gsh-link" onclick="jumpto JavaScriptView();">Command summary</span>
806 Golang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
807 Package structures
808 <a href="#import">import</a>
809 <a href="#struct">struct</a>
810 Main functions
811 <a href="#comexpansion">str-expansion</a> // macro processor
812 <a href="#finder">finder</a> // builtin find + du
813 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
814 <a href="#plugin">plugin</a> // plugin commands
815 <a href="#ex-commands">system</a> // external commands
816 <a href="#builtin">builtin</a> // builtin commands
817 <a href="#network">network</a> // socket handler
818 <a href="#remote-sh">remote-sh</a> // remote shell
819 <a href="#redirect">redirect</a> // StdIn/Out redirection
820 <a href="#history">history</a> // command history
821 <a href="#usage">usage</a> // resource usage
822 <a href="#encode">encode</a> // encode / decode
823 <a href="#IMP">IMP</a> // command line IMP
824 <a href="#getline">getline</a> // line editor
825 <a href="#scan">scanf</a> // string decomposer
826 <a href="#interpreter">interpreter</a> // command interpreter
827 <a href="#main">main</a>
828 </span>
829 JavaScript part
830 <a href="#script-src-view" class="gsh-link" onclick="jumpto JavaScriptView();">Source</a>
831 <a href="#gsh-data-frame" class="gsh-link" onclick="jumpto DataView();">Builtin data</a>
832 CSS part
833 <a href="#style-src-view" class="gsh-link" onclick="jumpto StyleView();">Source</a>
834 References
835 <a href="#gsh-link" class="gsh-link" onclick="jumpto WholeView();">Internal</a>
836 <a href="#gsh-reference" class="gsh-link" onclick="jumpto ReferenceView();">External</a>
837 Whole parts
838 <a href="#whole-src-view" class="gsh-link" onclick="jumpto WholeView();">Source</a>
839 <a href="#whole-src-view" class="gsh-link" onclick="jumpto WholeView();">Download</a>
840 <a href="#whole-src-view" class="gsh-link" onclick="jumpto WholeView();">Dump</a>
841 </div>
842 </details>
843
844 *//
845 <details id="gsh-gocode">
846 <summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
847 // gsh - Go lang based Shell
848 // (c) 2020 ITS more Co., Ltd.
849 // 2020-0807 created by SatoxITS (sato@its-more.jp)
850
851 package main // gsh main
852
853 // <a href="https://golang.org/pkg/">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
854 import (
855     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
856     "errors" // <a href="https://golang.org/pkg/errors/">errors</a>
857     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
858     "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
859     "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
860     "time" // <a href="https://golang.org/pkg/time/">time</a>
861     "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
862     "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
863     "os" // <a href="https://golang.org/pkg/os/">os</a>
864     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
865     "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
866     "net" // <a href="https://golang.org/pkg/net/">net</a>
867     "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
868     "html" // <a href="https://golang.org/pkg/html/">html</a>
869     "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
870     "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
871     "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
872     "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
873     "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
874     "gshdata" // gshell's id and source code
875     "hash/crc32" // <a href="https://golang.org/pkg/hash/crc32/">crc32</a>
876     "golang.org/x/net/websocket"
877     "runtime"
878 )
879
880 *//
881 #include <stdio.h> // </stdio.h> to be closed as HTML tag i-p
882 #ifdef WIN32
883 #include <windows.h> // </windows.h>

```

```

885 // 2020-1022 added -- terminal mode on Windows
886 // https://docs.microsoft.com/en-us/windows/console/setconsolemode
887 // https://docs.microsoft.com/en-us/windows/win32/inputdev/using-keyboard-input
888 int setTermRaw(){
889     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
890     DWORD tmode = 0;
891     if( GetConsoleMode(hStdin,&tmode) ){
892         DWORD xmode = tmode;
893         xmode &= ~ENABLE_ECHO_INPUT;
894         xmode &= ~ENABLE_LINE_INPUT;
895         xmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
896         if( SetConsoleMode(hStdin,xmode) ){
897             return tmode;
898         }
899     }
900     return 0;
901 }
902 int setTermMode(int tmode){
903     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
904     SetConsoleMode(hStdin,tmode);
905     return 0;
906 }
907 #else
908 int setTermRaw(){
909     return -1;
910 }
911 int setTermMode(int tmode){
912     return 0;
913 }
914 #endif
915 /*
916 import "C"
917
918 /*
919 // 2020-0906 added,
920 // <a href="https://golang.org/cmd/cgo/">CGO</a>
921 // #include "poll.h" // <poll.h> // </poll.h> to be closed as HTML tag :-p
922 // #typedef struct { struct pollfd fdv[8]; } pollFdv;
923 // int poll(pollFdv *fdv, int nfds, int timeout){
924 //     return poll(fdv->fdv,nfds,timeout);
925 // }
926 import "c"
927
928 // 2020-1021 replaced poll() with channel/select
929 // // 2020-0906 added,
930 // func CPollIn(epos File, timeoutUs int)(ready uintptr){
931 //     var fdv = C.pollFdv{
932 //         var nfds = 1
933 //         var timeout = timeoutUs/1000
934 //         fdv.fdv[0].fd = C.int(ep.Fd())
935 //         fdv.fdv[0].events = C.POLLIN
936 //         if( 0 < EventRecvFd {
937 //             fdv.fdv[1].fd = C.int(EventRecvFd)
938 //             fdv.fdv[1].events = C.POLLIN
939 //             nfds += 1
940 //         }
941 //         r := C.pollx(&fdv,C.int(nfds),C.int(timeout))
942 //         if( r <= 0 ){
943 //             return 0
944 //         }
945 //         if (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
946 //             //fprintf(stderr, "--De-- got Event\n");
947 //             return uintptr(EventFdOffset + fdv.fdv[1].fd)
948 //         }
949 //         if (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
950 //             return uintptr(NormalFdOffset + fdv.fdv[0].fd)
951 //         }
952 //         return 0
953 //     }
954 // }
955 /*
956
957 const (
958     NAME = "gsh"
959     VERSION = "0.8.5"
960     DATE = "2020-11-22"
961     AUTHOR = "SatoxITS("-)://"
962 )
963 var (
964     GSH_HOME = ".gsh" // under home directory
965     GSH_PORT = 9999
966     MaxStreamSize = int64(128*1024*1024*1024) // 128GiB is too large?
967     PROMPT = ">"
968     LINESIZE = (8*1024)
969     PATHSEP = "/" // should be ";" in Windows
970     DIRSEP = "/" // canbe \ in Windows
971     OnWindows = false;
972 )
973 func initGshEnv(){
974     if( runtime.GOOS == "windows" ){
975         PATHSEP = ";";
976         DIRSEP = "\\";
977         OnWindows = true;
978     }else{
979     }
980 }
981
982 // -X logging control
983 // --A-- all
984 // --I-- info.
985 // --D-- debug
986 // --T-- time and resource usage
987 // --W-- warning
988 // --E-- error
989 // --F-- fatal error
990 // --Xn- network
991
992 // <a name="struct">Structures</a>
993
994 // 2020-1022 Unix/Windows
995 // -----
996 //type aStat_t syscall.Stat_t;
997 //type aStat_t struct { syscall.Stat_t }
998 type aStat_t struct {
999     Size int64
1000     Mode os.FileMode
1001     Rdev int64
1002     Blocks int64
1003     Nlink int64
1004 }
1005 func aLstat(path string, astat *aStat_t)(error){
1006     /*
1007     sstat := syscall.Stat_t{};
1008     err := syscall.Lstat(path,&sstat);
1009     *astat = astat_t(sstat);
1010     */
1011     fi,err := os.Stat(path);
1012     if( err == nil ){
1013         astat.Mode = fi.Mode();
1014         astat.Size = fi.Size();
1015     }
1016     return err;
1017 }
1018
1019 func aFstat(fd int, astat *aStat_t)(error){
1020     /*
1021     sstat := syscall.Stat_t{};
1022     err := syscall.Fstat(fd,&sstat);
1023     *astat = astat_t(sstat);
1024     */
1025     err := errors.New("NotImplemented-Fstat");
1026     //fmt.Printf("----E-- fstat(%v)(%v)\n",fd,err);
1027     return err;
1028 }
1029
1030 func aAccess(path string, mode uint32)(error){
1031     //err := syscall.Access(path,mode);
1032     //err := errors.New("NotImplemented-Access");
1033     fi,err := os.Stat(path)
1034     //fmt.Printf("----Access(%v,%v)\n(%v)\n",path,mode,err);
1035     if( err == nil ){
1036         fmode := fi.Mode();
1037         if( fmode.IsRegular() ){
1038             perm := fmode.Perm();
1039             if( (uint32(perm) & mode) != 0 ){
1040                 return nil;
1041             }
1042             return errors.New("NotAccessible");
1043         }
1044         return errors.New("NotRegularFile");
1045     }
1046     return err;
1047 }
1048 // 2020-1022 Unix/Windows
1049 // -----
1050 type aRusage struct {
1051     syscall.Rusage
1052     Utime time.Duration
1053     Stime time.Duration
1054     //Sys interface{}
1055 }
1056
1057 /*
1058 const aRUSAGE_SELF = syscall.RUSAGE_SELF
1059 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
1060 */
1061 const aRUSAGE_SELF = 0

```

```

1062 const aRUSAGE_CHILDREN = 1
1063 func aGetrusage(sel int, ru *aRusage){
1064 /*
1065 sysru := syscall.Rusage{};
1066 syscall.Getrusage(sel, &sysru);
1067 ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
1068 ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
1069 */
1070 }
1071 func aSetrusage(ru *aRusage, ps *os.ProcessState){
1072 ru.Utime = ps.UserTime();
1073 ru.Stime = ps.SystemTime();
1074 }
1075 func showRusage(what string, argv []string, ru *aRusage){
1076 fmt.Printf("%s: ", what);
1077 //fmt.Printf("User=%d.%06ds", ru.Utime.Sec, ru.Utime.Usec)
1078 //fmt.Printf("Sys=%d.%06ds", ru.Stime.Sec, ru.Stime.Usec)
1079 fmt.Printf("User=%d.%06ds", ru.Utime/1000000000, (ru.Utime/1000)%1000000);
1080 fmt.Printf(" Sys=%d.%06ds", ru.Stime/1000000000, (ru.Stime/1000)%1000000);
1081 /*
1082 fmt.Printf(" Rss=%vB", ru.Maxrss)
1083 if isin("-l", argv) {
1084     fmt.Printf(" MinFlt=%v", ru.Minflt)
1085     fmt.Printf(" MajFlt=%v", ru.Majflt)
1086     fmt.Printf(" IdrSS=%vB", ru.IdrSS)
1087     fmt.Printf(" IdRSS=%vB", ru.IdrSS)
1088     fmt.Printf(" Nswap=%vB", ru.Nswap)
1089     fmt.Printf(" Read=%v", ru.Inblock)
1090     fmt.Printf(" Write=%v", ru.Oblock)
1091 }
1092 fmt.Printf(" Snd=%v", ru.Msgsnd)
1093 fmt.Printf(" Rcv=%v", ru.Msgrcv)
1094 //if isin("-l", argv) {
1095     fmt.Printf(" Sig=%v", ru.Nsignals)
1096 //}
1097 */
1098 }
1099 }
1100 }
1101 type GCommandHistory struct {
1102 StartAt time.Time // command line execution started at
1103 EndAt time.Time // command line execution ended at
1104 ResCode int // exit code of (external command)
1105 CmdError error // error string
1106 OutData *os.File // output of the command
1107 FoundFile []string // output - result of ufind
1108 Rusagev [2]aRusage // Resource consumption, CPU time or so
1109 CmdId int // maybe with identified with arguments or impact
1110 // redirection commands should not be the CmdId
1111 WorkDir string // working directory at start
1112 WorkDirX int // index in ChdirHistory
1113 CmdLine string // command line
1114 }
1115 type GChdirHistory struct {
1116 Dir string
1117 MovedAt time.Time
1118 CmdIndex int
1119 }
1120 type CmdMode struct {
1121 Background bool
1122 }
1123 type Event struct {
1124 when time.Time
1125 event int
1126 evarg int64
1127 CmdIndex int
1128 }
1129 var CmdIndex int
1130 var Events []Event
1131 type PluginInfo struct {
1132 Spec *plugin.Plugin
1133 Addr plugin.Symbol
1134 Name string // maybe relative
1135 Path string // this is in Plugin but hidden
1136 }
1137 type GServer struct {
1138 host string
1139 port string
1140 }
1141 }
1142 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
1143 const ( // SumType
1144 SUM_ITEMS = 0x000001 // items count
1145 SUM_SIZE = 0x000002 // data length (simply added)
1146 SUM_DATEHASH = 0x000004 // data length (hashed sequence)
1147 SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
1148 // also envelope attributes like time stamp can be a part of digest
1149 // hashed value of sizes or mod-date of files will be useful to detect changes
1150
1151 SUM_WORDS = 0x000010 // word count is a kind of digest
1152 SUM_LINES = 0x000020 // line count is a kind of digest
1153 SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
1154
1155 SUM_SUM32_BITS = 0x000100 // the number of true bits
1156 SUM_SUM32_2BYTE = 0x000200 // 16bits words
1157 SUM_SUM32_4BYTE = 0x000400 // 32bits words
1158 SUM_SUM32_8BYTE = 0x000800 // 64bits words
1159
1160 SUM_SUM16_BSD = 0x001000 // UNIXsum -sum -bsd
1161 SUM_SUM16_SYSV = 0x002000 // UNIXsum -sum -sysv
1162 SUM_UNIXFILE = 0x004000
1163 SUM_CRCIEEE = 0x008000
1164 )
1165 type CheckSum struct {
1166 Files int64 // the number of files (or data)
1167 Size int64 // content size
1168 Words int64 // word count
1169 Lines int64 // line count
1170 SumType int
1171 Sum64 uint64
1172 Crc32Table crc32.Table
1173 Crc32Val uint32
1174 Sum16 int
1175 Ctime time.Time
1176 Atime time.Time
1177 Mtime time.Time
1178 Start time.Time
1179 Done time.Time
1180 RusgAtStart [2]aRusage
1181 RusgAtEnd [2]aRusage
1182 }
1183 type ValueStack []string
1184 type GshContext struct {
1185 StartDir string // the current directory at the start
1186 Getline string // gsh-getline command as a input line editor
1187 ChdirHistory []GChdirHistory // the 1st entry is wd at the start
1188 //gshPA syscall.ProcAttr
1189 gshPA os.ProcAttr
1190 CommandHistory []GCommandHistory
1191 CmdCurrent GCommandHistory
1192 Background bool
1193 BackgroundJobs []os.ProcessState; ///[]int
1194 LastRusage aRusage
1195 GshHomeDir string
1196 TerminalId int
1197 CmdTrace bool // should be [map]
1198 CmdTime bool // should be [map]
1199 PluginFuncs []PluginInfo
1200 iValues []string
1201 iDelimiter string // field separator of print out
1202 iFormat string // default print format (of integer)
1203 iValStack ValueStack
1204 LastServer GServer
1205 RSERV string // [gsh://host[:port]
1206 RWD string // remote (target, there) working directory
1207 lastCheckSum CheckSum
1208 }
1209 }
1210 func nsleep(ns time.Duration){
1211 time.Sleep(ns)
1212 }
1213 func usleep(ns time.Duration){
1214 nsleep(ns*1000)
1215 }
1216 func msleep(ns time.Duration){
1217 nsleep(ns*1000000)
1218 }
1219 func sleep(ns time.Duration){
1220 nsleep(ns*1000000000)
1221 }
1222 }
1223 func strBegins(str, pat string)(bool){
1224 if len(pat) <= len(str){
1225     yes := str[0:len(pat)] == pat
1226     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n", str, pat, yes)
1227     return yes
1228 }
1229 //fmt.Printf("--D-- strBegins(%v,%v)=%v\n", str, pat, false)
1230 return false
1231 }
1232 }
1233 func isin(what string, list []string) bool {
1234 for _, v := range list {
1235     if v == what {
1236         return true
1237     }
1238 }
1239 return false
1240 }

```

```

1239 }
1240 func isinK(what string,list[]string)(int){
1241     for i,v := range list {
1242         if v == what {
1243             return i
1244         }
1245     }
1246     return -1
1247 }
1248
1249 func env(opts []string) {
1250     env := os.Environ()
1251     if isin("-s",opts){
1252         sort.Slice(env, func(i,j int) bool {
1253             return env[i] < env[j]
1254         })
1255     }
1256     for _, v := range env {
1257         fmt.Printf("%v\n",v)
1258     }
1259 }
1260
1261 // - rewriting should be context dependent
1262 // - should postpone until the real point of evaluation
1263 // - should rewrite only known notation of symbol
1264 func scanInt(str string)(val int,leng int){
1265     leng = -1
1266     for i,ch := range str {
1267         if '0' <= ch && ch <= '9' {
1268             leng = i+1
1269         }else{
1270             break
1271         }
1272     }
1273     if 0 < leng {
1274         ival := strconv.Atoi(str[0:leng])
1275         return ival,leng
1276     }else{
1277         return 0,0
1278     }
1279 }
1280 func substHistory(gshCtx *GshContext,str string,i int,rstr string)(leng int,rst string){
1281     if len(str[i+1:]) == 0 {
1282         return 0,rstr
1283     }
1284     hi := 0
1285     histlen := len(gshCtx.CommandHistory)
1286     if str[i+1] == '!' {
1287         hi = histlen - 1
1288         leng = 1
1289     }else{
1290         hi,leng = scanInt(str[i+1:])
1291         if leng == 0 {
1292             return 0,rstr
1293         }
1294         if hi < 0 {
1295             hi = histlen + hi
1296         }
1297     }
1298     if 0 <= hi && hi < histlen {
1299         var ext byte
1300         if 1 < len(str[i+leng:]) {
1301             ext = str[i+leng:][1]
1302         }
1303         //fmt.Printf("---D-- %v(%c)\n",str[i+leng:],str[i+leng])
1304         if ext == 'f' {
1305             leng += 1
1306             xlist := []string{}
1307             list := gshCtx.CommandHistory[hi].FoundFile
1308             for _,v := range list {
1309                 //list[i] = escapeWhiteSP(v)
1310                 xlist = append(xlist,escapeWhiteSP(v))
1311             }
1312             //rstr += strings.Join(list, " ")
1313             rstr += strings.Join(xlist, " ")
1314         }else{
1315             if ext == 'g' || ext == 'd' {
1316                 // /!N! .. workdir at the start of the command
1317                 leng += 1
1318                 rstr += gshCtx.CommandHistory[hi].WorkDir
1319             }else{
1320                 rstr += gshCtx.CommandHistory[hi].CmdLine
1321             }
1322         }else{
1323             leng = 0
1324         }
1325     }
1326     return leng,rstr
1327 }
1328 func escapeWhiteSP(str string)(string){
1329     if len(str) == 0 {
1330         return "\\z" // empty, to be ignored
1331     }
1332     rstr := ""
1333     for _,ch := range str {
1334         switch ch {
1335             case '\\': rstr += "\\\\"
1336             case ' ': rstr += "\\s"
1337             case '\t': rstr += "\\t"
1338             case '\n': rstr += "\\n"
1339             case '\r': rstr += "\\r"
1340             default: rstr += string(ch)
1341         }
1342     }
1343     return rstr
1344 }
1345 func unescapeWhiteSP(str string)(string){ // strip original escapes
1346     rstr := ""
1347     for i := 0; i < len(str); i++ {
1348         ch := str[i]
1349         if ch == '\\' {
1350             if i+1 < len(str) {
1351                 switch str[i+1] {
1352                     case 'z':
1353                         continue;
1354                 }
1355             }
1356             rstr += string(ch)
1357         }
1358     }
1359     return rstr
1360 }
1361 func unescapeWhiteSPV(strv []string)([]string){ // strip original escapes
1362     ustrv := []string{}
1363     for _,v := range strv {
1364         ustrv = append(ustrv,unescapeWhiteSP(v))
1365     }
1366     return ustrv
1367 }
1368 // <a name="comexpansion">str-expansion</a>
1369 // - this should be a macro processor
1370 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
1371     rbuff := []byte{}
1372     if false {
1373         //@@@ Unicode should be cared as a character
1374         return str
1375     }
1376     //rstr := ""
1377     inEsc = 0 // escape characer mode
1378     for i := 0; i < len(str); i++ {
1379         //fmt.Printf("---D--Subst %v:%v\n",i,str[i:])
1380         ch := str[i]
1381         if inEsc == 0 {
1382             if ch == '!' {
1383                 //leng,xrstr := substHistory(gshCtx,str,i,rstr)
1384                 leng,rs := substHistory(gshCtx,str,i,"")
1385                 if 0 < leng {
1386                     //rs := substHistory(gshCtx,str,i,"")
1387                     rbuff = append(rbuff,[]byte(rs)...)
1388                     i += leng
1389                     //rstr = xrstr
1390                     continue
1391                 }
1392             }
1393             switch ch {
1394                 case '\\': inEsc = '\\'; continue
1395                 //case '$': inEsc = '$'; continue
1396                 case '$':
1397             }
1398         }
1399         switch inEsc {
1400             case '\\':
1401                 switch ch {
1402                     case '\\': ch = '\\'
1403                     case 's': ch = ' '
1404                     case 't': ch = '\t'
1405                     case 'r': ch = '\r'
1406                     case 'n': ch = '\n'
1407                     case 'z': inEsc = 0; continue // empty, to be ignored
1408                 }
1409             case '$':
1410                 switch {
1411                     case ch == '$': ch = '$'
1412                     case ch == 'T':
1413                         //rstr = rstr + time.Now().Format(time.Stamp)
1414                     case ch == 't':
1415                         rs := time.Now().Format(time.Stamp)

```



```

1416 rbuff = append(rbuff,[]byte(rs)...)
1417         inEsc = 0
1418         continue;
1419         default:
1420             // postpone the interpretation
1421             //rstr = rstr + "%" + string(ch)
1422             rbuff = append(rbuff,ch)
1423             inEsc = 0
1424             continue;
1425         }
1426         inEsc = 0
1427     }
1428     //rstr = rstr + string(ch)
1429     rbuff = append(rbuff,ch)
1430 }
1431 //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
1432 return string(rbuff)
1433 //return rstr
1434 }
1435 func showFileInfo(path string, opts []string) {
1436     if isin("-l",opts) || isin("-ls",opts) {
1437         fi, err := os.Stat(path)
1438         if err != nil {
1439             fmt.Printf("----- (%s)",err)
1440         }else{
1441             mod := fi.ModTime()
1442             date := mod.Format(time.Stamp)
1443             fmt.Printf("%v %v %s %s",fi.Mode(),fi.Size(),date)
1444         }
1445     }
1446     fmt.Printf("%s",path)
1447     if isin("-sp",opts) {
1448         fmt.Printf(" ")
1449     }else
1450     if ! isin("-n",opts) {
1451         fmt.Printf("\n")
1452     }
1453 }
1454 func userHomeDir()(string,bool){
1455     /*
1456     homedir, _ = os.UserHomeDir() // not implemented in older Golang
1457     */
1458     homedir,found := os.LookupEnv("HOME")
1459     //fmt.Printf("--I-- HOME=%v(%v)\n",homedir,found)
1460     if !found {
1461         return "/tmp",found
1462     }
1463     return homedir,found
1464 }
1465 }
1466 func toFullpath(path string) (fullpath string) {
1467     if path[0] == '/' {
1468         return path
1469     }
1470     pathv := strings.Split(path,DIRSEP)
1471     switch {
1472     case pathv[0] == ".":
1473         pathv[0], _ = os.Getwd()
1474     case pathv[0] == "..": // all ones should be interpreted
1475         cwd, _ = os.Getwd()
1476         ppathv := strings.Split(cwd,DIRSEP)
1477         pathv[0] = strings.Join(ppathv,DIRSEP)
1478     case pathv[0] == "-":
1479         pathv[0], _ = userHomeDir()
1480     default:
1481         cwd, _ = os.Getwd()
1482         pathv[0] = cwd + DIRSEP + pathv[0]
1483     }
1484     return strings.Join(pathv,DIRSEP)
1485 }
1486 }
1487 func IsRegFile(path string)(bool){
1488     fi, err := os.Stat(path)
1489     if err != nil {
1490         fm := fi.Mode()
1491         return fm.IsRegular();
1492     }
1493     return false
1494 }
1495 }
1496 // <a name="encode">Encode / Decode</a>
1497 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
1498 func (gshCtx *GshContext)Enc(argv|string){
1499     file := os.Stdin
1500     buff := make([]byte,LINESIZE)
1501     li := 0
1502     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
1503     for li = 0; ; li++ {
1504         count, err := file.Read(buff)
1505         if count <= 0 {
1506             break
1507         }
1508         if err != nil {
1509             break
1510         }
1511         encoder.Write(buff[0:count])
1512     }
1513     encoder.Close()
1514 }
1515 func (gshCtx *GshContext)Dec(argv|string){
1516     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1517     li := 0
1518     buff := make([]byte,LINESIZE)
1519     for li = 0; ; li++ {
1520         count, err := decoder.Read(buff)
1521         if count <= 0 {
1522             break
1523         }
1524         if err != nil {
1525             break
1526         }
1527         os.Stdout.Write(buff[0:count])
1528     }
1529 }
1530 // lnspl [N] [-crlf][-C \\\
1531 func (gshCtx *GshContext)SplitLine(argv|string){
1532     strRep := isin("-str",argv) // "-"
1533     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1534     ni := 0
1535     toi := 0
1536     for ni = 0; ; ni++ {
1537         line, err := reader.ReadString('\n')
1538         if len(line) <= 0 {
1539             if err != nil {
1540                 fmt.Fprintf(os.Stderr,"--I-- lnspl %d to %d (%s)\n",ni,toi,err)
1541                 break
1542             }
1543         }
1544         off := 0
1545         ilen := len(line)
1546         rmlen := len(line)
1547         if strRep { os.Stdout.Write([]byte("\n")) }
1548         for oi = 0; 0 < rmlen; oi++ {
1549             olen := rmlen
1550             addnl := false
1551             if 72 < olen {
1552                 olen = 72
1553                 addnl = true
1554             }
1555             fmt.Fprintf(os.Stderr,"--D-- write %d [%d.%d] %d %d/%d\n",
1556                 toi,ni,oi,off,olen,rmlen,ilen)
1557             toi += 1
1558             os.Stdout.Write([]byte(line[0:olen]))
1559             if addnl {
1560                 if strRep {
1561                     os.Stdout.Write([]byte("\n\n"))
1562                 }else{
1563                     //os.Stdout.Write([]byte("\n\n"))
1564                     os.Stdout.Write([]byte("\n"))
1565                     os.Stdout.Write([]byte("\n"))
1566                 }
1567             }
1568             line = line[olen:]
1569             off += olen
1570             rmlen -= olen
1571         }
1572         if strRep { os.Stdout.Write([]byte("\n")) }
1573     }
1574     fmt.Fprintf(os.Stderr,"--I-- lnspl %d to %d\n",ni,toi)
1575 }
1576 }
1577 // CRC32 <a href="http://golang.jp/pkg/hash-crc32">crc32</a>
1578 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1579 var CRC32UNIX uint32 = uint32(0x94c11d97) // Unix cksum
1580 var CRC32IEEE uint32 = uint32(0xEDB88320)
1581 func byteCRC32add(crc uint32,str[]byte,len uint64)(uint32){
1582     var oi uint64
1583     for oi = 0; oi < len; oi++ {
1584         var oct = str[oi]
1585         for bi := 0; bi < 8; bi++ {
1586             //fmt.Fprintf(stderr,"--CRC32 %d %X (%d.%d)\n",crc,oct,oi,bi)
1587             ovf1 := (crc & 0x80000000) != 0
1588             ovf2 := (oct & 0x80) != 0
1589             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1590             oct <<= 1
1591             crc <<= 1
1592             if ovf { crc ^= CRC32UNIX }

```

```

1593     }
1594     //fprintf(stderr, "--CRC32 return %d %d\n", crc, len)
1595     return crc;
1596 }
1597 func byteCRC32end(crc uint32, len uint64)(uint32){
1598     var slen = make([]byte,4)
1599     var li = 0
1600     for li = 0; li < 4; {
1601         slen[li] = byte(len)
1602         li += 1
1603         len >= 8
1604         if( len == 0 ){
1605             break
1606         }
1607     }
1608     crc = byteCRC32add(crc, slen, uint64(li))
1609     crc ^= 0xFFFFFFFF
1610     return crc
1611 }
1612 func strCRC32(str string, len uint64)(crc uint32){
1613     crc = byteCRC32add(0, []byte(str), len)
1614     crc = byteCRC32end(crc, len)
1615     //fprintf(stderr, "--CRC32 %d %d\n", crc, len)
1616     return crc
1617 }
1618 }
1619 func CRC32Finish(crc uint32, table *crc32.Table, len uint64)(uint32){
1620     var slen = make([]byte,4)
1621     var li = 0
1622     for li = 0; li < 4; {
1623         slen[li] = byte(len & 0xFF)
1624         li += 1
1625         len >= 8
1626         if( len == 0 ){
1627             break
1628         }
1629     }
1630     crc = crc32.Update(crc, table, slen)
1631     crc ^= 0xFFFFFFFF
1632     return crc
1633 }
1634 }
1635 func (gsh*GshContext)xChecksum(path string, argv []string, sum*Checksum)(int64){
1636     if !isin("-type/f", argv) && !IsRegFile(path){
1637         return 0
1638     }
1639     if !isin("-type/d", argv) && IsRegFile(path){
1640         return 0
1641     }
1642     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1643     if err != nil {
1644         fmt.Printf("--E-- cksum %v (%v)\n", path, err)
1645         return -1
1646     }
1647     defer file.Close()
1648     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v (%v)\n", path, argv) }
1649     bi := 0
1650     var buff = make([]byte, 32*1024)
1651     var total int64 = 0
1652     var initTime = time.Time{}
1653     if sum.Start == initTime {
1654         sum.Start = time.Now()
1655     }
1656     for bi = 0; ; bi++ {
1657         count, err := file.Read(buff)
1658         if count <= 0 || err != nil {
1659             break
1660         }
1661         if (sum.SumType & SUM_SUM64) != 0 {
1662             s := sum.Sum64
1663             for _, c := range buff[0:count] {
1664                 s += uint64(c)
1665             }
1666             sum.Sum64 = s
1667         }
1668         if (sum.SumType & SUM_UNIXFILE) != 0 {
1669             sum.Crc32Val = byteCRC32add(sum.Crc32Val, buff, uint64(count))
1670         }
1671         if (sum.SumType & SUM_CRCIEEE) != 0 {
1672             sum.Crc32Val = crc32.Update(sum.Crc32Val, &sum.Crc32Table, buff[0:count])
1673         }
1674         // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
1675         if (sum.SumType & SUM_SUM16_BSD) != 0 {
1676             s := sum.Sum16
1677             for _, c := range buff[0:count] {
1678                 s = (s >> 1) + ((s & 1) << 15)
1679                 s += int(c)
1680                 s ^= 0xFFFF
1681             }
1682             //fmt.Printf("BSDsum: %d[%d] %d\n", sum.Size+int64(i), i, s)
1683             sum.Sum16 = s
1684         }
1685         if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1686             for bj := 0; bj < count; bj++ {
1687                 sum.Sum16 += int(buff[bj])
1688             }
1689             total += int64(count)
1690         }
1691     }
1692     sum.Done = time.Now()
1693     sum.Files += 1
1694     sum.Size += total
1695     if !isin("-s", argv) {
1696         fmt.Printf("%v ", total)
1697     }
1698     return 0
1699 }
1700 }
1701 // <a name="grep">grep</a>
1702 // "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
1703 // a*, lab, c, ... sequential combination of patterns
1704 // what "LINE" is should be definable
1705 // generic line-by-line processing
1706 // grep [-v]
1707 // cat -n -v
1708 // uniq [-c]
1709 // tail -f
1710 // sed s/x/y/ or awk
1711 // grep with line count like wc
1712 // rewrite contents if specified
1713 func (gsh*GshContext)xGrep(path string, rexp []string)(int){
1714     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1715     if err != nil {
1716         fmt.Printf("--E-- grep %v (%v)\n", path, err)
1717         return -1
1718     }
1719     defer file.Close()
1720     if gsh.CmdTrace { fmt.Printf("--I-- grep %v (%v)\n", path, rexp) }
1721     //reader := bufio.NewReaderSize(file, LINESIZE)
1722     reader := bufio.NewReaderSize(file, 80)
1723     li := 0
1724     found := 0
1725     for li = 0; ; li++ {
1726         line, err := reader.ReadString('\n')
1727         if len(line) <= 0 {
1728             break
1729         }
1730         if 150 < len(line) {
1731             // maybe binary
1732             break;
1733         }
1734         if err != nil {
1735             break
1736         }
1737         if 0 < strings.Index(string(line), rexp[0]) {
1738             found += 1
1739             fmt.Printf("%s:%d: %s", path, li, line)
1740         }
1741     }
1742     //fmt.Printf("total %d lines %s\n", li, path)
1743     //if( 0 < found ){ fmt.Printf("(found %d lines %s)\n", found, path); }
1744     return found
1745 }
1746 }
1747 // <a name="finder">Finder</a>
1748 // finding files with it name and contents
1749 // file names are Ored
1750 // show the content with %x fmt list
1751 // ls -R
1752 // tar command by adding output
1753 type FileSum struct {
1754     Err int64 // access error or so
1755     Size int64 // content size
1756     DupSize int64 // content size from hard links
1757     Blocks int64 // number of blocks (of 512 bytes)
1758     DupBlocks int64 // Blocks pointed from hard links
1759     HLinks int64 // hard links
1760     Words int64
1761     Lines int64
1762     Files int64
1763     Dirs int64 // the num. of directories
1764     SymLink int64
1765     Flats int64 // the num. of flat files
1766     MaxDepth int64
1767     MaxNameLen int64 // max. name length
1768     nextRepo time.Time
1769 }

```

```

1770 }
1771 func showFusage(dir string, fusage *fileSum) {
1772     bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1773     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1774
1775     fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1776         dir,
1777         fusage.Files,
1778         fusage.Dirs,
1779         fusage.Symlink,
1780         fusage.HLinks,
1781         float64(fusage.Size)/1000000.0, bsume);
1782 }
1783 const (
1784     S_IFMT    = 0170000
1785     S_IFCHR   = 0020000
1786     S_IFDIR   = 0040000
1787     S_IFREG   = 0100000
1788     S_IFLNK   = 0120000
1789     S_IFSOCK  = 0140000
1790 )
1791 func cumFinfo(fsum *fileSum, path string, stater error, fstat aStat_t, argv[]string, verb bool)(*fileSum){
1792     now := time.Now()
1793     if time.Second <= now.Sub(fsum.nextRepo) {
1794         if fsum.nextRepo.IsZero() {
1795             tstamp := now.Format(time.Stamp)
1796             showFusage(tstamp, fsum)
1797         }
1798         fsum.nextRepo = now.Add(time.Second)
1799     }
1800     if stater != nil {
1801         fsum.Err += 1
1802         return fsum
1803     }
1804     fsum.Files += 1
1805     if l < fstat.Nlink {
1806         // must count only once...
1807         // at least ignore ones in the same directory
1808         // if info.Mode() != Regular() {
1809             if (fstat.Mode & S_IFMT) == S_IFREG {
1810                 fsum.HLinks += 1
1811                 fsum.DupBlocks += int64(fstat.Blocks)
1812                 //fmt.Printf("---Dup HardLink %v %s\n", fstat.Nlink, path)
1813             }
1814         }
1815         //fsum.Size += finfo.Size()
1816         fsum.Size += fstat.Size
1817         fsum.Blocks += int64(fstat.Blocks)
1818         //if verb { fmt.Printf("%8dBk %s", fstat.Blocks/2, path) }
1819         if !isIn("-ls", argv) {
1820             //if verb { fmt.Printf("%4d %8d ", fstat.Blksize, fstat.Blocks) }
1821             //fmt.Printf("%d\t", fstat.Blocks/2)
1822         }
1823         //if finfo.IsDir()
1824         if (fstat.Mode & S_IFMT) == S_IFDIR {
1825             fsum.Dirs += 1
1826         }
1827         //if (finfo.Mode() & os.ModeSymlink) != 0
1828         if (fstat.Mode & S_IFMT) == S_IFLNK {
1829             //if verb { fmt.Printf("symlink(%v, %s)\n", fstat.Mode, finfo.Name()) }
1830             //if verb { fmt.Printf("symlink(%d, %s)\n", fstat.Mode, finfo.Name()) }
1831             fsum.Symlink += 1
1832         }
1833         return fsum
1834     }
1835 func (gsh*GshContext)xxFindEntv(depth int, total *fileSum, dir string, dstat aStat_t, ei int, entv []string, npatv[]string, argv[]string)(*fileSum){
1836     nols := isin("-grep", argv)
1837     // sort entv
1838     /*
1839     if isin("-t", argv){
1840         sort.Slice(filev, func(i, j int) bool {
1841             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1842         })
1843     }
1844     */
1845     /*
1846     if isin("-u", argv){
1847         sort.Slice(filev, func(i, j int) bool {
1848             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1849         })
1850     }
1851     if isin("-U", argv){
1852         sort.Slice(filev, func(i, j int) bool {
1853             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1854         })
1855     }
1856     */
1857     /*
1858     if isin("-s", argv){
1859         sort.Slice(filev, func(i, j int) bool {
1860             return filev[j].Size() < filev[i].Size()
1861         })
1862     }
1863     */
1864     for _, filename := range entv {
1865         for _, npat := range npatv {
1866             match := true
1867             if npat == "*" {
1868                 match = true
1869             } else {
1870                 match, _ = filepath.Match(npat, filename)
1871             }
1872             path := dir + DIRSEP + filename
1873             if !match {
1874                 continue
1875             }
1876             var fstat aStat_t
1877             stater := aLstat(path, &fstat)
1878             if stater != nil {
1879                 if !isin("-w", argv){fmt.Printf("ufind: %v\n", stater)}
1880                 continue;
1881             }
1882             if isin("-du", argv) && (fstat.Mode & S_IFMT) == S_IFDIR {
1883                 // should not show size of directory in "-du" mode ...
1884             } else {
1885                 if !nols && !isin("-s", argv) && (!isin("-du", argv) || !isin("-a", argv)) {
1886                     if !isin("-du", argv) {
1887                         fmt.Printf("%d\t", fstat.Blocks/2)
1888                     }
1889                     showFileInfo(path, argv)
1890                 }
1891                 if true { // && isin("-du", argv)
1892                     total = cumFinfo(total, path, stater, fstat, argv, false)
1893                 }
1894             }
1895             if isin("-wc", argv) {
1896                 /*
1897                 if gsh.lastCheckSum.SumType != 0 {
1898                     gsh.xCksum(path, argv, &gsh.lastCheckSum);
1899                 }
1900                 x := isinX("-grep", argv); // -grep will be convenient like -ls
1901                 if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1902                     if IsRegFile(path){
1903                         found := gsh.xGrep(path, argv[x+1])
1904                         if 0 < found {
1905                             foundv := gsh.CmdCurrent.FoundFile
1906                             if len(foundv) < 10 {
1907                                 gsh.CmdCurrent.FoundFile =
1908                                     append(gsh.CmdCurrent.FoundFile, path)
1909                             }
1910                         }
1911                     }
1912                 }
1913                 if !isin("-r0", argv) { // -d 0 in du, -depth n in find
1914                     //total.Depth += 1
1915                     if (fstat.Mode & S_IFMT) == S_IFLNK {
1916                         continue
1917                     }
1918                     if dstat.Rdev != fstat.Rdev {
1919                         fmt.Printf("--I- don't follow differnet device %v(%v) %v(%v)\n",
1920                             dir, dstat.Rdev, path, fstat.Rdev)
1921                     }
1922                     if (fstat.Mode & S_IFMT) == S_IFDIR {
1923                         total = gsh.xxFind(depth+1, total, path, npatv, argv)
1924                     }
1925                 }
1926             }
1927         }
1928     }
1929     return total
1930 }
1931 func (gsh*GshContext)xxFind(depth int, total *fileSum, dir string, npatv[]string, argv[]string)(*fileSum){
1932     nols := isin("-grep", argv)
1933     dirfile, oerr := os.OpenFile(dir, os.O_RDONLY, 0)
1934     if oerr == nil {
1935         //fmt.Printf("--I- %v(%v)[%d]\n", dir, dirfile, dirfile.Pd())
1936         defer dirfile.Close()
1937     } else {
1938     }
1939     prev := *total
1940     var dstat aStat_t
1941     stater := aLstat(dir, &dstat) // should be flstat
1942     if stater != nil {
1943         if !isin("-w", argv){fmt.Printf("ufind: %v\n", stater)}
1944         return total
1945     }

```

```

1947 }
1948 //filev,err := ioutil.ReadDir(dir)
1949 //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1950 /*
1951 if err != nil {
1952     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1953     return total
1954 }
1955 */
1956 if depth == 0 {
1957     total = cumFinfo(total,dir,staterr,dstat,argv,true)
1958     if !nols && !isin("-s",argv) && (!isin("-du",argv) || !isin("-a",argv)) {
1959         showFileinfo(dir,argv)
1960     }
1961 }
1962 // it is not a directory, just scan it and finish
1963
1964 for ei := 0; ei++ {
1965     envr,rderr := dirfile.Readdirnames(8*1024)
1966     if len(envr) == 0 || rderr != nil {
1967         //if rderr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(envr),rderr) }
1968         break
1969     }
1970     if 0 < ei {
1971         fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(envr),dir)
1972     }
1973     total = gsh.xxFindEntv(depth,total,dir,dstat,ei,envr,ndpatv,argv)
1974 }
1975 if !isin("-du",argv) {
1976     // if in "du" mode
1977     fmt.Printf("%d\t%s\n", (total.Blocks-prev.Blocks)/2,dir)
1978 }
1979 return total
1980 }
1981
1982 // {ufind|fu|ls} [Files] [Names] [-- Expressions]
1983 // Files is "-" by default
1984 // Names is "+" by default
1985 // Expressions is "-print" by default for "ufind", or "-du" for "fu" command
1986 func (gsh*GshContext).xFind(argv[]string){
1987     if 0 < len(argv) && strBegins(argv[0],"?"){
1988         showFound(gsh,argv)
1989         return
1990     }
1991     if !isin("-cksum",argv) || !isin("-sum",argv) {
1992         gsh.lastCheckSum = CheckSum{}
1993         if !isin("-sum",argv) && !isin("-add",argv) {
1994             gsh.lastCheckSum.SumType = SUM_SUM64
1995         }else{
1996             if !isin("-sum",argv) && !isin("-size",argv) {
1997                 gsh.lastCheckSum.SumType = SUM_SIZE
1998             }else{
1999                 if !isin("-sum",argv) && !isin("-bsd",argv) {
2000                     gsh.lastCheckSum.SumType = SUM_SUM16_BSD
2001                 }else{
2002                     if !isin("-sum",argv) && !isin("-sysv",argv) {
2003                         gsh.lastCheckSum.SumType = SUM_SUM16_SYSV
2004                     }else{
2005                         if !isin("-sum",argv) {
2006                             gsh.lastCheckSum.SumType = SUM_SUM64
2007                         }
2008                     }
2009                     if !isin("-unix",argv) {
2010                         gsh.lastCheckSum.SumType = SUM_UNIXFILE
2011                         gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32UNIX)
2012                     }
2013                     if !isin("-leee",argv){
2014                         gsh.lastCheckSum.SumType = SUM_CRCIEEE
2015                         gsh.lastCheckSum.Crc32Table = *Crc32.MakeTable(CRC32IEEE)
2016                     }
2017                     gsh.lastCheckSum.RusageAtStart = Getrusagev()
2018                 }
2019             }
2020             var total = filesSum{}
2021             npats = []string{}
2022             for v := range argv {
2023                 if 0 < len(v) && v[0] != '-' {
2024                     npats = append(npats,v)
2025                 }
2026                 if v == "/" { break }
2027                 if v == "--" { break }
2028                 if v == "-grep" { break }
2029                 if v == "-ls" { break }
2030             }
2031             if len(npats) == 0 {
2032                 npats = []string{"*"}
2033             }
2034             cwd := "."
2035             // if to be fullpath :: cwd, _ := os.Getwd()
2036             if len(npats) == 0 { npats = []string{"*"} }
2037             fusage := gsh.xxFind(0,total,cwd,npats,argv)
2038             if gsh.lastCheckSum.SumType != 0 {
2039                 var sumi uint64 = 0
2040                 sum := gsh.lastCheckSum
2041                 if (sum.SumType & SUM_SIZE) != 0 {
2042                     sumi = uint64(sum.Size)
2043                 }
2044                 if (sum.SumType & SUM_SUM64) != 0 {
2045                     sumi = sum.Sum64
2046                 }
2047                 if (sum.SumType & SUM_SUM16_SYSV) != 0 {
2048                     s := uint32(sum.Sum16)
2049                     r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
2050                     s = (r & 0xFFFF) + (r >> 16)
2051                     sum.Crc32Val = uint32(s)
2052                     sumi = uint64(s)
2053                 }
2054                 if (sum.SumType & SUM_SUM16_BSD) != 0 {
2055                     sum.Crc32Val = uint32(sum.Sum16)
2056                     sumi = uint64(sum.Sum16)
2057                 }
2058                 if (sum.SumType & SUM_UNIXFILE) != 0 {
2059                     sum.Crc32Val = byteCrc32end(sum.Crc32Val,uint64(sum.Size))
2060                     sumi = uint64(byteCrc32end(sum.Crc32Val,uint64(sum.Size)))
2061                 }
2062                 if l < sum.Files {
2063                     fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
2064                         sumi,sum.Size,
2065                         abszize(sum.Size),sum.Files,
2066                         abszize(sum.Size/sum.Files))
2067                 }else{
2068                     fmt.Printf("%v %v %v\n",
2069                         sumi,sum.Size,npats[0])
2070                 }
2071             }
2072             if !isin("-grep",argv) {
2073                 showFusage("total",fusage)
2074             }
2075             if !isin("-s",argv){
2076                 hits := len(gsh.CmdCurrent.FoundFile)
2077                 if 0 < hits {
2078                     fmt.Printf("--I-- %d files hits // can be refered with %df\n",
2079                         hits,len(gsh.CommandHistory))
2080                 }
2081             }
2082             if gsh.lastCheckSum.SumType != 0 {
2083                 if !isin("-ru",argv) {
2084                     sum := gsh.lastCheckSum
2085                     sum.Done = time.Now()
2086                     gsh.lastCheckSum.RusageAtEnd = Getrusagev()
2087                     elps := sum.Done.Sub(sum.Start)
2088                     fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\r\n",
2089                         sum.Size,abszize(sum.Size),sum.Files,abszize(sum.Size/sum.Files))
2090                     nanos := int64(elps)
2091                     dnanos := time.Duration(nanos)
2092                     fmt.Printf("--cksum-time: %v/total, %v/file, %lf files/s, %v\r\n",
2093                         abbtme(dnanos),
2094                         abbtme(time.Duration(nanos/sum.Files)),
2095                         (float64(sum.Files)*1000000000.0)/float64(nanos),
2096                         abbspd(sum.Size,nanos))
2097                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2098                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2099                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2100                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2101                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2102                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2103                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2104                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2105                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2106                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2107                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2108                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2109                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2110                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2111                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2112                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2113                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2114                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2115                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2116                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2117                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2118                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2119                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2120                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2121                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2122                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)
2123                     diff := RusageSubv(sum.RusageAtEnd,sum.RusageAtStart)

```

```

2124 }
2125 }
2126
2127 func showMatchFile(filev []os.FileInfo, npat, dir string, argv []string)(string, bool){
2128     fname := ""
2129     found := false
2130     for _, v := range filev {
2131         match, _ := filepath.Match(npat, v.Name())
2132         if match {
2133             fname = v.Name()
2134             found = true
2135             //fmt.Printf("[%d] %s\n", i, v.Name())
2136             showIfExecutable(fname, dir, argv)
2137         }
2138     }
2139     return fname, found
2140 }
2141
2142 func showIfExecutable(name, dir string, argv []string)(ffullpath string, ffound bool){
2143     var fullpath string
2144     if strBegins(name, DIRSEP){
2145         fullpath = name
2146     }else{
2147         if len(dir) == 0 {
2148             fullpath = name;
2149         }else{
2150             fullpath = dir + DIRSEP + name
2151         }
2152         fi, err := os.Stat(fullpath)
2153         //fmt.Printf("--Dp-- \"%v\" -- %v\n", fullpath, err);
2154         if err != nil {
2155             fullpath = ".exe";
2156             fi, err = os.Stat(fullpath)
2157         }
2158         if err != nil {
2159             fullpath = dir + DIRSEP + name + ".go"
2160             fi, err = os.Stat(fullpath)
2161         }
2162         if err == nil {
2163             fm := fi.Mode()
2164             if fm.IsRegular() {
2165                 // R_OK=4, W_OK=2, X_OK=1, F_OK=0
2166                 if aAccess(fullpath, 5) == nil {
2167                     ffullpath = fullpath
2168                     ffound = true
2169                     if ! isin("-s", argv) {
2170                         showFileInfo(fullpath, argv)
2171                     }
2172                 }
2173             }
2174             return ffullpath, ffound
2175         }
2176     }
2177     func which(list string, argv []string) (fullpathv []string, itis bool){
2178         if len(argv) <= 1 {
2179             fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
2180             return []string{"", false}
2181         }
2182         path := argv[1]
2183         if strBegins(path, "/") {
2184             // should check if executable?
2185             , exOK := showIfExecutable(path, "", argv)
2186             fmt.Printf("--D- %v exOK=%v\n", path, exOK)
2187             return []string{path}, exOK
2188         }
2189         pathenv, efound := os.LookupEnv(list)
2190         if ! efound {
2191             fmt.Printf("--E- which: no \"%s\" environment\n", list)
2192             return []string{"", false}
2193         }
2194         //fmt.Printf("PATH=%v\n", pathenv);
2195         showall := isin("-a", argv) || 0 < strings.Index(path, "*")
2196         dirv := strings.Split(pathenv, PATHSEP)
2197         ffound := false
2198         ffullpath := path
2199         for dir := range dirv {
2200             if 0 < strings.Index(path, "*") { // by wild-card
2201                 list, _ := ioutil.ReadDir(dir)
2202                 ffullpath, ffound = showMatchFile(list, path, dir, argv)
2203             }else{
2204                 ffullpath, ffound = showIfExecutable(path, dir, argv)
2205             }
2206             //if ffound && ! isin("-a", argv) {
2207             if ffound && ! showall {
2208                 break;
2209             }
2210         }
2211         return []string{ffullpath}, ffound
2212     }
2213 }
2214
2215 func stripLeadingWSParg(argv []string) ([]string){
2216     for ; 0 < len(argv); {
2217         if len(argv[0]) == 0 {
2218             argv = argv[1:]
2219         }else{
2220             break
2221         }
2222     }
2223     return argv
2224 }
2225
2226 func xEval(argv []string, nlend bool){
2227     argv = stripLeadingWSParg(argv)
2228     if len(argv) == 0 {
2229         fmt.Printf("eval [%format] [Go-expression]\n")
2230         return
2231     }
2232     pfmt := "%v"
2233     if argv[0][0] == '$' {
2234         pfmt = argv[0]
2235         argv = argv[1:]
2236     }
2237     if len(argv) == 0 {
2238         return
2239     }
2240     gocode := strings.Join(argv, " ");
2241     //fmt.Printf("eval [%v] [%v]\n", pfmt, gocode)
2242     fset := token.NewFileSet()
2243     rval, _ := types.Eval(fset, nil, token.NoPos, gocode)
2244     fmt.Printf(pfmt, rval.Value)
2245     if nlend { fmt.Printf("\n") }
2246 }
2247
2248 func getval(name string) (found bool, val int) {
2249     /* should expand the name here */
2250     if name == "gsh.pid" {
2251         return true, os.Getpid()
2252     }else{
2253         if name == "gsh.ppid" {
2254             return true, os.Getppid()
2255         }
2256     }
2257     return false, 0
2258 }
2259
2260 func echo(argv []string, nlend bool){
2261     for ai := 1; ai < len(argv); ai++ {
2262         if 1 < ai {
2263             fmt.Printf(" ");
2264         }
2265         arg := argv[ai]
2266         found, val := getval(arg)
2267         if found {
2268             fmt.Printf("%d", val)
2269         }else{
2270             fmt.Printf("%s", arg)
2271         }
2272     }
2273     if nlend {
2274         fmt.Printf("\n");
2275     }
2276 }
2277
2278 func resfile() string {
2279     return "gsh.tmp"
2280 }
2281
2282 //var resF *File
2283 func resmap() {
2284     // , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
2285     // https://developer.com/solution-to-golang-bad-file-descriptor-problem/
2286     // , err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
2287     if err != nil {
2288         fmt.Printf("refF could not open: %s\n", err)
2289     }else{
2290         fmt.Printf("refF opened\n")
2291     }
2292 }
2293
2294 // @2020-0821
2295 func gshScanArg(str string, strip int)(argv []string){
2296     var si = 0
2297     var sb = 0
2298     var inBracket = 0
2299     var argl = make([]byte, LINESIZE)
2300     var ax = 0
2301     debug := false
2302     for ; si < len(str); si++ {
2303         if str[si] != ' ' {
2304             break

```

```

2301     }
2302   }
2303   sb = si
2304   for ; si < len(str); si++ {
2305     if sb <= si {
2306       if debug {
2307         fmt.Printf("--Da- +%d %2d-%2d %s ... %s\n",
2308           inBracket, sb, si, arg1[0:ax], str[si:])
2309       }
2310     }
2311     ch := str[si]
2312     if ch == '{' {
2313       inBracket++
2314       if 0 < strip && inBracket <= strip {
2315         //fmt.Printf("stripLEV %d <= %d?\n", inBracket, strip)
2316         continue
2317       }
2318     }
2319     if 0 < inBracket {
2320       if ch == '}' {
2321         inBracket--
2322         if 0 < strip && inBracket < strip {
2323           //fmt.Printf("stripLEV %d < %d?\n", inBracket, strip)
2324           continue
2325         }
2326       }
2327       arg1[ax] = ch
2328       ax++
2329       continue
2330     }
2331     if str[si] == ' ' {
2332       argv = append(argv, string(arg1[0:ax]))
2333       if debug {
2334         fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2335           -1+len(argv), sb, si, str[sb:si], string(str[si:]))
2336       }
2337       sb = si+1
2338       ax = 0
2339       continue
2340     }
2341     arg1[ax] = ch
2342     ax++
2343   }
2344   if sb < si {
2345     argv = append(argv, string(arg1[0:ax]))
2346     if debug {
2347       fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2348         -1+len(argv), sb, si, string(arg1[0:ax]), string(str[si:]))
2349     }
2350   }
2351   if debug {
2352     fmt.Printf("--Da- %d [%s] => [%d]\n", strip, str, len(argv), argv)
2353   }
2354   return argv
2355 }
2356
2357 // should get stderr (into tmpfile ?) and return
2358 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
2359 //var pv = []int{-1,-1}
2360 //syscall.Pipe(pv)
2361
2362 xarg := gshScanArg(name,1)
2363 name = strings.Join(xarg, " ")
2364
2365 //pin = os.NewFile(uintptr(pv[0]),"StdoutOf-"+name+"")
2366 //pout = os.NewFile(uintptr(pv[1]),"StdinOf-"+name+"")
2367 pin,pout,_ = os.Pipe();
2368
2369 fdix := 0
2370 dir := ""
2371 if mode == "r" {
2372   dir = "<"
2373   fdix = 1 // read from the stdout of the process
2374 }else{
2375   dir = ">"
2376   fdix = 0 // write to the stdin of the process
2377 }
2378 gshPA := gsh.gshPA
2379 savfd := gshPA.Files[fdix]
2380
2381 var fd uintptr = 0
2382 if mode == "r" {
2383   //fd = pout.Fd()
2384   //gshPA.Files[fdix] = pout.Fd()
2385 }else{
2386   //fd = pin.Fd()
2387   //gshPA.Files[fdix] = pin.Fd()
2388   gshPA.Files[fdix] = pin;
2389 }
2390
2391 // should do this by Goroutine?
2392 if false {
2393   fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2394   fmt.Printf("--BED1 [%d,%d,%d]->[%d,%d,%d]\n",
2395     os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2396     pin.Fd(),pout.Fd(),pout.Fd())
2397 }
2398
2399 savi := os.Stdin
2400 savo := os.Stdout
2401 save := os.Stderr
2402 os.Stdin = pin
2403 os.Stdout = pout
2404 os.Stderr = pout
2405 gsh.Background = true
2406 gsh.gshellh(name)
2407 gsh.Background = false
2408 os.Stdin = savi
2409 os.Stdout = savo
2410 os.Stderr = save
2411 gshPA.Files[fdix] = savfd
2412 return pin,pout,false
2413 }
2414
2415 // <a name="ex-commands">External commands</a>
2416 func (gsh*GshContext)excommand(exec bool, argv []string) (notf bool,exit bool) {
2417   if gsh.CmdTrace { fmt.Printf("--I-- excommand[%v](%v)\n",exec,argv) }
2418 }
2419 gshPA := gsh.gshPA
2420 fullpath, itis := which("PATH",[]string{"which",argv[0],"-s"})
2421 if itis == false {
2422   return true,false
2423 }
2424 fullpath := fullpath[0]
2425 argv = unescapeWhiteSP(argv)
2426 if 0 < strings.Index(fullpath,"go") {
2427   nargv := argv // []string{}
2428   gofullpath, itis := which("PATH",[]string{"which","go","-s"})
2429   if itis == false {
2430     fmt.Printf("--P-- Go not found\n")
2431     return false,true
2432   }
2433   gofullpath := gofullpath[0]
2434   nargv = []string{ gofullpath, "run", fullpath }
2435   fmt.Printf("--I-- %s (%s %s %s)\n", gofullpath,
2436     nargv[0],nargv[1],nargv[2])
2437   if exec {
2438     syscall.Exec(gofullpath,nargv,os.Environ())
2439   }else{
2440     //pid, _ := syscall.ForkExec(gofullpath,nargv,&gshPA)
2441     proc, _ := os.StartProcess(gofullpath,nargv,&gshPA);
2442     pstat, _ := proc.Wait();
2443     pid := pstat.Pid();
2444     if gsh.Background {
2445       fmt.Fprint(stderr, "--Ip- in Background pid[%d] %d(%v)\n", pid, len(argv), nargv)
2446       //gsh.BackgroundJobs = append(gsh.BackgroundJobs, pid)
2447       gsh.BackgroundJobs = append(gsh.BackgroundJobs, *pstat)
2448     }else{
2449       /*
2450       rusage := aRusage {
2451         syscall.Wait4(pid,nil,0,*rusage)
2452       }
2453       gsh.LastRusage = rusage
2454       gsh.CmdCurrent.Rusage[1] = rusage
2455       */
2456     gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2457     gsh.CmdCurrent.Rusage[1] = *pstat.SysUsage().(*aRusage);
2458     /*
2459     aSetRusage(&gsh.LastRusage,pstat);
2460     gsh.CmdCurrent.Rusage[1] = gsh.LastRusage;
2461     }
2462     }
2463     }
2464     }
2465     }else{
2466     if exec {
2467       syscall.Exec(fullpath,argv,os.Environ())
2468     }else{
2469     //pid, _ := syscall.ForkExec(fullpath,argv,&gshPA)
2470     proc, _ := os.StartProcess(fullpath,argv,&gshPA);
2471     pstat, _ := proc.Wait();
2472     pid := pstat.Pid();
2473     //fmt.Printf("[%d]\n",pid); // 'e' to be background
2474     if false {
2475       fmt.Printf("Sys=%v\n", gshPA.Sys);
2476       if( gshPA.Sys != nil ){
2477         //fmt.Printf("inFG=%v\n", gshPA.Sys.Foreground);
2478       }
2479     }

```

```

2478 }
2479     if gsh.Background {
2480         fmt.Printf(stderr, "--Ip- in Background pid%d%d(%v)\n", pid, len(argv), argv)
2481         //gsh.BackgroundJobs = append(gsh.BackgroundJobs, pid)
2482         gsh.BackgroundJobs = append(gsh.BackgroundJobs, *pstat)
2483     } else {
2484         /*
2485         rusage := aRusage {}
2486         //
2487         syscall.Wait4(pid, nil, 0, rusage);
2488         gsh.LastRusage = rusage
2489         gsh.CmdCurrent.Rusagev[1] = rusage
2490         */
2491         /*
2492         gsh.LastRusage          = *pstat.SysUsage().(*aRusage);
2493         gsh.CmdCurrent.Rusagev[1] = *pstat.SysUsage().(*aRusage);
2494         */
2495         aSetRusage(&gsh.LastRusage, pstat);
2496         gsh.CmdCurrent.Rusagev[1] = gsh.LastRusage;
2497     }
2498 }
2499 return false, false
2500 }
2501
2502 // <a name="builtin">Builtin Commands</a>
2503 func (gshCtx *GshContext) sleep(argv []string) {
2504     if len(argv) < 2 {
2505         fmt.Printf("Sleep 100ms, 100us, 100ns, ... \n")
2506     }
2507     duration := argv[1];
2508     d, err := time.ParseDuration(duration)
2509     if err != nil {
2510         d, err = time.ParseDuration(duration+"s")
2511         if err != nil {
2512             fmt.Printf("duration ? %s (%s)\n", duration, err)
2513             return
2514         }
2515     }
2516     //fmt.Printf("Sleep %v\n", duration)
2517     time.Sleep(d)
2518     if 0 < len(argv[2:]) {
2519         gshCtx.gshelv(argv[2:])
2520     }
2521 }
2522 func (gshCtx *GshContext) repeat(argv []string) {
2523     if len(argv) < 2 {
2524         return
2525     }
2526     start0 := time.Now()
2527     for ri, := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2528         if 0 < len(argv[2:]) {
2529             //start := time.Now()
2530             gshCtx.gshelv(argv[2:])
2531             end := time.Now()
2532             elps = end.Sub(start0);
2533             if 100000000 < elps {
2534                 fmt.Printf("repeat%d %v\n", ri, elps);
2535             }
2536         }
2537     }
2538 }
2539 }
2540 func (gshCtx *GshContext) gen(argv []string) {
2541     gshPA := gshCtx.gshPA
2542     if len(argv) < 2 {
2543         fmt.Printf("Usage: %s N\n", argv[0])
2544         return
2545     }
2546     // should be repeated by "repeat" command
2547     count, := strconv.Atoi(argv[1])
2548     //fd := gshPA.Files[1] // Stdout
2549     //file := os.NewFile(fd, "internalStdout")
2550     file := gshPA.Files[1]; // Stdout
2551     fmt.Printf("-- Gen. Count%d to %d\n", count, file.Fd())
2552     //buf := []byte{}
2553     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
2554     for gi := 0; gi < count; gi++ {
2555         file.WriteString(outdata)
2556     }
2557     //file.WriteString("\n")
2558     fmt.Printf("\n(%d B)\n", count*len(outdata));
2559     //file.Close()
2560 }
2561
2562 // <a name="rexec">Remote Execution</a> // 2020-0820
2563 func Elapsed(from time.Time)(string){
2564     elps := time.Now().Sub(from)
2565     if 100000000 < elps {
2566         return fmt.Sprintf("%5d.%02ds", elps/1000000000, (elpls%1000000000)/10000000)
2567     } else
2568     if 1000000 < elps {
2569         return fmt.Sprintf("%3d.%03dms", elps/1000000, (elpls%1000000)/1000)
2570     } else{
2571         return fmt.Sprintf("%3d.%03dus", elps/1000, (elpls%1000))
2572     }
2573 }
2574 //func abtime(nanos int64)(string){
2575 func abtime(nanos time.Duration)(string){
2576     if 100000000 < nanos {
2577         return fmt.Sprintf("%d.%02ds", nanos/1000000000, (nanos%1000000000)/10000000)
2578     } else
2579     if 1000000 < nanos {
2580         return fmt.Sprintf("%d.%03dms", nanos/1000000, (nanos%1000000)/1000)
2581     } else{
2582         return fmt.Sprintf("%d.%03dus", nanos/1000, (nanos%1000))
2583     }
2584 }
2585 }
2586 func abszize(size int64)(string){
2587     fsz := float64(size)
2588     if 1024*1024*1024 < size {
2589         return fmt.Sprintf("%.2fGiB", fsz/(1024*1024*1024))
2590     } else
2591     if 1024*1024 < size {
2592         return fmt.Sprintf("%.3fMiB", fsz/(1024*1024))
2593     } else{
2594         return fmt.Sprintf("%.3fKiB", fsz/1024)
2595     }
2596 }
2597 func abszize(size int64)(string){
2598     fsz := float64(size)
2599     if 1024*1024*1024 < size {
2600         return fmt.Sprintf("%.2fGiB", fsz/(1024*1024*1024))
2601     } else
2602     if 1024*1024 < size {
2603         return fmt.Sprintf("%.3fMiB", fsz/(1024*1024))
2604     } else{
2605         return fmt.Sprintf("%.3fKiB", fsz/1024)
2606     }
2607 }
2608 func abspspeed(totalB int64, ns int64)(string){
2609     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2610     if 1000 <= MBs {
2611         return fmt.Sprintf("%6.3fGB/s", MBs/1000)
2612     }
2613     if 1 <= MBs {
2614         return fmt.Sprintf("%6.3fMB/s", MBs)
2615     } else{
2616         return fmt.Sprintf("%6.3fKB/s", MBs*1000)
2617     }
2618 }
2619 func abspspeed(totalB int64, ns time.Duration)(string){
2620     MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2621     if 1000 <= MBs {
2622         return fmt.Sprintf("%6.3fGBps", MBs/1000)
2623     }
2624     if 1 <= MBs {
2625         return fmt.Sprintf("%6.3fMBps", MBs)
2626     } else{
2627         return fmt.Sprintf("%6.3fKBps", MBs*1000)
2628     }
2629 }
2630 func fileRelay(what string, in*os.File, out*os.File, size int64, bsiz int)(wcount int64){
2631     Start := time.Now()
2632     buff := make([]byte, bsiz)
2633     var total int64 = 0
2634     var rem int64 = size
2635     nio := 0
2636     Prev := time.Now()
2637     var PrevSize int64 = 0
2638     fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) START\n",
2639         what, abszize(total), size, nio)
2640 }
2641
2642 for i:= 0; ; i++ {
2643     var len = bsiz
2644     if int(rem) < len {
2645         len = int(rem)
2646     }
2647     Now := time.Now()
2648     Elps := Now.Sub(Prev);
2649     if 1000000000 < Now.Sub(Prev) {
2650         fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %s\n",
2651             what, abszize(total), size, nio,
2652             abspspeed((total-PrevSize), Elps))
2653         Prev = Now;
2654         PrevSize = total
2655     }

```

```

2655     }
2656     rlen := len
2657     if in != nil {
2658         // should watch the disconnection of out
2659         rcc,err := in.Read(buf[0:rlen])
2660         if err != nil {
2661             fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<v\n",
2662                 what,rcc,err,in.Name())
2663             break
2664         }
2665         rlen = rcc
2666         if string(buf[0:rlen]) == "(SoftEOF " {
2667             var ecc int64 = 0
2668             fmt.Sscanf(string(buf),"(SoftEOF %v",&ecc)
2669             fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))\n",
2670                 what,ecc,total)
2671             if ecc == total {
2672                 break
2673             }
2674         }
2675     }
2676     wlen := rlen
2677     if out != nil {
2678         wcc,err := out.Write(buf[0:rlen])
2679         if err != nil {
2680             fmt.Printf(Elapsed(Start)+"--En- X: %s write(%v,%v)>v\n",
2681                 what,wcc,err,out.Name())
2682             break
2683         }
2684         wlen = wcc
2685     }
2686     if wlen < rlen {
2687         fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2688             what,wlen,rlen)
2689         break;
2690     }
2691     nio += 1
2692     total += int64(rlen)
2693     rem -= int64(rlen)
2694     if rem <= 0 {
2695         break
2696     }
2697 }
2698 }
2699 Done := time.Now()
2700 Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2701 TotalMB := float64(total)/1000000 //MB
2702 MBps := totalMB/Elps
2703 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %v %v\n",
2704     what,total,size,nio,absize(total),MBps)
2705 return total
2706 }
2707 func tcpPush(clnt *os.File){
2708     // shrink socket buffer and recover
2709     usleep(100);
2710 }
2711 func (gsh*GshContext)RexecServer(argv[[]string){
2712     debug := true
2713     Start0 := time.Now()
2714     Start := Start0
2715     // if local == ":" { local = "0.0.0.0:9999" }
2716     local := "0.0.0.0:9999"
2717     if 0 < len(argv) {
2718         if argv[0] == "-s" {
2719             debug = false
2720             argv = argv[1:]
2721         }
2722     }
2723     if 0 < len(argv) {
2724         argv = argv[1:]
2725     }
2726     port, err := net.ResolveTCPAddr("tcp",local);
2727     if err != nil {
2728         fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
2729         return
2730     }
2731     scconn, err := net.ListenTCP("tcp", port)
2732     if err != nil {
2733         fmt.Printf(Elapsed(Start)+"--In- S: Listen error: %s (%s)\n",local,err)
2734         return
2735     }
2736     reqbuf := make([]byte,LINESIZE)
2737     res := ""
2738     for {
2739         fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2740         acconn, err := scconn.AcceptTCP()
2741         Start = time.Now()
2742         if err != nil {
2743             fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2744             return
2745         }
2746         clnt, _ := acconn.File()
2747         fd := Clnt.Fd()
2748         ar := acconn.RemoteAddr()
2749         if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
2750             local,fd,ar) }
2751         res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
2752         fmt.Fprintln(clnt,"&res")
2753         if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
2754         count, err := clnt.Read(reqbuf)
2755         if err != nil {
2756             fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
2757                 count,err,string(reqbuf))
2758         }
2759         req := string(reqbuf[:count])
2760         if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
2761         reqv := strings.Split(string(req)," ")
2762         cmdv := strings.Split(reqv[0]," ")
2763         //cmdv := strings.Split(reqv[0]," ")
2764         switch cmdv[0] {
2765             case "HELO":
2766                 res = fmt.Sprintf("250 %v",req)
2767             case "GET":
2768                 // download {remotefile|-zN} [localfile]
2769                 var dsz int64 = 32*1024*1024
2770                 var bsize int = 64*1024
2771                 var fname string = ""
2772                 var in *os.File = nil
2773                 var pseudoEOF = false
2774                 if 1 < len(cmdv) {
2775                     fname = cmdv[1]
2776                     if strBegins(fname,"z") {
2777                         dsz = cmdv[2]
2778                     }
2779                 }
2780                 if strBegins(fname,"(") {
2781                     xln,xout,err := gsh.Popen(fname,"r")
2782                     if err {
2783                         xout.Close()
2784                         defer xln.Close()
2785                     }
2786                     in = xln
2787                     dsz = MaxStreamSize
2788                     pseudoEOF = true
2789                 }
2790                 }else{
2791                     xln,err := os.Open(fname)
2792                     if err != nil {
2793                         fmt.Printf("--En- GET (%v)\n",err)
2794                     }
2795                     defer xln.Close()
2796                     in = xln
2797                     fi, _ := xln.Stat()
2798                     dsz = fi.Size()
2799                 }
2800             }
2801         }
2802         //fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n",dsz,bsize)
2803         res = fmt.Sprintf("200 %v\r\n",dsz)
2804         fmt.Fprintln(clnt,"&res")
2805         tcpPush(clnt); // should be separated as line in receiver
2806         fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2807         wcount := fileRelay("SendGET",in,clnt,dsz,bsize)
2808         if pseudoEOF {
2809             in.Close() // pipe from the command
2810             // show end of stream data (its size) by OOB?
2811             SoftEOF := fmt.Sprintf("(SoftEOF %v)",wcount)
2812             fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n",SoftEOF)
2813         }
2814         tcpPush(clnt); // to let SoftEOF data apper at the top of received data
2815         fmt.Fprintln(clnt,"&v\n",SoftEOF)
2816         tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
2817         // with client generated random?
2818         //fmt.Printf("--In- L: close %v (%v)\n",in.Fd(),in.Name())
2819         }
2820         res = fmt.Sprintf("200 GET done\r\n")
2821     case "PUT":
2822         // upload {srcfile|-zN} [dstfile]
2823         var dsz int64 = 32*1024*1024
2824         var bsize int = 64*1024
2825         var fname string = ""
2826         var out *os.File = nil
2827         if 1 < len(cmdv) { // localfile
2828             fmt.Sscanf(cmdv[1],"%d",&dsz)

```



```

2832     }
2833     if 2 < len(cmdv) {
2834         fname = cmdv[2]
2835         if fname == "-" {
2836             // nul dev
2837             // nol dev
2838             if strBegins(fname, "{") {
2839                 xin, xout, err := gsh.Popen(fname, "w")
2840                 if err {
2841                     }else{
2842                         xin.Close()
2843                         defer xout.Close()
2844                         out = xout
2845                     }
2846                 }else{
2847                     // should write to temporary file
2848                     // should suppress "C on tty
2849                     xout, err := os.OpenFile(fname, os.O_CREATE|os.O_RDWR|os.O_TRUNC, 0600)
2850                     //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n", fname, xout, err)
2851                     if err != nil {
2852                         fmt.Printf("--En- PUT (%v)\n", err)
2853                     }else{
2854                         out = xout
2855                     }
2856                 }
2857                 fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
2858                     fname, local, err)
2859             }
2860             fmt.Printf(Elapsed(Start)+"--In- PUT %v (/v)\n", dsize, bsize)
2861             fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\r\n", dsize)
2862             fmt.Fprintf(clnt, "200 %v OK\r\n", dsize)
2863             fileRelay("RecvPUT", clnt, out, dsize, bsize)
2864             res = fmt.Sprintf("200 PUT done\r\n")
2865             default:
2866                 res = fmt.Sprintf("400 What? %v", req)
2867         }
2868         swcc, serr := clnt.Write([]byte(res))
2869         if serr != nil {
2870             fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v", swcc, serr, res)
2871         }else{
2872             fmt.Printf(Elapsed(Start)+"--In- S: %v", res)
2873         }
2874         aconn.Close();
2875         clnt.Close();
2876     }
2877     sconn.Close();
2878 }
2879 func (gsh*GshContext)RexecClient(argv []string)(int, string){
2880     debug := true
2881     Start := time.Now()
2882     if len(argv) == 1 {
2883         return -1, "EmptyARG"
2884     }
2885     argv = argv[1:]
2886     if argv[0] == "-serv" {
2887         gsh.RexecServer(argv[1:])
2888         return 0, "Server"
2889     }
2890     remote := "0.0.0.0:9999"
2891     if argv[0][0] == '#' {
2892         remote = argv[0][1:]
2893         argv = argv[1:]
2894     }
2895     if argv[0] == "-s" {
2896         debug = false
2897         argv = argv[1:]
2898     }
2899     dport, err := net.ResolveTCPAddr("tcp", remote);
2900     if err != nil {
2901         fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n", remote, err)
2902         return -1, "AddressError"
2903     }
2904     fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n", remote)
2905     serv, err := net.DialTCP("tcp", nil, dport)
2906     if err != nil {
2907         fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n", remote, err)
2908         return -1, "CannotConnect"
2909     }
2910     if debug {
2911         al := serv.LocalAddr()
2912         fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n", remote, al)
2913     }
2914     req := ""
2915     res := make([]byte, LINUXSIZE)
2916     count, err := serv.Read(res)
2917     if err != nil {
2918         fmt.Printf("--En- S: (%d,%v) %v", count, err, string(res))
2919     }
2920     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res)) }
2921     if argv[0] == "GET" {
2922         savPA := gsh.gshPA
2923         var bsize int = 64*1024
2924         req = fmt.Sprintf("%v\r\n", strings.Join(argv, " "))
2925         fmt.Printf(Elapsed(Start)+"--In- C: %v", req)
2926         fmt.Fprintf(serv, req)
2927         count, err = serv.Read(res)
2928         if err != nil {
2929             }else{
2930                 var dsize int64 = 0
2931                 var out *os.File = nil
2932                 var out_tobeclosed *os.File = nil
2933                 var fname string = ""
2934                 var rcode int = 0
2935                 var pid int = -1
2936                 var b := Scanf(string(res), "%d %d", &rcode, &dsize)
2937                 fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res[0:count]))
2938                 if 3 <= len(argv) {
2939                     fname = argv[2]
2940                     if strBegins(fname, "{") {
2941                         xin, xout, err := gsh.Popen(fname, "w")
2942                         if err {
2943                             }else{
2944                                 xin.Close()
2945                                 defer xout.Close()
2946                                 out = xout
2947                                 out_tobeclosed = xout
2948                                 pid = 0 // should be its pid
2949                             }
2950                         }else{
2951                             // should write to temporary file
2952                             // should suppress "C on tty
2953                             xout, err := os.OpenFile(fname, os.O_CREATE|os.O_RDWR|os.O_TRUNC, 0600)
2954                             if err != nil {
2955                                 fmt.Printf("--En- %v\n", err)
2956                             }
2957                             out = xout
2958                             //fmt.Printf("--In-- %d > %s\n", out.Fd(), fname)
2959                         }
2960                     }
2961                     in_ := serv.File()
2962                     fileRelay("RecvGET", in, out, dsize, bsize)
2963                     if 0 <= pid {
2964                         gsh.gshPA = savPA // recovery of Fd(), and more?
2965                         fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n", fname)
2966                         out_tobeclosed.Close()
2967                         //syscall.Wait4(pid, nil, 0, nil) //@@
2968                     }
2969                 }
2970             }
2971         }else
2972         if argv[0] == "PUT" {
2973             remote, := serv.File()
2974             var local *os.File = nil
2975             var dsize int64 = 32*1024*1024
2976             var bsize int = 64*1024
2977             var ofile string = "-"
2978             //fmt.Printf("--In- Rex %v\n", argv)
2979             if 1 < len(argv) {
2980                 fname := argv[1]
2981                 if strBegins(fname, "-z") {
2982                     fmt.Sscanf(fname[2:], "%d", &dsize)
2983                 }else
2984                 if strBegins(fname, "{") {
2985                     xin, xout, err := gsh.Popen(fname, "r")
2986                     if err {
2987                         }else{
2988                             xout.Close()
2989                             defer xin.Close()
2990                             //in = xin
2991                             local = xin
2992                             fmt.Printf("--In- [%d] < Upload output of %v\n",
2993                                 local.Fd(), fname)
2994                             ofile = "-from."+fname
2995                             dsize = MaxStreamSize
2996                         }
2997                     }else{
2998                         xlocal, err := os.Open(fname)
2999                         if err != nil {
3000                             fmt.Printf("--En- (%s)\n", err)
3001                             local = nil
3002                         }else{
3003                             local = xlocal
3004                             fi_ := local.Stat()
3005                             dsize = fi_.Size()
3006                             defer local.Close()
3007                             //fmt.Printf("--In- Rex in(%v / %v)\n", ofile, dsize)
3008                         }
3009                     }
3010                 }
3011             }
3012         }
3013     }

```

```

3009     }
3010     ofile = fname
3011     fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
3012         fname,dsize,local,err)
3013     }
3014     }
3015     if 2 < len(argv) && argv[2] != "" {
3016         ofile = argv[2]
3017         //fmt.Printf("(%d)%v B.ofile=%v\n",len(argv),argv,ofile)
3018     }
3019     //fmt.Printf(Elapsed(Start)+"--In- Rex out(%v)\n",ofile)
3020     fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n",dsize,bsize)
3021     req = fmt.Sprintf("PUT %v %v \n",dsize,ofile)
3022     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
3023     fmt.Fprintf(serv,"%v",req)
3024     count,err = serv.Read(res)
3025     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
3026     fileRelay("SendPUT",local,remote,dsize,bsize)
3027 }else{
3028     req = fmt.Sprintf("%v\n",strings.Join(argv," "))
3029     if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
3030     fmt.Fprintf(serv,"%v",req)
3031     //fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
3032 }
3033 //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
3034 count,err = serv.Read(res)
3035 res := ""
3036 if count == 0 {
3037     res = "(nil)\r\n"
3038 }else{
3039     res = string(res[:count])
3040 }
3041 if err != nil {
3042     fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,res)
3043 }else{
3044     fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
3045 }
3046 serv.Close()
3047 //conn.Close()
3048
3049 var stat string
3050 var rcode int
3051 fmt.Sscanf(res,"%d %s",&rcode,&stat)
3052 //fmt.Printf("--D-- Client: %v (%v)",rcode,stat)
3053 return rcode,res
3054 }
3055
3056 // <a name="remote-sh">Remote Shell</a>
3057 // gcp file [...] { [host]:[port]:[dir] | dir } // -p | -no-p
3058 func (gsh*GshContext)FileCopy(argv[]string){
3059     var host = ""
3060     var port = ""
3061     var upload = false
3062     var download = false
3063     var xargv = []string{"rex-gcp"}
3064     var srcv = []string{}
3065     var dstv = []string{}
3066     argv = argv[1:]
3067
3068     for ,v := range argv {
3069         /*
3070         if v[0] == '-' { // might be a pseudo file (generated date)
3071             continue
3072         }
3073         */
3074         obj := strings.Split(v,":")
3075         //fmt.Printf("%d %v %v\n",len(obj),v,obj)
3076         if 1 < len(obj) {
3077             host = obj[0]
3078             file := ""
3079             if 0 < len(host) {
3080                 gsh.LastServer.host = host
3081             }else{
3082                 host = gsh.LastServer.host
3083                 port = gsh.LastServer.port
3084             }
3085             if 2 < len(obj) {
3086                 port = obj[1]
3087                 if 0 < len(port) {
3088                     gsh.LastServer.port = port
3089                 }else{
3090                     port = gsh.LastServer.port
3091                 }
3092                 file = obj[2]
3093             }else{
3094                 file = obj[1]
3095             }
3096             if len(srcv) == 0 {
3097                 download = true
3098                 srcv = append(srcv,file)
3099             }
3100             upload = true
3101             dstv = append(dstv,file)
3102             continue
3103         }
3104         /*
3105         idx := strings.Index(v,":")
3106         if 0 <= idx {
3107             remote = v[0:idx]
3108             if len(srcv) == 0 {
3109                 download = true
3110                 srcv = append(srcv,v[idx+1:])
3111             }
3112             continue
3113         }
3114         upload = true
3115         dstv = append(dstv,v[idx+1:])
3116         continue
3117         */
3118         if download {
3119             dstv = append(dstv,v)
3120         }else{
3121             srcv = append(srcv,v)
3122         }
3123     }
3124     hostport := "@" + host + ":" + port
3125     if upload {
3126         if host != "" { xargv = append(xargv,hostport) }
3127         xargv = append(xargv,"PUT")
3128         xargv = append(xargv,srcv[0]...)
3129         xargv = append(xargv,dstv[0]...)
3130     }
3131     //fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v // %v\n",hostport,dstv,srcv,xargv)
3132     fmt.Printf("--I-- FileCopy PUT gsh://%s/%v < %v\n",hostport,dstv,srcv)
3133     gsh.RexecClient(xargv)
3134 }else
3135 if download {
3136     if host != "" { xargv = append(xargv,hostport) }
3137     xargv = append(xargv,"GET")
3138     xargv = append(xargv,srcv[0]...)
3139     xargv = append(xargv,dstv[0]...)
3140     //fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v // %v\n",hostport,srcv,dstv,xargv)
3141     fmt.Printf("--I-- FileCopy GET gsh://%v/%v > %v\n",hostport,srcv,dstv)
3142     gsh.RexecClient(xargv)
3143 }else{
3144 }
3145 }
3146
3147 // target
3148 func (gsh*GshContext)Trelpath(rloc string)(string){
3149     cwd, _ := os.Getwd()
3150     os.Chdir(gsh.RWD)
3151     os.Chdir(rloc)
3152     twd, _ := os.Getwd()
3153     os.Chdir(cwd)
3154
3155     tpath := twd + "/" + rloc
3156     return tpath
3157 }
3158
3159 // join to rmtote GShell - [user@]host[:port] or cd host[:port]:path
3160 func (gsh*GshContext)Rjoin(argv[]string){
3161     if len(argv) <= 1 {
3162         fmt.Printf("--I-- current server = %v\n",gsh.RSERV)
3163         return
3164     }
3165     serv := argv[1]
3166     servv := strings.Split(serv,":")
3167     if 1 <= len(servv) {
3168         if servv[0] == "lo" {
3169             servv[0] = "localhost"
3170         }
3171     }
3172     switch len(servv) {
3173     case 1:
3174         //if strings.Index(serv,":") < 0 {
3175             serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
3176         //}
3177     case 2: // host:port
3178         serv = strings.Join(servv,":")
3179     }
3180     xargv = []string{"rex-join", "@"+serv,"HELO"}
3181     rcode,stat := gsh.RexecClient(xargv)
3182     if (rcode / 100) == 2 {
3183         fmt.Printf("--I-- OK Joined (%v) %v\n",rcode,stat)
3184         gsh.RSERV = serv
3185     }else{
3186         fmt.Printf("--I-- NG, could not joined (%v) %v\n",rcode,stat)
3187     }

```

```

3186 }
3187 }
3188 func (gsh*GshContext)Rexec(argv []string){
3189     if len(argv) <= 1 {
3190         fmt.Printf("--I-- rexec command [ | {file | {command} ]\n",gsh.RSERV)
3191         return
3192     }
3193 }
3194 /*
3195     nargv := gshScanArg(strings.Join(argv, " "),0)
3196     fmt.Printf("--D-- nargc=%d %d\n",len(nargv),nargv)
3197     if nargv[1][0] != '{' {
3198         nargv[1] = "{" + nargv[1] + "}"
3199         fmt.Printf("--D-- nargc=%d %d\n",len(nargv),nargv)
3200     }
3201     argv = nargv
3202     */
3203     nargv := []string{}
3204     nargv = append(nargv, "{"+strings.Join(argv[1:], " ")+"}")
3205     fmt.Printf("--D-- nargc=%d %d\n",len(nargv),nargv)
3206     argv = nargv
3207 }
3208     xargv := []string{"rex-exec", "@"+gsh.RSERV, "GET"}
3209     xargv = append(xargv, argv...)
3210     xargv = append(xargv, "/dev/tty")
3211     rcode, stat := gsh.RexecClient(xargv)
3212     if (rcode / 100) == 2 {
3213         fmt.Printf("--I-- OK Rexec (%v) %v\n", rcode, stat)
3214     } else {
3215         fmt.Printf("--I-- NG Rexec (%v) %v\n", rcode, stat)
3216     }
3217 }
3218 func (gsh*GshContext)Rchdir(argv []string){
3219     if len(argv) <= 1 {
3220         return
3221     }
3222     cwd, _ := os.Getwd()
3223     os.Chdir(gsh.RWD)
3224     os.Chdir(argv[1])
3225     twd, _ := os.Getwd()
3226     gsh.RWD = twd
3227     fmt.Printf("--I-- JWD=%v\n", twd)
3228     os.Chdir(cwd)
3229 }
3230 func (gsh*GshContext)Rpwd(argv []string){
3231     fmt.Printf("%v\n", gsh.RWD)
3232 }
3233 func (gsh*GshContext)Rls(argv []string){
3234     cwd, _ := os.Getwd()
3235     os.Chdir(gsh.RWD)
3236     argv[0] = "-ls"
3237     gsh.xFind(argv)
3238     os.Chdir(cwd)
3239 }
3240 func (gsh*GshContext)Rput(argv []string){
3241     var local string = ""
3242     var remote string = ""
3243     if 1 < len(argv) {
3244         local = argv[1]
3245         remote = local // base name
3246     }
3247     if 2 < len(argv) {
3248         remote = argv[2]
3249     }
3250     fmt.Printf("--I-- jput from=%v to=%v\n", local, gsh.Trelpath(remote))
3251 }
3252 func (gsh*GshContext)Rget(argv []string){
3253     var remote string = ""
3254     var local string = ""
3255     if 1 < len(argv) {
3256         remote = argv[1]
3257         local = remote // base name
3258     }
3259     if 2 < len(argv) {
3260         local = argv[2]
3261     }
3262     fmt.Printf("--I-- jget from=%v to=%v\n", gsh.Trelpath(remote), local)
3263 }
3264 }
3265 // <a name="network">network</a>
3266 // -s, -sl, -sso // bi-directional, source, sync (maybe socket)
3267 func (gshCtx*GshContext)ssconnect(inTCP bool, argv []string) {
3268     gshPA := gshCtx.gshPA
3269     if len(argv) < 2 {
3270         fmt.Printf("Usage: -s [host]:[port.[udp]]\n")
3271         return
3272     }
3273     remote := argv[1]
3274     if remote == ":" { remote = "0.0.0.0:9999" }
3275 }
3276     if inTCP { // TCP
3277         dport, err := net.ResolveTCPAddr("tcp", remote);
3278         if err != nil {
3279             fmt.Printf("Address error: %s (%s)\n", remote, err)
3280             return
3281         }
3282         conn, err := net.DialTCP("tcp", nil, dport)
3283         if err != nil {
3284             fmt.Printf("Connection error: %s (%s)\n", remote, err)
3285             return
3286         }
3287         file, _ := conn.File();
3288         //fd := file.Fd()
3289         //fmt.Printf("Socket: connected to %s, socket[%d]\n", remote, fd)
3290         fmt.Printf("Socket: connected to %s, socket[%d]\n", remote, file.Fd())
3291     }
3292     savfd := gshPA.Files[1]
3293     //gshPA.Files[1] = fd;
3294     gshPA.Files[1] = file;
3295     gshCtx.gshellv(argv[2:1])
3296     gshPA.Files[1] = savfd
3297     file.Close()
3298     conn.Close()
3299 } else {
3300     //dport, err := net.ResolveUDPAddr("udp4", remote);
3301     dport, err := net.ResolveUDPAddr("udp", remote);
3302     if err != nil {
3303         fmt.Printf("Address error: %s (%s)\n", remote, err)
3304         return
3305     }
3306     //conn, err := net.DialUDP("udp4", nil, dport)
3307     conn, err := net.DialUDP("udp", nil, dport)
3308     if err != nil {
3309         fmt.Printf("Connection error: %s (%s)\n", remote, err)
3310         return
3311     }
3312     file, _ := conn.File();
3313     //fd := file.Fd()
3314 }
3315     ar := conn.RemoteAddr()
3316     //al := conn.LocalAddr()
3317     fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
3318         //remote, ar.String(), fd)
3319         remote, ar.String(), file.Fd())
3320 }
3321     savfd := gshPA.Files[1]
3322     //gshPA.Files[1] = fd;
3323     gshPA.Files[1] = file;
3324     gshCtx.gshellv(argv[2:1])
3325     gshPA.Files[1] = savfd
3326     file.Close()
3327     conn.Close()
3328 }
3329 }
3330 func (gshCtx*GshContext)ssaccept(inTCP bool, argv []string) {
3331     gshPA := gshCtx.gshPA
3332     if len(argv) < 2 {
3333         fmt.Printf("Usage: -ac [host]:[port.[udp]]\n")
3334         return
3335     }
3336     local := argv[1]
3337     if local == ":" { local = "0.0.0.0:9999" }
3338     if inTCP { // TCP
3339         port, err := net.ResolveTCPAddr("tcp", local);
3340         if err != nil {
3341             fmt.Printf("Address error: %s (%s)\n", local, err)
3342             return
3343         }
3344         //fmt.Printf("Listen at %s...\n", local);
3345         sconn, err := net.ListenTCP("tcp", port)
3346         if err != nil {
3347             fmt.Printf("Listen error: %s (%s)\n", local, err)
3348             return
3349         }
3350         //fmt.Printf("Accepting at %s...\n", local);
3351         acconn, err := sconn.AcceptTCP()
3352         if err != nil {
3353             fmt.Printf("Accept error: %s (%s)\n", local, err)
3354             return
3355         }
3356         file, _ := acconn.File()
3357         //fd := file.Fd()
3358         //fmt.Printf("Accepted TCP at %s [%d]\n", local, fd)
3359         fmt.Printf("Accepted TCP at %s [%d]\n", local, file.Fd())
3360     }
3361     savfd := gshPA.Files[0]
3362     //gshPA.Files[0] = fd;

```

```

3363     gshPA.Files[0] = file;
3364     gshCtx.gshellv(argv[2:])
3365     gshPA.Files[0] = savfd
3366
3367     sconn.Close();
3368     sconn.Close();
3369     file.Close();
3370 }else{
3371     //port, err := net.ResolveUDPAddr("udp4",local);
3372     port, err := net.ResolveUDPAddr("udp",local);
3373     if err != nil {
3374         fmt.Printf("Address error: %s (%s)\n",local,err)
3375         return
3376     }
3377     fmt.Printf("Listen UDP at %s...\n",local);
3378     //uconn, err := net.ListenUDP("udp4", port)
3379     uconn, err := net.ListenUDP("udp", port)
3380     if err != nil {
3381         fmt.Printf("Listen error: %s (%s)\n",local,err)
3382         return
3383     }
3384     file, _ := uconn.File()
3385     //fd := file.Fd()
3386     ar := uconn.RemoteAddr()
3387     remote :=
3388     if ar != nil { remote = ar.String() }
3389     if remote == "" { remote = "?" }
3390
3391     // not yet received
3392     //fmt.Printf("Accepted at %s [%d] <- %s\n",local,fd,")
3393
3394     savfd := gshPA.Files[0]
3395     //gshPA.Files[0] = fd;
3396     gshPA.Files[0] = file;
3397     savenv := gshPA.Env
3398     gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
3399     gshCtx.gshellv(argv[2:])
3400     gshPA.Env = savenv
3401     gshPA.Files[0] = savfd
3402
3403     uconn.Close();
3404     file.Close();
3405 }
3406 }
3407
3408 // empty line command
3409 func (gshCtx*GshContext)xPwd(argv[]string){
3410     // execute context command, pwd + date
3411     // context notation, representation scheme, to be resumed at re-login
3412     cwd, _ := os.Getwd()
3413     switch {
3414     case isin("-a",argv):
3415         gshCtx.ShowChdirHistory(argv)
3416     case isin("-ls",argv):
3417         showFileInfo(cwd,argv)
3418     default:
3419         fmt.Printf("%s\n",cwd)
3420     case isin("-v",argv): // obsolete empty command
3421         t := time.Now()
3422         date := t.Format(time.UnixDate)
3423         exe, _ := os.Executable()
3424         host, _ := os.Hostname()
3425         fmt.Printf("PWD=\"%s\" ",cwd)
3426         fmt.Printf("HOST=\"%s\" ",host)
3427         fmt.Printf("DATE=\"%s\" ",date)
3428         fmt.Printf("TIME=\"%s\" ",t.String())
3429         fmt.Printf("PID=\"%d\" ",os.Getpid())
3430         fmt.Printf("EXE=\"%s\" ",exe)
3431         fmt.Printf("\n")
3432     }
3433 }
3434
3435 // <a name="history">History</a>
3436 // these should be browsed and edited by HTTP browser
3437 // show the time of command with -t and directory with -ls
3438 // openfile-history, sort by -a -m -c
3439 // sort by elapsed time by -t -s
3440 // search by "more" like interface
3441 // edit history
3442 // sort history, and wc or uniq
3443 // CPU and other resource consumptions
3444 // limit showing range (by time or so)
3445 // export / import history
3446 func (gshCtx *GshContext)xHistory(argv []string){
3447     atWorkDirX := -1
3448     if 1 < len(argv) && strBegins(argv[1],"@") {
3449         atWorkDirX, _ = strconv.Atoi(argv[1][1:])
3450     }
3451     //fmt.Printf("--D-- showHistory(%v)\n",argv)
3452     for i, v := range gshCtx.CommandHistory {
3453         // exclude commands not to be listed by default
3454         // internal commands may be suppressed by default
3455         if v.CmdLine == "" && !isin("-a",argv) {
3456             continue;
3457         }
3458         if 0 <= atWorkDirX {
3459             if v.WorkDirX != atWorkDirX {
3460                 continue
3461             }
3462         }
3463         if !isin("-n",argv){ // like "fc"
3464             fmt.Printf("%s-2d ",i)
3465         }
3466         if !isin("-v",argv){
3467             fmt.Println(v) // should be with it date
3468         }else{
3469             if !isin("-l",argv) || !isin("-l0",argv) {
3470                 elps := v.EndAt.Sub(v.StartAt);
3471                 start := v.StartAt.Format(time.Stamp)
3472                 fmt.Printf("%d ",v.WorkDirX)
3473                 fmt.Printf("[%v] %11v/t ",start,elps)
3474             }
3475             if !isin("-l",argv) && !isin("-l0",argv){
3476                 fmt.Printf("%v",Rusageof("%t %t// %s",argv,v.Rusagev))
3477             }
3478             if !isin("-at",argv) { // !isin("-ls",argv){
3479                 dhi := v.WorkDirX // workdir history index
3480                 fmt.Printf("%d %s\t",dhi,v.WorkDirX)
3481                 // show the FileInfo of the output command??
3482             }
3483             fmt.Printf("%s",v.CmdLine)
3484             fmt.Printf("\n")
3485         }
3486     }
3487 }
3488 // ln - history index
3489 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
3490     if gline[0] == 'l' {
3491         hix, err := strconv.Atoi(gline[1:])
3492         if err != nil {
3493             fmt.Printf("--E-- (%s : range)\n",hix)
3494             return "", false, true
3495         }
3496         if hix < 0 || len(gshCtx.CommandHistory) <= hix {
3497             fmt.Printf("--E-- (%d : out of range)\n",hix)
3498             return "", false, true
3499         }
3500         return gshCtx.CommandHistory[hix].CmdLine, false, false
3501     }
3502     // search
3503     //for i, v := range gshCtx.CommandHistory {
3504     //}
3505     return gline, false, false
3506 }
3507 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
3508     if 0 <= hix && hix < len(gsh.CommandHistory) {
3509         return gsh.CommandHistory[hix].CmdLine,true
3510     }
3511     return "",false
3512 }
3513
3514 // temporary adding to PATH environment
3515 // cd name -lib for LD_LIBRARY_PATH
3516 // chdir with directory history (date + full-path)
3517 // -s for sort option (by visit date or so)
3518 func (gsh*GshContext)ShowChdirHistory(l int,v GChdirHistory, argv []string){
3519     fmt.Printf("%s-2d ",v.CmdIndex) // the first command at this WorkDir
3520     fmt.Printf("%d ",i)
3521     fmt.Printf("[%v] ",v.MovedAt.Format(time.Stamp))
3522     showFileInfo(v.Dir,argv)
3523 }
3524 func (gsh*GshContext)ShowChdirHistory(argv []string){
3525     for i, v := range gsh.ChdirHistory {
3526         gsh.ShowChdirHistory1(i,v,argv)
3527     }
3528 }
3529 func skipOpts(argv[]string)(int){
3530     for i,v := range argv {
3531         if strBegins(v,"-") {
3532             }else{
3533                 return i
3534             }
3535     }
3536     return -1
3537 }
3538 func (gshCtx*GshContext)xChdir(argv []string){
3539     cdhist := gshCtx.ChdirHistory

```

```

3540 if isin("?",argv) || isin("-",argv) || isin("-",argv) {
3541     gshCtx.ShowChdirHistory(argv)
3542     return
3543 }
3544 pwd, _ := os.Getwd()
3545 dir := ""
3546 if len(argv) <= 1 {
3547     dir = toFullPath("-")
3548 }else{
3549     i := skipOpts(argv[1:])
3550     if i < 0 {
3551         dir = toFullPath("-")
3552     }else{
3553         dir = argv[i+1]
3554     }
3555 }
3556 if strBegins(dir,"@") {
3557     if dir == "@0" { // obsolete
3558         dir = gshCtx.StartDir
3559     }else
3560     if dir == "@1" {
3561         index := len(cdhist) - 1
3562         if 0 < index { index -= 1 }
3563         dir = cdhist[index].Dir
3564     }else{
3565         index, err := strconv.Atoi(dir[1:])
3566         if err != nil {
3567             fmt.Printf("--E-- xChdir(%v)\n",err)
3568             dir = "?"
3569         }else
3570         if len(gshCtx.ChdirHistory) <= index {
3571             fmt.Printf("--E-- xChdir(history range error)\n")
3572             dir = "?"
3573         }else{
3574             dir = cdhist[index].Dir
3575         }
3576     }
3577 }
3578 if dir != "?" {
3579     err := os.Chdir(dir)
3580     if err != nil {
3581         fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
3582     }else{
3583         cwd, _ := os.Getwd()
3584         if cwd != pwd {
3585             hist1 := GChdirHistory { }
3586             hist1.Dir = cwd
3587             hist1.MovedAt = time.Now()
3588             hist1.CmdIndex = len(gshCtx.CommandHistory)+1
3589             gshCtx.ChdirHistory = append(cdhist,hist1)
3590             if !isin("-",argv){
3591                 //cwd, _ := os.Getwd()
3592                 //fmt.Printf("%s\n",cwd)
3593                 ix := len(gshCtx.ChdirHistory)-1
3594                 gshCtx.ShowChdirHistoryl(ix,hist1,argv)
3595             }
3596         }
3597     }
3598 }
3599 if isin("-ls",argv){
3600     cwd, _ := os.Getwd()
3601     showFileInfo(cwd,argv);
3602 }
3603 }
3604 func TimeValSub(tv1 *time.Duration, tv2 *time.Duration){
3605     //tv1 := syscall.NsecToTimeval(tv1.Nano()) - tv2.Nano()
3606     *tv1 -= *tv2;
3607 }
3608 func RusageSubv(ru1, ru2 [2]aRusage){(2)aRusage){
3609     TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
3610     TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
3611     TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
3612     TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
3613 }
3614 }
3615 func TimeValAdd(tv1 time.Duration, tv2 time.Duration)(time.Duration){
3616     //tv1 := syscall.NsecToTimeval(tv1.Nano()) + tv2.Nano()
3617     tv1 += tv2;
3618     return tv1;
3619 }
3620 //
3621 func RusageAddv(ru1, ru2 [2]aRusage){(2)aRusage){
3622     TimeValAdd(&ru1[0].Utime,&ru2[0].Utime)
3623     TimeValAdd(&ru1[0].Stime,&ru2[0].Stime)
3624     TimeValAdd(&ru1[1].Utime,&ru2[1].Utime)
3625     TimeValAdd(&ru1[1].Stime,&ru2[1].Stime)
3626 }
3627 }
3628 //
3629 // <a name="rusage">Resource Usage</a>
3630 func showRusage(fmtspeg string, argv []string, ru [2]aRusage)(string){
3631     // ru[0] self , ru[1] children
3632     ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3633     st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3634     //uu := (int64(ut.Sec)*1000000 + int64(ut.Usec)) * 1000
3635     //su := (int64(st.Sec)*1000000 + int64(st.Usec)) * 1000
3636     uu := ut; // in nano sec
3637     su := st; // in nano sec
3638     tu := uu + su
3639     ret := fmt.Sprintf("%v/sum",abotime(tu))
3640     ret += fmt.Sprintf(", %v/usr",abotime(uu))
3641     ret += fmt.Sprintf(", %v/sys",abotime(su))
3642     return ret
3643 }
3644 }
3645 func Rusagef(fmtspeg string, argv []string, ru [2]aRusage)(string){
3646     ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3647     st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3648     //fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3649     //fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
3650     fmt.Printf("%d.%06ds/u ",ut/1000000000,(ut/1000)&1000000);
3651     fmt.Printf("%d.%06ds/s ",st/1000000000,(st/1000)&1000000);
3652     return ""
3653 }
3654 }
3655 func GetRusagev(){(2)aRusage){
3656     var ruv = [2]aRusage{}
3657     aGetRusage(aRUSAGE_SELF,&ruv[0])
3658     aGetRusage(aRUSAGE_CHILDREN,&ruv[1])
3659     return ruv
3660 }
3661 }
3662 func (gshCtx *GshContext)xTime(argv[]string)(bool){
3663     if 2 <= len(argv){
3664         gshCtx.LastRusage = aRusage{}
3665         rusagev1 := GetRusagev()
3666         fin := gshCtx.gshellv(argv[1:])
3667         rusagev2 := GetRusagev()
3668         showRusage(argv[1],argv,&gshCtx.LastRusage)
3669         rusagev := RusageSubv(rusagev2,rusagev1)
3670         showRusage("self",argv,&rusagev[0])
3671         showRusage("chld",argv,&rusagev[1])
3672         return fin
3673     }else{
3674         rusage := aRusage { }
3675         aGetRusage(aRUSAGE_SELF,&rusage)
3676         showRusage("self",argv,&rusage)
3677         aGetRusage(aRUSAGE_CHILDREN,&rusage)
3678         showRusage("chld",argv,&rusage)
3679         return false
3680     }
3681 }
3682 }
3683 func (gshCtx *GshContext)xJobs(argv[]string){
3684     fmt.Printf("%d Jobs\n",len(gshCtx.BackGroundJobs))
3685     for ji, pid := range gshCtx.BackGroundJobs {
3686         //wstat := syscall.WaitStatus {0}
3687         rusage := aRusage { }
3688         //wpid, err := syscall.Wait4(pid,&wstat,syscall.WNOHANG,&rusage);
3689         //wpid, err := syscall.Wait4(pid,nil,syscall.WNOHANG,&rusage);
3690         wpid := pid.Pid();
3691         err := errors.New("stab_NoError");
3692         if err != nil {
3693             fmt.Printf("--E-- %%%d [%d] (%v)\n",ji,wpid,err)
3694         }else{
3695             fmt.Printf("%%d [%d]\n",ji,wpid)
3696             showRusage("chld",argv,&rusage)
3697         }
3698     }
3699 }
3700 }
3701 func (gshCtx *GshContext)inBackground(argv[]string)(bool){
3702     if gsh.CmdTrace { fmt.Printf("--I-- inBackground(%v)\n",argv) }
3703     gsh.BackGround = true // set background option
3704     xfin := false
3705     xfin = gsh.gshellv(argv)
3706     gsh.BackGround = false
3707     return xfin
3708 }
3709 }
3710 // -o file without command means just opening it and refer by #N
3711 // should be listed by "files" command
3712 func (gshCtx *GshContext)xOpen(argv[]string){
3713     //var pv = []int{-1,-1}
3714     //err := syscall.Pipe(pv)
3715     //fmt.Printf("--I-- pipe()=[#d,#d](%v)\n",pv[0],pv[1],err)
3716     pin,pout,err := os.Pipe();
3717     fmt.Printf("--I-- pipe()=[#d,#d](%v)\n",pin.Fd(),pout.Fd(),err)
3718 }

```

```

3717 func (gshCtx*GshContext)fromPipe(argv[]string){
3718 }
3719 func (gshCtx*GshContext)xClose(argv[]string){
3720 }
3721 }
3722 // <a name="redirect">redirect</a>
3723 func (gshCtx*GshContext)redirect(argv[]string)(bool){
3724 if len(argv) < 2 {
3725 return false
3726 }
3727 }
3728 cmd := argv[0]
3729 fname := argv[1]
3730 var file *os.File = nil
3731 }
3732 fdix := 0
3733 mode := os.O_RDONLY
3734 }
3735 switch {
3736 case cmd == "-i" || cmd == "<":
3737 fdix = 0
3738 mode = os.O_RDONLY
3739 case cmd == "-o" || cmd == ">":
3740 fdix = 1
3741 mode = os.O_RDWR | os.O_CREATE
3742 case cmd == "-a" || cmd == ">>":
3743 fdix = 1
3744 mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3745 }
3746 if fname[0] == '#' {
3747 fd, err := strconv.Atoi(fname[1:])
3748 if err != nil {
3749 fmt.Printf("--E-- (%v)\n",err)
3750 return false
3751 }
3752 file = os.NewFile(uintptr(fd),"MaybePipe")
3753 }else{
3754 xfile, err := os.OpenFile(argv[1], mode, 0600)
3755 if err != nil {
3756 fmt.Printf("--E-- (%s)\n",err)
3757 return false
3758 }
3759 file = xfile
3760 }
3761 gshPA := gshCtx.gshPA
3762 savfd := gshPA.Files[fdix]
3763 //gshPA.Files[fdix] = file.Fd()
3764 gshPA.Files[fdix] = file;
3765 fmt.Printf("--I-- Opened [%d] %s\n",file.Fd(),argv[1])
3766 gshCtx.gshellv(argv[2:])
3767 gshPA.Files[fdix] = savfd
3768 }
3769 return false
3770 }
3771 }
3772 //fmt.Fprintf(res, "Gshell Status: %q", html.EscapeString(req.URL.Path))
3773 func httpHandler(res http.ResponseWriter, req *http.Request){
3774 path := req.URL.Path
3775 fmt.Printf("--I-- Got HTTP Request(%s)\n",path)
3776 {
3777 gshCtxBuf, _ := setupGshContext()
3778 gshCtx := *gshCtxBuf
3779 fmt.Printf("--I-- %s\n",path[1:])
3780 gshCtx.tgshell(path[1:])
3781 }
3782 }
3783 }
3784 func (gshCtx *GshContext) httpServer(argv []string){
3785 http.HandleFunc("/", httpHandler)
3786 accport := "localhost:9999"
3787 fmt.Printf("--I-- HTTP Server Start at [%s]\n",accport)
3788 http.ListenAndServe(accport,nil)
3789 }
3790 func (gshCtx *GshContext)xGo(argv[]string){
3791 go gshCtx.gshellv(argv[1:]);
3792 }
3793 func (gshCtx *GshContext) xPs(argv[]string){}
3794 }
3795 }
3796 // <a name="plugin">Plugin</a>
3797 // plugin [-ls [names]] to list plugins
3798 // Reference: <a href="https://github.com/src/plugin/">https://github.com/src/plugin/</a> source code
3799 func (gshCtx *GshContext) whichPlugin(name string,argv[]string)(pi *PluginInfo){
3800 pi = nil
3801 for _,p := range gshCtx.PluginFuncs {
3802 if p.Name == name && pi == nil {
3803 pi = *p
3804 }
3805 if !isin("-s",argv){
3806 //fmt.Printf("%v %v ",i,p)
3807 if !isin("-ls",argv){
3808 showFileInfo(p.Path,argv)
3809 }else{
3810 fmt.Printf("%s\n",p.Name)
3811 }
3812 }
3813 }
3814 return pi
3815 }
3816 }
3817 func (gshCtx *GshContext) xPlugin(argv[]string) (error) {
3818 if len(argv) == 0 || argv[0] == "-ls" {
3819 gshCtx.whichPlugin("",argv)
3820 return nil
3821 }
3822 name := argv[0]
3823 pin := gshCtx.whichPlugin(name,[]string{"-s"})
3824 if pin != nil {
3825 os.Args = argv // should be recovered?
3826 pin.Addr.func()()
3827 return nil
3828 }
3829 sofile := toPullpath(argv[0] + ".so") // or find it by which($PATH)
3830 p, err := plugin.Open(sofile)
3831 if err != nil {
3832 fmt.Printf("--E-- plugin.Open(%s)(%v)\n",sofile,err)
3833 return err
3834 }
3835 fname := "Main"
3836 f, err := p.Lookup(fname)
3837 if err != nil {
3838 fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n",fname,err)
3839 return err
3840 }
3841 pin := PluginInfo {p,f,name,sofile}
3842 gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pin)
3843 fmt.Printf("--I-- added (%d)\n",len(gshCtx.PluginFuncs))
3844 }
3845 //fmt.Printf("--I-- first call(%s:%s)%v\n",sofile,fname,argv)
3846 os.Args = argv
3847 f.(func())()
3848 return err
3849 }
3850 }
3851 func (gshCtx*GshContext)Args(argv[]string){
3852 for i,v := range os.Args {
3853 fmt.Printf("[%v] %v\n",i,v)
3854 }
3855 }
3856 }
3857 func (gshCtx *GshContext) showVersion(argv[]string){
3858 if !isin("-l",argv) {
3859 fmt.Printf("%v%v (%v)",NAME,VERSION,DATE);
3860 }else{
3861 fmt.Printf("%v",VERSION);
3862 }
3863 if !isin("-a",argv) {
3864 fmt.Printf("%s",AUTHOR)
3865 }
3866 if !isin("-n",argv) {
3867 fmt.Printf("\n")
3868 }
3869 }
3870 }
3871 // <a name="scanf">Scanf</a> // string decomposer
3872 // scanf [format] [input]
3873 func scanf(sstr string)(strv[]string){
3874 strv = strings.Split(sstr, " ")
3875 return strv
3876 }
3877 }
3878 func scanfUntil(src,end string)(rstr string,leng int){
3879 idx = strings.Index(src,end)
3880 if 0 <= idx {
3881 rstr = src[0:idx]
3882 return rstr,idx+len(end)
3883 }
3884 return src,0
3885 }
3886 }
3887 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3888 func (gsh*GshContext)printVal(fmts string, vstr string, optv[]string){
3889 //vint,err := strconv.Atoi(vstr)
3890 var ival int64 = 0
3891 n := 0
3892 err := error(nil)
3893 if strBegins(vstr, "-") {
3894 vx,_ := strconv.Atoi(vstr[1:])
3895 if vx < len(gsh.iValues) {
3896 vstr = gsh.iValues[vx]
3897 }
3898 }

```

```

3894     }else{
3895     }
3896     }
3897     // should use Eval()
3898     if strBegins(vstr,"0x") {
3899     n,err = fmt.Sscanf(vstr[2:], "%x", &ival)
3900     }else{
3901     n,err = fmt.Sscanf(vstr, "%d", &ival)
3902     //fmt.Printf("--D-- n=%d err=(%v) (%s)=%v\n", n, err, vstr, ival)
3903     }
3904     if n == 1 && err == nil {
3905     //fmt.Printf("--D-- formatn(%v) ival(%v)\n", fmts, ival)
3906     fmt.Printf("%s"+fmts, ival)
3907     }else{
3908     if isin("-bn", optv){
3909     fmt.Printf("%s"+fmts, filepath.Base(vstr))
3910     }else{
3911     fmt.Printf("%s"+fmts, vstr)
3912     }
3913     }
3914     }
3915     func (gsh+GshContext)Printfv(fmts,div string,argv[ ]string,optv[ ]string,list[ ]string){
3916     //fmt.Printf("%d", len(list))
3917     //curfmt := ""
3918     outlen := 0
3919     curfmt := gsh.iFormat
3920     if 0 < len(fmts) {
3921     for xi := 0; xi < len(fmts); xi++ {
3922     fch := fmts[xi]
3923     if fch == '%' {
3924     if xi+1 < len(fmts) {
3925     curfmt = string(fmts[xi+1])
3926     gsh.iFormat = curfmt
3927     xi += 1
3928     if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3929     vals, leng := scanUntil(fmts[xi+2:], ")")
3930     //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n", curfmt, vals, leng)
3931     gsh.printVal(curfmt, vals, optv)
3932     xi += 2+leng-1
3933     outlen += 1
3934     }
3935     }
3936     }
3937     }
3938     }
3939     if fch == '.' {
3940     hi, leng := scanInt(fmts[xi+1:])
3941     if 0 < leng {
3942     if hi < len(gsh.iValues) {
3943     gsh.printVal(curfmt, gsh.iValues[hi], optv)
3944     outlen += 1 // should be the real length
3945     }else{
3946     fmt.Printf("(out-range)")
3947     }
3948     xi += leng
3949     continue;
3950     }
3951     }
3952     }
3953     }
3954     }
3955     }else{
3956     //fmt.Printf("--D-- print (%s)\n")
3957     for i,v := range list {
3958     if 0 < i {
3959     fmt.Printf(div)
3960     }
3961     gsh.printVal(curfmt, v, optv)
3962     outlen += 1
3963     }
3964     }
3965     if 0 < outlen {
3966     fmt.Printf("\n")
3967     }
3968     }
3969     }
3970     func (gsh+GshContext)Scanv(argv[ ]string){
3971     //fmt.Printf("--D-- Scanv(%v)\n", argv)
3972     if len(argv) == 1 {
3973     return
3974     }
3975     argv = argv[1:]
3976     fmts := ""
3977     if strBegins(argv[0], "-F") {
3978     fmts = argv[0]
3979     gsh.iDelimiter = fmts
3980     argv = argv[1:]
3981     }
3982     input := strings.Join(argv, " ")
3983     if fmts == "" { // simple decomposition
3984     v := scanv(input)
3985     gsh.iValues = v
3986     //fmt.Printf("%v\n", strings.Join(v, ","))
3987     }else{
3988     v := make( []string, 8)
3989     n, err := fmt.Sscanf(input, fmts, &v[0], &v[1], &v[2], &v[3])
3990     //fmt.Printf("--D-- Sscanf ->(%v) n=%d err=(%v)\n", v, n, err)
3991     gsh.iValues = v
3992     }
3993     }
3994     func (gsh+GshContext)Printv(argv[ ]string){
3995     if false { //@@@
3996     fmt.Printf("%v\n", strings.Join(argv[1:], " "))
3997     return
3998     }
3999     //fmt.Printf("--D-- Printv(%v)\n", argv)
4000     //fmt.Printf("%v\n", strings.Join(gsh.iValues, ","))
4001     div = gsh.iDelimiter
4002     fmts := ""
4003     argv = argv[1:]
4004     if 0 < len(argv) {
4005     if strBegins(argv[0], "-F") {
4006     div = argv[0][2:]
4007     argv = argv[1:]
4008     }
4009     }
4010     optv := []string{}
4011     for _,v := range argv {
4012     if strBegins(v, "-") {
4013     optv = append(optv, v)
4014     argv = argv[1:]
4015     }else{
4016     break;
4017     }
4018     }
4019     if 0 < len(argv) {
4020     fmts = strings.Join(argv, " ")
4021     }
4022     gsh.printfv(fmts, div, argv, optv, gsh.iValues)
4023     }
4024     func (gsh+GshContext)Basename(argv[ ]string){
4025     for i,v := range gsh.iValues {
4026     gsh.iValues[i] = filepath.Base(v)
4027     }
4028     }
4029     func (gsh+GshContext)Sortv(argv[ ]string){
4030     sv := gsh.iValues
4031     sort.Slice(sv, func(i,j int) bool {
4032     return sv[i] < sv[j]
4033     })
4034     }
4035     func (gsh+GshContext)Shiftv(argv[ ]string){
4036     vi := len(gsh.iValues)
4037     if 0 < vi {
4038     if isin("-r", argv) {
4039     top := gsh.iValues[0]
4040     gsh.iValues = append(gsh.iValues[1:], top)
4041     }else{
4042     gsh.iValues = gsh.iValues[1:]
4043     }
4044     }
4045     }
4046     }
4047     func (gsh+GshContext)Enqv(argv[ ]string){
4048     }
4049     func (gsh+GshContext)Deqv(argv[ ]string){
4050     }
4051     func (gsh+GshContext)Push(argv[ ]string){
4052     gsh.iValStack = append(gsh.iValStack, argv[1:])
4053     fmt.Printf("depth=%d\n", len(gsh.iValStack))
4054     }
4055     func (gsh+GshContext)Dump(argv[ ]string){
4056     for i,v := range gsh.iValStack {
4057     fmt.Printf("%d %v\n", i, v)
4058     }
4059     }
4060     func (gsh+GshContext)Pop(argv[ ]string){
4061     depth := len(gsh.iValStack)
4062     if 0 < depth {
4063     v := gsh.iValStack[depth-1]
4064     if isin("-cat", argv){
4065     gsh.iValues = append(gsh.iValues, v...)
4066     }else{
4067     gsh.iValues = v
4068     }
4069     gsh.iValStack = gsh.iValStack[0:depth-1]
4070     fmt.Printf("depth=%d %s\n", len(gsh.iValStack), gsh.iValues)

```

```

4071 }else{
4072     fmt.Printf("depth=%d\n",depth)
4073 }
4074 }
4075 // <a name="interpreter">Command Interpreter</a>
4076 func (gshCtx+GshContext).gshellv(argv []string) (fin bool) {
4077     fin = false
4078
4079     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
4080     if len(argv) <= 0 {
4081         return false
4082     }
4083     xargv := []string{}
4084     for ai := 0; ai < len(argv); ai++ {
4085         xargv = append(xargv,subst(gshCtx,argv[ai],false))
4086     }
4087     argv = xargv
4088     if false {
4089         for ai := 0; ai < len(argv); ai++ {
4090             fmt.Printf("%d %s %d\n",ai,argv[ai],len(argv[ai]))
4091         }
4092     }
4093 }
4094
4095 cmd := argv[0]
4096 if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv),argv) }
4097 switch { // https://tour.golang.org/flowcontrol/11
4098 case cmd == "":
4099     gshCtx.xPvd([]string{}); // empty command
4100 case cmd == "-x":
4101     gshCtx.CmdTrace = 1 - gshCtx.CmdTrace
4102 case cmd == "-xt":
4103     gshCtx.CmdTime = 1 - gshCtx.CmdTime
4104 case cmd == "-ot":
4105     gshCtx.sconnect(true, argv)
4106 case cmd == "-ou":
4107     gshCtx.sconnect(false, argv)
4108 case cmd == "-lt":
4109     gshCtx.saccept(true, argv)
4110 case cmd == "-iu":
4111     gshCtx.saccept(false, argv)
4112 case cmd == "-l" || cmd == "<" || cmd == ">" || cmd == "-a" || cmd == ">>" || cmd == "-s" || cmd == "><":
4113     gshCtx.redirect(argv)
4114 case cmd == "|":
4115     gshCtx.fromPipe(argv)
4116 case cmd == "args":
4117     gshCtx.Args(argv)
4118 case cmd == "bg" || cmd == "-bg":
4119     rfin := gshCtx.inBackground(argv[1:])
4120     return rfin
4121 case cmd == "-bn":
4122     gshCtx.Basename(argv)
4123 case cmd == "call":
4124     // = gshCtx.excommand(false,argv[1:])
4125 case cmd == "cd" || cmd == "chdir":
4126     gshCtx.xChdir(argv);
4127 case cmd == "-cksum":
4128     gshCtx.xFind(argv)
4129 case cmd == "-sum":
4130     gshCtx.xFind(argv)
4131 case cmd == "-sumtest":
4132     str := ""
4133     if 1 < len(argv) { str = argv[1] }
4134     crc := strCRC32(str,uint64(len(str)))
4135     fprintf(stderr,"%v %v\n",crc,len(str))
4136 case cmd == "close":
4137     gshCtx.xClose(argv)
4138 case cmd == "gcp":
4139     gshCtx.FileCopy(argv)
4140 case cmd == "dec" || cmd == "decode":
4141     gshCtx.Dec(argv)
4142 case cmd == "#define":
4143 case cmd == "dic" || cmd == "d":
4144     xDic(argv)
4145 case cmd == "dump":
4146     gshCtx.Dump(argv)
4147 case cmd == "echo" || cmd == "e":
4148     echo(argv,true)
4149 case cmd == "enc" || cmd == "encode":
4150     gshCtx.Enc(argv)
4151 case cmd == "env":
4152     env(argv)
4153 case cmd == "eval":
4154     xEval(argv[1:],true)
4155 case cmd == "ev" || cmd == "events":
4156     dumpEvents(argv)
4157 case cmd == "exec":
4158     // = gshCtx.excommand(true,argv[1:])
4159     // should not return here
4160 case cmd == "exit" || cmd == "quit":
4161     // write Result code EXIT to 3>
4162     return true
4163 case cmd == "fdls":
4164     // dump the attributes of fds (of other process)
4165 case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
4166     gshCtx.xFind(argv[1:])
4167 case cmd == "fu":
4168     gshCtx.xFind(argv[1:])
4169 case cmd == "fork":
4170     // mainly for a server
4171 case cmd == "-gen":
4172     gshCtx.gen(argv)
4173 case cmd == "-go":
4174     gshCtx.xGo(argv)
4175 case cmd == "-grep":
4176     gshCtx.xFind(argv)
4177 case cmd == "gseq":
4178     gshCtx.Deq(argv)
4179 case cmd == "genq":
4180     gshCtx.Enc(argv)
4181 case cmd == "gpop":
4182     gshCtx.Pop(argv)
4183 case cmd == "gpush":
4184     gshCtx.Push(argv)
4185 case cmd == "history" || cmd == "hi": // hi should be alias
4186     gshCtx.xHistory(argv)
4187 case cmd == "jobs":
4188     gshCtx.xJobs(argv)
4189 case cmd == "lisp" || cmd == "nlisp":
4190     gshCtx.SplitLine(argv)
4191 case cmd == "-ls":
4192     gshCtx.xFind(argv)
4193 case cmd == "nop":
4194     // do nothing
4195 case cmd == "pipe":
4196     gshCtx.xOpen(argv)
4197 case cmd == "plug" || cmd == "plugin" || cmd == "pin":
4198     gshCtx.xPlugin(argv[1:])
4199 case cmd == "print" || cmd == "-pr":
4200     // output internal slice // also sprintf should be
4201     gshCtx.Printv(argv)
4202 case cmd == "ps":
4203     gshCtx.xPs(argv)
4204 case cmd == "pstitle":
4205     // to be gsh.title
4206 case cmd == "rexecl" || cmd == "rexd":
4207     gshCtx.RexeclServer(argv)
4208 case cmd == "rexecl" || cmd == "rex":
4209     gshCtx.RexeclClient(argv)
4210 case cmd == "repeat" || cmd == "rep": // repeat cond command
4211     gshCtx.repeat(argv)
4212 case cmd == "replay":
4213     gshCtx.xReplay(argv)
4214 case cmd == "scan":
4215     // scan input (or so in fscanf) to internal slice (like Files or map)
4216     gshCtx.Scanv(argv)
4217 case cmd == "set":
4218     // set name ...
4219 case cmd == "serv":
4220     gshCtx.httpServer(argv)
4221 case cmd == "shift":
4222     gshCtx.Shiftv(argv)
4223 case cmd == "sleep":
4224     gshCtx.sleep(argv)
4225 case cmd == "-sort":
4226     gshCtx.Sortv(argv)
4227
4228 case cmd == "j" || cmd == "join":
4229     gshCtx.RJoin(argv)
4230 case cmd == "a" || cmd == "alpa":
4231     gshCtx.Rexecl(argv)
4232 case cmd == "jcd" || cmd == "jchdir":
4233     gshCtx.Rchdir(argv)
4234 case cmd == "jget":
4235     gshCtx.Rget(argv)
4236 case cmd == "jls":
4237     gshCtx.Rls(argv)
4238 case cmd == "jput":
4239     gshCtx.Rput(argv)
4240 case cmd == "jpwd":
4241     gshCtx.Rpwd(argv)
4242
4243 case cmd == "time":
4244     fin = gshCtx.xTime(argv)
4245 case cmd == "ungets":
4246     if 1 < len(argv) {
4247         ungets(argv[1]+"\\n")
4248     }

```



```

4248     }else{
4249     case cmd == "pwd":
4250     gshCtx.xPwd(argv);
4251     case cmd == "ver" || cmd == "-ver" || cmd == "version":
4252     gshCtx.showVersion(argv)
4253     case cmd == "where":
4254     // data file or so?
4255     case cmd == "which":
4256     which("PATH",argv);
4257     case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
4258     go gj_server(argv[1:]);
4259     case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
4260     go gj_server(argv[1:]);
4261     case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
4262     go gj_client(argv[1:]);
4263     case cmd == "gj":
4264     }send(argv);
4265     case cmd == "jsend":
4266     }send(argv);
4267     default:
4268     if gshCtx.whichPlugin(cmd,[]string{"-s"}) != nil {
4269     gshCtx.xPlugin(argv)
4270     }else{
4271     notfound, _ := gshCtx.excommand(false,argv)
4272     if notfound {
4273     fmt.Printf("--E-- command not found (%v)\n",cmd)
4274     }
4275     }
4276     }
4277     return fin
4278     }
4279 }
4280
4281 func (gsh*GshContext)gshell(gline string) (rfin bool) {
4282 argv := strings.Split(string(gline)," ")
4283 fin := gsh.gshellv(argv)
4284 return fin
4285 }
4286 func (gsh*GshContext)tgshell(gline string)(xfin bool){
4287 start := time.Now()
4288 fin := gsh.gshell(gline)
4289 end := time.Now()
4290 elps := end.Sub(start);
4291 if gsh.CmdTime {
4292     fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + " (%d.%09ds)\n",
4293     elps/100000000,elps%100000000)
4294 }
4295 return fin
4296 }
4297 func Ttyid() (int) {
4298 fi, err := os.Stdin.Stat()
4299 if err != nil {
4300 return 0;
4301 }
4302 //fmt.Printf("Stdin: %v Dev=%d\n",
4303 // fi.Mode(),fi.Mode()&os.ModeDevice)
4304 if (fi.Mode() & os.ModeDevices) != 0 {
4305 stat := aStat_t{};
4306 err := aStat(0,&stat)
4307 if err != nil {
4308 //fmt.Printf("--I-- Stdin: (%v)\n",err)
4309 }else{
4310 //fmt.Printf("--I-- Stdin: rdev=%d %d\n",
4311 // stat.Rdev&0xFF,stat.Rdev);
4312 //fmt.Printf("--I-- Stdin: tty%d\n",stat.Rdev&0xFF);
4313 return int(stat.Rdev & 0xFF)
4314 }
4315 }
4316 return 0
4317 }
4318 func (gshCtx *GshContext) ttyfile() string {
4319 //fmt.Printf("--I-- GSH_HOME=%s\n",gshCtx.GshHomeDir)
4320 ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
4321     fmt.Sprintf("%02d",gshCtx.TerminalId)
4322 //strconv.Itoa(gshCtx.TerminalId)
4323 //fmt.Printf("--I-- ttyfile=%s\n",ttyfile)
4324 return ttyfile
4325 }
4326 func (gshCtx *GshContext) ttyline()(*os.File){
4327 file, err := os.OpenFile(gshCtx.ttyfile(),os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4328 if err != nil {
4329     fmt.Printf("--P-- cannot open %s (%s)\n",gshCtx.ttyfile(),err)
4330     return file;
4331 }
4332 return file
4333 }
4334 func (gshCtx *GshContext)getline(hix int, skipping bool, prevline string) (string) {
4335 if( skipping ) {
4336     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4337     line, _ := reader.Readline()
4338     return string(line)
4339 }else{
4340 if true {
4341     return xgetline(hix,prevline,gshCtx)
4342 }
4343 /*
4344 else
4345 if( with_exgetline && gshCtx.Getline != "" ){
4346     //var xhix int64 = int64(hix); // cast
4347     newenv := os.Environ()
4348     newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix),10) )
4349     tty := gshCtx.ttyline()
4350     tty.WriteString(prevline)
4351     Pa := os.ProcAttr {
4352     // start dir
4353     newenv, //os.Environ(),
4354     []os.File{os.Stdin,os.Stdout,os.Stderr,tty},
4355     nil,
4356     }
4357     //fmt.Printf("--I-- getline=%s // %s\n",gsh_getlinev[0],gshCtx.Getline)
4358     proc, err := os.StartProcess(gsh_getlinev[0],[]string{"getline","getline"},&Pa)
4359     if err != nil {
4360     fmt.Printf("--P-- getline process error (%v)\n",err)
4361     // for ; ; {
4362     return "exit (getline program failed)"
4363     }
4364     //stat, err := proc.Wait()
4365     proc.Wait()
4366     buf := make([]byte,LINESIZE)
4367     count, err := tty.Read(buf)
4368     //_, err = tty.Read(buf)
4369     //fmt.Printf("--D-- getline (%d)\n",count)
4370     if err != nil {
4371     if ! (count == 0) { // && err.String() == "EOF" } {
4372     fmt.Printf("--E-- getline error (%s)\n",err)
4373     }
4374     }else{
4375     //fmt.Printf("--I-- getline OK \"%s\"\n",buf)
4376     }
4377     }
4378     tty.Close()
4379     gline := string(buf[0:count])
4380     return gline
4381     }else
4382     /*
4383     {
4384     // if isatty {"%d",hix)
4385     fmt.Printf("%d",hix)
4386     fmt.Print(PROMPT)
4387     // }
4388     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4389     line, _ := reader.Readline()
4390     return string(line)
4391     }
4392     }
4393 }
4394 //== begin ===== getline
4395 /*
4396 * getline.c
4397 * 2020-0819 extracted from dog.c
4398 * getline.go
4399 * 2020-0822 ported to Go
4400 */
4401 /*
4402 package main // getline main
4403 import (
4404     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
4405     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
4406     "os" // <a href="https://golang.org/pkg/os/">os</a>
4407     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
4408     "bytes" // <a href="https://golang.org/pkg/bytes/">bytes</a>
4409     "os/exec" // <a href="https://golang.org/pkg/os/">os</a>
4410 )
4411 */
4412
4413 // C language compatibility functions
4414 var errno = 0
4415 var stdin *os.File = os.Stdin
4416 var stdout *os.File = os.Stdout
4417 var stderr *os.File = os.Stderr
4418 var EOF = -1
4419 var NULL = 0
4420 type FILE os.File
4421 type StrBuf []byte
4422 var NULL_FP os.File = nil
4423 var NULLSP = 0
4424 //var LINESIZE = 1024

```

```

4425
4426 func system(cmdstr string)(int){
4427 //PA := syscall.ProcAttr {
4428 PA := os.ProcAttr {
4429     "", // the starting directory
4430     os.Environ(),
4431     []uintptr{os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()},
4432     []os.File{os.Stdin,os.Stdout,os.Stderr},
4433     nil,
4434 }
4435 argv := strings.Split(cmdstr, " ")
4436 //pid,err := syscall.ForkExec(argv[0],argv,*PA)
4437 proc,err := os.StartProcess(argv[0],argv,*PA);
4438 if err != nil {
4439     //fmt.Printf("E-- system(%v)\n(%v)\n",cmdstr,err);
4440     return -1;
4441 }
4442 pstat, _ := proc.Wait();
4443 pid := pstat.Pid();
4444 if( err != nil ){
4445     fmt.Printf("E-- pid=%v syscall(%v) err(%v)\n",pid,cmdstr,err)
4446 }
4447 //syscall.Wait4(pid,nil,0,nil)
4448 //fmt.Printf("E==E-- pid=%d exit=%v stat=%v\n",pid,pstat.Exited(),pstat.ExitCode());
4449
4450 /*
4451 argv := strings.Split(cmdstr, " ")
4452 fmt.Printf(os.Stderr, "E-- system(%v)\n",argv)
4453 //cmd := exec.Command(argv[0],...)
4454 cmd := exec.Command(argv[0],argv[1],argv[2])
4455 cmd.Stdin = strings.NewReader("output of system")
4456 var out bytes.Buffer
4457 cmd.Stdout = &out
4458 var serr bytes.Buffer
4459 cmd.Stderr = &serr
4460 err := cmd.Run()
4461 if err != nil {
4462     fmt.Printf(os.Stderr, "E-- system(%v)err(%v)\n",argv,err)
4463     fmt.Printf("ERR:%s\n",serr.String())
4464 }else{
4465     fmt.Printf("%s",out.String())
4466 }
4467 */
4468 return 0
4469 }
4470 func atoi(str string)(ret int){
4471 ret,err := fmt.Scanf(str,"%d",ret)
4472 if err == nil {
4473     return ret
4474 }else{
4475     // should set errno
4476     return 0
4477 }
4478 }
4479 func getenv(name string)(string){
4480 val, got := os.LookupEnv(name)
4481 if got {
4482     return val
4483 }else{
4484     return "?"
4485 }
4486 }
4487 func strcpy(dst StrBuff, src string){
4488     var i int
4489     srcb := []byte(src)
4490     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4491         dst[i] = srcb[i]
4492     }
4493     dst[i] = 0
4494 }
4495 func xstrcpy(dst StrBuff, src StrBuff){
4496     dst = src
4497 }
4498 func strcat(dst StrBuff, src StrBuff){
4499     dst = append(dst,src...)
4500 }
4501 func strdup(str StrBuff)(string){
4502     return string(str[:strlen(str)])
4503 }
4504 func strlen(str string)(int){
4505     return len(str)
4506 }
4507 func strlen(str StrBuff)(int){
4508     var i int
4509     for i = 0; i < len(str) && str[i] != 0; i++ {
4510     }
4511     return i
4512 }
4513 func sizeof(data StrBuff)(int){
4514     return len(data)
4515 }
4516 func isatty(fd int)(ret int){
4517     return 1
4518 }
4519
4520 func fopen(file string,mode string)(fp*os.File){
4521     if mode == "r" {
4522         fp,err := os.Open(file)
4523         if err != nil {
4524             fmt.Printf("E-- fopen(%s,%s)=(%v)\n",file,mode,err)
4525             return NULL_FP;
4526         }
4527         return fp;
4528     }else{
4529         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4530         if( err != nil ){
4531             return NULL_FP;
4532         }
4533         return fp;
4534     }
4535 }
4536
4537 func fclose(fp*os.File){
4538     fp.Close()
4539 }
4540
4541 func fflush(fp *os.File)(int){
4542     return 0
4543 }
4544
4545 func fgetc(fp*os.File)(int){
4546     var buf [1]byte
4547     _,err := fp.Read(buf[0:1])
4548     if( err != nil ){
4549         return EOF;
4550     }else{
4551         return int(buf[0])
4552     }
4553 }
4554
4555 func fgets(str*string, size int, fp*os.File)(int){
4556     buf := make(StrBuff,size)
4557     var ch int
4558     var i int
4559     for i = 0; i < len(buf)-1; i++ {
4560         ch = fgetc(fp)
4561         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
4562         if( ch == EOF ){
4563             break;
4564         }
4565         buf[i] = byte(ch);
4566         if( ch == '\n' ){
4567             break;
4568         }
4569     }
4570     buf[i] = 0
4571     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4572     return i
4573 }
4574
4575 func fgets(buf StrBuff, size int, fp*os.File)(int){
4576     var ch int
4577     var i int
4578     for i = 0; i < len(buf)-1; i++ {
4579         ch = fgetc(fp)
4580         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
4581         if( ch == EOF ){
4582             break;
4583         }
4584         buf[i] = byte(ch);
4585         if( ch == '\n' ){
4586             break;
4587         }
4588     }
4589     buf[i] = 0
4590     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4591     return i
4592 }
4593
4594 func fputc(ch int, fp*os.File)(int){
4595     var buf [1]byte
4596     buf[0] = byte(ch)
4597     fp.Write(buf[0:1])
4598     return 0
4599 }
4600
4601 func fputs(buf StrBuff, fp*os.File)(int){
4602     fp.Write(buf)
4603     return 0
4604 }
4605
4606 func fputs(str string, fp*os.File)(int){
4607     return fputs([]byte(str),fp)
4608 }
4609
4610 func sscanf(str StrBuff,fmts string, params ...interface{})(int){

```

```

4602     fmt.Sprintf(string(str[:strlen(str)]),fmts,params...)
4603     return 0
4604 }
4605 func fprintf(fp*os.File,fmts string, params ...interface{})(int){
4606     fmt.Fprintf(fp,fmts,params...)
4607     return 0
4608 }
4609
4610 // <a name="IME">Command Line IME</a>
4611 //----- MyIME
4612 var MyIMEVER = "MyIME/0.0.2";
4613 type RomKana struct {
4614     dic string // dictionary ID
4615     pat string // input pattern
4616     out string // output pattern
4617     hit int64 // count of hit and used
4618 }
4619 var dicents = 0
4620 var romkana [1024]RomKana
4621 var Romkan []RomKana
4622
4623 func isinDic(str string)(int){
4624     for i,v := range Romkan {
4625         if v.pat == str {
4626             return i
4627         }
4628     }
4629     return -1
4630 }
4631 const {
4632     DIC_COM_LOAD = "im"
4633     DIC_COM_DUMP = "s"
4634     DIC_COM_LIST = "ls"
4635     DIC_COM_ENA = "en"
4636     DIC_COM_DIS = "di"
4637 }
4638 func helpDic(argv []string){
4639     out := stderr
4640     cmd := ""
4641     if 0 < len(argv) { cmd = argv[0] }
4642     fprintf(out,"--- %v Usage\n",cmd)
4643     fprintf(out,"--- Commands\n")
4644     fprintf(out,"... %v %v [dicName] [dicURL] -- Import dictionary\n",cmd,DIC_COM_LOAD)
4645     fprintf(out,"... %v %v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
4646     fprintf(out,"... %v %v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
4647     fprintf(out,"... %v %v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
4648     fprintf(out,"... %v %v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
4649     fprintf(out,"... Keys ... %v\n","ESC can be used for '\\''")
4650     fprintf(out,"... \\c -- Reverse the case of the last character\n",)
4651     fprintf(out,"... \\i -- Replace input with translated text\n",)
4652     fprintf(out,"... \\j -- On/Off translation mode\n",)
4653     fprintf(out,"... \\l -- Force Lower Case\n",)
4654     fprintf(out,"... \\u -- Force Upper Case (software CapsLock)\n",)
4655     fprintf(out,"... \\v -- Show translation actions\n",)
4656     fprintf(out,"... \\x -- Replace the last input character with it Hexa-Decimal\n",)
4657 }
4658 func xDic(argv[]string){
4659     if len(argv) <= 1 {
4660         helpDic(argv)
4661         return
4662     }
4663     argv = argv[1:]
4664     var debug = false
4665     var info = false
4666     var silent = false
4667     var dump = false
4668     var builtin = false
4669     cmd := argv[0]
4670     argv = argv[1:]
4671     opt := ""
4672     arg := ""
4673
4674     if 0 < len(argv) {
4675         arg1 := argv[0]
4676         if arg1[0] == '-' {
4677             switch arg1 {
4678                 default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4679                 return
4680                 case "-b": builtin = true
4681                 case "-d": debug = true
4682                 case "-s": silent = true
4683                 case "-v": info = true
4684             }
4685             opt = arg1
4686             argv = argv[1:]
4687         }
4688     }
4689     dicName := ""
4690     dicURL := ""
4691     if 0 < len(argv) {
4692         arg = argv[0]
4693         dicName = arg
4694         argv = argv[1:]
4695     }
4696     if 0 < len(argv) {
4697         dicURL = argv[0]
4698         argv = argv[1:]
4699     }
4700     if false {
4701         fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4702     }
4703     if cmd == DIC_COM_LOAD {
4704         //dicType := ""
4705         dicBody := ""
4706         if !builtin && dicName != "" && dicURL == "" {
4707             f,err := os.Open(dicName)
4708             if err == nil {
4709                 dicURL = dicName
4710             }else{
4711                 f,err = os.Open(dicName+".html")
4712                 if err == nil {
4713                     dicURL = dicName+".html"
4714                 }else{
4715                     f,err = os.Open("gshdic-"+dicName+".html")
4716                     if err == nil {
4717                         dicURL = "gshdic-"+dicName+".html"
4718                     }
4719                 }
4720             }
4721             if err == nil {
4722                 var buf = make([]byte,128*1024)
4723                 count,err := f.Read(buf)
4724                 f.Close()
4725                 if info {
4726                     fprintf(stderr,"--Id-- ReadDic(%v,%v)\n",count,err)
4727                 }
4728                 dicBody = string(buf[0:count])
4729             }
4730         }
4731     }
4732     if dicBody == "" {
4733         switch arg {
4734             default:
4735                 dicName = "WorldDic"
4736                 dicURL = WorldDic
4737                 if info {
4738                     fprintf(stderr,"--Id-- default dictionary \"%v\"\n",
4739                         dicName);
4740                 }
4741             case "wnn":
4742                 dicName = "WnnDic"
4743                 dicURL = WnnDic
4744             case "sumomo":
4745                 dicName = "SumomoDic"
4746                 dicURL = SumomoDic
4747             case "sijimi":
4748                 dicName = "SijimiDic"
4749                 dicURL = SijimiDic
4750             case "jkl":
4751                 dicName = "JKLJaDic"
4752                 dicURL = JA_JKLJaDic
4753         }
4754         if debug {
4755             fprintf(stderr,"--Id-- %v URL=%v\n",dicName,dicURL);
4756         }
4757         dicv := strings.Split(dicURL,",")
4758         if debug {
4759             fprintf(stderr,"--Id-- %v encoded data...\n",dicName)
4760             fprintf(stderr,"Type: %v\n",dicv[0])
4761             fprintf(stderr,"Body: %v\n",dicv[1])
4762             fprintf(stderr,"")
4763         }
4764         body, _ := base64.StdEncoding.DecodeString(dicv[1])
4765         dicBody = string(body)
4766     }
4767     if info {
4768         fmt.Printf("--Id-- %v %v\n",dicName,dicURL)
4769         fmt.Printf("%s\n",dicBody)
4770     }
4771     if debug {
4772         fprintf(stderr,"--Id-- dicName %v text...\n",dicName)
4773         fprintf(stderr,"%v\n",string(dicBody))
4774     }
4775     entv := strings.Split(dicBody,"\n");
4776     if info {
4777         fprintf(stderr,"--Id-- %v scan...\n",dicName);
4778     }

```

```

4779 var added int = 0
4780 var dup int = 0
4781 for i,v := range entv {
4782     var pat string
4783     var out string
4784     fmt.Sscanf(v,"%s %s", &pat, &out)
4785     if len(pat) <= 0 {
4786     }else{
4787         if 0 <= isinDic(pat) {
4788             dup += 1
4789             continue
4790         }
4791         romkana[dictents] = RomKana{dicName, pat, out, 0}
4792         dictents += 1
4793         added += 1
4794         Romkan = append(Romkan, RomKana{dicName, pat, out, 0})
4795         if debug {
4796             fmt.Printf("%3v: (%2v)%s-%s (%2v)%s\n",
4797                 i, len(pat), pat, len(out), out)
4798         }
4799     }
4800 }
4801 if !silent {
4802     url := dicURL
4803     if strBegins(url, "data:") {
4804         url = "builtin"
4805     }
4806     fprintf(stderr, "--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4807         dicName, added, dup, len(Romkan), url);
4808 }
4809 // should sort by pattern length for complete match, for performance
4810 if debug {
4811     arg = "" // search pattern
4812     dump = true
4813 }
4814 }
4815 if cmd == DIC_COM_DUMP || dump {
4816     fprintf(stderr, "--Id-- %v dump... %v entries:\n", dicName, len(Romkan));
4817     var match = 0
4818     for i := 0; i < len(Romkan); i++ {
4819         dic := Romkan[i].dic
4820         pat := Romkan[i].pat
4821         out := Romkan[i].out
4822         if arg == "" || 0 <= strings.Index(pat, arg) || 0 <= strings.Index(out, arg) {
4823             fmt.Printf("%3v: (%2v)%s-%s (%2v)%s\n",
4824                 i, dic, len(pat), pat, len(out), out)
4825             match += 1
4826         }
4827     }
4828     fprintf(stderr, "--Id-- %v matched %v / %v entries:\n", arg, match, len(Romkan));
4829 }
4830 }
4831 func loadDefaultDic(dic int){
4832     if( 0 < len(Romkan) ){
4833         return
4834     }
4835     //fprintf(stderr, "\r\n")
4836     xDic([]string{"dic", DIC_COM_LOAD});
4837 }
4838 var info = false
4839 if info {
4840     fprintf(stderr, "--Id-- Congraturations!! WorldDic is now activated.\r\n")
4841     fprintf(stderr, "--Id-- enter \"dic\" command for help.\r\n")
4842 }
4843 }
4844 func readDic()(int){
4845     /*
4846     var rk *os.File;
4847     var dic = "MyIME-dic.txt";
4848     //rk = fopen("romkana.txt", "r");
4849     //rk = fopen("JK-JA-morse-dic.txt", "r");
4850     rk = fopen(dic, "r");
4851     if( rk == NULL_FP ){
4852         if true {
4853             fprintf(stderr, "--s-- Could not load %s\n", MyIMEVER, dic);
4854         }
4855         return -1;
4856     }
4857     if( true ){
4858         var di int;
4859         var line = make(StrBuff, 1024);
4860         var pat string
4861         var out string
4862         for di = 0; di < 1024; di++ {
4863             if( fgets(line, sizeof(line), rk) == NULLSP ){
4864                 break;
4865             }
4866             fmt.Sscanf(string(line[0:strlen(line)]), "%s %s", &pat, &out);
4867             //scanf(line, "%s %s", &pat, &out);
4868             romkana[di].pat = pat;
4869             romkana[di].out = out;
4870             //fprintf(stderr, "--Dd- %s-%s\n", pat, out)
4871         }
4872         dictents += di
4873         if( false ){
4874             fprintf(stderr, "--s-- loaded romkana.txt [%d]\n", MyIMEVER, di);
4875             for di = 0; di < dictents; di++ {
4876                 fprintf(stderr,
4877                     "%s %s\n", romkana[di].pat, romkana[di].out);
4878             }
4879         }
4880     }
4881     fclose(rk);
4882 }
4883 //romkana[dictents].pat = "//ddump"
4884 //romkana[dictents].pat = "//ddump" // dump the dic. and clean the command input
4885 }
4886 return 0;
4887 }
4888 func matchlen(str1 string, pat1 string)(int){
4889     if strBegins(str1, pat1) {
4890         return len(pat1)
4891     }else{
4892         return 0
4893     }
4894 }
4895 }
4896 func convs(src string)(string){
4897     var si int;
4898     var sx = len(src);
4899     var di int;
4900     var mi int;
4901     var dstb []byte
4902     for si = 0; si < sx; { // search max. match from the position
4903         if strBegins(src[si:], "x") {
4904             // %x/integer/ // s/a/b/
4905             ix := strings.Index(src[si+3:], "/")
4906             if 0 < ix {
4907                 var iv int = 0
4908                 //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4909                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4910                 sval := fmt.Sprintf("%x", iv)
4911                 bval := []byte(sval)
4912                 dstb = append(dstb, bval...)
4913                 si = si+3+ix+1
4914                 continue
4915             }
4916         }
4917         if strBegins(src[si:], "%d") {
4918             // %d/integer/ // s/a/b/
4919             ix := strings.Index(src[si+3:], "/")
4920             if 0 < ix {
4921                 var iv int = 0
4922                 fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4923                 sval := fmt.Sprintf("%d", iv)
4924                 bval := []byte(sval)
4925                 dstb = append(dstb, bval...)
4926                 si = si+3+ix+1
4927                 continue
4928             }
4929         }
4930         if strBegins(src[si:], "%t") {
4931             now := time.Now()
4932             if true {
4933                 date := now.Format(time.Stamp)
4934                 dstb = append(dstb, []byte(date)...)
4935                 si = si+3
4936             }
4937             continue
4938         }
4939         var maxlen int = 0;
4940         var len int;
4941         mi = -1;
4942         for di = 0; di < dictents; di++ {
4943             len = matchlen(src[si:], romkana[di].pat);
4944             if( maxlen < len ){
4945                 maxlen = len;
4946                 mi = di;
4947             }
4948         }
4949         if( 0 < maxlen ){
4950             out := romkana[mi].out;
4951             dstb = append(dstb, []byte(out)...);
4952             si += maxlen;
4953         }else{
4954             dstb = append(dstb, src[si])
4955             si += 1;

```

```

4956     }
4957     return string(dstb)
4958 }
4959 }
4960 func trans(src string)(int){
4961     dst := convs(src);
4962     xputc(dst,stderr);
4963     return 0;
4964 }
4965 }
4966 //----- LINEEDIT
4967 // "?" at the top of the line means searching history
4968 }
4969 // should be compatilbe with Telnet
4970 const (
4971     EV_MODE      = 255
4972     EV_IDLE     = 254
4973     EV_TIMEOUT  = 253
4974
4975     GO_UP       = 252 // k
4976     GO_DOWN    = 251 // j
4977     GO_RIGHT   = 250 // l
4978     GO_LEFT    = 249 // h
4979     DEL_RIGHT  = 248 // x
4980     GO_TOPL   = 'A'-0x40 // 0
4981     GO_ENDL   = 'E'-0x40 // $
4982
4983     GO_TOPW    = 239 // b
4984     GO_ENDW   = 238 // e
4985     GO_NEXTW  = 237 // w
4986
4987     GO_FORWCH  = 229 // f
4988     GO_PAIRCH = 228 // %
4989
4990     GO_DEL     = 219 // d
4991
4992     HI_SRCH_FW = 209 // /
4993     HI_SRCH_BK = 208 // ?
4994     HI_SRCH_FW = 207 // n
4995     HI_SRCH_RBK = 206 // N
4996 )
4997
4998 // should return number of octets ready to be read immediately
4999 //fprintf(stderr, "\n--Select(%v %v)\n", err, r.Bits[0])
5000
5001
5002 var EventRecvFd = -1 // file descriptor
5003 var EventSendFd = -1
5004 const EventFdOffset = 1000000
5005 const NormalFdOffset = 100
5006
5007 /* 2020-1021 replaced poll() with channel/select
5008 func putKeyinEvent(event int, evarg int){
5009     if true {
5010         if EventRecvFd < 0 {
5011             var pv = [1]int{-1,-1}
5012             syscall.Pipe(pv)
5013             EventRecvFd = pv[0]
5014             EventSendFd = pv[1]
5015             //fmt.Printf("--De-- EventPipe created(%v,%v)\n", EventRecvFd, EventSendFd)
5016         }
5017     }else{
5018         if EventRecvFd < 0 {
5019             // the document differs from this spec
5020             // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
5021             sv, err := syscall.Socketpair(syscall.AF_UNIX, syscall.SOCK_STREAM, 0)
5022             EventRecvFd = sv[0]
5023             EventSendFd = sv[1]
5024             if err != nil {
5025                 //fmt.Printf("--De-- EventSock created(%v,%v)(%v)\n",
5026                     EventRecvFd, EventSendFd, err)
5027             }
5028         }
5029     }
5030     var buf = []byte{ byte(event)}
5031     n, err := syscall.Write(EventSendFd, buf)
5032     if err != nil {
5033         //fmt.Printf("--De-- putEvent(%v)(%v)(%v)\n", EventSendFd, event, n, err)
5034     }
5035 }
5036 */
5037 func ungets(str string){
5038     for _, ch := range str {
5039         putKeyinEvent(int(ch), 0)
5040     }
5041 }
5042 func (gsh*GshContext)xReplay(argv []string){
5043     hix := 0
5044     tempo := 1.0
5045     xtempo := 1.0
5046     repeat := 1
5047
5048     for _, a := range argv { // tempo
5049         if strBegins(a, "x") {
5050             //fmt.Sscanf(a[1:], "%f", &xtempo)
5051             tempo = 1 / xtempo
5052             //fprintf(stderr, "--Dr-- tempo=%(v)%v\n", a[2:], tempo);
5053         }else{
5054             if strBegins(a, "r") { // repeat
5055                 //fmt.Sscanf(a[1:], "%v", &repeat)
5056             }else{
5057                 if strBegins(a, "!") {
5058                     //fmt.Sscanf(a[1:], "%d", &hix)
5059                 }else{
5060                     //fmt.Sscanf(a, "%d", &hix)
5061                 }
5062             }
5063             if hix == 0 || len(argv) <= 1 {
5064                 hix = len(gsh.CommandHistory)-1
5065             }
5066             //fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n", hix, xtempo, repeat)
5067             //dumpEvents(hix)
5068             //gsh.xScanReplay(hix, false, repeat, tempo, argv)
5069             go gsh.xScanReplay(hix, true, repeat, tempo, argv)
5070
5071             runtime.Gosched(); // wait xScanReplay is launched
5072             //fmt.Printf("--Ir-- Replay set\n");
5073         }
5074     }
5075     // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select(</a>
5076     // 2020-0827 GShell-0.2.3
5077     /*
5078     func FpollInl(fp *os.File, usec int)(uintptr){
5079         nfd := 1
5080
5081         rdv := syscall.FdSet {}
5082         fd1 := fp.Fd()
5083         bank1 := fd1/32
5084         mask1 := int32(1 << fd1)
5085         rdv.Bits[bank1] = mask1
5086
5087         fd2 := -1
5088         bank2 := -1
5089         var mask2 int32 = 0
5090
5091         if 0 <= EventRecvFd {
5092             fd2 = EventRecvFd
5093             nfd = fd2 + 1
5094             bank2 = fd2/32
5095             mask2 = int32(1 << fd2)
5096             rdv.Bits[bank2] |= mask2
5097             //fmt.Printf("--De-- EventPoll mask added (%d)|(%v)|(%v)\n", fd2, bank2, mask2)
5098         }
5099
5100         tout := syscall.NsecToTimeval(int64(usec*1000))
5101         //n, err := syscall.Select(nfd, &rdv, nil, nil, &tout) // spec. mismatch
5102         err := syscall.Select(nfd, &rdv, nil, nil, &tout)
5103         if err != nil {
5104             //fmt.Printf("--De-- select() err(%v)\n", err)
5105         }
5106         if err == nil {
5107             if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
5108                 if false {
5109                     //fmt.Printf("--De-- got Event\n")
5110                 }
5111                 return uintptr(EventFdOffset + fd2)
5112             }else{
5113                 if (rdv.Bits[bank1] & mask1) != 0 {
5114                     return uintptr(NormalFdOffset + fd1)
5115                 }else{
5116                     return 1
5117                 }
5118             }else{
5119                 return 0
5120             }
5121         }
5122     }
5123     /*
5124     func fgetcTimeout1(fp *os.File, usec int)(int){
5125     READ:
5126         //readyFd := FpollInl(fp, usec)
5127         readyFd := FpollInl(fp, usec)
5128         if readyFd < 100 {
5129             return EV_TIMEOUT
5130         }
5131         var buf [1]byte
5132     }

```

```

5133
5134 if EventFdOffset <= readyFd {
5135     fd := int(readyFd-EventFdOffset)
5136     _,err := syscall.Read(fd,buf[0:1])
5137     if (err != nil){
5138         return EOF;
5139     }else{
5140         if buf[0] == EV_MODE {
5141             recvKeyEvent(fd)
5142             goto READ1
5143         }
5144         return int(buf[0])
5145     }
5146 }
5147
5148 _,err := fp.Read(buf[0:1])
5149 if( err != nil ){
5150     return EOF;
5151 }else{
5152     return int(buf[0])
5153 }
5154 }
5155 */
5156
5157 func visibleChar(ch int)(string){
5158     switch {
5159     case 'l' <= ch && ch <= '-':
5160         return string(ch)
5161     }
5162     switch ch {
5163     case '\a': return "\\a"
5164     case '\n': return "\\n"
5165     case '\r': return "\\r"
5166     case '\t': return "\\t"
5167     }
5168     switch ch {
5169     case 0x00: return "NUL"
5170     case 0x07: return "BEL"
5171     case 0x08: return "BS"
5172     case 0x0E: return "SO"
5173     case 0x0F: return "SI"
5174     case 0x1B: return "ESC"
5175     case 0x7F: return "DEL"
5176 }
5177     switch ch {
5178     case EV_IDLE: return fmt.Sprintf("IDLE")
5179     case EV_MODE: return fmt.Sprintf("MODE")
5180     }
5181     return fmt.Sprintf("%X",ch)
5182 }
5183 /*
5184 func recvKeyEvent(fd int){
5185     var buf = make([]byte,1)
5186     _,_ = syscall.Read(fd,buf[0:1])
5187     if( buf[0] != 0 ){
5188         romkanmode = true
5189     }else{
5190         romkanmode = false
5191     }
5192 }
5193 */
5194 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
5195     var Start time.Time
5196     var events = []Event{}
5197     for _,e := range Events {
5198         if hix == 0 || e.CmdIndex == hix {
5199             events = append(events,e)
5200         }
5201     }
5202     elen := len(events)
5203     if 0 < elen {
5204         if events[elen-1].event == EV_IDLE {
5205             events = events[0:elen-1]
5206         }
5207     }
5208     for r := 0; r < repeat; r++ {
5209         for i,e := range events {
5210             nano := e.when.Nanosecond()
5211             micro := nano / 1000
5212             if Start.Second() == 0 {
5213                 Start = time.Now()
5214             }
5215             diff := time.Now().Sub(Start)
5216             if replay {
5217                 if e.event == EV_IDLE {
5218                     //fmt.Printf("--replay %v / %v event=%X\n",i,len(events),e.event);
5219                     putKeyEvent(e.event,0)
5220                     if e.event == EV_MODE { // event with arg
5221                         putKeyEvent(int(e.evarg),0)
5222                     }
5223                 }else{
5224                     //fmt.Printf("--replay %v / %v idle=%X\n",i,len(events),e.event);
5225                 }
5226             }else{
5227                 fmt.Printf("%7.3fms #%-3v %-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",
5228                     float64(diff)/1000000.0,
5229                     i,
5230                     e.CmdIndex,
5231                     e.when.Format(time.Stamp),micro,
5232                     e.event,e.event.visibleChar(e.event),
5233                     float64(e.evarg)/1000000.0)
5234             }
5235             if e.event == EV_IDLE {
5236                 //fmt.Printf("--replay %v / %v delay\n",i,len(events));
5237                 d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
5238                 //n.sleep(time.Duration(e.evarg))
5239                 n.sleep(d)
5240             }
5241         }
5242     }
5243 }
5244 func dumpEvents(argv[]string){
5245     hix := 0
5246     if 1 < len(argv) {
5247         fmt.Sscanf(argv[1],"%d",&hix)
5248     }
5249     for i,e := range Events {
5250         nano := e.when.Nanosecond()
5251         micro := nano / 1000
5252         //if e.event != EV_TIMEOUT {
5253             if hix == 0 || e.CmdIndex == hix {
5254                 fmt.Printf("#%-3v %-3v [%v.%06d] %3v %02X %-4v %10.3fms\n",i,
5255                     e.CmdIndex,
5256                     e.when.Format(time.Stamp),micro,
5257                     e.event,e.event.visibleChar(e.event),float64(e.evarg)/1000000.0)
5258             }
5259         }
5260     }
5261 }
5262 /*
5263 func fgetcTimeout(fp *os.File,usec int)(int){
5264     ch := fgetcTimeout1(fp,usec)
5265     if ch != EV_TIMEOUT {
5266         now := time.Now()
5267         if 0 < len(Events) {
5268             last := Events[len(Events)-1]
5269             dura := int64(now.Sub(last.when))
5270             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5271         }
5272         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5273     }
5274     return ch
5275 }
5276 */
5277 // 2020-1021 replaced poll() with channel/select
5278 var Kbd = make(chan int);
5279 var Kbinit = false;
5280 var evQ = make(chan int);
5281 /*
5282 func keyInput(kbd chan int, fp *os.File){
5283     for {
5284         ch := C.getc(C.stdin);
5285         if (ch == C.EOF){
5286             break;
5287         }
5288         kbd <- int(ch);
5289     }
5290 }
5291 */
5292 // https://godoc.org/golang.org/x/crypto/ssh/terminal
5293 // https://stackoverflow.com/questions/14094190/function-similar-to-getchar
5294 func keyInput(kbd chan int, tty *os.File){
5295     tmode := C.setTermRaw();
5296     defer func(){ C.setTermMode(tmode); }();
5297     if( !OnWindows ){
5298         system("/bin/stty -echo -icanon");
5299         defer func(){ system("/bin/stty echo sane"); }();
5300     }
5301     for {
5302         var rbuf []byte = make([]byte,1);
5303         if( OnWindows ){
5304             C.setTermRaw();
5305         }
5306         rerr := tty.Read(rbuf);
5307         if( rerr != nil ){
5308             break;
5309         }

```

```

5310     }
5311     //fmt.Printf("++KBD[%X]\n", rbuf[0]);
5312     Kbd <- int(rbuf[0]);
5313 }
5314 if( !OnWindows ){ system("/bin/stty echo sane"); }
5315 }
5316 func fgetcTimeout(fp *os.File, usec int)(int){
5317     if( !Kbinit ){
5318         Kbinit = true;
5319         go keyInput(Kbd, fp);
5320     }
5321     for {
5322         select {
5323             case <- time.After(time.Duration(usec*1000)):
5324                 //fmt.Printf("--Timeout %v us\n", usec);
5325                 return EV_TIMEOUT;
5326             case ch := <- Kbd:
5327                 //fmt.Printf("--KBD[%X]\n", ch);
5328                 // record a KeyIn(ch) Event
5329                 {
5330                     now := time.Now()
5331                     if 0 < len(Events) {
5332                         last := Events[len(Events)-1]
5333                         dura := int64(now.Sub(last.when))
5334                         Events = append(Events, Event{last.when, EV_IDLE, dura, last.CmdIndex})
5335                     }
5336                     Events = append(Events, Event{time.Now(), ch, 0, CmdIndex})
5337                 }
5338                 return ch;
5339             case ch := <- evQ:
5340                 if( ch == EV_MODE ){
5341                     recvKeyEvent()
5342                 }else{
5343                     return ch;
5344                 }
5345             }
5346     }
5347 }
5348 func putKeyEvent(event int, evarg int){
5349     evQ <- event;
5350 }
5351 func recvKeyEvent(){
5352     ch := <- evQ;
5353     if( ch != 0 ){
5354         romkanmode = true
5355     }else{
5356         romkanmode = false
5357     }
5358 }
5359 }
5360 var AtConsoleLineTop = true
5361 var TtyMaxCol = 72 // to be obtained by ioctl?
5362 var EscTimeout = (100*1000)
5363 var (
5364     MODE_VicMode    bool // vi compatible command mode
5365     MODE_ShowMode   bool
5366     romkanmode      bool // shown translation mode, the mode to be retained
5367     MODE_Recursive  bool // recursive translation
5368     MODE_CapsLock   bool // software CapsLock
5369     MODE_LowerLock  bool // force lower-case character lock
5370     MODE_ViInsert   int // visible insert mode, should be like "I" icon in X Window
5371     MODE_ViTrace    bool // output newline before translation
5372 )
5373 type IInput struct {
5374     lno int
5375     lastlno int
5376     pch []int // input queue
5377     prompt string
5378     line string
5379     right string
5380     inMode bool
5381     pinMode bool
5382     waitingMeta string // waiting meta character
5383     LastCmd string
5384 }
5385 func (iin*IInput)Getc(timeoutUs int)(int){
5386     ch1 := EOF
5387     ch2 := EOF
5388     ch3 := EOF
5389     if( 0 < len(iin.pch) ){ // deQ
5390         ch1 = iin.pch[0]
5391         iin.pch = iin.pch[1:]
5392     }else{
5393         ch1 = fgetcTimeout(stdin, timeoutUs);
5394     }
5395     if( ch1 == 033 ){ // escape sequence
5396         ch2 = fgetcTimeout(stdin, EscTimeout);
5397         if( ch2 == EV_TIMEOUT ){
5398             }else{
5399                 ch3 = fgetcTimeout(stdin, EscTimeout);
5400                 if( ch3 == EV_TIMEOUT ){
5401                     iin.pch = append(iin.pch, ch2) // enQ
5402                 }else{
5403                     switch( ch2 ){
5404                         default:
5405                             iin.pch = append(iin.pch, ch2) // enQ
5406                             iin.pch = append(iin.pch, ch3) // enQ
5407                         case '[':
5408                             switch( ch3 ){
5409                                 case 'A': ch1 = GO_UP; // ^
5410                                 case 'B': ch1 = GO_DOWN; // v
5411                                 case 'C': ch1 = GO_RIGHT; // >
5412                                 case 'D': ch1 = GO_LEFT; // <
5413                                 case '3':
5414                                     ch4 := fgetcTimeout(stdin, EscTimeout);
5415                                     if( ch4 == '-' ){
5416                                         //fprintf(stderr, "%02X %02X %02X\n", ch1, ch2, ch3, ch4);
5417                                         ch1 = DEL_RIGHT
5418                                     }
5419                                 case '\\':
5420                                     //ch4 := fgetcTimeout(stdin, EscTimeout);
5421                                     //fprintf(stderr, "%02X %02X %02X\n", ch1, ch2, ch3, ch4);
5422                                     switch( ch3 ){
5423                                         case '-': ch1 = DEL_RIGHT
5424                                     }
5425                                 }
5426                             }
5427                     }
5428                 }
5429             }
5430         }
5431     }
5432     return ch1
5433 }
5434 func (iin*IInput)clearline(){
5435     var i int
5436     fprintf(stderr, "\r");
5437     // should be ANSI ESC sequence
5438     for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
5439         fputc(' ', os.Stderr);
5440     }
5441     fprintf(stderr, "\r");
5442 }
5443 func (iin*IInput)Redraw(){
5444     redraw(iin, iin.lno, iin.line, iin.right)
5445 }
5446 func redraw(iin *IInput, lno int, line string, right string){
5447     inMeta := false
5448     showMode := ""
5449     showMeta := "" // visible Meta_mode on the cursor position
5450     showlno := fmt.Sprintf("%d", lno)
5451     InsertMark := "" // in visible insert mode
5452 }
5453 if MODE_VicMode {
5454     if 0 < len(iin.right) {
5455         InsertMark = " "
5456     }
5457 }
5458 if( 0 < len(iin.waitingMeta) ){
5459     inMeta = true
5460     if iin.waitingMeta[0] != 033 {
5461         showMeta = iin.waitingMeta
5462     }
5463 }
5464 if( romkanmode ){
5465     //romkanmark = " * ";
5466 }else{
5467     //romkanmark = "";
5468 }
5469 if MODE_ShowMode {
5470     romkan := "-"
5471     inmeta := "-"
5472     inveri := ""
5473     if MODE_CapsLock {
5474         inmeta = "A"
5475     }
5476     if MODE_LowerLock {
5477         inmeta = "a"
5478     }
5479     if MODE_ViTrace {
5480         inveri = "v"
5481     }
5482     if MODE_VicMode {
5483         inveri = ":"
5484     }
5485     if romkanmode {
5486         romkan = "\343\201\202"
5487         if MODE_CapsLock {

```

```

5487         inmeta = "R"
5488     }else{
5489         inmeta = "r"
5490     }
5491 }
5492 if inMeta {
5493     inmeta = "\\ "
5494 }
5495 showMode = "["+romkan+inmeta+inveri+"]";
5496 }
5497 Pre := "r" + showMode + showLineo
5498 Output := ""
5499 Left := ""
5500 Right := ""
5501 if romkanmode {
5502     Left = convs(line)
5503     Right = InsertMark+convs(right)
5504 }else{
5505     Left = line
5506     Right = InsertMark+right
5507 }
5508 Output = Pre+Left
5509 if MODE_ViTrace {
5510     Output += iin.LastCmd
5511 }
5512 Output += showMeta+Right
5513 for len(Output) < TtyMaxCol { // to the max. position that may be dirty
5514     Output += " "
5515     // should be ANSI ESC sequence
5516     // not necessary just after newline
5517 }
5518 Output += Pre+Left+showMeta // to set the cursor to the current input position
5519 fprintf(stderr,"%s",Output)
5520
5521 if MODE_ViTrace {
5522     if 0 < len(iin.LastCmd) {
5523         iin.LastCmd = ""
5524         fprintf(stderr,"\r\n")
5525     }
5526 }
5527 AtConsoleLineTop = false
5528 //fmt.Printf("(Redraw(%v)(%v))\n",len(line),len(right));
5529 }
5530 // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
5531 func delHeadChar(str string)(rline string,head string){
5532     clen := utf8.DecodeRune([]byte(str))
5533     head = string(str[0:clen])
5534     return str[clen:],head
5535 }
5536 func delTailChar(str string)(rline string, last string){
5537     var i = 0
5538     var clen = 0
5539     for {
5540         ,siz := utf8.DecodeRune([]byte(str)[i:])
5541         if siz <= 0 { break }
5542         clen = siz
5543         i += siz
5544     }
5545     last = str[len(str)-clen:]
5546     return str[0:len(str)-clen],last
5547 }
5548
5549 // > for output and history
5550 // < for keylog?
5551 // <a name="getline">Command Line Editor</a>
5552 func xgetline(lno int, prevline string, gsh*GshContext)(string){
5553     var iin*Input
5554     iin.lino = lno
5555     iin.lno = lno
5556
5557     CmdIndex = len(gsh.CommandHistory)
5558     if( isatty(0) == 0 ){
5559         if( sfgets(&iin.line,LINESIZE,stdin) == NULL ){
5560             iin.line = "exit\n";
5561         }else{
5562             return iin.line
5563         }
5564     }
5565     if( true ){
5566         //var pts string;
5567         //pts = ptsname(0);
5568         //pts = ttyname(0);
5569         //fprintf(stderr,"--pts[0] = %s\n,pts?pts:");
5570     }
5571     if( false ){
5572         fprintf(stderr,"! ");
5573         fflush(stderr);
5574         sfgets(&iin.line,LINESIZE,stdin);
5575         return iin.line
5576     }
5577     if( !OnWindows ){ system("/bin/stty -echo -icanon"); }
5578     xline = iin.xgetline(prevline,gsh)
5579     if( !OnWindows ){system("/bin/stty echo sane"); }
5580     return xline
5581 }
5582 func (iin*Input)Translate(cmdch int){
5583     romkanmode = !romkanmode;
5584     if MODE_ViTrace {
5585         fprintf(stderr,"%v\r\n",string(cmdch));
5586     }else{
5587         if( cmdch == 'J' ){
5588             fprintf(stderr,"J\r\n");
5589             iin.lnmode = true
5590         }
5591         iin.Redraw();
5592         loadDefaultDic(cmdch);
5593         iin.Redraw();
5594     }
5595     func (iin*Input)Replace(cmdch int){
5596         iin.LastCmd = fmt.Sprintf("%v",string(cmdch))
5597         iin.Redraw();
5598         loadDefaultDic(cmdch);
5599         dst := convs(iin.line+iin.right);
5600         iin.line = dst
5601         iin.right = ""
5602     }
5603     if( cmdch == 'I' ){
5604         fprintf(stderr,"I\r\n");
5605         iin.lnmode = true
5606     }
5607     iin.Redraw();
5608 }
5609 // aa 12 alal
5610 func isAlpha(ch rune)(bool){
5611     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5612         return true
5613     }
5614     return false
5615 }
5616 func isAlnum(ch rune)(bool){
5617     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5618         return true
5619     }
5620     if '0' <= ch && ch <= '9' {
5621         return true
5622     }
5623     return false
5624 }
5625 // 0.2.8 2020-0901 created
5626 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5627 func (iin*Input)GotoTOPW(){
5628     str iin.line
5629     i := len(str)
5630     if i <= 0 {
5631         return
5632     }
5633     //i0 := i
5634     i -= 1
5635     lastSize := 0
5636     var lastRune rune
5637     var found = -1
5638     for 0 < i { // skip preamble spaces
5639         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5640         if !isAlnum(lastRune) { // character, type, or string to be searched
5641             i -= lastSize
5642             continue
5643         }
5644         break
5645     }
5646     for 0 < i {
5647         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5648         if lastSize <= 0 { continue } // not the character top
5649         if !isAlnum(lastRune) { // character, type, or string to be searched
5650             found = i
5651             break
5652         }
5653         i -= lastSize
5654     }
5655     if found < 0 && i == 0 {
5656         found = 0
5657     }
5658     if 0 <= found {
5659         if !isAlnum(lastRune) { // or non-kana character
5660             //else{ // when positioning to the top o the word
5661                 i += lastSize
5662             }
5663         iin.right = str[i:] + iin.right

```



```

5664         if 0 < i {
5665             iin.line = str[0:i]
5666         }else{
5667             iin.line = ""
5668         }
5669     }
5670     //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i0,i,found,iin.line,iin.right)
5671     //fmt.Printf("") // set debug messae at the end of line
5672 }
5673 // 0.2.8 2020-0901 created
5674 func (iin*Input)GotoENDW(){
5675     str := iin.right
5676     if len(str) <= 0 {
5677         return
5678     }
5679     lastSize := 0
5680     var lastRune rune
5681     var lastW = 0
5682     i := 0
5683     inWord := false
5684
5685     lastRune,lastSize = utf8.DecodeRuneInString(str[0:])
5686     if !isAlnum(lastRune) {
5687         str = utf8.DecodeRuneInString(str[lastSize:])
5688         if 0 < z && !isAlnum(r) {
5689             inWord = true
5690         }
5691     }
5692     for i < len(str) {
5693         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5694         if lastSize <= 0 { break } // broken data?
5695         if !isAlnum(lastRune) { // character, type, or string to be searched
5696             break
5697         }
5698         lastW = i // the last alnum if in alnum word
5699         i += lastSize
5700     }
5701     if inWord {
5702         goto DISP
5703     }
5704     for i < len(str) {
5705         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5706         if lastSize <= 0 { break } // broken data?
5707         if !isAlnum(lastRune) { // character, type, or string to be searched
5708             break
5709         }
5710         i += lastSize
5711     }
5712     for i < len(str) {
5713         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5714         if lastSize <= 0 { break } // broken data?
5715         if !isAlnum(lastRune) { // character, type, or string to be searched
5716             break
5717         }
5718         lastW = i
5719         i += lastSize
5720     }
5721 DISP:
5722     if 0 < lastW {
5723         iin.line = iin.line + str[0:lastW]
5724         iin.right = str[lastW:]
5725     }
5726     //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
5727     //fmt.Printf("") // set debug messae at the end of line
5728 }
5729 // 0.2.8 2020-0901 created
5730 func (iin*Input)GotoNEXTW(){
5731     str := iin.right
5732     if len(str) <= 0 {
5733         return
5734     }
5735     lastSize := 0
5736     var lastRune rune
5737     var found = -1
5738     i := 1
5739     for i < len(str) {
5740         lastRune,lastSize = utf8.DecodeRuneInString(str[i:])
5741         if lastSize <= 0 { break } // broken data?
5742         if !isAlnum(lastRune) { // character, type, or string to be searched
5743             found = i
5744             break
5745         }
5746         i += lastSize
5747     }
5748     if 0 < found {
5749         if !isAlnum(lastRune) { // or non-kana character
5750             // when positioning to the top o the word
5751             found += lastSize
5752         }
5753         iin.line = iin.line + str[0:found]
5754         if 0 < found {
5755             iin.right = str[found:]
5756         }else{
5757             iin.right = ""
5758         }
5759     }
5760     //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
5761     //fmt.Printf("") // set debug messae at the end of line
5762 }
5763 // 0.2.8 2020-0902 created
5764 func (iin*Input)GotoPAIRCH(){
5765     str := iin.right
5766     if len(str) <= 0 {
5767         return
5768     }
5769     lastRune,lastSize := utf8.DecodeRuneInString(str[0:])
5770     if lastSize <= 0 {
5771         return
5772     }
5773     forw := false
5774     back := false
5775     pair := ""
5776     switch string(lastRune){
5777     case "(": pair = ")"; forw = true
5778     case ")": pair = "("; back = true
5779     case "(": pair = ")"; forw = true
5780     case ")": pair = "("; back = true
5781     case "[": pair = "]"; forw = true
5782     case "]": pair = "["; back = true
5783     case "<": pair = ">"; forw = true
5784     case ">": pair = "<"; back = true
5785     case "\"": pair = "\""; // context depednet, can be f" or back-double quote
5786     case "'": pair = "'"; // context depednet, can be f' or back-quote
5787     // case Japanese Kakkos
5788     }
5789     if forw {
5790         iin.SearchForward(pair)
5791     }
5792     if back {
5793         iin.SearchBackward(pair)
5794     }
5795 }
5796 // 0.2.8 2020-0902 created
5797 func (iin*Input)SearchForward(pat string)(bool){
5798     right := iin.right
5799     found := -1
5800     i := 0
5801     if strBegins(right,pat) {
5802         z := utf8.DecodeRuneInString(right[i:])
5803         if 0 < z {
5804             i += z
5805         }
5806     }
5807     for i < len(right) {
5808         if strBegins(right[i:],pat) {
5809             found = i
5810             break
5811         }
5812         z := utf8.DecodeRuneInString(right[i:])
5813         if z <= 0 { break }
5814         i += z
5815     }
5816     if 0 <= found {
5817         iin.line = iin.line + right[0:found]
5818         iin.right = iin.right[found:]
5819         return true
5820     }else{
5821         return false
5822     }
5823 }
5824 // 0.2.8 2020-0902 created
5825 func (iin*Input)SearchBackward(pat string)(bool){
5826     line := iin.line
5827     found := -1
5828     i := len(line)-1
5829     for i = i; 0 <= i; i-- {
5830         z := utf8.DecodeRuneInString(line[i:])
5831         if z <= 0 {
5832             continue
5833         }
5834         //fprinf(stderr,"-- %v %v\n",pat,line[i:])
5835         if strBegins(line[i:],pat) {
5836             found = i
5837             break
5838         }
5839     }
5840     //fprinf(stderr,"--%d\n",found)

```

```

5841 if 0 <= found {
5842     iin.right = line[found:] + iin.right
5843     iin.line = line[0:found]
5844     return true
5845 }else{
5846     return false
5847 }
5848 }
5849 // 0.2.8 2020-0902 created
5850 // search from top, end, or current position
5851 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
5852     if forw {
5853         for v := range gsh.CommandHistory {
5854             if 0 <= strings.Index(v.CmdLine,pat) {
5855                 //fprintf(stderr,"n--- found !%v [%v]\n",i,pat,v.CmdLine)
5856                 return true,v.CmdLine
5857             }
5858         }
5859     }else{
5860         hlen := len(gsh.CommandHistory)
5861         for i := hlen-1; 0 <= i; i--{
5862             v := gsh.CommandHistory[i]
5863             if 0 <= strings.Index(v.CmdLine,pat) {
5864                 //fprintf(stderr,"n--- found !%v [%v]\n",i,pat,v.CmdLine)
5865                 return true,v.CmdLine
5866             }
5867         }
5868     }
5869     //fprintf(stderr,"n--- not-found(%v)\n",pat)
5870     return false,"(Not Found in History)"
5871 }
5872 // 0.2.8 2020-0902 created
5873 func (iin*IInput)GotoFORWSTR(pat string,gsh*GshContext){
5874     found := false
5875     if 0 < len(iin.right) {
5876         found = iin.SearchForward(pat)
5877     }
5878     if !found {
5879         found,line := gsh.SearchHistory(pat,true)
5880         if found {
5881             iin.line = line
5882             iin.right = ""
5883         }
5884     }
5885 }
5886 func (iin*IInput)GotoBACKSTR(pat string, gsh*GshContext){
5887     found := false
5888     if 0 < len(iin.line) {
5889         found = iin.SearchBackward(pat)
5890     }
5891     if !found {
5892         found,line := gsh.SearchHistory(pat,false)
5893         if found {
5894             iin.line = line
5895             iin.right = ""
5896         }
5897     }
5898 }
5899 func (iin*IInput)getString(prompt string)(string) // should be editable
5900 iin.clearline()
5901 fprintf(stderr,"r\v",prompt)
5902 str := ""
5903 for {
5904     ch := iin.Getc(10*1000*1000)
5905     if ch == '\n' || ch == '\r' {
5906         break
5907     }
5908     sch := string(ch)
5909     str += sch
5910     fprintf(stderr,"%s",sch)
5911 }
5912 return str
5913 }
5914
5915 // search pattern must be an array and selectable with 'N/P'
5916 var SearchPat = ""
5917 var SearchForw = true
5918
5919 func (iin*IInput)xgetline(prevline string, gsh*GshContext)(string){
5920     var ch int;
5921
5922     MODE_ShowMode = false
5923     MODE_VicMode = false
5924     iin.Redraw();
5925     first := true
5926
5927     for cix := 0; ; cix++ {
5928         iin.plinJmode = iin.inJmode
5929         iin.inJmode = false
5930
5931         ch = iin.Getc(1000*1000)
5932
5933         if ch != EV_TIMEOUT && first {
5934             first = false
5935             mode := 0
5936             if romkanmode {
5937                 mode = 1
5938             }
5939             now := time.Now()
5940             Events = append(Events,Event{now,EV_MODE,int64(mode),CmdIndex})
5941         }
5942         if ch == 033 {
5943             MODE_ShowMode = true
5944             MODE_VicMode = !MODE_VicMode
5945             iin.Redraw();
5946             continue
5947         }
5948         if MODE_VicMode {
5949             swiTch ch {
5950                 case 'o': ch = GO_TOPL
5951                 case '$': ch = GO_ENDL
5952                 case 'b': ch = GO_TOPW
5953                 case 'e': ch = GO_ENDW
5954                 case 'v': ch = GO_NEXTW
5955                 case '$': ch = GO_PAIRCH
5956
5957                 case 'j': ch = GO_DOWN
5958                 case 'k': ch = GO_UP
5959                 case 'h': ch = GO_LEFT
5960                 case 'l': ch = GO_RIGHT
5961                 case 'x': ch = DEL_RIGHT
5962                 case 'a': MODE_VicMode = !MODE_VicMode
5963                 ch = GO_RIGHT
5964                 case 'l': MODE_VicMode = !MODE_VicMode
5965                 iin.Redraw();
5966                 continue
5967                 case '-':
5968                     right,head := delHeadChar(iin.right)
5969                     if len([]byte(head)) == 1 {
5970                         ch = int(head[0])
5971                         if ('a' <= ch && ch <= 'z') {
5972                             ch = ch + 'A'-a'
5973                         }else
5974                             if ('A' <= ch && ch <= 'Z') {
5975                                 ch = ch + 'a'-A'
5976                             }
5977                         iin.right = string(ch) + right
5978                     }
5979                     iin.Redraw();
5980                     continue
5981                 case 'f': // GO_FORWCH
5982                     iin.Redraw();
5983                     ch = iin.Getc(3*1000*1000)
5984                     if ch == EV_TIMEOUT {
5985                         iin.Redraw();
5986                         continue
5987                     }
5988                     SearchPat = string(ch)
5989                     SearchForw = true
5990                     iin.GotoFORWSTR(SearchPat,gsh)
5991                     iin.Redraw();
5992                     continue
5993                 case '/':
5994                     SearchPat = iin.getString("/") // should be editable
5995                     SearchForw = true
5996                     iin.GotoFORWSTR(SearchPat,gsh)
5997                     iin.Redraw();
5998                     continue
5999                 case '?':
6000                     SearchPat = iin.getString("?") // should be editable
6001                     SearchForw = false
6002                     iin.GotoBACKSTR(SearchPat,gsh)
6003                     iin.Redraw();
6004                     continue
6005                 case 'n':
6006                     if SearchForw {
6007                         iin.GotoFORWSTR(SearchPat,gsh)
6008                     }else{
6009                         iin.GotoBACKSTR(SearchPat,gsh)
6010                     }
6011                     iin.Redraw();
6012                     continue
6013                 case 'N':
6014                     if !SearchForw {
6015                         iin.GotoFORWSTR(SearchPat,gsh)
6016                     }else{
6017                         iin.GotoBACKSTR(SearchPat,gsh)

```

```

6018         }
6019         iin.Redraw();
6020         continue
6021     }
6022 }
6023 switch ch {
6024 case GO_TOPW:
6025     iin.GotoTOPW()
6026     iin.Redraw();
6027     continue
6028 case GO_ENDW:
6029     iin.GotoENDW()
6030     iin.Redraw();
6031     continue
6032 case GO_NEXTW:
6033     // To next space then
6034     iin.GotoNEXTW()
6035     iin.Redraw();
6036     continue
6037 case GO_PAIRCH:
6038     iin.GotoPAIRCH()
6039     iin.Redraw();
6040     continue
6041 }
6042 //fprintf(stderr,"A[%02X]\n",ch);
6043 if (ch == '\\\ | | ch == 033 ){
6044     MODE_ShowMode = true
6045     metach := ch
6046     iin.waitingMeta = string(ch)
6047     iin.Redraw();
6048     // set cursor //fprintf(stderr,"???\b\b\b")
6049     ch = fgetcTimeout(stdin,2000*1000)
6050     // reset cursor
6051     iin.waitingMeta = ""
6052 }
6053 cmdch := ch
6054 if (ch == EV_TIMEOUT){
6055     if metach == 033 {
6056         continue
6057     }
6058     ch = metach
6059 }else
6060 /*
6061 if (ch == 'm' || ch == 'M' ){
6062     mch := fgetcTimeout(stdin,1000*1000)
6063     if mch == 'r' {
6064         romkanmode = true
6065     }else{
6066         romkanmode = false
6067     }
6068     continue
6069 }else
6070 /*
6071 if (ch == 'k' || ch == 'K' ){
6072     MODE_Recursive = !MODE_Recursive
6073     iin.Translate(cmdch);
6074     continue
6075 }else
6076 if (ch == 'j' || ch == 'J' ){
6077     iin.Translate(cmdch);
6078     continue
6079 }else
6080 if (ch == 'i' || ch == 'I' ){
6081     iin.Replace(cmdch);
6082     continue
6083 }else
6084 if (ch == 'l' || ch == 'L' ){
6085     MODE_LowerLock = !MODE_LowerLock
6086     MODE_CapsLock = false
6087     if MODE_ViTrace {
6088         fprintf(stderr,"%v\n",string(cmdch));
6089     }
6090     iin.Redraw();
6091     continue
6092 }else
6093 if (ch == 'u' || ch == 'U' ){
6094     MODE_CapsLock = !MODE_CapsLock
6095     MODE_LowerLock = false
6096     if MODE_ViTrace {
6097         fprintf(stderr,"%v\n",string(cmdch));
6098     }
6099     iin.Redraw();
6100     continue
6101 }else
6102 if (ch == 'v' || ch == 'V' ){
6103     MODE_ViTrace = !MODE_ViTrace
6104     if MODE_ViTrace {
6105         fprintf(stderr,"%v\n",string(cmdch));
6106     }
6107     iin.Redraw();
6108     continue
6109 }else
6110 if (ch == 'c' || ch == 'C' ){
6111     if 0 < len(iin.line) {
6112         xline,tail := delTailChar(iin.line)
6113         if len([]byte(tail)) == 1 {
6114             ch = int(tail[0])
6115             if 'a' <= ch && ch <= 'z' {
6116                 ch = ch + 'A'-'a'
6117             }else
6118                 if 'A' <= ch && ch <= 'Z' {
6119                     ch = ch + 'a'-'A'
6120                 }
6121             iin.line = xline + string(ch)
6122         }
6123     }
6124     if MODE_ViTrace {
6125         fprintf(stderr,"%v\n",string(cmdch));
6126     }
6127     iin.Redraw();
6128     continue
6129 }else{
6130     iin.pch = append(iin.pch,ch) // push
6131     ch = '\\'
6132 }
6133 }
6134 }
6135 switch( ch ){
6136 case 'P'-0x40: ch = GO_UP
6137 case 'N'-0x40: ch = GO_DOWN
6138 case 'B'-0x40: ch = GO_LEFT
6139 case 'F'-0x40: ch = GO_RIGHT
6140 }
6141 //fprintf(stderr,"B[%02X]\n",ch);
6142 switch( ch ){
6143 case 0:
6144     continue;
6145 }
6146 case '\t':
6147     iin.Replace('j');
6148     continue
6149 case 'X'-0x40:
6150     iin.Replace('j');
6151     continue
6152 }
6153 case EV_TIMEOUT:
6154     iin.Redraw();
6155     if iin.pinmode {
6156         fprintf(stderr,"\\j\n")
6157         iin.inmode = true
6158     }
6159     continue
6160 case GO_UP:
6161     if iin.lno == 1 {
6162         continue
6163     }
6164     cmd,ok := gsh.cmdStringInHistory(iin.lno-1)
6165     if ok {
6166         iin.line = cmd
6167         iin.right = ""
6168         iin.lno = iin.lno - 1
6169     }
6170     iin.Redraw();
6171     continue
6172 case GO_DOWN:
6173     cmd,ok := gsh.cmdStringInHistory(iin.lno+1)
6174     if ok {
6175         iin.line = cmd
6176         iin.right = ""
6177         iin.lno = iin.lno + 1
6178     }else{
6179         iin.line = ""
6180         iin.right = ""
6181         if iin.lno == iin.lastlno-1 {
6182             iin.lno = iin.lno + 1
6183         }
6184     }
6185     iin.Redraw();
6186     continue
6187 case GO_LEFT:
6188     if 0 < len(iin.line) {
6189         xline,tail := delTailChar(iin.line)
6190         iin.line = xline
6191         iin.right = tail + iin.right
6192     }
6193     iin.Redraw();
6194     continue;

```

```

6195 case GO_RIGHT:
6196     if( 0 < len(iin.right) && iin.right[0] != 0 ){
6197         xright.head := delHeadChar(iin.right)
6198         iin.right = xright
6199         iin.line += head
6200     }
6201     iin.Redraw();
6202     continue;
6203 case EOF:
6204     goto EXIT;
6205 case 'R'-0x40: // replace
6206     dst := convs(iin.line+iin.right);
6207     iin.line = dst
6208     iin.right = ""
6209     iin.Redraw();
6210     continue;
6211 case 'T'-0x40: // just show the result
6212     readD1C();
6213     romkanmode = !romkanmode;
6214     iin.Redraw();
6215     continue;
6216 case 'L'-0x40:
6217     iin.Redraw();
6218     continue;
6219 case 'K'-0x40:
6220     iin.right = ""
6221     iin.Redraw();
6222     continue;
6223 case 'E'-0x40:
6224     iin.line += iin.right
6225     iin.right = ""
6226     iin.Redraw();
6227     continue;
6228 case 'A'-0x40:
6229     iin.right = iin.line + iin.right
6230     iin.line = ""
6231     iin.Redraw();
6232     continue;
6233 case 'U'-0x40:
6234     iin.line = ""
6235     iin.right = ""
6236     iin.clearline();
6237     iin.Redraw();
6238     continue;
6239 case DEL_RIGHT:
6240     if( 0 < len(iin.right) ){
6241         iin.right,_ = delHeadChar(iin.right)
6242         iin.Redraw();
6243     }
6244     continue;
6245 case 0x7F: // BS? not DEL
6246     if( 0 < len(iin.line) ){
6247         iin.line,_ = delTailChar(iin.line)
6248         iin.Redraw();
6249     }
6250     /*
6251     else
6252     if( 0 < len(iin.right) ){
6253         iin.right,_ = delHeadChar(iin.right)
6254         iin.Redraw();
6255     }
6256     */
6257     continue;
6258 case 'H'-0x40:
6259     if( 0 < len(iin.line) ){
6260         iin.line,_ = delTailChar(iin.line)
6261         iin.Redraw();
6262     }
6263     continue;
6264 }
6265 if( OnWindows && ch == '\n' ){
6266     continue;
6267 }
6268 if( ch == '\n' || ch == '\r' ){
6269     iin.line += iin.right;
6270     iin.right = ""
6271     iin.Redraw();
6272     //fputc(ch,stderr);
6273     fprintf(stderr, "\r\n"); // NL on Unix, CR on Windows
6274     AtConsoleLineTop = true
6275     break;
6276 }
6277 if MODE_CapsLock {
6278     if 'a' <= ch && ch <= 'z' {
6279         ch = ch+'A'-'a'
6280     }
6281 }
6282 if MODE_LowerLock {
6283     if 'A' <= ch && ch <= 'Z' {
6284         ch = ch+'a'-'A'
6285     }
6286 }
6287 iin.line += string(ch);
6288 iin.Redraw();
6289 }
6290 EXIT:
6291 return iin.line + iin.right;
6292 }
6293 }
6294 func getline_main(){
6295     line := xgetline(0,"",nil)
6296     fprintf(stderr, "%s\n",line);
6297 /*
6298     dp = strpbk(line, "\r\n");
6299     if( dp != NULL ){
6300         *dp = 0;
6301     }
6302     if( 0 ){
6303         fprintf(stderr, "\n(%d)\n",int(strlen(line)));
6304     }
6305     if( lseek(3,0,0) == 0 ){
6306         if romkanmode {
6307             var buf [81024]byte;
6308             convs(line,buf);
6309             strcpy(line,buf);
6310         }
6311         write(3,line,strlen(line));
6312         ftruncate(3,lseek(3,0,SEEK_CUR));
6313         //fprintf(stderr, "outsize=%d\n", (int)lseek(3,0,SEEK_END));
6314         lseek(3,0,SEEK_SET);
6315         close(3);
6316     }else{
6317         fprintf(stderr, "\r\n gotline: ");
6318         trans(line);
6319         //printf("%s\n",line);
6320         printf("%s\n",line);
6321     }
6322 */
6323 }
6324 }
6325 //== end =====
6326 }
6327 //
6328 // $USERHOME/.gsh/
6329 // gsh-rc.txt, or gsh-configure.txt
6330 // gsh-history.txt
6331 // gsh-aliases.txt // should be conditional?
6332 //
6333 func (gshCtx *GshContext)gshSetupHomeDir()(bool) {
6334     homedir,found := userHomeDir()
6335     if !found {
6336         fmt.Printf("--E-- You have no UserHomeDir\n")
6337         return true
6338     }
6339     gshhome := homedir + "/" + GSH_HOME
6340     err2 := os.Stat(gshhome)
6341     if err2 != nil {
6342         err3 := os.Mkdir(gshhome,0700)
6343         if err3 != nil {
6344             fmt.Printf("--E-- Could not Create %s (%s)\n",
6345                 gshhome,err3)
6346             return true
6347         }
6348         fmt.Printf("--I-- Created %s\n",gshhome)
6349     }
6350     gshCtx.GshHomeDir = gshhome
6351     return false
6352 }
6353 func setupGshContext()(GshContext,bool){
6354     //gshPA := syscall.ProcAttr {
6355     gshPA := os.ProcAttr {
6356         "", // the staring directory
6357         os.Environ(), // environ[]
6358         //[]uintptr(os.Stdin.Fd()),os.Stdout.Fd(),os.Stderr.Fd(),
6359         []os.File{os.Stdin,os.Stdout,os.Stderr},
6360         nil, // OS specific
6361     }
6362     cwd,_ := os.Getwd()
6363     gshCtx := GshContext {
6364         cwd, // StartDir
6365         "", // GetLine
6366         []GchdirHistory { {cwd,time.Now(),0} }, // ChdirHistory
6367         gshPA,
6368         []GCommandHistory{}, //something for invokation?
6369         GCommandHistory{}, // CmdCurrent
6370         false,
6371         []os.ProcessState{}, //[]int{}

```

```

6372     aRusage{},
6373     "", // GshHomeDir
6374     Ttyid(),
6375     false,
6376     false,
6377     []PluginInfo{},
6378     []string{},
6379     "",
6380     "v",
6381     ValueStack{},
6382     GServer{"", ""}, // LastServer
6383     "", // RSERV
6384     cwd, // RWD
6385     CheckSum{},
6386 }
6387 err := gshCtx.gshSetupHomedir()
6388 return gshCtx, err
6389 }
6390 func (gsh*GshContext)gshelllh(gline string)(bool){
6391     ghist := gsh.CmdCurrent
6392     ghist.WorKDir_ = os.Getwd()
6393     ghist.WorKDirX = len(gsh.ChdirHistory)-1
6394     //fmt.Printf("--D--ChdirHistory(%#d)\n", len(gsh.ChdirHistory))
6395     ghist.StartAt = time.Now()
6396     rusagev1 := Getrusagev()
6397     gsh.CmdCurrent.FoundFile = []string{}
6398     fin := gsh.tgshelllh(gline)
6399     rusagev2 := Getrusagev()
6400     ghist.Rusagev = RusageSubv(rusagev2, rusagev1)
6401     ghist.EndAt = time.Now()
6402     ghist.CmdLine = gline
6403     ghist.FoundFile = gsh.CmdCurrent.FoundFile
6404
6405     /* record it but not show in list by default
6406     if len(gline) == 0 {
6407         continue
6408     }
6409     if gline == "hi" || gline == "history" { // don't record it
6410         continue
6411     }
6412     */
6413     gsh.CommandHistory = append(gsh.CommandHistory, ghist)
6414     return fin
6415 }
6416 // <a name="main">Main loop</a>
6417 func script(gshCtxGIVEN *GshContext) (_ GshContext) {
6418     gshCtxBuf, err0 := setupGshContext()
6419     if err0 {
6420         return gshCtxBuf;
6421     }
6422     gshCtx := *gshCtxBuf
6423     //fmt.Printf("--I-- GSH_HOME=%s\n", gshCtx.GshHomeDir)
6424     //resmap()
6425
6426     /*
6427     if false {
6428         gsh_getlinev, with_exgetline :=
6429             which("PATH", []string{"which", "gsh-getline", "-s"})
6430         if with_exgetline {
6431             gsh_getlinev[0] = toFullpath(gsh_getlinev[0])
6432             gshCtx.GetLine = toFullpath(gsh_getlinev[0])
6433         }else{
6434             fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6435         }
6436     }
6437     */
6438
6439     ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
6440     gshCtx.CommandHistory = append(gshCtx.CommandHistory, ghist0)
6441
6442     prevline := ""
6443     skipping := false
6444     for hix := len(gshCtx.CommandHistory); ; {
6445         gline := gshCtx.getline(hix, skipping, prevline)
6446         if skipping {
6447             if strings.Index(gline, "fi") == 0 {
6448                 fmt.Printf("fi\n");
6449                 skipping = false;
6450             }else{
6451                 //fmt.Printf("%s\n", gline);
6452             }
6453             continue
6454         }
6455         if strings.Index(gline, "if") == 0 {
6456             //fmt.Printf("--D-- if start: %s\n", gline);
6457             skipping = true;
6458             continue
6459         }
6460         if false {
6461             os.Stdout.Write([]byte("gotline:"))
6462             os.Stdout.Write([]byte(gline))
6463             os.Stdout.Write([]byte("\n"))
6464         }
6465         gline = strsubst(gshCtx, gline, true)
6466         if false {
6467             fmt.Printf("fmt.Printf %v - %v\n", gline)
6468             fmt.Printf("fmt.Printf %s - %s\n", gline)
6469             fmt.Printf("fmt.Printf %x - %s\n", gline)
6470             fmt.Printf("fmt.Printf %U - %s\n", gline)
6471             fmt.Printf("Stout.Write -")
6472             os.Stdout.Write([]byte(gline))
6473             fmt.Printf("\n")
6474         }
6475     }
6476     /*
6477     // should be cared in substitution ?
6478     if 0 < len(gline) && gline[0] == '!': {
6479         xgline, set, err := searchHistory(gshCtx, gline)
6480         if err {
6481             continue
6482         }
6483         if set {
6484             // set the line in command line editor
6485         }
6486         gline = xgline
6487     }
6488     */
6489     fin := gshCtx.gshelllh(gline)
6490     if fin {
6491         break;
6492     }
6493     prevline = gline;
6494     hix++;
6495 }
6496 return *gshCtx
6497 }
6498 func ftest(where, path string){
6499     //fi_err := os.Stat(path);
6500     //fmt.Printf("-- %v os.Stat(%v)=(%v)%v\n", where, path, err, fi);
6501 }
6502 func main() {
6503     openGshLog();
6504     initGshEnv();
6505     ftest("gsh-main", ".");
6506     ftest("gsh-main", "gsh.go");
6507     ftest("gsh-main", "gsh.exe");
6508     ftest("gsh-main", "gsh");
6509
6510     gshCtxBuf := GshContext{}
6511     gsh := *gshCtxBuf
6512     argv := os.Args
6513
6514     if( isin("wss", argv) ){
6515         gj_server(argv[1]);
6516         return;
6517     }
6518     if( isin("wsc", argv) ){
6519         gj_client(argv[1]);
6520         return;
6521     }
6522     if 1 < len(argv) {
6523         if isin("version", argv){
6524             gsh.showVersion(argv)
6525             return
6526         }
6527         if argv[1] == "gj" {
6528             if argv[2] == "listen" { go gj_server(argv[2]); }
6529             if argv[2] == "server" { go gj_server(argv[2]); }
6530             if argv[2] == "serve" { go gj_server(argv[2]); }
6531             if argv[2] == "client" { go gj_client(argv[2]); }
6532             if argv[2] == "join" { go gj_client(argv[2]); }
6533         }
6534         comx := isinX("-c", argv)
6535         if 0 < comx {
6536             gshCtxBuf, err := setupGshContext()
6537             gsh := *gshCtxBuf
6538             if !err {
6539                 gsh.gshellv(argv[comx+1])
6540             }
6541             return
6542         }
6543     }
6544     if 1 < len(argv) && isin("-s", argv) {
6545     }else{
6546         gsh.showVersion(append(argv, []string{"-l", "-a"}...))
6547     }
6548     script(nil)

```













```

7434 white-space:nowrap;
7435 color:#fff; background-color:rgba(0,0,64,0.7);
7436 text-align:left;
7437 vertical-align:middle;
7438 }
7439 </style>
7440
7441 <script id="gsh-script">
7442 // 2020-0909 added, permanet local storage
7443 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7444 var MyHistory = '';
7445 Permanent = localStorage;
7446 MyHistory = Permanent.getItem('MyHistory')
7447 if( MyHistory == null ){ MyHistory = '' }
7448 d = new Date()
7449 MyHistory = d.getTime()/1000+ "document.URL+"\n" + MyHistory
7450 Permanent.setItem('MyHistory',MyHistory)
7451 //Permanent.setItem('MyWindow',window)
7452
7453 var GJLog_Win = null
7454 var GJLog_Tab = null
7455 var GJLog_Stat = null
7456 var GJLog_Text = null
7457 var GJWin_Mode = null
7458 var FProductInterval = 0
7459
7460 var GJ_FactoryID = -1
7461 var GJFactory = null
7462 if( e = document.getElementById('GJFactory_0') ){
7463   GJFactory_l.height = 0
7464   GJFactory = e
7465   e.setAttribute('class','GJFactory')
7466   var GJ_FactoryID = 0
7467 }else{
7468   GJFactory = GJFactory_1
7469   var GJ_FactoryID = 1
7470 }
7471
7472 function GJFactory_Destroy(){
7473   gjf = GJFactory
7474   //gjf = document.getElementById('GJFactory')
7475   //alert('gjf'=gjf)
7476   if( gjf != null ){
7477     if( gjf.childNodes != null ){
7478       for( i = 0; i < gjf.childNodes.length; i++ ){
7479         gjf.removeChild(gjf.childNodes[i])
7480       }
7481       gjf.innerHTML = ''
7482       gjf.style.width = 0
7483       gjf.style.height = 0
7484       gjf.removeAttribute('style')
7485       GJLog_Win GJLog_Tab GJLog_Stat = GJLog_Text = GJWin_Mode = null
7486       window.clearInterval(FProductInterval)
7487       return '-- Destroy: work product destroyed'
7488     }else{
7489       return '-- Destroy: work product not exist'
7490     }
7491   }
7492 }
7493
7494 var TransMode = false
7495 var OnKeyControl = false
7496 var OnKeyShift = false
7497 var OnKeyAlt = false
7498 var OnKeyJ = false
7499 var OnKeyK = false
7500 var OnKeyL = false
7501
7502 function GJWin_OnKeyUp(ev){
7503   keycode = ev.code;
7504   if( keycode == 'ShiftLeft' ){
7505     OnKeyShift = false
7506   }else
7507   if( keycode == 'ControlLeft' ){
7508     OnKeyControl = false
7509   }else
7510   if( keycode == 'AltLeft' ){
7511     OnKeyAlt = false
7512   }else
7513   if( keycode == 'KeyJ' ){ OnKeyJ = false }else
7514   if( keycode == 'KeyK' ){ OnKeyK = false }else
7515   if( keycode == 'KeyL' ){ OnKeyL = false }else
7516   {
7517     ev.preventDefault()
7518   }
7519 }
7520 function and(a,b){ if(a){ if(b){ return true; } return false; } }
7521 function GJWin_OnKeyDown(ev){
7522   keycode = ev.code;
7523   mode = ''
7524   key = ''
7525   if( keycode == 'ControlLeft' ){
7526     OnKeyControl = true
7527     ev.preventDefault()
7528     return;
7529   }else
7530   if( keycode == 'ShiftLeft' ){
7531     OnKeyShift = true
7532     ev.preventDefault()
7533     return;
7534   }else
7535   if( keycode == 'AltLeft' ){
7536     ev.preventDefault()
7537     OnKeyAlt = true
7538     return;
7539   }else
7540   if( keycode == 'Backquote' ){
7541     TransMode = !TransMode
7542     ev.preventDefault()
7543   }else
7544   if( and(keycode == 'Space', OnKeyShift) ){
7545     TransMode = !TransMode
7546     ev.preventDefault()
7547   }else
7548   if( keycode == 'ShiftRight' ){
7549     TransMode = !TransMode
7550   }else
7551   if( keycode == 'Escape' ){
7552     TransMode = true
7553     ev.preventDefault()
7554   }else
7555   if( keycode == 'Enter' ){
7556     TransMode = false
7557     //ev.preventDefault()
7558   }
7559   if( keycode == 'KeyJ' ){ OnKeyJ = true }else
7560   if( keycode == 'KeyK' ){ OnKeyK = true }else
7561   if( keycode == 'KeyL' ){ OnKeyL = true }else
7562   {
7563     }
7564
7565   if( ev.altKey ){ key += 'Alt+' }
7566   if( onKeyControl ){ key += 'Ctrl+' }
7567   if( OnKeyShift ){ key += 'Shift+' }
7568   if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
7569   if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
7570   if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
7571   key += keycode
7572
7573   if( TransMode ){
7574     //mode = "[343\201\202r]"
7575     JaUtf8 = new Uint8Array([0343,0201,0202]);
7576     utf8dec = new TextDecoder();
7577     JaA = utf8dec.decode(JaUtf8);
7578     mode = "[" + JaA + "r]";
7579   }else{
7580     mode = "[---]"
7581   }
7582   }
7583   //[[[ /gmode.innerHTML = "[---]" + key
7584   GJWin_Mode.innerHTML = mode + ' ' + key
7585   //alert('Key: '+keycode)
7586   ev.stopPropagation()
7587   //ev.preventDefault()
7588 }
7589 function GJWin_OnScroll(ev){
7590   x = DragStartX = gsh.getBoudingClientRect().left.toFixed(0)
7591   y = DragStartY = gsh.getBoudingClientRect().top.toFixed(0)
7592   GJLog_append('OnScroll: x='+x+',y='+y)
7593 }
7594 document.addEventListener('scroll',GJWin_OnScroll)
7595 function GJWin_OnResize(ev){
7596   w = window.innerWidth
7597   h = window.innerHeight
7598   GJLog_append('OnResize: w='+w+',h='+h)
7599 }
7600 window.addEventListener('resize',GJWin_OnResize)
7601
7602 var DragStartX = 0
7603 var DragStartY = 0
7604 function GJWin_DragStart(ev){
7605   // maybe this is the grabbing position
7606   this.style.position = 'fixed'
7607   x = DragStartX = this.getBoudingClientRect().left.toFixed(0)
7608   y = DragStartY = this.getBoudingClientRect().top.toFixed(0)
7609   GJLog_Stat.value = 'DragStart: x='+x+',y='+y
7610 }

```

```

7611 function GJWin_Drag(ev){
7612   x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
7613   this.style.left = x - DragStartX
7614   this.style.top = y - DragStartY
7615   this.style.zIndex = '30000'
7616   this.style.position = 'fixed'
7617   x = this.getBoundingClientRect().left.toFixed(0)
7618   y = this.getBoundingClientRect().top.toFixed(0)
7619   GJLog_Stat.value = 'x'+x+',y'+y
7620   ev.preventDefault()
7621   ev.stopPropagation()
7622 }
7623 function GJWin_DragEnd(ev){
7624   x = ev.clientX; y = ev.clientY
7625   //x = ev.pageX; y = ev.pageY
7626   this.style.left = x - DragStartX
7627   this.style.top = y - DragStartY
7628   this.style.zIndex = '30000'
7629   this.style.position = 'fixed'
7630   if( true ){
7631     console.log("Dropped: "+this.nodeName+'#'+this.id+' x'+x+' y'+y
7632       + ' parent='+this.parentNode.id)
7633   }
7634   x = this.getBoundingClientRect().left.toFixed(0)
7635   y = this.getBoundingClientRect().top.toFixed(0)
7636   GJLog_Stat.value = 'x'+x+',y'+y
7637   ev.preventDefault()
7638   ev.stopPropagation()
7639 }
7640 function GJWin_DragIgnore(ev){
7641   ev.preventDefault()
7642   ev.stopPropagation()
7643 }
7644 // 2020-09-15 let every object have console view!
7645 var GJ_ConsoleID = 0
7646 var PrevReport = new Date()
7647 function GJLog_StatUpdate(){
7648   txa = GJLog_Stat;
7649   if( txa == null ){
7650     return;
7651   }
7652   tmlap0 = new Date();
7653   p = txa.parentNode;
7654   pw = txa.getBoundingClientRect().width;
7655   ph = txa.getBoundingClientRect().height;
7656   //txa.value += '#'+p.id+' pw'+pw+' ph'+ph+'\n';
7657   txl = '#'+p.id+' pw'+pw+' ph'+ph+'\n';
7658
7659   w = txa.getBoundingClientRect().width;
7660   h = txa.getBoundingClientRect().height;
7661   //txa.value += 'w'+w+', h'+h+'\n';
7662   txl += 'w'+w+', h'+h+'\n';
7663
7664   //txa.value += '\n';
7665   //txa.value += DateShort() + '\n';
7666   txl += '\n';
7667   txl += DateShort() + '\n';
7668   tmlap1 = new Date();
7669
7670   txa.value += txl;
7671   tmlap2 = new Date();
7672
7673   // vertical centering of the last line
7674   sHeight = txa.scrollHeight - 30; // depends on the font-size
7675   tmlap3 = new Date();
7676
7677   txa.scrollTop = sHeight; // depends on the font-size
7678   tmlap4 = new Date();
7679
7680   now = tmlap0.getTime();
7681   if( PrevReport == 0 || 10000 <= now-PrevReport ){
7682     PrevReport = now;
7683     console.log('StatBarUpdate: '
7684       + ' Leng= ' + txa.value.length + ' byte, '
7685       + ' times= ' + (tmlap4 - tmlap0) + ' ms {
7686       + ' tadd= ' + (tmlap2 - tmlap1) + ', '
7687       + ' hcal= ' + (tmlap3 - tmlap2) + ', '
7688       + ' scri= ' + (tmlap4 - tmlap3) + ' }'
7689     );
7690   }
7691 }
7692 GJWin_StatUpdate = GJLog_StatUpdate;
7693 function GJ_showTimeI(wid){
7694   //e = document.getElementById(wid);
7695   //console.log(wid.id+'.value.length'+wid.value.length)
7696   if( e != null ){
7697     //e.value = DateShort();
7698   }else{
7699     // should remove the Listener
7700   }
7701 }
7702 function GJWin_OnResizeTextarea(ev){
7703   this.value += 'resized: ' + '\n'
7704 }
7705 function GJ_NewConsole(wname){
7706   wid = wname + ' ' + GJ_ConsoleID
7707   GJ_ConsoleID += 1
7708
7709   GJFactory.style.setProperty('width',360+'px'); //GJFsize
7710   GJFactory.style.setProperty('height',320+'px')
7711   e = GJFactory;
7712   console.log('GJFa #'+'e.id' from w='+e.style.width+', h='+e.style.height)
7713
7714   if( GJFactory.innerHTML == "" ){
7715     GJFactory.innerHTML = '<'+'H3>GJ Factory_' + GJ_FactoryID + '<'+'#3><'+'hr>\n'
7716   }else{
7717     GJFactory.innerHTML += '<'+'hr>\n'
7718   }
7719
7720   gjwin = GJLog_Win = document.createElement('span')
7721   gjwin.id = wid
7722   gjwin.setAttribute('class', 'GJWin')
7723   gjwin.setAttribute('draggable', 'true')
7724   gjwin.addEventListener('dragstart', GJWin_DragStart)
7725   gjwin.addEventListener('drag', GJWin_Drag)
7726   gjwin.addEventListener('dragend', GJWin_Drag)
7727   gjwin.addEventListener('dragover', GJWin_DragIgnore)
7728   gjwin.addEventListener('dragenter', GJWin_DragIgnore)
7729   gjwin.addEventListener('dragleave', GJWin_DragIgnore)
7730   gjwin.addEventListener('dragexit', GJWin_DragIgnore)
7731   gjwin.addEventListener('drop', GJWin_DragIgnore)
7732   gjwin.addEventListener('keydown', GJWin_OnKeyDown)
7733
7734   gjtab = GJLog_Tab = document.createElement('textarea')
7735   gjtab.addEventListener('keydown', GJWin_OnKeyDown)
7736   gjtab.style.readonly = true
7737   gjtab.contentEditable = false
7738   gjtab.value = wid
7739   gjtab.id = wid + ' Tab'
7740   gjtab.setAttribute('class', 'GJTab')
7741   gjtab.setAttribute('spellcheck', 'false')
7742   gjwin.appendChild(gjtab)
7743
7744   gjstat = GJLog_Stat = document.createElement('textarea')
7745   gjstat.addEventListener('keydown', GJWin_OnKeyDown)
7746   gjstat.id = wid + ' Stat'
7747   gjstat.value = DateShort()
7748   gjstat.setAttribute('class', 'GJStat')
7749   gjstat.setAttribute('spellcheck', 'false')
7750   gjwin.appendChild(gjstat)
7751
7752   gjicon = document.createElement('span')
7753   gjicon.addEventListener('keydown', GJWin_OnKeyDown)
7754   gjicon.id = wid + ' Icon'
7755   gjicon.innerHTML = '<font color=#f44>J</font>'
7756   gjicon.setAttribute('class', 'GJIcon')
7757   gjicon.setAttribute('spellcheck', 'false')
7758   gjwin.appendChild(gjicon)
7759
7760   gjtext = GJLog_Text = document.createElement('textarea')
7761   gjtext.addEventListener('keydown', GJWin_OnKeyDown)
7762   gjtext.addEventListener('keyup', GJWin_OnKeyUp)
7763   gjtext.addEventListener('resize', GJWin_OnResizeTextarea)
7764   gjtext.id = wid + ' Text'
7765   gjtext.setAttribute('class', 'GJText')
7766   gjtext.setAttribute('spellcheck', 'false')
7767   gjwin.appendChild(gjtext)
7768
7769
7770   // user's mode as of INE
7771   gjmode = GJWin_Mode = document.createElement('textarea')
7772   gjmode.addEventListener('keydown', GJWin_OnKeyDown)
7773   gjmode.addEventListener('keyup', GJWin_OnKeyUp)
7774   gjmode.id = wid + ' Mode'
7775   gjmode.setAttribute('class', 'GJMode')
7776   gjmode.setAttribute('spellcheck', 'false')
7777   gjmode.innerHTML = '[--]'
7778   gjwin.appendChild(gjmode)
7779
7780   gjwin.zIndex = 30000
7781   GJFactory.appendChild(gjwin)
7782
7783   gjtab.scrollTop = 0
7784   gjstat.scrollTop = 0
7785
7786   //x = gjwin.getBoundingClientRect().left.toFixed(0)
7787   //y = gjwin.getBoundingClientRect().top.toFixed(0)

```

```

7788 //gJwin.style.position = 'static'
7789 //gJwin.style.left = 0
7790 //gJwin.style.top = 0
7791
7792 //update = '{'+wid+''.value=DateShort()}',
7793 update = '{GJ_showTime1('+wid+')}',
7794 // 2020-09-19 this causes memory leaks
7795 //FProductInterval = window.setInterval(update,200)
7796 //FProductInterval = window.setInterval(GJWin_StatUpdate,200)
7797 //FProductInterval = window.setInterval(GJ_showTime1,200,wid);
7798 FProductInterval = window.setInterval(GJ_showTime1,200,gJstat);
7799 return update
7800 }
7801 function xxxGJF_StripeClass(){
7802 GJLog_Win.style.removeProperty('width')
7803 GJLog_Tab.style.removeProperty('width')
7804 GJLog_Stat.style.removeProperty('width')
7805 GJLog_Text.style.removeProperty('width')
7806 return "Stripped classes"
7807 }
7808 function isElem(id){
7809 return document.getElementById(id) != null
7810 }
7811 function GJLog_append(...args){
7812 txt = GJLog_Text;
7813 if( txt == null ){
7814 return; // maybe GJLog element is removed
7815 }
7816 logs = args.join(' ');
7817 txt.value += logs + '\n'
7818 txt.scrollTop = txt.scrollHeight
7819 //GJLog_Stat.value = DateShort()
7820 }
7821 //window.addEventListener('time',GJLog_StatUpdate)
7822 function test_GJ_Console(){
7823 window.setInterval(GJLog_StatUpdate,1000);
7824 GJ_NewConsole('GJ_Console')
7825 e = GJFactory;
7826 console.log('GJF0 #'+'e.id+' from w='+e.style.width+', h='+e.style.height)
7827 e.style.width = 360; //GJFSize
7828 e.style.height = 320;
7829 console.log('GJF0 #'+'e.id+' to w='+e.style.width+', h='+e.style.height)
7830 }
7831 // test_GJ_Console();
7832
7833 var StopConsoleLog = true
7834 // 2020-09-15 added,
7835 // log should be saved to permanent memory
7836 // const px = new Proxy(console.log,{ alert() })
7837 console_log = console.log
7838 __console_info = console.info
7839 __console_warn = console.warn
7840 __console_error = console.error
7841 __console_exception = console.exception
7842 // should pop callstack info.
7843 console.exception = function(...args){
7844 console_exception(...args)
7845 alert('-- got console.exception(""+args+"")')
7846 }
7847 console.error = function(...args){
7848 __console_error(...args)
7849 alert('-- got console.error(""+args+"")')
7850 }
7851 console.warn = function(...args){
7852 __console_warn(...args)
7853 alert('-- got console.warn(""+args+"")')
7854 }
7855 console.info = function(...args){
7856 alert('-- got console.info(""+args+"")')
7857 __console_info(...args)
7858 }
7859 console.xxxlog = function(...args){ // rewrite xxxlog to log to enable it
7860 console_log(...args)
7861 if( StopConsoleLog ){
7862 return;
7863 }
7864 if( 0 <= args[0].indexOf('.') ){
7865 //alert('-- got console.log(""+args+"")')
7866 }
7867 GJLog_append(...args)
7868 }
7869
7870 //document.getElementById('GshFaviconURL').href = GShellFavicon
7871 //document.getElementById('GshFaviconURL').href = GShellInsideIcon
7872 //document.getElementById('GshFaviconURL').href = ITSmoreQR
7873 //document.getElementById('GshFaviconURL').href = GShellLogo
7874
7875 // id of GShell HTML elemets
7876 var E_BANNER = "GshBanner" // banner element in HTML
7877 var E_FOOTER = "GshFooter" // footer element in HTML
7878 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7879 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7880 var E_TODO = "gsh-todo" // TODO of GShell
7881 var E_DICT = "gsh-dict" // Dictionary of GShell
7882
7883 function bannerElem(){ return document.getElementById(E_BANNER); }
7884 function bannerStyleFunc(){ return bannerElem().style; }
7885 var bannerStyle = bannerStyleFunc()
7886 function GshSetImages(){
7887 document.getElementById('GshFaviconURL').href = GShellInsideIcon
7888 bannerStyle.backgroundImage = "url("+GShellLogo+")";
7889 //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
7890 //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
7891 //GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7892 //showFooter();
7893 }
7894 function GshInsideIconSetup(){
7895 GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7896 GMenu.style.zIndex = 1000000;
7897 //GMenu.style.left = window.innerWidth - 100
7898 GMenu.style.left = 0;
7899 GMenu.style.top = window.innerHeight - 90; // - 200
7900 window.addEventListener('resize',GshInsideIconSetup);
7901 }
7902
7903 function footerElem(){ return document.getElementById(E_FOOTER); }
7904 function footerStyle(){ return footerElem().style; }
7905 //footerElem().style.backgroundImage="url("+ITSmoreQR+")";
7906 //footerStyle().backgroundImage = "url("+ITSmoreQR+")";
7907
7908 function html_fold(e){
7909 if( e.innerHTML == "Fold" ){
7910 e.innerHTML = "Unfold"
7911 document.getElementById('gsh-menu-exit').innerHTML=""
7912 document.getElementById('GshStatement').open=false
7913 GshFeatures.open = false
7914 document.getElementById('html-src').open=false
7915 document.getElementById(E_GINDEX).open=false
7916 document.getElementById(E_GOCODE).open=false
7917 document.getElementById(E_TODO).open=false
7918 document.getElementById('References').open=false
7919 }else{
7920 e.innerHTML = "Fold"
7921 document.getElementById('GshStatement').open=true
7922 GshFeatures.open = true
7923 document.getElementById(E_GINDEX).open=true
7924 document.getElementById(E_GOCODE).open=true
7925 document.getElementById(E_TODO).open=true
7926 document.getElementById('References').open=true
7927 }
7928 }
7929 function html_pure(e){
7930 if( e.innerHTML == "Pure" ){
7931 document.getElementById('gsh').style.display=true
7932 //document.style.display = false
7933 e.innerHTML = "Unpure"
7934 }else{
7935 document.getElementById('gsh').style.display=false
7936 //document.style.display = true
7937 e.innerHTML = "Pure"
7938 }
7939 }
7940
7941 var bannerIsStopping = false
7942 //NOTE: .com/JSREF/prop_style_backgroundposition.asp
7943 function shiftBG(){
7944 bannerIsStopping = !bannerIsStopping
7945 bannerStyle.backgroundPosition = "0 0";
7946 }
7947 // status should be inherited on Window Fork(), so use the status in DOM
7948 function html_stop(e,toggle){
7949 if( toggle ){
7950 if( e.innerHTML == "Stop" ){
7951 bannerIsStopping = true
7952 e.innerHTML = "Start"
7953 }else{
7954 bannerIsStopping = false
7955 e.innerHTML = "Stop"
7956 }
7957 }else{
7958 // update JavaScript variable from DOM status
7959 if( e.innerHTML == "Stop" ){ // shown if it's running
7960 bannerIsStopping = false
7961 }else{
7962 bannerIsStopping = true
7963 }
7964 }

```

```

7965 }
7966 html_stop(document.getElementById('GshMenuStop'),false) // oninit.
7967 //html_stop(bannerElem(),false) // oninit.
7968
7969 //https://www.w3schools.com/jsref/met_win_getinterval.asp
7970 var banNshft = 0;
7971 function console(str){
7972 //console.log(str);
7973 }
7974 function shiftBanner(){
7975 var now = new Date().getTime();
7976 bpos = ((now/10)%10000).toFixed(0)+'px' + " 0px";
7977 if( !bannerIsStopping ){
7978 bannerStyle.backgroundColor = bpos;
7979 //GshBanner.style.setProperty('background-position',bpos,'important');
7980 banNshft += 1;
7981 console.log('shiftBanner <'+GshBanner.nodeName+'> '+banNshft
7982 + ' now='+now%10
7983 //+ ' stop='+bannerIsStopping
7984 + ' pos='+bpos
7985 + ' -> '+bannerStyle.backgroundColor);
7986 }
7987 }
7988 function Banner init(){
7989 console.log('-- Banner Shift init. ');
7990 window.setInterval(shiftBanner,10); // onInit.
7991 }
7992
7993 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open</a>
7994 // from embedded html to standalone page
7995 var MyChildren = 0
7996 function html_fork(){
7997 ResetPerfMon();
7998 ResetAfxView();
7999 Reset_ShadingCanvas();
8000 GJFactory_Destroy();
8001 MyChildren += 1
8002 WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
8003 newwin = window.open(""+WinId,"");
8004 src = document.getElementById('gsh');
8005 srchtml = src.outerHTML
8006 newwin.document.write("<"+srchtml>");
8007 newwin.document.write(srchtml);
8008 newwin.document.write("<"+"/html>");
8009 newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
8010 newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
8011 newwin.document.close();
8012 newwin.focus();
8013 }
8014 function html_close(){
8015 window.close()
8016 }
8017 function win_jump(win){
8018 //win = window.top;
8019 win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window/opener
8020 if( win == null ){
8021 console.log("jump to window.opener("+win+") (Error)\n")
8022 }else{
8023 console.log("jump to window.opener("+win+")\n")
8024 win.Focus();
8025 }
8026 }
8027
8028 // 0.2.9 2020-0902 created checksum of HTML
8029 CRC32UNIX = 0x04C11DB7 // Unix cksum
8030 function byteCRC32add(bigcrc,octstr,octlen){
8031 var crc = new Uint32Array(1)
8032 crc[0] = bigcrc
8033
8034 let oi = 0
8035 for( ; oi < octlen; oi++){
8036 var oct = new Uint8Array(1)
8037 oct[0] = octstr[oi]
8038 for( bi = 0; bi < 8; bi++){
8039 //console.log("--CRC32 "+crc[0]+" "+oct[0].toString(16)+" ["+oi+"."+bi+"]\n")
8040 ovf1 = crc[0] < 0 ? 1 : 0
8041 ovf2 = oct[0] < 0 ? 1 : 0
8042 ovf = ovf1 ^ ovf2
8043 oct[0] <<= 1
8044 crc[0] <<= 1
8045 if( ovf ){ crc[0] ^= CRC32UNIX }
8046 }
8047 }
8048 //console.log("--CRC32 byteAdd return crc="+crc[0]+" "+oi+"/"+"octlen+"\n")
8049 return crc[0];
8050 }
8051 function strCRC32add(bigcrc,stri,strlen){
8052 var crc = new Uint32Array(1)
8053 crc[0] = bigcrc
8054 var code = new Uint8Array(strlen);
8055 for( i = 0; i < strlen; i++){
8056 code[i] = str.charCodeAt(i) // not charAt() !!!!
8057 //console.log("== "+code[i].toString(16)+" <="+stri[i]+"")
8058 }
8059 crc[0] = byteCRC32add(crc,code,strlen)
8060 //console.log("--CRC32 strAdd return crc="+crc[0]+"")
8061 return crc[0];
8062 }
8063 function byteCRC32end(bigcrc, len){
8064 var crc = new Uint32Array(1)
8065 crc[0] = bigcrc
8066 var slen = new Uint8Array(4)
8067 let li = 0
8068 for( ; li < 4; ){
8069 slen[li] = len
8070 li += 1
8071 len >>= 8
8072 if( len == 0 ){
8073 break
8074 }
8075 }
8076 crc[0] = byteCRC32add(crc[0],slen,li)
8077 crc[0] ^= 0xFFFFFFFF
8078 return crc[0];
8079 }
8080 function strCRC32(stri,len){
8081 var crc = new Uint32Array(1)
8082 crc[0] = 0
8083 crc[0] = strCRC32add(0,stri,len)
8084 crc[0] = byteCRC32end(crc[0],len)
8085 //console.log("--CRC32 "+crc[0]+" "+len+"\n")
8086 return crc[0];
8087 }
8088
8089 DestroyGJLink = null; // to be replaced
8090 DestroyFooter = null; // to be defined
8091 DestroyEventSharingCodeview = function dummy(){}
8092 Destroy_VirtualDesktop = function(){}
8093 DestroyNavButtons = function(){}
8094
8095 function getSourceText(){
8096 if( DestroyFooter != null ) DestroyFooter();
8097 version = document.getElementById('GshVersion').innerHTML
8098 sfavico = document.getElementById('GshFavicoURL').href;
8099 sbanner = document.getElementById('GshBanner').style.backgroundColor;
8100 spositi = document.getElementById('GshBanner').style.backgroundColor;
8101
8102 if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
8103 if( DestroyGJLink != null ) DestroyGJLink();
8104 DestroyEventSharingCodeview();
8105 Destroy_VirtualDesktop();
8106 GshTopbar.innerHTML = "";
8107 DestroyIndexBar();
8108 DestroyNavButtons();
8109 ResetPerfMon();
8110 ResetAfxView();
8111 Reset_ShadingCanvas();
8112
8113 // these should be removed by CSS selector or class, after seavaed to non-printed attribute
8114 GshBanner.removeAttribute('style');
8115 document.getElementById('GshMenuSign').removeAttribute("style");
8116 styleGMenu = GMenu.removeAttribute("style");
8117 GMenu.removeAttribute("style");
8118 styleGStat = GStat.getAttribute("style")
8119 GStat.removeAttribute("style");
8120 styleGTop = GTop.getAttribute("style")
8121 GTop.removeAttribute("style");
8122 styleGshGrid = GshGrid.getAttribute("style")
8123 GshGrid.removeAttribute("style");
8124 //styleGPos = GPos.getAttribute("style");
8125 //GPos.removeAttribute("style");
8126 //GPos.innerHTML = "";
8127 //styleGLog = GLog.getAttribute("style");
8128 //GLog.removeAttribute("style");
8129 //GLog.innerHTML = "";
8130 styleGShellPlane = GShellPlane.getAttribute("style")
8131 GShellPlane.removeAttribute("style");
8132 styleRawTextViewer = RawTextViewer.getAttribute("style")
8133 RawTextViewer.removeAttribute("style")
8134 styleRawTextViewerClose = RawTextViewerClose.getAttribute("style")
8135 RawTextViewerClose.removeAttribute("style")
8136
8137 GshFavicoURL.href = "";
8138 if( !selem('ConfigIcon') ) ConfigIcon.src = "";
8139
8140 //it seems that interHTML and outerHTML generate style="" for these (??)
8141 //GshBanner.removeAttribute('style');

```

```

8142 //GshFooter.removeAttribute('style');
8143 //GshMenuSign.removeAttribute('style');
8144 GshBanner.style=""
8145 GshMenuSign.style=""
8146
8147 textarea = document.createElement("textarea")
8148 srchtml = document.getElementById("gsh").outerHTML;
8149 //textarea = document.createElement("textarea")
8150 // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
8151 // with Chromium/ after reloading from file:///
8152 textarea.innerHTML = srchtml
8153 // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
8154 var rawtext = textarea.value
8155 //textarea.destroy()
8156 //rawtext = gsh.textContent // this removes #include <FILENAME> too
8157 var orgtext = ""
8158 + "<"+html>\n" // lost preamble text
8159 + rawtext
8160 + "<"+html>\n" // lost trail text
8161 ;
8162
8163 tlen = orgtext.length
8164 //console.log("getSourceText: length="+tlen+"\n")
8165 document.getElementById("GshFaviconURL").href = sfavico;
8166
8167 document.getElementById("GshBanner").style.backgroundImage = sbanner;
8168 document.getElementById("GshBanner").style.backgroundPosition = spositi;
8169
8170 GStat.setAttribute("style",styleGStat)
8171 GMenu.setAttribute("style",styleGMenu)
8172 GTop.setAttribute("style",styleGTop)
8173 //Glog.setAttribute("style",styleGlog)
8174 //GPos.setAttribute("style",styleGPos)
8175 GshGrid.setAttribute("style",styleGshGrid)
8176 GshellPlane.setAttribute("style",styleGshellPlane)
8177 RawTextViewer.setAttribute("style",styleRawTextViewer)
8178 RawTextViewerClose.setAttribute("style",styleRawTextViewerClose)
8179 canonontext = orgtext.replace(' style=""');
8180 // open" too
8181 return canonontext
8182 }
8183
8184 function getDigest(){
8185 var text = ""
8186 text = getSourceText()
8187 var digest = ""
8188 tlen = text.length
8189 digest = strCRC32(text,tlen) + " " + tlen
8190 return { text, digest }
8191 }
8192
8193 function html_digest(){
8194 version = document.getElementById('GshVersion').innerHTML
8195 let {text, digest} = getDigest()
8196 alert("cksum: " + digest + " " + version)
8197 }
8198
8199 function charsin(str,char){
8200 ln = 0;
8201 for(i = 0; i < str.length; i++){
8202 if( str.charCodeAt(i) == char.charCodeAt(0) )
8203 ln++;
8204 }
8205 return ln;
8206 }
8207
8208 //class digestElement extends HTMLElement { }
8209 //< script>customElements.define('digest',digestElement)< /script>
8210 function showDigest(e){
8211 result = 'version=' + GshVersion.innerHTML + '\n'
8212 result += 'lines=' + e.dataset.lines + '\n'
8213 + 'length=' + e.dataset.length + '\n'
8214 + 'crc32u=' + e.dataset.crc32u + '\n'
8215 + 'time=' + e.dataset.time + '\n';
8216
8217 alert(result)
8218 }
8219
8220 function html_sign(e){
8221 if( RawTextViewer.style.zIndex == 1000 ){
8222 hideRawTextViewer()
8223 return
8224 }
8225 GshTopbar.innerHTML = "";
8226 ResetParMon();
8227 ResetAffView();
8228 Reset_ShadingCanvas();
8229 DestroyIndexBar();
8230 DestroyNavButtons();
8231 DestroyEventSharingCodeview();
8232 Destroy_WirtualDesktop();
8233 GJFactory_Destroy()
8234 if( DestroyGJLink != null ) DestroyGJLink();
8235 //gsh_digest_innerHTML = "";
8236 text = getSourceText() // the original text
8237 tlen = text.length
8238 digest = strCRC32(text,tlen)
8239 //gsh_digest_innerHTML = digest + " " + tlen
8240 //txt = getSourceText() // the text with its digest
8241 Lines = charsin(text,'\n')
8242
8243 name = "gsh"
8244 sid = name + "-digest"
8245 d = new Date()
8246 signedAt = d.getTime()
8247
8248 sign = '<'+<'<span>\n'
8249 + ' id="'+sid + "\n'
8250 + ' class="digest-\n'
8251 + ' data-target-id="'+name+"\n'
8252 + ' data-crc32u="'+ digest + "\n'
8253 + ' data-length="'+ tlen + "\n'
8254 + ' data-lines="'+ Lines + "\n'
8255 + ' data-time="'+ signedAt + "\n'
8256 + '>'<'+<'</span>\n'+<'</\n'
8257
8258 text = sign + text
8259
8260 txthtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td'
8261 + '<' + 'textarea cols=5 rows=' + Lines + ' class="LineNumber">
8262 for( i = 1; i <= Lines; i++){
8263 txthtml += i.toString() + '\n'
8264 }
8265 txthtml += ""
8266 + '<' + '/textarea'
8267 + '<' + '/td><' + 'td'
8268 + '<' + 'textarea cols=150 rows=' + Lines + ' spellcheck="false"
8269 + ' class="LineNumbered">
8270 + text + '<' + '/textarea'
8271 + '<' + '/td><' + 'tr><' + '/table>'
8272
8273 for( i = 1; i <= 30; i++){
8274 txthtml += '<br>\n'
8275 }
8276 RawTextViewer.innerHTML = txthtml
8277 RawTextViewer.spellcheck = false // (spelcheck above seems ineffective)
8278
8279 btn = e
8280 e.style.color = "rgba(128,128,255,0.9)";
8281 y = e.getBoundingClientRect().top.toFixed(0)
8282 //h = e.getBoundingClientRect().height.toFixed(0)
8283 RawTextViewer.style.top = Number(y) + 30
8284 RawTextViewer.style.left = 100;
8285 RawTextViewer.style.height = window.innerHeight - 20;
8286 //RawTextViewer.style.Opacity = 1.0;
8287 //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
8288 RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
8289 RawTextViewer.style.zindex = 1000;
8290 RawTextViewer.style.display = true;
8291
8292 if( RawTextViewerClose.style == null ){
8293 RawTextViewerClose.style = "";
8294 }
8295 RawTextViewerClose.style.top = Number(y) + 10
8296 RawTextViewerClose.style.left = 100;
8297 RawTextViewerClose.style.zindex = 1001;
8298 ScrollToElement(CurElement,RawTextViewerClose)
8299 }
8300
8301 function hideRawTextViewer(){
8302 RawTextViewer.style.left = 10000;
8303 RawTextViewer.style.zindex = -100;
8304 RawTextViewer.style.Opacity = 0.0;
8305 RawTextViewer.style = null
8306 RawTextViewer.innerHTML = "";
8307
8308 GshMenuSign.style.color = "rgba(255,128,128,1.0)";
8309 RawTextViewerClose.style.top = 0;
8310 RawTextViewerClose.style = null
8311 }
8312
8313 // source code view
8314 function frame_close(){
8315 srcframe = document.getElementById("src-frame");
8316 srcframe.innerHTML = "";
8317 //srcframe.style.cols = 1;
8318 srcframe.style.rows = 1;
8319 srcframe.style.height = 0;
8320 srcframe.style.display = false;
8321 src = document.getElementById("SrcTextarea");

```

```

8319 src.innerHTML = ""
8320 //src.cols = 0
8321 src.rows = 0
8322 src.display = false
8323 //alert("--closed--")
8324 }
8325 //<!-- | <span onclick="html_view();">Source</span> -->
8326 //<!-- | <span onclick="frame_close();">SourceClose</span> -->
8327 //<!-- | <span>Download</span> -->
8328 function frame_open(){
8329   GshTopbar.InnerHTML = "";
8330   ResetPerfMon();
8331   ResetAffView();
8332   Reset_ShadingCanvas();
8333   DestroyIndexBar();
8334   DestroyNavButtons();
8335   if( DestroyFooter != null ) DestroyFooter();
8336   document.getElementById('GshFaviconURL').href = "";
8337   oldsrc = document.getElementById("GENSRC");
8338   if( oldsrc != null ){
8339     //alert("--I--(raising old text)")
8340     oldsrc.innterHTML = "";
8341     return
8342   }else{
8343     //alert("--I--(no old text)")
8344   }
8345   styleBanner = GshBanner.getAttribute("style")
8346   GshBanner.removeAttribute("style")
8347   if( document.getElementById("GJC_1") ){ GJC_1.remove() }
8348
8349   GshFaviconURL.href = "";
8350   if( iselem('ConfigIcon') ) ConfigIcon.src = "";
8351   GStat.removeAttribute('style')
8352   GshGrid.removeAttribute('style')
8353   GshMenuSign.removeAttribute('style')
8354   //GPos.removeAttribute('style')
8355   //GPos.InnerHTML = "";
8356   //GLog.removeAttribute('style')
8357   //GLog.InnerHTML = "";
8358   GMenu.removeAttribute('style')
8359   GTop.removeAttribute('style')
8360   GShellPlane.removeAttribute('style')
8361   RawTextViewer.removeAttribute('style')
8362   RawTextViewerClose.removeAttribute('style')
8363
8364   if( DestroyGJLink != null ) DestroyGJLink();
8365   GJFactory_Destroy()
8366   Destroy_VirtualDesktop();
8367   DestroyEventSharingCodeview();
8368
8369   src = document.getElementById("gsh");
8370   srchtml = src.outerHTML
8371   srcframe = document.getElementById("src-frame");
8372   srcframe.innerHTML = "";
8373   + "<+>cite id=\"GENSRC\">\n"
8374   + "<+>style>\n"
8375   + "#GENSRC textarea{tab-size:4;}\n"
8376   + "#GENSRC textarea{-o-tab-size:4;}\n"
8377   + "#GENSRC textarea{-moz-tab-size:4;}\n"
8378   + "#GENSRC textarea{spellcheck:false;}\n"
8379   + "</+>style>\n"
8380   + "<+>textarea id=\"SrcTextarea\" cols=100 rows=20 class=\"gsh-code\" spellcheck=\"false\">"
8381   + "</+>html>\n" // lost preamble text
8382   + srchtml
8383   + "<+>/html>\n" // lost trail text
8384   + "</+>textarea>\n"
8385   + "</+>cite><!-- GENSRC -->\n";
8386
8387   //srcframe.style.cols = 80;
8388   //srcframe.style.rows = 80;
8389
8390   GshBanner.setAttribute('style',styleBanner)
8391 }
8392 function fill_CSSView(){
8393   part = document.getElementById('GshStyleDef')
8394   view = document.getElementById('gsh-style-view')
8395   view.innerHTML =
8396   + "<+>textarea cols=100 rows=20 class=\"gsh-code\">"
8397   + part.innerHTML
8398   + "<+>/textarea>"
8399 }
8400 function fill_JavaScriptView(){
8401   jpart = document.getElementById('gsh-script')
8402   view = document.getElementById('gsh-script-view')
8403   view.innerHTML =
8404   + "<+>textarea cols=100 rows=20 class=\"gsh-code\">"
8405   + jpart.innerHTML
8406   + "<+>/textarea>"
8407 }
8408 function fill_DataView(){
8409   part = document.getElementById('gsh-data')
8410   view = document.getElementById('gsh-data-view')
8411   view.innerHTML =
8412   + "<+>textarea cols=100 rows=20 class=\"gsh-code\">"
8413   + part.innerHTML
8414   + "<+>/textarea>"
8415 }
8416 function jumpto_StyleView(){
8417   jview = document.getElementById('html-src')
8418   jview.open = true
8419   jview = document.getElementById('gsh-style-frame')
8420   jview.open = true
8421   fill_CSSView()
8422 }
8423 function jumpto_JavaScriptView(){
8424   jview = document.getElementById('html-src')
8425   jview.open = true
8426   jview = document.getElementById('gsh-script-frame')
8427   jview.open = true
8428   fill_JavaScriptView()
8429 }
8430 function jumpto_DataView(){
8431   jview = document.getElementById('html-src')
8432   jview.open = true
8433   jview = document.getElementById('gsh-data-frame')
8434   jview.open = true
8435   fill_DataView()
8436 }
8437 function jumpto_WholeView(){
8438   jview = document.getElementById('html-src')
8439   jview.open = true
8440   jview = document.getElementById('gsh-whole-view')
8441   jview.open = true
8442   frame_open()
8443 }
8444 function html_view(){
8445   html_stop();
8446
8447   banner = document.getElementById('GshBanner').style.backgroundImage;
8448   footer = document.getElementById('GshFooter').style.backgroundImage;
8449   document.getElementById('GshBanner').style.backgroundImage = "";
8450   document.getElementById('GshBanner').style.backgroundPosition = "";
8451   document.getElementById('GshFooter').style.backgroundImage = "";
8452
8453   //srcwin = window.open("", "CodeView2", "");
8454   srcwin = window.open("", "t", "");
8455   srcwin.document.write("<span id=\"gsh\">\n");
8456
8457   src = document.getElementById("gsh");
8458   srcwin.document.write("<+>style>\n");
8459   srcwin.document.write("textarea{tab-size:4;}\n");
8460   srcwin.document.write("textarea{-o-tab-size:4;}\n");
8461   srcwin.document.write("textarea{-moz-tab-size:4;}\n");
8462   srcwin.document.write("</style>\n");
8463   srcwin.document.write("<h2>\n");
8464   srcwin.document.write("<+>span onclick=\"window_close();\">Close</span> | \n");
8465   //srcwin.document.write("<+>span onclick=\"html_stop();\">Run</span>\n");
8466   srcwin.document.write("<h2>\n");
8467   srcwin.document.write("<+>span id=\"gsh-src-src\" cols=100 rows=60>");
8468   srcwin.document.write("</+>html>\n");
8469   srcwin.document.write("<+>span id=\"gsh\">");
8470   srcwin.document.write("<+>span<+>/html>\n");
8471   srcwin.document.write("<+>textarea>\n");
8472
8473   document.getElementById('GshBanner').style.backgroundImage = banner;
8474   document.getElementById('GshFooter').style.backgroundImage = footer
8475
8476   sty = document.getElementById("GshStyleDef");
8477   srcwin.document.write("<+>style>\n");
8478   srcwin.document.write(sty.innerHTML);
8479   srcwin.document.write("<+>/style>\n");
8480
8481   run = document.getElementById("gsh-script");
8482   srcwin.document.write("<+>script>\n");
8483   srcwin.document.write(run.innerHTML);
8484   srcwin.document.write("<+>/script>\n");
8485
8486   srcwin.document.write("<+>/span>+>/html>\n"); // gsh span
8487   srcwin.document.close();
8488   srcwin.focus();
8489 }
8490 GSH = document.getElementById("gsh")
8491
8492 //GSH.onclick = "alert('Ouch!')"
8493 //GSH.css = "{background-color:#eef;}"
8494 //GSH.style = "background-color:#eef;"

```



```

8496 //GSH.style.display = false;
8497 //alert('Ouch0!')
8498 //GSH.style.display = true;
8499
8500 // 2020-0904 created, tentative
8501 //document.addEventListener('keydown', jgshCommand);
8502 //CurElement = GshStatement
8503 CurElement = GshMenu
8504 MemElement = GshMenu
8505
8506 function nextSib(e){
8507     n = e.nextSibling();
8508     for( i = 0; i < 100; i++ ){
8509         if( n == null ){
8510             break;
8511         }
8512         if( n.nodeName == "DETAILS" ){
8513             return n;
8514         }
8515         n = n.nextSibling();
8516     }
8517     return null;
8518 }
8519 function prevSib(e){
8520     n = e.previousSibling();
8521     for( i = 0; i < 100; i++ ){
8522         if( n == null ){
8523             break;
8524         }
8525         if( n.nodeName == "DETAILS" ){
8526             return n;
8527         }
8528         n = n.previousSibling();
8529     }
8530     return null;
8531 }
8532 function setColor(e,eName,eColor){
8533     if( e.hasChildNodes() ){
8534         s = e.childNodes();
8535         if( s != null ){
8536             for( ci = 0; ci < s.length; ci++ ){
8537                 if( s[ci].nodeName == eName ){
8538                     s[ci].style.color = eColor;
8539                     //s[ci].style.backgroundColor = eColor;
8540                     break;
8541                 }
8542             }
8543         }
8544     }
8545 }
8546
8547 // https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
8548 function showCurElementPosition(ev){
8549     // if( document.getElementById("GPos") == null ){
8550     //     return;
8551     // }
8552     // if( GPos == null ){
8553     //     return;
8554     // }
8555     e = CurElement
8556     y = e.getBoundingClientRect().top.toFixed(0)
8557     x = e.getBoundingClientRect().left.toFixed(0)
8558
8559     h = ev + " "
8560     h += "y="+y+", " + "x="+x+" -- "
8561     h += "w=" + window.innerWidth + " , h=" + window.innerHeight + " -- "
8562     //GPos.textContent = h
8563     //GPos.innerHTML = h
8564     // GPos.innerHTML = h
8565 }
8566
8567 function zero2(n){
8568     if( n < 10 ){
8569         return '0' + n;
8570     }else{
8571         return n;
8572     }
8573 }
8574 function DateHourMin(){
8575     d = new Date();
8576     //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes());
8577     return zero2(d.getHours()) + ':' + zero2(d.getMinutes());
8578 }
8579 function DateShort0(d){
8580     return d.getFullYear()
8581     + '/' + zero2(d.getMonth())
8582     + '/' + zero2(d.getDate())
8583     + ':' + zero2(d.getHours())
8584     + ':' + zero2(d.getMinutes())
8585     + ':' + zero2(d.getSeconds())
8586 }
8587 function DateShort(){
8588     return DateShort0(new Date());
8589 }
8590 function DateLong0(ms){
8591     d = new Date();
8592     d.setTime(ms);
8593     return DateShort0(d)
8594     + '.' + d.getMilliseconds()
8595     + '.' + d.getTimezoneOffset()/60
8596     + '.' + d.getTime() + '.' + d.getMilliseconds()
8597 }
8598 function DateLong(){
8599     return DateLong0(new Date());
8600 }
8601 function GShellMenu(e){
8602     //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
8603     //showGShellPlane()
8604     ConfigClick();
8605 }
8606 // placements of planes
8607 function GShellResizeX(ev){
8608     //if( document.getElementById("GMenu") != null ){
8609         //GshInsideIconSetup();
8610         //GMenu.style.left = window.innerWidth - 100
8611         //GMenu.style.top = window.innerHeight - 90 - 200
8612         //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
8613     //}
8614     //}
8615     GStat.style.width = window.innerWidth
8616     //if( document.getElementById("GPos") != null ){
8617         //GPos.style.width = window.innerWidth
8618         //GPos.style.top = window.innerHeight - 30; //GPos.style.height
8619     //}
8620     //}
8621     //if( document.getElementById("GLog") != null ){
8622         //GLog.style.width = window.innerWidth
8623         //GLog.innerHTML = ""
8624     //}
8625     //if( document.getElementById("GLog") != null ){
8626         //GLog.innerHTML = "Resize: w=" + window.innerWidth +
8627         //', h=" + window.innerHeight
8628     //}
8629     showCurElementPosition(ev)
8630 }
8631 function GShellResize(){
8632     GShellResizeX("RESIZE")
8633 }
8634 window.onresize = GShellResize
8635 var prevNode = null
8636 var LogMouseMoveOverElement = false;
8637 function GJSH_OnMouseMove(ev){
8638     if( LogMouseMoveOverElement == false ){
8639         return;
8640     }
8641     x = ev.clientX
8642     y = ev.clientY
8643     d = new Date()
8644     t = d.getTime() / 1000
8645     if( document.elementFromPoint(x,y)
8646         e = document.elementFromPoint(x,y)
8647         if( e != null ){
8648             if( e == prevNode ){
8649                 console.log('Mo-'+t+'('+x+', '+y+') '
8650                     +e.nodeType+' '+e.tagName+'#'+e.id)
8651                 prevNode = e
8652             }
8653             }else{
8654                 console.log(t+'('+x+', '+y+') no element')
8655             }
8656         }else{
8657             console.log(t+'('+x+', '+y+') no elementFromPoint')
8658         }
8659     }
8660 window.addEventListener('mousemove', GJSH_OnMouseMove);
8661
8662 function GJSH_OnMouseMoveScreen(ev){
8663     x = ev.screenX
8664     y = ev.screenY
8665     d = new Date()
8666     t = d.getTime() / 1000
8667     console.log(t+'('+x+', '+y+') no elementFromPoint')
8668 }
8669 //screen.addEventListener('mousemove', GJSH_OnMouseMoveScreen);
8670
8671 function ScrollToElement(oe,ne){
8672     ne.scrollIntoView()

```

```

8673 ny = ne.getBoundingClientRect().top.toFixed(0)
8674 nx = ne.getBoundingClientRect().left.toFixed(0)
8675 //GLog.innerHTML = "[+ny+"," +nx+]"
8676 //window.scrollTo(0,0)
8677
8678 GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
8679 GshGrid.style.left = "250px";
8680 GshGrid.style.zIndex = 0
8681 if( false ){
8682   cy = oe.getBoundingClientRect().top.toFixed(0)
8683   ox = oe.getBoundingClientRect().left.toFixed(0)
8684   y = e.getBoundingClientRect().top.toFixed(0)
8685   x = e.getBoundingClientRect().left.toFixed(0)
8686   window.scrollTo(ox,y)
8687   ny = e.getBoundingClientRect().top.toFixed(0)
8688   nx = e.getBoundingClientRect().left.toFixed(0)
8689   //GLog.innerHTML = "[+oy+"," +ox+]"->["+y+"," +x+]"->["+ny+"," +nx+]"
8690 }
8691 }
8692 function showGShellPlane(){
8693   if( GShellPlane.style.zIndex == 0 ){
8694     GShellPlane.style.zIndex = 1000;
8695     GShellPlane.style.left = 30;
8696     GShellPlane.style.height = 320;
8697     GShellPlane.innerHTML = DateLong() + "<br>" +
8698       "-- History --<br>" + MyHistory;
8699   }else{
8700     GShellPlane.style.zIndex = 0;
8701     GShellPlane.style.left = 0;
8702     GShellPlane.style.height = 50;
8703     GShellPlane.innerHTML = "";
8704   }
8705 }
8706 var SuppressGJShell = false
8707 function jgshCommand(kevent){
8708   if( SuppressGJShell ){
8709     return
8710   }
8711   key = kevent
8712   keycode = key.code
8713   //GStat.style.width = window.innerWidth
8714   GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
8715
8716   console.log("JSGsh-Key:"+keycode+"(~)"/)
8717   if( keycode == "Slash" ){
8718     console.log( "[+x+"," +y+]" )
8719     e = document.elementFromPoint(x,y)
8720     console.log( "[+x+"," +y+]" +e.nodeType+ ' '+e.tagName+'#'+e.id )
8721   }else
8722   if( keycode == "Digit0" ){ // fold side-bar
8723     // "zero page"
8724     showGShellPlane();
8725   }else
8726   if( keycode == "Digit1" ){ // fold side-bar
8727     primary.style.width = "94%"
8728     secondary.style.width = "0%"
8729     secondary.style.opacity = 0
8730     GStat.innerHTML = "[Single Column View]"
8731   }else
8732   if( keycode == "Digit2" ){ // unfold side-bar
8733     primary.style.width = "58%"
8734     secondary.style.width = "36%"
8735     secondary.style.opacity = 1
8736     GStat.innerHTML = "[Double Column View]"
8737   }else
8738   if( keycode == "KeyU" ){ // fold/unfold all
8739     html_fold(GshMenuFold);
8740     location.href = "#"+CurElement.id;
8741   }else
8742   if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8743     CurElement.open = !CurElement.open;
8744   }else
8745   if( keycode == "ArrowRight" ){ // unfold the element
8746     CurElement.open = true
8747   }else
8748   if( keycode == "ArrowLeft" ){ // unfold the element
8749     CurElement.open = false
8750   }else
8751   if( keycode == "KeyI" ){ // inspect the element
8752     e = CurElement
8753     //GLog.innerHTML =
8754     GJLog_append("Current Element: " + e + "<br>"
8755       + "name="+e.nodeName + ", "
8756       + "id="+e.id + ", "
8757       + "children="+e.childNodes.length + ", "
8758       + "parent="+e.parentNode.id + "<br>"
8759       + "text="+e.textContent )
8760     GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8761     return
8762   }else
8763   if( keycode == "KeyM" ){ // memory the position
8764     MemElement = CurElement
8765   }else
8766   if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8767     e = nextSib(CurElement)
8768     if( e != null ){
8769       setColor(CurElement, "SUMMARY", "#fff")
8770       setColor(e, "SUMMARY", "#8f8") // should be complement ?
8771       oe = CurElement
8772       CurElement = e
8773       //location.href = "#"+e.id;
8774       scrollToElement(oe,e)
8775     }
8776   }else
8777   if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8778     oe = CurElement
8779     e = prevSib(CurElement)
8780     if( e != null ){
8781       setColor(CurElement, "SUMMARY", "#fff")
8782       setColor(e, "SUMMARY", "#8f8") // should be complement ?
8783       CurElement = e
8784       //location.href = "#"+e.id;
8785       scrollToElement(oe,e)
8786     }else{
8787       e = document.getElementById("GshBanner")
8788       if( e != null ){
8789         setColor(CurElement, "SUMMARY", "#fff")
8790         CurElement = e
8791         scrollToElement(oe,e)
8792       }else{
8793         e = document.getElementById("primary")
8794         if( e != null ){
8795           setColor(CurElement, "SUMMARY", "#fff")
8796           CurElement = e
8797           scrollToElement(oe,e)
8798         }
8799       }
8800     }
8801   }else
8802   if( keycode == "KeyR" ){
8803     location.reload()
8804   }else
8805   if( keycode == "KeyJ" ){
8806     GshGrid.style.top = '120px';
8807     GshGrid.innerHTML = '<>_<{Down}';
8808   }else
8809   if( keycode == "KeyK" ){
8810     GshGrid.style.top = '0px';
8811     GshGrid.innerHTML = '(~_){Up}';
8812   }else
8813   if( keycode == "KeyH" ){
8814     GshGrid.style.left = '0px';
8815     GshGrid.innerHTML = '{_~}{Left}';
8816   }else
8817   if( keycode == "KeyL" ){
8818     //GLog.innerHTML +=
8819     GJLog_append(
8820       'screen'+screen.width+'px'+<br>'+
8821       'window'+window.innerWidth+'px'+<br>'+
8822     )
8823     GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
8824     GshGrid.innerHTML = '({_~){Right}';
8825   }else
8826   if( keycode == "KeyS" ){
8827     html_stop(GshMenuStop,true)
8828   }else
8829   if( keycode == "KeyF" ){
8830     html_fork()
8831   }else
8832   if( keycode == "KeyC" ){
8833     window.close()
8834   }else
8835   if( keycode == "KeyD" ){
8836     html_digest()
8837   }else
8838   if( keycode == "KeyV" ){
8839     e = document.getElementById('gsh-digest')
8840     if( e != null ){
8841       showDigest(e)
8842     }
8843   }
8844 }
8845 showCurElementPosition("[+key.code+] --");
8846 //if( document.getElementById("GPos") != null ){
8847 //  //GPos.innerHTML = "[+key.code+] --"
8848 //}
8849 //GShellResizeX("[+key.code+] --");

```

```

8850 }
8851 var initGSKC = false;
8852 function GShell_initKeyCommands(){
8853   if( initGSKC ){ return; } initGSKC = true;
8854
8855   GShellResizeX("[INIT]");
8856   DisplaySize = " - Display: "
8857     + "screen="+screen.width+"px, " + "window="+window.innerWidth+"px";
8858
8859   let {text, digest} = getDigest()
8860   //GLog.innerHTML +=
8861   GLog.append(
8862     "-- GShell: " + GshVersion.innerHTML + "\n" +
8863     "-- Digest: " + digest + "\n" +
8864     DisplaySize
8865     //+ "<br>" + "-- LastVisit:<br>" + MyHistory
8866   )
8867   GShellResizeX(null);
8868 }
8869 //GShell_initKeyCommands();
8870
8871 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8872 //Convert a string into an ArrayBuffer
8873 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8874 function str2ab(str){
8875   const buf = new ArrayBuffer(str.length);
8876   const bufView = new Uint8Array(buf);
8877   for (let i = 0, strLen = str.length; i < strLen; i++) {
8878     bufView[i] = str.charCodeAt(i);
8879   }
8880   return buf;
8881 }
8882 function importPrivateKey(pem) {
8883   const binaryDerString = window.atob(pemContents);
8884   const binaryDer = str2ab(binaryDerString);
8885   return window.crypto.subtle.importKey(
8886     "pkcs8",
8887     binaryDer,
8888     {
8889       name: "RSA-PSS",
8890       modulusLength: 2048,
8891       publicExponent: new Uint8Array([1, 0, 1]),
8892       hash: "SHA-256",
8893     },
8894     true,
8895     ["sign"]
8896   );
8897 }
8898 //importPrivateKey(ppem)
8899
8900 //key = {}
8901 //buf = "abc"
8902 //enc = "xyzxxxxx"; //crypto.publicEncrypt(key,buf)
8903 //b64 = btoa(enc)
8904 //dec = atob(b64)
8905 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8906 </script>
8907 */
8908
8909 //<!-- Work { ----- -->
8910 </span id="PackmonGo_WorkCodeSpan">
8911 /*
8912 <details id="PackmonGo_Section"><summary id="PackmonGo_Summary">PackmonGo</summary>
8913 </----- PackmonGo // 2020-1025 SatOct17S { -->
8914 <h2>PackmonGo</h2>
8915 <div id="PackmonGo_1" class="PackmonGo">
8916 <canvas id="PackmonGo_1_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8917 </div>
8918 <div id="PackmonGo_2" class="PackmonGo">
8919 <canvas id="PackmonGo_2_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8920 </div>
8921 <div id="PackmonGo_3" class="PackmonGo">
8922 <canvas id="PackmonGo_3_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8923 </div>
8924 <div id="PackmonGo_4" class="PackmonGo">
8925 <canvas id="PackmonGo_4_Canvas" class="PackmonGo_Canvas" width="400px" height="400px" draggable="true"></canvas>
8926 </div>
8927 <style>
8928 .PackmonGo {
8929   position:fixed !important;
8930   background-color:rgba(0,0,0,0.0);
8931 }
8932 .PackmonGo_Canvas {
8933   background-color:rgba(0,0,0,0.0);
8934 }
8935 </style>
8936 <script>
8937 var stopPackmonFlag = false;
8938 var PackmonInit = false;
8939 function stopPackmon(){
8940   stopPackmonFlag = !stopPackmonFlag;
8941 }
8942 function spawnPackmonGo(){
8943   if( PackmonInit == false ){
8944     pgw = 100;
8945     pgh = 100;
8946     pgo = 0.5;
8947
8948     ctx = PackmonGo_1_Canvas.getContext('2d');
8949     ctx.fillStyle = 'rgba(255,255,0,"pgo")';
8950     ctx.fillRect(0,0,pgw,pgh);
8951     base = PackmonGo_1;
8952     gsh.appendChild(base);
8953     base.style.zindex = 1000;
8954     base.style.position = "fixed";
8955     base.style.left = 0 + 'px';
8956     base.style.top = 0 + 'px';
8957     base.xinc = 4;
8958     base.yinc = 4;
8959     base.scrx = 0;
8960     base.scry = 0;
8961     base.pgw = 100;
8962     base.pgh = 100;
8963     movePWindow(PackmonGo_1);
8964
8965     ctx = PackmonGo_2_Canvas.getContext('2d');
8966     ctx.fillStyle = 'rgba(127,255,127,"pgo")';
8967     ctx.fillRect(0,0,pgw,pgh);
8968     base = PackmonGo_2;
8969     gsh.appendChild(base);
8970     base.style.zindex = 1001;
8971     base.style.position = "fixed";
8972     base.style.left = 200 + 'px';
8973     base.style.top = 0 + 'px';
8974     base.xinc = 4;
8975     base.yinc = 4;
8976     base.scrx = 0;
8977     base.scry = 0;
8978     base.pgw = 100;
8979     base.pgh = 100;
8980     movePWindow(PackmonGo_2);
8981
8982     ctx = PackmonGo_3_Canvas.getContext('2d');
8983     pgo = 0.3;
8984     ctx.fillStyle = 'rgba(64,64,255,"pgo")';
8985     ctx.fillRect(0,0,pgw,pgh);
8986     base = PackmonGo_3;
8987     gsh.appendChild(base);
8988     base.style.zindex = 1002;
8989     base.style.position = "fixed";
8990     base.style.left = 200 + 'px';
8991     base.style.top = 0 + 'px';
8992     base.xinc = 4;
8993     base.yinc = 4;
8994     base.scrx = 0;
8995     base.scry = 0;
8996     base.soff = 0;
8997     base.pgw = 100;
8998     base.pgh = 100;
8999     movePWindow(PackmonGo_3);
9000
9001     ctx = PackmonGo_4_Canvas.getContext('2d');
9002     pgo = pgh = 400;
9003     ctx.beginPath();
9004     ctx.strokeStyle = 'rgba(0,0,0,0)';
9005     ctx.arc(200,200,200,0,Math.PI*2,true);
9006     ctx.stroke();
9007     pgo = 0.4;
9008     ctx.fillStyle = 'rgba(255,31,32,"pgo")';
9009     ctx.fill();
9010     base = PackmonGo_4;
9011     gsh.appendChild(base);
9012     base.style.zindex = 1002;
9013     base.style.position = "fixed";
9014     base.style.left = 200 + 'px';
9015     base.style.top = 0 + 'px';
9016     base.xinc = 4;
9017     base.yinc = 4;
9018     base.scrx = 0;
9019     base.scry = 0;
9020     base.soff = 5000;
9021     base.pgw = pgw;
9022     base.pgh = pgh;
9023     movePWindow(PackmonGo_4);
9024   }
9025   function movePWindow(base){
9026     if( stopPackmonFlag ){

```

```

9027     return;
9028 }
9029 x = parseInt(base.style.left);
9030 y = parseInt(base.style.top);
9031 w = window.innerWidth;
9032 h = window.innerHeight;
9033 if( x < 0 || w-base.pgw < x ){
9034     base.xinc = -base.xinc;
9035 }
9036 x += base.xinc;
9037 if( y < 0 || h-base.pgh < y ){
9038     //yinc = -yinc;
9039     base.yinc = -base.yinc;
9040 }
9041 y += base.yinc;
9042 base.style.left = x + 'px';
9043 base.style.top = y + 'px';
9044 //console.log('PG x='+x+',y='+y);
9045 //window.setTimeout(movePG,10);
9046 }
9047 function movePGscreen(base){
9048     if( stopPackmonFlag ){
9049         return;
9050     }
9051     sw = screen.width;
9052     sh = screen.height;
9053     d = new Date();
9054     s = d.getTime(); // milli-seconds
9055     sx = ((s+base.soff) % 10000) * (screen.width / 10000);
9056     sy = ((s+base.soff) % 5000) * (screen.height / 5000);
9057     x = sx - window.screenX;
9058     y = sy - window.screenY;
9059     base.style.left = x + 'px';
9060     base.style.top = y + 'px';
9061 }
9062 function movePGscreenCircle(base){
9063     if( stopPackmonFlag ){
9064         return;
9065     }
9066     sw = screen.width;
9067     sh = screen.height;
9068     pgw = base.pgw;
9069     pgh = base.pgh;
9070     d = new Date();
9071     s = d.getTime(); // milli-seconds
9072     ds = s + base.soff; // Delay
9073     vsw = pgw + sw + pgw;
9074     vsh = pgh + sh + pgh;
9075     sx = -pgw + vsw * ((ds % 10000)/10000);
9076     sy = -pgh + vsh * ((ds % 10000)/10000);
9077     x = sx - window.screenX;
9078     y = sy - window.screenY;
9079     base.style.left = x + 'px';
9080     base.style.top = y + 'px';
9081 }
9082 if( PackmonInit == false ){
9083     //window.setTimeout(movePG,mm300);
9084     window.setInterval(movePGwindow,10,PackmonGo_1);
9085     window.setInterval(movePGwindow,10,PackmonGo_2);
9086     window.setInterval(movePGscreen,10,PackmonGo_3);
9087     window.setInterval(movePGscreen,10,PackmonGo_4);
9088     window.setInterval(movePGscreenCircle,10,PackmonGo_4);
9089     window.addEventListener('click',stopPackmon);
9090     PackmonInit = true;
9091     stopPackmonFlag = false;
9092 }
9093 }
9094 function PackmonGo_Setup(e){
9095     stopPackmon();
9096     spawnPackmonGo();
9097     if( e != null ){
9098         e.stopPropagation();
9099     }
9100 }
9101 PackmonGo_Summary.addEventListener('click',PackmonGo_Setup);
9102 if( PackmonGo_Section.open == true ){
9103     PackmonGo_Setup();
9104 }
9105 </script>
9106
9107 <input id="PackmonGo_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
9108 <input id="PackmonGo_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
9109 <input id="PackmonGo_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
9110 <span id="PackmonGo_WorkCodeView"></span>
9111 <script id="PackmonGo_WorkScript">
9112 function PackmonGo_openWorkCodeView(){
9113     function PackmonGo_showWorkCode(){
9114         showHtmlCode(PackmonGo_WorkCodeView,PackmonGo_WorkCodeSpan);
9115     }
9116     PackmonGo_WorkCodeViewOpen.addEventListener('click',PackmonGo_showWorkCode);
9117 }
9118 PackmonGo_openWorkCodeView(); // should be invoked by an event
9119 </script>
9120 </details>
9121 <!-- Template_WorkCodeSpan -->
9122 <!-- //span -->
9123 <!-- ===== Work } ===== -->
9124
9125
9126
9127 <!-- ===== Work { ===== -->
9128 <span id="SightGlass_WorkCodeSpan">
9129 /
9130 <details id="SightGlass"><summary>SightGlass</summary>
9131 <!-- ----- SightGlass // 2020-1023 SatoxITS -->
9132 <h2>SightGlass</h2>
9133 <details><summary>enter</summary>
9134 <div id="SightGlass_l_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
9135 <div id="SightGlass_l_BGScroffset" class="SightGlass_TextData">(0000, 0000)</div>
9136 <div id="SightGlass_l_BPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
9137 <div id="SightGlass_l_BGPosition" class="SightGlass_TextData">(0000, 0000)(0000 x 0000) </div>
9138 <div id="SightGlass_l_Wheel" class="SightGlass_TextData">Wheel</div>
9139 </details>
9140 <div id="SightGlass_l_Window" class="SightGlass_Window" draggable="true"></div>
9141 <style>
9142 .SightGlass_TextData {
9143     color:#000;
9144     font-size:10pt;
9145 }
9146 .SightGlass_Window {
9147     xzcom:2;
9148     resize:both;
9149     width:600px;
9150     height:320px;
9151     background-color:rgba(200,200,200,0.5);
9152     background-size:200%;
9153     xbackground-size:1280px 720px;
9154     background-image:url(WD-WallPaper03.png);
9155 }
9156 xbody {
9157     scroll-behavior:smooth;
9158 }
9159 </style>
9160 <script>
9161 //
9162 // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
9163 var SGScrInitX = 0;
9164 var SGScrInitY = 0;
9165 var SG_initX = 0;
9166 var SG_initY = 0;
9167 function SG_adjust(){
9168     xy = window.screenX + ' ' + window.screenY;
9169     wh = window.innerWidth + ' x ' + window.innerHeight;
9170     xywh = 'xy(' + xy + ') wh(' + wh + ')';
9171     if( SightGlass.open) SightGlass_l_BPosition.innerHTML = 'Browser: ' + xywh;
9172 }
9173
9174 sg = SightGlass_l_Window;
9175 sgr = sg.getBoundingClientRect();
9176 xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
9177 wh = sgr.width + ' x ' + sgr.height;
9178 xywh = 'xy(' + xy + ') wh(' + wh + ')';
9179 if( SightGlass.open) SightGlass_l_GPosition.innerHTML = 'SiGlass: ' + xywh;
9180 //SightGlass_l_Window.style.backgroundColor = sgr.left+'px ' + sgr.top+'px';
9181 if( SG_initX == 0 ){
9182     SGScrInitX = window.screenX;
9183     SGScrInitY = window.screenY;
9184     //SightGlass_l_Window.style.backgroundColor = '200%';
9185     //SightGlass_l_Window.style.backgroundImage = 'url("WD-WallPaper03.png")';
9186     SG_initX = sgr.left;
9187     SG_initY = sgr.top;
9188 }
9189 dx = SG_initX - sgr.left;
9190 dy = SG_initY - sgr.top;
9191
9192 dsx = SGScrInitX - window.screenX;
9193 dsy = SGScrInitY - window.screenY;
9194 scroff = 'Screen: '+sdx+'dsx '+sy+'dsy';
9195 if( SightGlass.open) SightGlass_l_BGScroffset.innerHTML = scroff;
9196 dx += dsx;
9197 dy += dsy;
9198
9199 bgpos = dx+'px ' + dy+'px';
9200 SightGlass_l_Window.style.backgroundColor = bgpos;
9201 if( SightGlass.open) SightGlass_l_BGPosition.innerHTML = 'BGround:'
9202     + SightGlass_l_Window.style.backgroundColor;
9203 }

```

```

9204 var wheels = 0;
9205 function SG_wheel(e){
9206   wheels += 1;
9207   SightClass_1.Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
9208   if( e.target.id == 'SightClass_1_Window' ){
9209     e.preventDefault();
9210   }
9211 }
9212 function SG_adjust(){
9213   SG_adjust();
9214   //sampling = 0;
9215   sampling = 20;
9216   //sampling = 200;
9217   window.setTimeout(SG_adjust, sampling);
9218 }
9219 function SightClass_Setup(){
9220   SG_adjust();
9221   window.addEventListener('resize', SG_adjust);
9222   window.addEventListener('mousemove', SG_adjust);
9223   window.addEventListener('scroll', SG_adjust);
9224   window.addEventListener('wheel', SG_wheel, {passive:false});
9225   //window.moveTo(0,0);
9226
9227   // if focused
9228   //window.setInterval(SG_adjust, 1);
9229   window.setTimeout(SG_adjust, 1);
9230 }
9231 // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
9232 function Electron_Setup(){
9233   const { BrowserWindow } = require('electron');
9234   const currentWindow = BrowserWindow.getFocusedWindow();
9235   //currentWindow.on('move', function(){ SG_adjust(); });
9236   currentWindow.addEventListener('move', SG_adjust);
9237 }
9238 </script>
9239
9240 <input id="SightClass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
9241 <input id="SightClass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
9242 <input id="SightClass_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
9243 <span id="SightClass_WorkCodeView"></span>
9244 <script id="SightClass_WorkCodeView">
9245 function SightClass_openWorkCodeView(){
9246   function SightClass_showWorkCode(){
9247     showHtmlCode(SightClass_WorkCodeView, SightClass_WorkCodeSpan);
9248   }
9249   SightClass_WorkCodeViewOpen.addEventListener('click', SightClass_showWorkCode);
9250 }
9251 SightClass_openWorkCodeView(); // should be invoked by an event
9252 </script>
9253 </details>
9254 <!-- SightClass_WorkCodeSpan -->
9255 *//</span>
9256 <!-- ===== Work ] ===== -->
9257
9258
9259 /*
9260 <!-- ----- GJConsole BEGIN { ----- -->
9261 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
9262 <details><summary>GJ Console</summary>
9263 <p>
9264 <span id="GJE_RootNode0"></span>
9265 <span id="GJC_Container"></span>
9266 </p>
9267 <style id="GJConsoleStyle">
9268 .GJConsole {
9269   z-index:1000;
9270   width:400px; height:200px;
9271   margin:2px;
9272   color:#fff; background-color:#66a;
9273   font-size:12px; font-family:monospace,Courier New;
9274 }
9275 </style>
9276
9277 <script id="GJConsoleScript" class="GJConsole">
9278 var PS1 = "% "
9279 function GJC_KeyDown(keyevent){
9280   key = keyevent.code
9281   if( key == "Enter" ){
9282     GJC_Command(this)
9283     this.value += "\n" + PS1 // prompt
9284   }else
9285   if( key == "Escape"){
9286     SuppressGJShell = false
9287     GshMenu.focus() // should be previous focus
9288   }
9289 }
9290
9291 var GJC_SessionId
9292 function GJC_SetSessionId(){
9293   var xd = new Date()
9294   GJC_SessionId = xd.getTime() / 1000
9295 }
9296 GJC_SetSessionId()
9297 function GJC_Memory(mem,args,text){
9298   argv = args.split(" ")
9299   cmd = argv[0]
9300   argv.shift()
9301   args = argv.join(' ')
9302   ret = ""
9303
9304   if( cmd == 'clear' ){
9305     Permanent.setItem(mem, '')
9306   }else
9307   if( cmd == 'read' ){
9308     ret = Permanent.getItem(mem)
9309   }else
9310   if( cmd == 'save' ){
9311     val = Permanent.getItem(mem)
9312     if( val == null ){ val = "" }
9313     d = new Date()
9314     val += d.getTime()/1000+ " +GJC_SessionId+ " +document.URL+ "+args+"\n"
9315     val += text.value
9316     Permanent.setItem(mem, val)
9317   }else
9318   if( cmd == 'write' ){
9319     val = Permanent.getItem(mem)
9320     if( val == null ){ val = "" }
9321     d = new Date()
9322     val += d.getTime()/1000+ " +GJC_SessionId+ " +document.URL+ "+args+"\n"
9323     Permanent.setItem(mem, val)
9324   }else{
9325     ret = "Commands: write | read | save | clear"
9326   }
9327   return ret
9328 }
9329 // -- 2020-09-14 added TableEditor
9330 var GJE_CurElement = null; //GJE_RootNode
9331 GJE_NodeSaved = null
9332 GJE_TableNo = 1
9333 function GJE_StyleKeyCommand(kev){
9334   keycode = Kev.code
9335   console.log('GJE-Key: '+keycode)
9336   if( keycode == 'Escape' ){
9337     GJE_SetStyle(this);
9338   }
9339   kev.stopPropagation()
9340   // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
9341 }
9342 var GJE_CommandMode = false
9343 function GJE_TableKeyCommand(kev, tab){
9344   wasCmdMode = GJE_CommandMode
9345   key = kev.code
9346   if( key == 'Escape' ){
9347     console.log("To command mode: "+tab.nodeName+"#"+tab.id)
9348     //tab.setAttribute('contenteditable', 'false')
9349     tab.style.caretColor = 'blue'
9350     GJE_CommandMode = true
9351   }else
9352   if( key == "KeyA" ){
9353     tab.style.caretColor = "red"
9354     GJE_CommandMode = false
9355   }else
9356   if( key == "KeyI" ){
9357     tab.style.caretColor = "red"
9358     GJE_CommandMode = false
9359   }else
9360   if( key == "KeyO" ){
9361     tab.style.caretColor = "red"
9362     GJE_CommandMode = false
9363   }else
9364   if( key == "KeyJ" ){
9365     console.log("ROW-DOWN")
9366   }else
9367   if( key == "KeyK" ){
9368     console.log("ROW-UP")
9369   }else
9370   if( key == "KeyW" ){
9371     console.log("COL-FORW")
9372   }else
9373   if( key == "KeyB" ){
9374     console.log("COL-BACK")
9375   }
9376
9377   kev.stopPropagation()
9378   if( wasCmdMode ){
9379     kev.preventDefault()
9380   }

```

```

9381 }
9382 function GJE_DragEvent(ev,elem){
9383     x = ev.clientX
9384     y = ev.clientY
9385     console.log("Dragged: "+this.nodeName+'#'+this.id+' x='+x+' y="+y)
9386 }
9387 // https://developer.mozilla.org/en-US/docs/Web/API/DragEvent
9388 // https://www.w3.org/TR/uievents/#events-mouseevents
9389 function GJE_DropEvent(ev,elem){
9390     x = ev.clientX
9391     y = ev.clientY
9392     this.style.x = x
9393     this.style.y = y
9394     this.style.position = 'absolute' // 'fixed'
9395     this.parentNode = gsh // just for test
9396     console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y="+y
9397     + " parent="+this.parentNode.id)
9398 }
9399 function GJE_SetTableStyle(ev){
9400     this.innerHTML = this.value; // sync. for external representation?
9401     if(false){
9402         stid = this.parentNode.id+this.id
9403         // and remove "span" at the end
9404         e = document.getElementById(stid)
9405         //alert('SetTableStyle #' + stid + '\n'+this.value)
9406         if ( e != null ){
9407             e.innerHTML = this.value
9408         }else{
9409             console.log('Style Not found: '+stid)
9410         }
9411     }
9412     //alert('event StopPropagation: '+ev)
9413 }
9414 function setCSSofClass(cclass,cstyle){
9415     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
9416     rlen = ss.cssRules.length;
9417     let tabrule = null;
9418     rulex = -1
9419
9420     // should skip white space at the top of cstyle
9421     sel = cstyle.charAt(0);
9422     selector = sel+cclass;
9423     console.log('-- search style rule for '+selector)
9424
9425     for(let i = 0; i < rlen; i++){
9426         cr = ss.cssRules[i];
9427         console.log('CSS rule [' + i + '/' + rlen + ']' + cr.selectorText);
9428         if (cr.selectorText == selector ){ // css class selector
9429             tabrule = ss.cssRules[i];
9430             console.log('CSS rule found for: [' + i + '/' + rlen + ']' + selector);
9431             ss.deleteRule(i);
9432             //rlen = ss.cssRules.length;
9433             rulex = i
9434             // should search and replace the property here
9435         }
9436     }
9437     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
9438     if( tabrule == null ){
9439         console.log('CSS rule NOT found for: [' + rlen + ']' + selector);
9440         ss.insertRule(cstyle,rlen);
9441         ss.insertRule(cstyle,0); // override by 0?
9442         console.log('CSS rule inserted: [' + (rlen+1) + ']' + cstyle);
9443     }else{
9444         ss.insertRule(cstyle,rlen);
9445         ss.insertRule(cstyle,0);
9446         console.log('CSS rule replaced: [' + (rlen+1) + ']' + cstyle);
9447     }
9448 }
9449 function GJE_SetStyle(te){
9450     console.log('Apply the style to: '+te.id+'\n');
9451     console.log('Apply the style to: '+te.parentNode.id+'\n');
9452     console.log('Apply the style to: '+te.parentNode.class+'\n');
9453     cclass = te.parentNode.class;
9454     setCSSofClass(cclass,te.value); // should get selector part from
9455     // setCSSofClass { rules }
9456
9457     if(false){
9458         //console.log('Apply the style:')
9459         //stid = this.parentNode.id+this.id+"
9460         //stid = this.id+".style"
9461         css = te.value
9462         stid = te.parentNode.id+".style"
9463         e = document.getElementById(stid)
9464         if( e != null ){
9465             //console.log('Apply the style: '+e.id+'\n'+te.value);
9466             console.log('Apply the style: '+e.id+'\n'+css);
9467             e.innerHTML = css; //te.value;
9468             //ncss = e.sheet;
9469             //ncss.insertRule(te.value,ncss.cssRules.length);
9470         }else{
9471             console.log('No element to Apply the style: '+stid)
9472         }
9473         tblid = te.parentNode.id+".table";
9474         e = document.getElementById(tblid);
9475         if( e != null ){
9476             //e.setAttribute('style',css);
9477             e.setProperty('style',css,'important');
9478         }
9479     }
9480 }
9481 function makeTable(argv){
9482     //tid = "
9483     //cwe = GJE_CurElement
9484     cwe = GJCI_Container;
9485     //cwd = GJFactory;
9486     tid = 'table' + GJE_TableNo
9487
9488     nt = new Text('\n')
9489     cwe.appendChild(nt)
9490
9491     ne = document.createElement('span'); // the container
9492     cwe.appendChild(ne)
9493     ne.id = tid + "-span"
9494     ne.setAttribute('contenteditable',true)
9495
9496     htspan = document.createElement('span'); // html part
9497     //htspan.id = tid + "-html"
9498     //ne.innerHTML = '\n'
9499     nt = new Text('\n')
9500     ne.appendChild(nt)
9501     ne.appendChild(htspan)
9502
9503     htspan.id = tid
9504     htspan.setAttribute('class',tid)
9505
9506     ne.setAttribute('draggable', 'true')
9507     ne.addEventListener('drag',GJE_DragEvent);
9508     ne.addEventListener('dragend',GJE_DropEvent);
9509
9510     var col = 3
9511     var row = 2
9512     if( argv[0] != null ){
9513         col = argv[0]
9514         argv.shift()
9515     }
9516     if( argv[0] != null ){
9517         row = argv[0]
9518         argv.shift()
9519     }
9520
9521     //ne.setAttribute('class',tid)
9522     ht = "\n"
9523     //ht += '<'+table' + 'id="'+tid+'"' + ' class="'+tid+'"'
9524     ht += '<'+table'
9525         + " onkeydown='GJE_TableKeyCommand(event,this)'"
9526         + " ondrag='GJE_DragEvent(event,this)'"
9527         + " ondragend='GJE_DropEvent(event,this)'"
9528         + " draggable='true'"
9529         + " contenteditable='true'"
9530         + ">\n"
9531     ht += '<'+tbody>\n';
9532     for( r = 0; r < row; r++ ){
9533         ht += '<'+tr>\n'
9534         for( c = 0; c < col; c++ ){
9535             ht += '<'+td>'
9536             ht += "ABCDEFGHIJKLMNOPQRSTUVWXYZ".charAt(c) + r
9537             ht += '<'+td>\n'
9538         }
9539         ht += '<'+tr>\n'
9540     }
9541     ht += '<'+tbody>\n';
9542     ht += '<'+table>\n';
9543     htspan.innerHTML = ht;
9544     nt = new Text('\n')
9545     ne.appendChild(nt)
9546
9547     st = '#'+tid+' *\n' // # for instanse specific
9548         + '*border:1px solid #aaa;\n'
9549         + '*background-color:#efe;\n'
9550         + '*color:#222;\n'
9551         + '*font-size:#4pt !important;\n'
9552         + '*font-family:monospace,Courier New !important;\n'
9553         + '/* hit ESC to apply '+tid+'\n'
9554
9555     // wish script to be included
9556     //nj = document.createElement('script')
9557     //ne.appendChild(nj)

```

```

9558 //ne.innerHTML = 'function SetStyle(e){}'
9559
9560 // selector seems lost in dynamic style appending
9561 if(false){
9562   ns = document.createElement('style')
9563   ne.appendChild(ns)
9564   ns.id = tid + ".style"
9565   ns.innerHTML = '\n'+st
9566   nt = new Text('\n')
9567   ne.appendChild(nt)
9568 }
9569 setCSSofClass(tid,st); // should be in JavaScript script?
9570
9571 nx = document.createElement('textarea')
9572 ne.appendChild(nx)
9573 nx.id = tid + "-style_def"
9574 nx.setAttribute('class','GJ_StyleEditor')
9575 nx.spellcheck = false
9576 nx.cols = 60
9577 nx.rows = 10
9578 nx.innerHTML = '\n'+st
9579 nx.addEventListener('change',GJE_SetTableStyle);
9580 nx.addEventListener('keydown',GJE_StyleKeyCommand);
9581 //nx.addEventListener('click',GJE_SetTableStyle);
9582
9583 nt = new Text('\n')
9584 cwe.appendChild(nt)
9585
9586 GJE_TableNo += 1
9587 return 'created TABLE id="'+tid+'"'
9588 }
9589 function GJE_NodeEdit(argv){
9590   cwe = GJE_CurElement
9591   cmd = argv[0]
9592   argv.shift()
9593   args = argv.join(' ')
9594   ret = ""
9595
9596   if( cmd == 'u' || cmd == 'un' || cmd == 'undo' ){
9597     if( GJE_NodesSaved != null ){
9598       xn = GJE_RootNode
9599       GJE_RootNode = GJE_NodesSaved
9600       GJE_NodesSaved = xn
9601       ret = '-- did undo'
9602     }else{
9603       ret = '-- could not undo'
9604     }
9605     return ret
9606   }
9607   GJE_NodesSaved = GJE_RootNode.cloneNode()
9608   if( cmd == 'c' || cmd == 'cd' || cmd == 'cd' ){
9609     if( argv[0] == null ){
9610       ne = GJE_RootNode
9611     }else
9612     if( argv[0] == '..' ){
9613       ne = cwe.parentNode
9614     }else{
9615       ne = document.getElementById(argv[0])
9616     }
9617     if( ne != null ){
9618       GJE_CurElement = ne
9619       ret = "-- current node: " + ne.id
9620     }else{
9621       ret = "-- not found: " + argv[0]
9622     }
9623   }else
9624   if( cmd == 'mkt' || cmd == 'mktable' ){
9625     makeTable(argv)
9626   }else
9627   if( cmd == 'm' || cmd == 'mk' || cmd == 'mk' ){
9628     ne = document.createElement(argv[0])
9629     //ne.id = argv[0]
9630     ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
9631     cwe.appendChild(ne)
9632     if( cmd == 'm' || cmd == 'mk' ){
9633       GJE_CurElement = ne
9634     }
9635   }else
9636   if( cmd == 'n' || cmd == 'nm' || cmd == 'nm' ){
9637     cwe.id = argv[0]
9638   }else
9639   if( cmd == 'r' || cmd == 'rm' || cmd == 'rm' ){
9640   }else
9641   if( cmd == 'h' || cmd == 'sh' || cmd == 'sh' ){
9642     s = argv.join(' ')
9643     cwe.innerHTML = s
9644   }else
9645   if( cmd == 'a' || cmd == 'sa' || cmd == 'sa' ){
9646     cwe.setAttribute(argv[0],argv[1])
9647   }else
9648   if( cmd == 'l' ){
9649   }else
9650   if( cmd == 'i' || cmd == 'ih' || cmd == 'ih' ){
9651     ret = cwe.innerHTML
9652   }else
9653   if( cmd == 'p' || cmd == 'pw' || cmd == 'pw' ){
9654     ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
9655     for( we = cwe.parentNode; we != null; ){
9656       ret += "\n" + we.nodeType + " " + we.tagName + " " + we.id
9657       we = we.parentNode
9658     }
9659   }else
9660   {
9661     ret = "Command: mk | rm \n"
9662     ret += " pw -- print current node\n"
9663     ret += " mk type -- make node with name and type\n"
9664     ret += " nm name -- set the id #name of current node\n"
9665     ret += " rm name -- remove named node\n"
9666     ret += " cd name -- change current node\n"
9667   }
9668   //alert(ret)
9669   return ret
9670 }
9671 function GJC_Command(text){
9672   lines = text.value.split('\n')
9673   line = lines[lines.length-1]
9674   argv = line.split(' ')
9675   text.value = '\n'
9676   if( argv[0] == '$' ){ argv.shift() }
9677   args0 = argv.join(' ')
9678   cmd = argv[0]
9679   argv.shift()
9680   args = argv.join(' ')
9681
9682   if( cmd == 'nolog' ){
9683     StopConsoleLog = true
9684   }else
9685   if( cmd == 'new' ){
9686     if( argv[0] == 'table' ){
9687       argv.shift()
9688       console.log('argv='+argv)
9689       text.value += makeTable(argv)
9690     }else
9691     if( argv[0] == 'console' ){
9692       text.value += GJ_NewConsole('GJ_Console')
9693     }else{
9694       text.value += '-- new { console | table }'
9695     }
9696   }else
9697   if( cmd == 'strip' ){
9698     //text.value += GJF_StripeClass()
9699   }else
9700   if( cmd == 'css' ){
9701     sel = '#table l'
9702     if( argv[0] == '0' ){
9703       rule1 = sel+'{color:#00 !important; background-color:#fff !important;}';
9704     }else
9705     rule1 = sel+'{color:#00 !important; background-color:#eef !important;}';
9706     document.styleSheets[3].deleteRule(0);
9707     document.styleSheets[3].insertRule(rule1,0);
9708     text.value += 'CSS rule added: '+rule1
9709   }else
9710   if( cmd == 'print' ){
9711     e = null;
9712     if( e == null ){
9713       e = document.getElementById('GJFactory_0')
9714     }
9715     if( e == null ){
9716       e = document.getElementById('GJFactory_1')
9717     }
9718     if( argv[0] != null ){
9719       id = argv[0]
9720       if( id == 't' ){
9721         //e = document.getElementById('GJE_RootNode');
9722       }else{
9723         e = document.getElementById(id)
9724       }
9725       if( e != null ){
9726         text.value += e.outerHTML
9727       }else{
9728         text.value += "Not found: " + id
9729       }
9730     }else{
9731       text.value += GJE_RootNode.outerHTML
9732       //text.value += e.innerHTML
9733     }
9734   }else

```

```

9735 if( cmd == 'destroy' ){
9736     text.value += GJFactory_Destroy()
9737 }else
9738 if( cmd == 'save' ){
9739     e = document.getElementById('GJFactory')
9740     Permanent.setItem('GJFactory-1',e.innerHTML)
9741     text.value += "-- Saved GJFactory"
9742 }else
9743 if( cmd == 'load' ){
9744     gjf = Permanent.getItem('GJFactory-1')
9745     e = document.getElementById('GJFactory')
9746     e.innerHTML = gjf
9747     // must restore EventListener
9748     text.value += "-- EventListener was not restored"
9749 }else
9750 if( cmd.charAt(0) == '.' ){
9751     argv0 = argv0.split(' ')
9752     text.value += GJE_NodeEdit(argv0)
9753 }else
9754 if( cmd == 'cont' ){
9755     bannerIsStopping = false
9756     GshMenuStop.innerHTML = "Stop"
9757 }else
9758 if( cmd == 'date' ){
9759     text.value += DateLong()
9760 }else
9761 if( cmd == 'echo' ){
9762     text.value += args
9763 }else
9764 if( cmd == 'fork' ){
9765     html_fork()
9766 }else
9767 if( cmd == 'last' ){
9768     text.value += MyHistory
9769     //h = document.createElement("span")
9770     //h.innerHTML = MyHistory
9771     //text.value += h.innerHTML
9772     //tx = MyHistory.replace("\n","")
9773     //text.value += tx.replace("<'+>","<br>","\n") + "xxxx<'+>yyyy"
9774 }else
9775 if( cmd == 'ne' ){
9776     text.value += GJE_NodeEdit(argv)
9777 }else
9778 if( cmd == 'reload' ){
9779     location.reload()
9780 }else
9781 if( cmd == 'mem' ){
9782     text.value += GJC_Memory('GJC_Storage',args,text)
9783 }else
9784 if( cmd == 'stop' ){
9785     bannerIsStopping = true
9786     GshMenuStop.innerHTML = "Start"
9787 }else
9788 if( cmd == 'who' ){
9789     text.value += "SessionId="+GJC_SessionId+" "+document.URL
9790 }else
9791 if( cmd == 'wall' ){
9792     text.value += GJC_Memory('GJC_Wall','write',text)
9793 }else
9794 {
9795     text.value += "Commands: help | echo | date | last \n"
9796     + '      new | save | load | mem \n'
9797     + '      who | wall | fork | nife'
9798 }
9799 }
9800
9801 function GJC_Input(){
9802     if( this.value.endsWith("\n") ){ // remove NL added by textarea
9803         this.value = this.value.slice(0,this.value.length-1)
9804     }
9805 }
9806
9807 var GCJ_Id = null
9808 function GJC_Resize(){
9809     GCJ_Id.style.zIndex = 20000
9810     //GCJ_Id.style.width = window.innerWidth - 16
9811     GCJ_Id.style.width = '100%';
9812     GCJ_Id.style.height = 300;
9813     GCJ_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9814     GCJ_Id.style.color = "rgba(255,255,255,1.0)"
9815 }
9816 function GJC_FocusIn(){
9817     this.spellCheck = false
9818     SuppressGJShell = true
9819     this.onkeydown = GJC_Keydown
9820     GJC_Resize()
9821 }
9822 function GJC_FocusOut(){
9823     SuppressGJShell = false
9824     this.removeEventListener('keydown',GJC_Keydown);
9825 }
9826 window.addEventListener('resize',GJC_Resize);
9827
9828 function GJC_OnStorage(e){
9829     //alert('Got Message')
9830     //GCJ.value += "\n((ReceivedMessage))\n"
9831 }
9832 window.addEventListener('storage',GJC_OnStorage);
9833 //window.addEventListener('storage',()=>{alert('GotMessage')})
9834
9835 function GJC_Setup(gjcId){
9836     //gjcId.style.width = gsh.getBoundingClientRect().width
9837     gjcId.style.width = '100%';
9838     gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
9839     //gjcId.value += "Date: " + DateLong() + "\n"
9840     gjcId.value += PS1
9841     gjcId.onfocus = GJC_FocusIn
9842     gjcId.addEventListener('input',GJC_Input);
9843     gjcId.addEventListener('focusout',GJC_FocusOut);
9844     GCJ_Id = gjcId
9845 }
9846 function GJC_Clear(id){
9847 }
9848 function GJConsole_initConsole(){
9849     if( document.getElementById("GJC_0") != null ){
9850         GJC_Setup(GJC_0)
9851     }else{
9852         GJCL_Container.innerHTML = '<'
9853             + "textarea id='GJC_1' class='GJConsole'><'+>/textarea>";
9854         GJC_Setup(GJC_1)
9855         factory = document.createElement('span');
9856         gsh.appendChild(factory);
9857         GJE_RootNode = factory;
9858         GJE_CurElement = GJE_RootNode;
9859     }
9860 }
9861 var initGJCF = false;
9862 function GJConsole_initFactory(){
9863     if( initGJCF ){ return; } initGJCF = true;
9864     GShell_initKeyCommands();
9865     GJConsole_initConsole();
9866 }
9867 //GJConsole_initFactory();
9868 // TODO: focus handling
9869 </script>
9870 <style>
9871 .GJ_StyleEditor {
9872     font-size:9pt !important;
9873     font-family:Courier New, monospace !important;
9874 }
9875 </style>
9876
9877 </details>
9878 </span>
9879 <!-- ----- GJConsole END ----- -->
9880 */
9881
9882 /*
9883 <span id="BlinderText">
9884 <style id="BlinderTextStyle">
9885 #GJLinkView {
9886     xposition:absolute; z-index:5000;
9887     position:relative;
9888     display:block;
9889     left:8px;
9890     color:#fff;
9891     width:80px; height:300px; resize:both;
9892     margin:0px; padding:4px;
9893     background-color:rgba(200,200,200,0.5) !important;
9894 }
9895 .MsgText {
9896     width:578px !important;
9897     resize:both !important;
9898     color:#000 !important;
9899 }
9900 .GJNote {
9901     font-family:Georgia !important;
9902     font-size:13pt !important;
9903     color:#22a !important;
9904 }
9905 .textField {
9906     display:inline;
9907     border:0.5px solid #444;
9908     border-radius:3px;
9909     color:#000; background-color:#fff;
9910     width:106pt; height:18pt;
9911     margin:2px;

```



```

9912 padding:2px;
9913 resize:none;
9914 vertical-align:middle;
9915 font-size:10pt; font-family:Courier New;
9916 }
9917 .textLabel {
9918 border:0px solid #000 !important;
9919 background-color:rgba(0,0,0,0);
9920 }
9921 .textURL {
9922 width:300pt !important;
9923 border:0px solid #000 !important;
9924 background-color:rgba(0,0,0,0);
9925 }
9926 .VisibleText {
9927 }
9928 .BlinderText {
9929 color:#000; background-color:#eee;
9930 }
9931 .joinButton {
9932 font-family:Georgia !important;
9933 font-size:11pt;
9934 line-height:1.1;
9935 height:15pt;
9936 width:50pt;
9937 padding:3px !important;
9938 text-align:center !important;
9939 border-color:#aaa !important;
9940 border-radius:5px;
9941 color:#fff; background-color:#4a4 !important;
9942 vertical-align:middle !important;
9943 }
9944 .SendButton {
9945 vertical-align:top;
9946 }
9947 .ws0 log {
9948 font-size:10pt;
9949 color:#000 !important;
9950 line-height:1.0;
9951 background-color:rgba(255,255,255,0.7) !important;
9952 font-family:Courier New,monospace !important;
9953 width:99.3%;
9954 white-space:pre;
9955 }
9956 </style>
9957
9958 <!-- Form autofill test
9959 Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
9960 <form method="POST" id="xxform" action="https://192.168.10.1/boafm/formLogin">
9961 dest? <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
9962 -->
9963 <details><summary>Form Auto. Filling</summary>
9964 <style>
9965 .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9966 display:inline !important; font-size:10pt !important; padding:1px !important;
9967 }
9968 </style>
9969 <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
9970 <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
9971 Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
9972 Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
9973 Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
9974 Sessionid:<input id="xxssid" class="xxinput" name="SESSION_ID" type="text" size="80">
9975 <input id="xxsum" class="xxinput" type="submit" value="Submit"></form>
9976 </span>
9977 <script>
9978 function XXSetFormAction(){
9979 xxform.setAttribute('action',xxserv.value);
9980 }
9981 xxform.setAttribute('action',xxserv.value);
9982 xxserv.addEventListener('change',XXSetFormAction);
9983 //xxserv.value = location.href;
9984 </script>
9985 </details>
9986
9987
9988 /*
9989 <details id="BlinderTextClass"><summary>BlinderText</summary>
9990 <span class="gsh-src">
9991 <span id="BlinderTextScript">
9992 // https://w3c.github.io/uevents/#event-type-keydown
9993 //
9994 // 2020-09-21 class BlinderText - textarea element not to be readable
9995 //
9996 // BlinderText attributes
9997 // bl_plainText - null
9998 // bl_hideChecksum - [false]
9999 // bl_showLength - [false]
10000 // bl_visible - [false]
10001 // data-bl_config - []
10002 // - min. length
10003 // - max. length
10004 // - acceptable charset in generate text
10005 //
10006 function BlinderChecksum(text){
10007 plain = text.bl_plainText;
10008 return strCRC32(plain,plain.length).toFixed(0);
10009 }
10010 function BlinderKeydown(ev){
10011 pass = ev.target
10012 if( ev.code == 'Enter' ){
10013 ev.preventDefault();
10014 }
10015 ev.stopPropagation()
10016 }
10017 function BlinderKeyUp(ev){
10018 blind = ev.target;
10019 if( ev.code == 'Backspace' ){
10020 blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
10021 }else
10022 if( and(ev.code == 'KeyV', ev.ctrlKey) ){
10023 blind.bl_visible = !blind.bl_visible;
10024 }else
10025 if( and(ev.code == 'KeyL', ev.ctrlKey) ){
10026 blind.bl_showLength = !blind.bl_showLength;
10027 }else
10028 if( and(ev.code == 'KeyU', ev.ctrlKey) ){
10029 blind.bl_plainText = "";
10030 }else
10031 if( and(ev.code == 'KeyR', ev.ctrlKey) ){
10032 checksum = BlinderChecksum(blind);
10033 blind.bl_plainText = checksum; // toString(32);
10034 }else
10035 if( ev.code == 'Enter' ){
10036 ev.stopPropagation();
10037 ev.preventDefault();
10038 return;
10039 }else
10040 if( ev.key.length != 1 ){
10041 console.log('KeyUp: '+ev.code+'/'+ev.key);
10042 return;
10043 }else{
10044 blind.bl_plainText += ev.key;
10045 }
10046 }
10047 leng = blind.bl_plainText.length;
10048 //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
10049 checksum = BlinderChecksum(blind) % 10; // show last one digit only
10050
10051 visual = '';
10052 if( !blind.bl_hideChecksum || blind.bl_showLength ){
10053 visual += '|';
10054 }
10055 if( !blind.bl_hideChecksum ){
10056 visual += '#' + checksum.toString(10);
10057 }
10058 if( blind.bl_showLength ){
10059 visual += '/' + leng;
10060 }
10061 if( !blind.bl_hideChecksum || blind.bl_showLength ){
10062 visual += '|';
10063 }
10064 if( blind.bl_visible ){
10065 visual += blind.bl_plainText;
10066 }else{
10067 visual += '*'.repeat(leng);
10068 }
10069 blind.value = visual;
10070 }
10071 function BlinderKeyUp(ev){
10072 BlinderKeyUp(ev);
10073 ev.stopPropagation();
10074 }
10075 // https://w3c.github.io/uevents/#keyboardevent
10076 // https://w3c.github.io/uevents/#uievent
10077 // https://dom.spec.whatwg.org/event
10078 function BlinderTextEvent(){
10079 ev = event;
10080 blind = ev.target;
10081 console.log('Event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
10082 if( ev.type == 'keyup' ){
10083 BlinderKeyUp(ev);
10084 }else
10085 if( ev.type == 'keydown' ){
10086 BlinderKeydown(ev);
10087 }else{
10088 console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)

```

```

10089 }
10090 }
10091 }
10092 //< textarea hidden id="BlinderTextClassDef" class="textField"
10093 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
10094 // spellcheck="false"></textarea>
10095 //< textarea hidden id="gj_pass1"
10096 // class="textField BlinderText"
10097 // placeholder="PassWord"
10098 // onkeydown="BlinderTextEvent()"
10099 // onkeyup="BlinderTextEvent()"
10100 // spellcheck="false"></textarea>
10101 function SetupBlinderText(parent,txa,phold){
10102     if( txa == null ){
10103         txa = document.createElement('textarea');
10104         //txa.id = id;
10105     }
10106     txa.setAttribute('class','textField BlinderText');
10107     txa.setAttribute('placeholder',phold);
10108     txa.setAttribute('onkeydown','BlinderTextEvent()');
10109     txa.setAttribute('spellcheck','false');
10110     //txa.setAttribute('bl_plainText','false');
10111     txa.bl_plainText = '';
10112     //parent.appendChild(txa);
10113 }
10114 function DestroyBlinderText(txa){
10115     txa.removeAttribute('class');
10116     txa.removeAttribute('placeholder');
10117     txa.removeAttribute('onkeydown');
10118     txa.removeAttribute('onkeyup');
10119     txa.removeAttribute('spellcheck');
10120     txa.bl_plainText = '';
10121 }
10122 //
10123 // visible textarea like Username
10124 //
10125 function VisibleTextEvent(){
10126     if( event.code == 'Enter' ){
10127         if( event.target.NoEnter ){
10128             event.preventDefault();
10129         }
10130     }
10131     event.stopPropagation();
10132 }
10133 function SetupVisibleText(parent,txa,phold){
10134     if( false ){
10135         txa.setAttribute('class','textField VisibleText');
10136     }else{
10137         newclass = txa.getAttribute('class');
10138         if( and(newclass != null, newclass != '') ){
10139             newclass += ' ';
10140         }
10141         newclass += 'VisibleText';
10142         txa.setAttribute('class',newclass);
10143     }
10144     //console.log('SetupVisibleText class='+txa.class);
10145     txa.setAttribute('placeholder',phold);
10146     txa.setAttribute('onkeydown','VisibleTextEvent()');
10147     txa.setAttribute('onkeyup','VisibleTextEvent()');
10148     txa.setAttribute('spellcheck','false');
10149     cols = txa.getAttribute('cols');
10150     if( cols != null ){
10151         txa.style.width = '580px';
10152         //console.log('VisualText'+txa.id+' cols='+cols)
10153     }else{
10154         //console.log('VisualText'+txa.id+' NO cols')
10155     }
10156     rows = txa.getAttribute('rows');
10157     if( rows != null ){
10158         txa.style.height = '30px';
10159         txa.style.resize = 'both';
10160         txa.NoEnter = false;
10161     }else{
10162         txa.NoEnter = true;
10163     }
10164 }
10165 function DestroyVisibleText(txa){
10166     txa.removeAttribute('class');
10167     txa.removeAttribute('placeholder');
10168     txa.removeAttribute('onkeydown');
10169     txa.removeAttribute('onkeyup');
10170     txa.removeAttribute('spellcheck');
10171     cols = txa.removeAttribute('cols');
10172 }
10173 </span>
10174 <script>
10175 js = document.getElementById('BlinderTextScript');
10176 eval(js.innerHTML);
10177 //js.outerHTML =
10178 </script>
10179 </span><!-- end of class="gsh-src" -->
10180 </details>
10181 </span>
10182 *
10183 *
10184 *
10185 *
10186 <script id="GJLinkScript">
10187 function gj_key_hash(text){
10188     return strCRC32(text, text.length) % 0x10000;
10189 }
10190 function gj_addlog(e,msg){
10191     now = (new Date()).getTime() / 1000.toFixed(3);
10192     tstp = '['+now+'';
10193     e.value += tstp + msg;
10194     e.scrollTop = e.scrollHeight;
10195 }
10196 function gj_addlog_cl(msg){
10197     ws0_log.value += '(console.log) ' + msg + '\n';
10198 }
10199 var GJ_Channel = null;
10200 var GJ_Log = null;
10201 var gjx; // the global variable
10202 function GJ_join(){
10203     target = gj_join;
10204     if( target.value == 'Leave' ){
10205         GJ_Channel.close();
10206         GJ_Channel = null;
10207         target.value = 'Join';
10208         return;
10209     }
10210 }
10211 var ws0;
10212 var ws0_log;
10213
10214 sav_console_log = console.error
10215 console.error = gj_addlog_cl
10216 ws0 = new WebSocket(gj_serv.innerHTMLHTML);
10217 console.error = sav_console_log
10218
10219 GJ_Channel = ws0;
10220 ws0_log = document.getElementById('ws0_log');
10221 GJ_Log = ws0_log;
10222
10223 now = (new Date()).getTime() / 1000.toFixed(3);
10224 const wsstats = ["CONNECTING","OPEN","CLOSING","CLOSED"];
10225 cst = wsstats[ws0.readyState];
10226 gj_addlog(ws0_log,'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');
10227
10228 ws0.addEventListener('error', function(event){
10229     gj_addlog(ws0_log,'stat error : transport error?\n');
10230 });
10231 ws0.addEventListener('open', function(event){
10232     GJLinkView.style.zIndex = 10000;
10233     //console.log('#'+GJLinkView.id+'.zIndex'+GJLinkView.style.zIndex);
10234     datel = new Date().getTime();
10235     date2 = (datel / 1000).toFixed(3);
10236     seed = datel.toString(16);
10237
10238     // user name and key
10239     user = document.getElementById('gj_user').value;
10240     if( user.length == 0 ){
10241         gj_user.value = 'nemo';
10242         user = 'nemo';
10243     }
10244     key1 = document.getElementById('gj_ukey').bl_plainText;
10245     ukey = gjkey_hash(seed+user+key1).toString(16);
10246
10247     // session name and key
10248     chan = document.getElementById('gj_chan').value;
10249     if( chan.length == 0 ){
10250         gj_chan.value = 'main';
10251         chan = 'main';
10252     }
10253     key2 = document.getElementById('gj_ckey').bl_plainText;
10254     ckey = gjkey_hash(seed+chan+key2).toString(16);
10255
10256     msg = date2 + ' JOIN ' + user + '|' + chan + ' ' + ukey + '|' + ckey;
10257     gj_addlog(ws0_log,'send '+msg+'\n');
10258     ws0.send(msg);
10259
10260     target.value = 'Leave';
10261     //console.log('[ '+date2+' ] #'+target.id+' '+target.value+'\n');
10262     //gj_addlog(ws0_log,'label '+target.value+'\n');
10263 });
10264 ws0.addEventListener('message', function(event){
10265     now = (new Date()).getTime() / 1000.toFixed(3);

```

```

10266 msg = event.data;
10267 if ( false ){
10268     gj_addlog(ws0_log,'recv '+msg+'\n');
10269 }
10270 argv = msg.split(' ');
10271 tstamp = argv[0];
10272 argv.shift();
10273 if ( argv[0] == 'reload' ){
10274     location.reload()
10275 }
10276 gjcmd = argv[0];
10277 otstamp = '';
10278 if ( gjcmd == 'CAST' ){ // from reflector
10279     otstamp = argv[0];
10280     argv.shift(); // original time stamp
10281     ofrom = argv[0];
10282     argv.shift(); // original from
10283 }
10284 argv.shift(); // command
10285 from = argv[0];
10286 argv.shift(); // from/to
10287 cmdl = argv[0];
10288 argv.shift(); // xxxx command
10289
10290 if ( false ){
10291     gj_addlog(ws0_log,'--'
10292         + ' tstamp='+tstamp
10293         + ' gjcmd='+gjcmd
10294         + ' from='+from
10295         + ' cmdl='+cmdl+' '
10296         + ' '+argv+'\n');
10297 }
10298 if ( cmdl == 'auth' ){
10299     // doing authorization required
10300 }
10301 if ( cmdl == 'echo' ){
10302     now = (new Date()).getTime() / 1000).toFixed(3);
10303     msg = now+ ' ' + 'RESP '+argv.join(' ');
10304     gj_addlog(ws0_log,'send '+msg+'\n');
10305     ws0.send(msg);
10306 }
10307 if ( cmdl == 'eval' ){
10308     argv.shift();
10309     js = argv.join(' ');
10310     ret = eval(js); // <----- eval()
10311     gj_addlog(ws0_log,'eval '+js+' = '+ret+'\n');
10312     now = (new Date()).getTime() / 1000).toFixed(3);
10313     msg = now + ' ' + 'RESP ' + ret;
10314     ws0.send(msg);
10315     gj_addlog(ws0_log,'send '+msg+'\n')
10316 }
10317 if ( cmdl == 'DRAW' ){
10318     if ( false ){
10319         gj_addlog(ws0_log,'DRAW '+argv[0]+'\n')
10320     }
10321     Pointillism_RemoteDraw(argv[0]);
10322 }
10323 if ( cmdl == 'MOUSEPOSITION' ){
10324     XYeyes_recvMousePosition(argv[0]);
10325 }
10326 });
10327 ws0.addEventListener('close', function(event){
10328     if ( GJ_Channel == null ){
10329         gj_addlog(ws0_log,'stat OK : GJ UnLinked\n');
10330         return;
10331     }
10332     GJ_Channel.close();
10333     GJ_Channel = null;
10334     target.value = 'Join';
10335     gj_addlog(ws0_log,'stat error : close : GJ UnLinked unexpectedly\n');
10336 });
10337
10338 function GJ_BcastMessageUserPass(user,chan,msgbody){
10339     now = (new Date()).getTime() / 1000).toFixed(3);
10340     msg = now + ' BCAST ' +user+' '+chan+' '+msgbody;
10341     if ( false ){
10342         gj_addlog(GJ_Log,'send '+msg+'\n');
10343     }
10344     GJ_Channel.send(msg);
10345 }
10346
10347 function GJ_BcastMessage(msgbody){
10348     if ( GJ_Channel == null ){
10349         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
10350         return;
10351     }
10352     //target = event.target;
10353     user = document.getElementById('gj_user').value;
10354     chan = document.getElementById('gj_chan').value;
10355     GJ_BcastMessageUserPass(user,chan,msgbody);
10356 }
10357
10358 function GJ_SendMessageUserPass(user,chan,msgbody){
10359     now = (new Date()).getTime() / 1000).toFixed(3);
10360     msg = now + ' SBY '+user+' '+chan+' '+msgbody;
10361     gj_addlog(GJ_Log,'send '+msg+'\n');
10362     GJ_Channel.send(msg);
10363 }
10364
10365 function GJ_SendMessage(msgbody){
10366     if ( GJ_Channel == null ){
10367         gj_addlog(ws0_log,'stat error : send : GJ not Linked\n');
10368         return;
10369     }
10370     //target = event.target;
10371     user = document.getElementById('gj_user').value;
10372     chan = document.getElementById('gj_chan').value;
10373     GJ_SendMessageUserPass(user,chan,msgbody);
10374 }
10375
10376 function GJ_Send(){
10377     msgbody = gj_sendText.value;
10378     GJ_SendMessage(msgbody);
10379 }
10380
10381 </script>
10382
10383 <!-- ----- GJLINK ----- -->
10384
10385 - User can subscribe to a channel
10386 - A channel will be broadcasted
10387 - A channel can be a pattern (regular expression)
10388 - User is like from:(me) and channel is like To: or Recipient:
10389 - like VIABUS
10390 - watch message with SENDME, WATCH, CATCH, HEAR, or so
10391 - routing with path expression or name pattern (with routing with DNS like system)
10392 -->
10393
10394
10395
10396
10397
10398
10399
10400
10401
10402
10403
10404
10405
10406
10407
10408
10409
10410
10411
10412
10413
10414
10415
10416
10417
10418
10419
10420
10421
10422
10423
10424
10425
10426
10427
10428
10429
10430
10431
10432
10433
10434
10435
10436
10437
10438
10439
10440
10441
10442
10443
10444
10445
10446
10447
10448
10449
10450
10451
10452
10453
10454
10455
10456
10457
10458
10459
10460
10461
10462
10463
10464
10465
10466
10467
10468
10469
10470
10471
10472
10473
10474
10475
10476
10477
10478
10479
10480
10481
10482
10483
10484
10485
10486
10487
10488
10489
10490
10491
10492
10493
10494
10495
10496
10497
10498
10499
10500
10501
10502
10503
10504
10505
10506
10507
10508
10509
10510
10511
10512
10513
10514
10515
10516
10517
10518
10519
10520
10521
10522
10523
10524
10525
10526
10527
10528
10529
10530
10531
10532
10533
10534
10535
10536
10537
10538
10539
10540
10541
10542
10543
10544
10545
10546
10547
10548
10549
10550
10551
10552
10553
10554
10555
10556
10557
10558
10559
10560
10561
10562
10563
10564
10565
10566
10567
10568
10569
10570
10571
10572
10573
10574
10575
10576
10577
10578
10579
10580
10581
10582
10583
10584
10585
10586
10587
10588
10589
10590
10591
10592
10593
10594
10595
10596
10597
10598
10599
10600
10601
10602
10603
10604
10605
10606
10607
10608
10609
10610
10611
10612
10613
10614
10615
10616
10617
10618
10619
10620
10621
10622
10623
10624
10625
10626
10627
10628
10629
10630
10631
10632
10633
10634
10635
10636
10637
10638
10639
10640
10641
10642
10643
10644
10645
10646
10647
10648
10649
10650
10651
10652
10653
10654
10655
10656
10657
10658
10659
10660
10661
10662
10663
10664
10665
10666
10667
10668
10669
10670
10671
10672
10673
10674
10675
10676
10677
10678
10679
10680
10681
10682
10683
10684
10685
10686
10687
10688
10689
10690
10691
10692
10693
10694
10695
10696
10697
10698
10699
10700
10701
10702
10703
10704
10705
10706
10707
10708
10709
10710
10711
10712
10713
10714
10715
10716
10717
10718
10719
10720
10721
10722
10723
10724
10725
10726
10727
10728
10729
10730
10731
10732
10733
10734
10735
10736
10737
10738
10739
10740
10741
10742
10743
10744
10745
10746
10747
10748
10749
10750
10751
10752
10753
10754
10755
10756
10757
10758
10759
10760
10761
10762
10763
10764
10765
10766
10767
10768
10769
10770
10771
10772
10773
10774
10775
10776
10777
10778
10779
10780
10781
10782
10783
10784
10785
10786
10787
10788
10789
10790
10791
10792
10793
10794
10795
10796
10797
10798
10799
10800
10801
10802
10803
10804
10805
10806
10807
10808
10809
10810
10811
10812
10813
10814
10815
10816
10817
10818
10819
10820
10821
10822
10823
10824
10825
10826
10827
10828
10829
10830
10831
10832
10833
10834
10835
10836
10837
10838
10839
10840
10841
10842
10843
10844
10845
10846
10847
10848
10849
10850
10851
10852
10853
10854
10855
10856
10857
10858
10859
10860
10861
10862
10863
10864
10865
10866
10867
10868
10869
10870
10871
10872
10873
10874
10875
10876
10877
10878
10879
10880
10881
10882
10883
10884
10885
10886
10887
10888
10889
10890
10891
10892
10893
10894
10895
10896
10897
10898
10899
10900
10901
10902
10903
10904
10905
10906
10907
10908
10909
10910
10911
10912
10913
10914
10915
10916
10917
10918
10919
10920
10921
10922
10923
10924
10925
10926
10927
10928
10929
10930
10931
10932
10933
10934
10935
10936
10937
10938
10939
10940
10941
10942
10943
10944
10945
10946
10947
10948
10949
10950
10951
10952
10953
10954
10955
10956
10957
10958
10959
10960
10961
10962
10963
10964
10965
10966
10967
10968
10969
10970
10971
10972
10973
10974
10975
10976
10977
10978
10979
10980
10981
10982
10983
10984
10985
10986
10987
10988
10989
10990
10991
10992
10993
10994
10995
10996
10997
10998
10999
11000

```

```

10443     fmt.Fprintf(GshLog,fmts+"\n",params...)
10444 }
10445 }
10446
10447 var WSV = []*websocket.Conn{}
10448 func jsend(argv []string){
10449     if len(argv) <= 1 {
10450         fmt.Printf("--i] %v [-m command arguments\n",argv[0])
10451         return
10452     }
10453     argv = argv[1:]
10454     if( len(WSV) == 0 ){
10455         fmt.Printf("--Ej-- No link now\n")
10456         return
10457     }
10458     if( 1 < len(WSV) ){
10459         fmt.Printf("--j-- multiple links (%v)\n",len(WSV))
10460     }
10461
10462     multicast := false // should be filtered with regexp
10463     if( 0 < len(argv) && argv[0] == "-m" ){
10464         multicast = true
10465         argv = argv[1:]
10466     }
10467     args := strings.Join(argv, " ")
10468
10469     now := time.Now()
10470     msec := now.UnixNano() / 1000000;
10471     tstamp := fmtstring("%.3f",float64(msec)/1000.0)
10472     msg := fmtstring("%v SEND gshell* %v",tstamp,args)
10473
10474     if( multicast ){
10475         for i,ws := range WSV {
10476             wn,werr := ws.Write([]byte(msg))
10477             if( werr != nil ){
10478                 fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10479             }
10480             glog("SQ",fmtstring("(%v) %v",wn,msg))
10481         }
10482     }else{
10483         i = 0
10484         ws := WSV[i]
10485         wn,werr := ws.Write([]byte(msg))
10486         if( werr != nil ){
10487             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10488         }
10489         glog("SQ",fmtstring("(%v) %v",wn,msg))
10490     }
10491 }
10492
10493 func ws_broadcast(msg string)(wn int,werr error){
10494     for i,ws := range WSV {
10495         wn,werr = ws.Write([]byte(msg))
10496         if( werr != nil ){
10497             fmt.Printf("[%v] wn=%v, werr=%v\n",i,wn,werr)
10498         }
10499         glog("SQ",fmtstring("(%v) %v",wn,msg))
10500     }
10501     return wn,werr;
10502 }
10503
10504 func servl(ws *websocket.Conn) {
10505     WSV = append(WSV,ws)
10506     //fmt.Print("\n")
10507     glog("CO",accepted connections[%v]",len(WSV))
10508     //remoteAddr := ws.RemoteAddr
10509     //fmt.Printf("-- accepted %v\n",remoteAddr)
10510     //fmt.Printf("-- accepted %v\n",ws.Config().Header)
10511     //fmt.Printf("-- accepted %v // %v\n",ws,servl)
10512
10513     var reqb = make([]byte, GSHWS_MSGSIZE)
10514     for {
10515         rn, rerr := ws.Read(reqb)
10516         if( rerr != nil || rn <= 0 ){
10517             glog("SQ",fmtstring("(%v,%v)",rn,rerr))
10518             break
10519         }
10520         req := string(reqb[0:rn])
10521         glog("SQ",fmtstring("(%v) %v",rn,req))
10522         margv = strings.Split(req, " ")
10523         margv = margv[1:]
10524         if( 0 < len(margv) ){
10525             if( margv[0] == "RESP" ){
10526                 // should forward to the destination
10527                 continue;
10528             }
10529         }
10530         now := time.Now()
10531         msec := now.UnixNano() / 1000000;
10532         tstamp := fmtstring("%.3f",float64(msec)/1000.0)
10533         res := fmtstring("%v "+ "CAST"+ "%v",tstamp,req)
10534
10535         wn := 0;
10536         werr := error(nil);
10537         if( 0 < len(margv) && margv[0] == "BCAST" ){
10538             wn, werr = ws_broadcast(res);
10539         }else{
10540             wn, werr = ws.Write([]byte(res))
10541         }
10542         gchk("SE",werr)
10543         glog("SR",fmtstring("(%v) %v",wn,string(res)))
10544     }
10545     glog("SP", "WS response finish")
10546
10547     wsv := []*websocket.Conn{}
10548     wsx := 0
10549     for i, v := range WSV {
10550         if( v != ws ){
10551             wsx = i
10552             wsv = append(wsv,v)
10553         }
10554     }
10555     WSV = wsv
10556     //glog("CO", "closed %v",ws)
10557     glog("CO", "closed connection [%v/%v]",wsx+1,len(WSV)+1)
10558     ws.Close()
10559 }
10560
10561 // url := [scheme://host[:port]][/path]
10562 func decomp_URL(url string){
10563 }
10564
10565 func full_wURL(){
10566 }
10567
10568 func gj_server(argv []string) {
10569     gjserv := gshws_url
10570     gjport := gshws_server
10571     gjpath := gshws_path
10572     gjscheme := "ws"
10573
10574     //cmd := argv[0]
10575     argv = argv[1:]
10576     if( 1 < len(argv) ){
10577         serv := argv[0]
10578         if( 0 < strings.Index(serv,"://") ){
10579             schemev := strings.Split(serv,"://")
10580             gjscheme = schemev[0]
10581             serv = schemev[1]
10582         }
10583         if( 0 < strings.Index(serv,"/") ){
10584             pathv := strings.Split(serv,"/")
10585             serv = pathv[0]
10586             gjpath = pathv[1]
10587         }
10588         servv := strings.Split(serv,":")
10589         host := "localhost"
10590         port := 9999
10591         if( servv[0] != "" ){
10592             host = servv[0]
10593         }
10594         if( len(servv) == 2 ){
10595             fmt.Sscanf(servv[1],"%d",&port)
10596         }
10597         //glog("LC", "hostport=%v (%v : %v)",servv,host,port)
10598         gjport = fmt.Sprintf("%v:%v",host,port)
10599         gjserv = gjscheme + "://" + gjport + "/" + gjpath
10600     }
10601     glog("LS",fmtstring("listening at %v",gjserv))
10602     http.Handle("/"+gjpath,websocket.Handler(serv))
10603     err := error(nil)
10604     if( gjscheme == "wss" ){
10605         // https://golang.org/pkg/net/http/#ListenAndServeTLS
10606         //err = http.ListenAndServeTLS(gjport,nil)
10607     }else{
10608         err = http.ListenAndServe(gjport,nil)
10609     }
10610     gchk("LE",err)
10611 }
10612
10613 func gj_client(argv []string) {
10614     glog("CS",fmtstring("connecting to %v",gshws_url))
10615     ws, err := websocket.Dial(gshws_url,"",gshws_origin)
10616     gchk("C",err)
10617
10618     var resb = make([]byte, GSHWS_MSGSIZE)
10619     for ql := 0; ql < 3; ql++ {
10620         req := fmtstring("Hello, GShell! (%v)",ql)
10621         wn, werr := ws.Write([]byte(req))
10622         glog("QM",fmtstring("(%v) %v",wn,req))
10623     }

```

```

10620     gchk("QE",werr)
10621     rn, rerr := ws.Read(resb)
10622     gchk("RE",rerr)
10623     glog("RM",fmtstring("%v %v",rn,string(resb)))
10624     }
10625     glog("CF", "WS request finish")
10626 }
10627 //</span><!-- end of class="gsh-src" -->
10628 </details></span>
10629 *
10630 <details id="GJLink Section"><summary>GJ Link</summary>
10631 <span id="GJLinkView" class="GJLinkView">
10632 <p>
10633 <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
10634 </p>
10635 </span>
10636 <span id="GJLink 1">
10637 <div id="GJLink ServerSet"></div>
10638 </span>
10639 <div>
10640 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
10641 <span id="GJLink_Account"></span>
10642 </div>
10643 <div>
10644 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
10645 <span id="GJLink_SendArea"></span>
10646 </div>
10647 <div id="ws0_log_container"></div>
10648 </span>
10649 </span>
10650 </script>
10651 function setupGJLinkArea(){
10652     GJLink_ServerSet.innerHTML = '<'+span id="gj_serv_label"'
10653     + ' class="textField textLabel">Server: <'+/span>'
10654     + '<'+span id="gj_serv" class="textField textLabel" contenteditable><'+/span>';
10655 }
10656 GJLink_Account.innerHTML = '<'+textarea id="gj_user" class="textField"><'+/textarea>'
10657 + '<'+textarea id="gj_ukey" class="textField"><'+/textarea>'
10658 + '<'+textarea id="gj_chan" class="textField"><'+/textarea>'
10659 + '<'+textarea id="gj_ckey" class="textField"><'+/textarea>';
10660 GJLink_SendArea.innerHTML =
10661 '<'+textarea id="gj_sendText" class="textField MsgText" cols=60 rows=2><'+/textarea>';
10662 ws0_log_container.innerHTML = '<'+textarea id="ws0_log" class="ws0_log"'
10663 + ' cols=100 rows=10 spellcheck="false"><'+/textarea>';
10664 }
10665 function clearGJLinkArea(){
10666     GJLink_ServerSet.innerHTML = "";
10667     GJLink_Account.innerHTML = "";
10668     GJLink_SendArea.innerHTML = "";
10669     ws0_log_container.innerHTML = "";
10670 }
10671 </script>
10672 <script>
10673 function SetupGJLink(){
10674     setupGJLinkArea();
10675     SetupVisibleText(GJLink_l_gj_serv, 'GJLinkSv');
10676     SetupVisibleText(GJLink_l_gj_user, 'UserName');
10677     SetupBlinderText(GJLink_l_gj_ukey, 'UserKey');
10678     SetupVisibleText(GJLink_l_gj_chan, 'ChannelName');
10679     SetupBlinderText(GJLink_l_gj_ckey, 'ChannelKey');
10680     SetupVisibleText(GJLink_l_gj_sendText, 'Message');
10681 }
10682 console.log('location.href: '+location.href);
10683 console.log('location.host: '+location.host);
10684 wshost = location.host;
10685 gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
10686 //gj_serv.innerHTML = 'ws://'+wshost+':9999/gjlink1' // 2020-1117
10687 }
10688 function GJLink_init(){
10689     SetupGJLink();
10690 }
10691 function iselem(eid){
10692     return document.getElementById(eid);
10693 }
10694 function DestroyGJLink1(){
10695     clearGJLinkArea();
10696     if( iselem('gj_user') ){
10697         return;
10698     }
10699     if( gj_serv_label.parentNode != gj_user ){
10700         return;
10701     }
10702     if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
10703     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
10704     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
10705     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
10706     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
10707     if( iselem('gj_ckey') ) gj_ckey.parentNode.removeChild(gj_ckey);
10708     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
10709     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
10710 }
10711 DestroyGJLink = DestroyGJLink1;
10712 </script>
10713 </details>
10714 *
10715 </style>
10716 <script id="HtmlCodeview-script">
10717 function showNodeAsHtmlSourceX(otxa,code, prefix, postfix, sign){
10718     txa = document.createElement('textarea');
10719     txa.id = otxa.id;
10720     txa.setAttribute('class', 'HtmlCodeviewText');
10721     otxa.parentNode.replaceChild(txa,otxa);
10722     txa.setAttribute('spellcheck', 'false');
10723     //txa.value = code.innerHTML;
10724     //txa.innerHTML = code.innerHTML;
10725     txa.innerHTML = prefix + code.outerHTML + postfix;
10726     if( sign ){
10727         tlen = txa.value.length;
10728         digest = strCRC32(text,tlen) + ' ' + tlen
10729         + ' ' + code.id + ' (' + DateShort() + ')';
10730         //alert('digest: '+digest);
10731         console.log('digest: '+digest);
10732         txa.innerHTML += '<'+span class="GJDigest">'+digest+'<'+/span>\n';
10733     }
10734     txa.style.display = "block";
10735     txa.style.width = "100%";
10736     txa.style.height = "300px";
10737 }
10738 function showHtmlCodeX(otxa,code, prefix, postfix, sign){
10739     if( event.target.value == 'ShowCode' ){
10740         showNodeAsHtmlSourceX(otxa,code, prefix, postfix, sign);
10741         event.target.value = 'HideCode';
10742     }else{
10743         otxa.style.display = "none";
10744         event.target.value = 'ShowCode';
10745     }
10746 }
10747 function showNodeAsHtmlSource(otxa,code){
10748     showNodeAsHtmlSourceX(otxa,code, '','');
10749 }
10750 function showHtmlCode(otxa,code){
10751     if( event.target.value == 'ShowCode' ){
10752         showNodeAsHtmlSource(otxa,code);
10753         event.target.value = 'HideCode';
10754     }else{
10755         otxa.style.display = "none";
10756         event.target.value = 'ShowCode';
10757     }
10758 }
10759 </script>
10760 <style id="HtmlCodeview-style">
10761 .HtmlCodeviewText {
10762     font-size:10pt;
10763     font-family:Courier New;
10764     white-space:pre;
10765 }
10766 .HtmlCodeviewButton {
10767     padding:2pt 10important;
10768     line-height:1.1 10important;
10769     border:2px inset #bbb 10important;
10770     font-size:11pt 10important;
10771     font-weight:normal 10important;
10772     font-family:Georgia 10important;
10773     border-radius:3px 10important;
10774     color:#ddd; background-color:#228 10important;
10775 }
10776 </style>
10777 *
10778 </script>
10779 <summary>Live HTML Snapshot</summary>
10780 <span id="LiveHTML">
10781 <!-- ----- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->

```

```

1079 <div class="GshMenu">
1079 <span class="GshMenu" onclick="html_edit();">Edit</span>
1079 <span class="GshMenu" onclick="html_save();">Save</span>
1080 <span class="GshMenu" onclick="html_load();">Load</span>
1080 <span class="GshMenu" onclick="html_ver0();">Vers</span>
1080 </div>
1080 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLcode()">
1080 <span id="LiveHTML_Codeview"></span>
1080 </div>
1080 <script id="LiveHtmlScript">
1080 function showLiveHTMLcode(){
1080   showHtmlCode(LiveHTML_Codeview,LiveHTML);
1081 }
1081 var _editable = false;
1081 var savSuppressGJShell = false;
1081 function ToggleEditMode(){
1081   _editable = !_editable;
1081   if( _editable ){
1081     savSuppressGJShell = SuppressGJShell;
1081     SuppressGJShell = true;
1081     gsh.setAttribute('contenteditable','true');
1081     GshMenuEdit.innerHTML = 'Lock';
1081     GshMenuEdit.style.color = 'rgba(255,0,0,1)';
1081     GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
1082   }else{
1082     SuppressGJShell = savSuppressGJShell;
1082     gsh.setAttribute('contenteditable','false');
1082     GshMenuEdit.innerHTML = 'Edit';
1082     GshMenuEdit.style.color = 'rgba(16,160,16,1)';
1082     GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
1082   }
1082 }
1083 function html_edit(){
1083   ToggleEditMode();
1083 }
1083 // Live HTML (DOM) Snapshot onto browser's localStorage
1083 // 2020-0923 SatoxITS
1083 var htRoot = gsh // -- Element-ID, should be selectable
1083 const snappedHTML = "SnappedHTML"; // Item-ID of the HTML data in localStogate
1083 // -- should be a [map] of URL
1083 // -- should be with CSSOM as inline script
1084 const htVersionTag = "VersionTag"; // VesionTag Element-ID in the HTML (in DOM)
1084 function showVersion(note,w,v,u,t){
1084   w.alert(note); // + v + '\n'
1084   + '-- URL: ' + u + '\n'
1084   + '-- Time: ' + t + ' (' + DateLong(t*1000) + ')';
1084 }
1084 function html_save(){
1084   u = document.URL;
1084   t = new Date().getTime() / 1000;
1084   v = '<'+_span id="+htVersionTag+" data-url="+u+" data-time="+t+">';
1084   v += '<'+_span>\n';
1084   h += htRoot.outerHTML;
1084   localStorage.setItem(snappedHTML,h);
1084   showVersion("Saved",window,v,u,t);
1084 }
1084 function html_load(){
1084   h = localStorage.getItem(snappedHTML);
1084   if( h == null ){
1084     alert('No snapshot taken yet');
1084   }
1084   return;
1084 }
1084 w = window.open('','');
1084 d = w.document;
1084 d.write(h);
1084 w.focus();
1084 html_ver1("Loaded",w,d);
1084 }
1084 function html_ver1(note,w,d){
1084   if( (v = d.getElementById(htVersionTag)) != null ){
1084     h = v.outerHTML;
1084     u = v.getAttribute('data-url');
1084     t = v.getAttribute('data-time');
1084   }else{
1084     h = 'No version info. in the page';
1084     u = '';
1084     t = 0;
1084   }
1084   showVersion(note,w,v,u,t);
1084 }
1084 function html_ver0(){
1084   html_ver1("Version",window,document);
1084 }
1084 </script>
1084 <!-- LiveHTML -->
1084 </span>
1084 </details>
1084 */
1084 *
1084 <details><summary>Event sharing</summary>
1084 <span id="EventSharingCodeSpan">
1085 <!-- ----- Event sharing // 2020-0925 SatoxITS { -->
1085
1085 <div id="iftestTemplate" class="iftest" hidden="">
1085 <style>.iftestbody{ color:#f22;font-family:Georgia;font-size:10pt;}</style>
1085 <span id="frameBody" class="iftestbody" onclick="frameClick()"><script>
1085 function docadd(txt){
1085   document.body.append(txt);
1085   window.scrollTo(0,100000);
1085 }
1085 function frameClick(){
1085   xy = 'x'+event.x + ' y'+event.y+'';
1085   //docadd('Got Click on #' +event.target.id+' +xy+ '\n');
1085   docadd('Got Click on #' +Fid.value+' , +xy+ '\n');
1085   window.scrollTo(0,100000);
1085   window.parent.postMessage('OnClick: '+xy, '*');
1085 }
1085 function frameMousemove(){
1085   if( false ){
1085     document.body.append('Mousemove on #' +event.target.id+' '
1085     + 'x'+event.x + ' y'+event.y + '\n');
1085     peerWin = window.frames.iframe1;
1085     document.body.append('Send to peer #' +peerWin+' ' + '\n');
1085     window.scrollTo(0,100000);
1085     peerWin.postMessage('Hi!', '*');
1085   }
1085 }
1085 function frameKeydown(){
1085   msg = 'Got Keydown: #' +Fid.value+' , (' +event.code+' )';
1085   docadd(msg+ '\n');
1085   window.parent.postMessage(msg, '*');
1085 }
1085 function frameOnMessage(){
1085   docadd('Message ' + event.data + '\n');
1085   window.scrollTo(0,100000);
1085 }
1085 if( document.getElementById('Fid') ){
1085   frameBody.id = Fid.value;
1085   h = '';
1085   h += '<'+_style>';
1085   h += 'font-size:10pt;white-space:pre-wrap;';
1085   h += 'font-family:Courier New;';
1085   h += '<'+_style>';
1085   h += 'id am '+Fid.value+' \n';
1085   document.write(h);
1085   window.addEventListener('click',frameClick);
1085   window.addEventListener('keydown',frameKeydown);
1085   window.addEventListener('message',frameOnMessage);
1085   window.addEventListener('mousemove',frameMousemove);
1085   window.parent.postMessage('Hi parent, I am '+Fid.value, '*');
1085 }
1085 </script></span></div>
1085
1085 <style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
1085 <h2>Inter-window communication</h2>
1085 <note>
1085 frame0 >>> frame1 and frame2<br>
1085 frame1 >>> frame0 and frame2<br>
1085 frame2 >>> frame0 and frame1<br>
1085 </note>
1085 <div id="iframe-test">
1085 <pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
1085 <iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
1085 <iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
1085 <iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
1085 </div>
1085
1085 <script id="if0-test-script">
1085 function InterFrameComm_init(){
1085   setupFrames0();
1085   setupFrames1();
1085 }
1085 function setFrameSrcdoc(dst,src){
1085   if( true ){
1085     dst.contentWindow.document.write(src);
1085     // this makes browser write close, and crash if accumulated !?
1085     // so it should be closed after write
1085     dst.contentWindow.document.close();
1085   }else{
1085     // to be erased before source dump
1085     // but should be set for live snapshot
1085     dst.srcdoc = src;
1085 }

```

```

10974 }
10975 }
10976 function setupFrames0(){
10977     body = iframe0.contentWindow.document.body;
10978     iframe0.style.width = "755px"
10979     //iframeHost.innerHTML += "Message exchange at iframes' host:\n";
10980     window.addEventListener('message', messageFromChild);
10981
10982     if0 = '';
10983     if0 += '<pre style="font-family:Courier New;">';
10984     if0 += '<input id="Fid" value="iframe0">';
10985     if0 += iftestTemplate.innerHTML;
10986     setFrameSrcdoc(iframe0,if0);
10987
10988     function clickOnChild(){
10989         console.log('clickOn #' + this.id);
10990     }
10991     function moveOnChild(){
10992         console.log('moveOn #' + this.id);
10993     }
10994     iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
10995     iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
10996 }
10997 function setupFrames12(){
10998     if1 = '<input id="Fid" value="iframe1">';
10999     if1 += iftestTemplate.innerHTML;
11000     setFrameSrcdoc(iframe1,if1);
11001     //iframe1.name = 'iframe1'; // this seems break contentWindow
11002
11003     if2 = '<input id="Fid" value="iframe2">';
11004     if2 += iftestTemplate.innerHTML;
11005     setFrameSrcdoc(iframe2,if2);
11006
11007     iframe1.addEventListener('message',messageFromChild);
11008     //iframe1.addEventListener('mouseover',moveOnChild);
11009     iframe2.addEventListener('message',messageFromChild);
11010     //iframe2.addEventListener('mouseover',moveOnChild);
11011     iframe1.contentWindow.postMessage(' [parent0] Hi iframe1 -- from parent.', '*');
11012     //iframe1.contentWindow.postMessage(' Your peer is '+iframe2.contentWindow, '*');
11013     iframe2.contentWindow.postMessage(' [parent0] Hi iframe2 -- from parent.', '*');
11014     //iframe2.contentWindow.postMessage(' Your peer is '+iframe1.contentWindow, '*');
11015
11016     function messageFromChild(){
11017         from = null;
11018         forw = null;
11019         if( event.source == iframe0.contentWindow ){
11020             from = 'iframe0';
11021             forw = 'iframe2';
11022         }else
11023         if( event.source == iframe1.contentWindow ){
11024             from = 'iframe1';
11025             forw = 'iframe2';
11026         }else
11027         if( event.source == iframe2.contentWindow ){
11028             from = 'iframe2';
11029             forw = 'iframe1';
11030         }else
11031         {
11032             iframeHost.innerHTML += "Message [unknown] "
11033             + " orig=" + event.origin
11034             + " data=" + event.data
11035             //+ " from=" + event.source
11036             ;
11037         }
11038         msglog1 = from + event.data + ' -- '
11039         + " from=" + event.source
11040         + " orig=" + event.origin
11041         + " name=" + event.source.name
11042         //+ " port=" + event.porta
11043         //+ " evid=" + event.lastEventId
11044         + "\n";
11045
11046         if( true ){
11047             if( forw == 'iframe1' || forw == 'iframe2' ){
11048                 iframe1.contentWindow.postMessage(from+event.data);
11049             }
11050             if( forw == 'iframe2' || forw == 'iframe1' ){
11051                 iframe2.contentWindow.postMessage(from+event.data);
11052             }
11053         }
11054         txtadd0(msglog1);
11055     }
11056     function txtadd0(txt){
11057         iframe0.contentWindow.document.body.append(txt);
11058         iframe0.contentWindow.scrollTo(0,100000);
11059     }
11060 }
11061 function es_ShowSelf(){
11062     iframe1.setAttribute('src',document.URL);
11063     iframe2.setAttribute('src',document.URL);
11064 }
11065 </script>
11066
11067 <input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
11068 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
11069 <span id="EventSharingCodeview"></span>
11070 <script id="EventSharingScript">
11071 function es_showHtmlCode(){
11072     showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
11073 }
11074 DestroyEventSharingCodeview = function(){
11075     //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
11076     EventSharingCodeview.innerHTML = "";
11077     iframe0.style = "";
11078     //iframe0.srcdoc = "erased";
11079     //iframe1.srcdoc = "erased";
11080     //iframe2.srcdoc = "erased";
11081 }
11082 </script>
11083 <!-- EventSharing -->
11084 </span>
11085 </details>
11086 */
11087
11088 /*
11089 <!-- ----- "GShell Inside" Notification { -->
11090 <script id="script-gshell-inside">
11091 var notices = 0;
11092 function noticeGShellInside(){
11093     ver = '';
11094     if( ver = document.getElementById('GshVersion') ){
11095         ver = ver.innerHTML;
11096     }
11097     console.log('GShell Inside (^-^)' + ver);
11098     notices = 1;
11099     if( 2 <= notices ){
11100         document.removeEventListener('mousemove',noticeGShellInside);
11101     }
11102 }
11103 document.addEventListener('mousemove',noticeGShellInside);
11104 noticeGShellInside();
11105
11106 const FooterName = 'GshFooter'
11107 function DestroyFooter(){
11108     if( footer = document.getElementById(FooterName) != null ){
11109         //footer.parentNode.removeChild(footer);
11110         empty = document.createElement('div');
11111         empty.id = 'GshFooter0';
11112         footer.parentNode.replaceChild(empty, footer);
11113     }
11114 }
11115 function showFooter(){
11116     footer = document.createElement('div');
11117     footer.id = FooterName;
11118     footer.style.backgroundImage = "url('+ITSmoreQR+')";
11119     //GshFooter0.parentNode.appendChild(footer);
11120     if( document.getElementById('GshFooter0') != null ){
11121         GshFooter0.parentNode.replaceChild(footer,GshFooter0);
11122     }
11123 }
11124 </script>
11125 <!-- } -->
11126 <!--
11127 <!--
11128 border:20px inset #888;
11129 -->
11130 </span id="VirtualDesktopCodeSpan">
11131 /*
11132 <details id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
11133 <!-- ----- Web Virtual Desktop // 2020-0927 SatoxITS { -->
11134 <style>
11135 .VirtualSpace {
11136     z-index:0;
11137     width:1280px !important; xheight:720px !important;
11138     width:5120px; height:2880px;
11139     border-width:0px;
11140     xbackground-color:rgba(32,32,160,0.8);
11141     xbackground-image:url('WD-WallPaper03.png');
11142     xbackground-size:100% 100%;
11143     color:#22a;font-family:Georgia;font-size:10pt;
11144     xoverflow:scroll;
11145 }
11146 .VirtualGrid {
11147     z-index:0;
11148     position:absolute;
11149     width:800px; height:500px;

```

```

11151 border:1px inset #fff;
11152 color:rgba(192,255,192,0.8);
11153 font-family:Georgia, Courier New;
11154 text-align:right;
11155 vertical-align:middle;
11156 font-size:200px;
11157 text-shadow:4px 4px #ccf;
11158 }
11159 .WD_GridScroll {
11160 z-index:100000;
11161 background-color:rgba(200,200,200,0.1);
11162 }
11163 .VirtualDesktop {
11164 z-index:0;
11165 position:relative;
11166 resize:both !important;
11167 overflow:scroll;
11168 display:block;
11169 min-width:120px !important; min-height:60px !important;
11170 width:800px;
11171 height:500px;
11172 border:10px inset #228;
11173 border-width:30px; border-radius:20px;
11174 background-image:url("WD-WallPaper03.png");
11175 background-size:100% 100%;
11176 color:#22a;font-family:Georgia;font-size:10pt;
11177 }
11178 .comment {
11179 // overflow-scroll seems to bound childrens' view in the element span
11180 // specifying overflow seems fix the position of the element
11181 }
11182 .VirtualBrowserSpan {
11183 z-index:10;
11184 xxxborder:0.5px dashed #fff !important;
11185 border-color:rgba(255,255,255,0.5) !important;
11186 position:relative;
11187 left:100px;
11188 top:100px;
11189 display:block;
11190 resize:both !important;
11191 width:540px;
11192 height:540px;
11193 min-width:40px !important; min-height:20px !important;
11194 max-width:5120px !important; max-height:2880px !important;
11195 background-color:rgba(255,200,255,0.1);
11196 xoverflow:scroll;
11197 }
11198 .xVirtualBrowserLocationBar:focus {
11199 color:#f00;
11200 background-color:rgba(255,128,128,0.2);
11201 }
11202 .xVirtualBrowserLocationBar:active {
11203 color:#f00;
11204 background-color:rgba(128,255,128,0.2);
11205 }
11206 .a.VirtualBrowserLocation {
11207 color:#ccc !important;
11208 text-decoration:none !important;
11209 }
11210 .a.VirtualBrowserLocation:hover {
11211 color:#fff !important;
11212 text-decoration:underline;
11213 }
11214 .VirtualBrowserLocationBar {
11215 position:absolute;
11216 z-index:100000;
11217 display:block;
11218 width:400px;
11219 height:20px;
11220 padding-left:2px;
11221 line-height:1.1;
11222 vertical-align:middle;
11223 font-size:14px;
11224 color:#fff;
11225 background-color:rgba(128,128,128,0.2);
11226 font-family:Georgia;
11227 }
11228 .VirtualBrowserCommandBar {
11229 position:absolute;
11230 z-index:200000;
11231 xxxdisplay:inline;
11232 display:block;
11233 width:60px;
11234 height:20px;
11235 line-height:1.1;
11236 vertical-align:middle;
11237 font-size:14px;
11238 color:#f64;
11239 background-color:rgba(128,128,128,0.1);
11240 font-family:Georgia;
11241 text-align:left;
11242 left:404px;
11243 }
11244 .VirtualBrowserFrame {
11245 xposition:relative;
11246 position:absolute;
11247 xxxdisplay:inline;
11248 display:block;
11249 z-index:10;
11250 resize:both !important;
11251 width:480px; height:240px;
11252 min-width:60px; min-height:30px;
11253 max-width:5120px; max-height:2880px;
11254 border-radius:6px;
11255 background-color:rgba(255,255,255,0.9);
11256 border-top:20px solid;
11257 border-right:4px solid;
11258 border-bottom:10px solid;
11259 }
11260 .WinFavicon {
11261 width:16px;
11262 height:16px;
11263 margin:1px;
11264 margin-right:3px;
11265 vertical-align:middle;
11266 background-color:rgba(255,255,255,1.0);
11267 }
11268 .VirtualDesktopMenuBar {
11269 xposition:absolute;
11270 color:#ff7;
11271 font-size:7pt;
11272 text-align:right;
11273 padding-right:4px;
11274 background-color:rgba(128,128,128,0.7);
11275 }
11276 .VirtualDesktopCalendar {
11277 color:#ff7;
11278 font-size:22pt;
11279 text-align:right;
11280 padding-right:4px;
11281 xbackground-color:rgba(255,255,255,0.2);
11282 }
11283 .xxxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
11284 display:inline !important; font-size:10pt !important; padding:1px !important;
11285 }
11286 .WD_Config {
11287 display:inline !important;
11288 padding:2px !important;
11289 font-size:10pt !important;
11290 width:60pt !important;
11291 height:12pt !important;
11292 line-height:1.0pt !important;
11293 height:15pt !important;
11294 }
11295 .WD_Button {
11296 display:inline !important;
11297 padding:2px !important;
11298 color:#fff !important;
11299 background-color:#228 !important;
11300 font-size:10pt !important;
11301 width:60pt !important;
11302 height:12pt !important;
11303 line-height:1.0pt !important;
11304 height:16pt !important;
11305 border:2px inset #44a !important;
11306 }
11307 .WD_Href {
11308 display:inline !important;
11309 padding:2px !important;
11310 font-size:9pt !important;
11311 width:120pt !important;
11312 height:12pt !important;
11313 line-height:1.0pt !important;
11314 height:15pt !important;
11315 }
11316 .LiveHtmlCodeviewText {
11317 font-size:10pt;
11318 font-family:Courier New;
11319 xwhite-space:pre;
11320 }
11321 }
11322 }
11323 .WD_Panel {
11324 x-index:100 !important;
11325 color:#000 !important;
11326 margin-left:25px !important;
11327 width:800px !important;

```



```

11328 padding:4px !important;
11329 border:1px solid #888 !important;
11330 border-radius:6px !important;
11331 background-color:rgba(220,220,220,0.9) !important;
11332 font-size:9pt;
11333 font-family:Courier New;
11334 }
11335 .WD_Help {
11336 font-size:10pt !important;
11337 font-family:Courier New;
11338 line-height:1.2 !important;
11339 color:#000 !important;
11340 width:100% !important;
11341 background-color:rgba(240,240,255,0.8) !important;
11342 }
11343
11344 .WB_Zoom {
11345 }
11346 }
11347 </style>
11348 <menu>
11349 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
11350 g ... grid on/off<br>
11351 i ... zoom in<br>
11352 o ... zoom out<br>
11353 s ... save current scope<br>
11354 r ... restore saved scope<br>
11355 </span>
11356 </menu>
11357 <div class="WD_Panel" draggable="true">
11358 <p>!-- should be on the frame of the WD -->
11359 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
11360 <input id="WD_Width_1" class="WD_Config" type="text">
11361 x <input id="WD_Height_1" class="WD_Config" type="text">
11362 wall-paper: <a href="WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
11363 </p>
11364 Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
11365 <input id="WS_1_Width" class="WD_Config" type="text">
11366 x <input id="WS_1_Height" class="WD_Config" type="text">
11367 </p>
11368 <p>
11369 Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
11370 <input id="WD_Zoom_1_XY" class="WD_Config" type="text" value="1.0">
11371 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
11372 <input id="WD_Zoom_1_OUTP" class="WD_Button" type="button" value="ZoomOut">
11373 <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
11374 </p>
11375 <p>
11376 Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
11377 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
11378 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
11379 shift+wheel for horizontal scroll
11380 </p>
11381 <p>
11382 ScopeX <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
11383 <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
11384 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
11385 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
11386 </p>
11387 <p>
11388 ScopeY <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
11389 <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
11390 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
11391 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
11392 </p>
11393 <p>
11394 ScopeZ <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
11395 <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
11396 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
11397 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
11398 </p>
11399 <p>
11400 Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
11401 </p>
11402 <p>
11403 Overflo <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
11404 "scroll" imprisons windows inside the display
11405 </p>
11406 </div>
11407
11408 <div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
11409 <div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
11410 <div id="VirtualDesktop_1_Clock" class="VirtualDesktopClock"></div>
11411 <div id="VirtualDesktop_1_Calendar" class="VirtualDesktopCalendar">00:00</div>
11412 <div align="right"><div id="VirtualSpace_1" class="VirtualSpace">
11413 <div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11414 <div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
11415 <div id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</div>
11416 <div id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></div>
11417 </div>
11418 <div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11419 <div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
11420 <div id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</div>
11421 <div id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></div>
11422 </div>
11423 <div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11424 <div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
11425 <div id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</div>
11426 <div id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></div>
11427 </div>
11428 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
11429 </div>
11430 </div>
11431
11432 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
11433 <span id="VirtualDesktopCodeView"></span>
11434 <script id="VirtualDesktopCodeView">
11435 function vd_showHtmlCode(){
11436 codespan = document.getElementById("VirtualDesktopCodeSpan");
11437 showHtmlCode(VirtualDesktopCodeView.codespan);
11438 VirtualDesktopCodeView.setAttribute("class", "LiveHtmlCodeViewText");
11439 }
11440 DestroyEventSharingCodeView = function(){
11441 VirtualDesktopCodeView.innerHTML = "";
11442 }
11443
11444 function wdlog(log){
11445 if (GJ_Channel != null ){
11446 GJ_SendMessage('WD '+log);
11447 }
11448 console.log(log);
11449 }
11450
11451 var topMostWin = 10000;
11452 function onEnterWin(e){
11453 t = e.target;
11454 oindex = t.style.zIndex;
11455 //if( oindex == '' ) oindex = 0;
11456 //t.saved_zIndex = oindex;
11457 //t.style.zIndex = 10000;
11458 topMostWin += 1;
11459 t.style.zIndex = topMostWin;
11460 nindex = t.style.zIndex;
11461 wdlog("Enter "+t.id+" ('+oindex+'->'+nindex+')");
11462 e.stopPropagation();
11463 e.preventDefault();
11464 }
11465 function onClickWin(e){ // can detect click on the thick border? t = e.target;
11466 oindex = t.style.zIndex;
11467 topMostWin = 1;
11468 t.style.zIndex = topMostWin;
11469 nindex = t.style.zIndex;
11470 wdlog("Click "+t.id+" ('+oindex+'->'+nindex+')");
11471 //e.stopPropagation();
11472 //e.preventDefault();
11473 }
11474 function onLeaveWin(e){
11475 t = e.target;
11476 //oindex = t.style.zIndex;
11477 //nindex = t.saved_zIndex;
11478 //t.style.zIndex = oindex;
11479 //wdlog("Leave "+e.target.id+" ('+oindex+'->'+nindex+')");
11480 e.stopPropagation();
11481 e.preventDefault();
11482 }
11483
11484 var WinDragstartX; // event
11485 var WinDragstartY;
11486 var WinDragstartTX; // target
11487 var WinDragstartTY;
11488
11489 function onWinDragstart(e){
11490 WinDragstartX = e.x;
11491 WinDragstartY = e.y;
11492 t = e.target;
11493 //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11494 //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11495 if( t.style.left == '' ){
11496 WinDragstartTX = x0 = 0;

```

```

11505     t.style.left = '0px';
11506 }else{
11507     //WinDragstartTX = x0 = Number(t.style.left);
11508     WinDragstartTX = x0 = parseInt(t.style.left);
11509 }
11510 if( t.style.top == '' ){
11511     WinDragstartTY = y0 = 0;
11512     t.style.top = '0px';
11513 }else{
11514     //WinDragstartTY = y0 = Number(t.style.top);
11515     WinDragstartTY = y0 = parseInt(t.style.top);
11516 }
11517 if( true ){ // to be undo
11518     t.wasATX = WinDragstartTX;
11519     t.wasATY = WinDragstartTY;
11520 }
11521 wlog('DragSTA #' + t.id
11522 + ' event('+e.x+', '+e.y+')'
11523 + ' position=' + t.style.position
11524 + ' style left,top('+t.style.left+', '+t.style.top+')'
11525 );
11526 e.stopPropagation();
11527 //e.preventDefault();
11528 return true;
11529 }
11530 function onWinDragEvent(wh,e,set,dolog){
11531     t = e.target;
11532     dx = e.x - WinDragstartX;
11533     dy = e.y - WinDragstartY;
11534     nx = WinDragstartTX + dx;
11535     ny = WinDragstartTY + dy;
11536     log = ' Drag '+wh+' #' + t.id
11537 + ' event0('+WinDragstartX+', '+WinDragstartY+')'
11538 + ' event('+e.x+', '+e.y+')'
11539 + ' dif('+dx+', '+dy+')'
11540 + ' (' + nx + ', ' + ny + ')';
11541 + ' (' + t.style.left + ', ' + t.style.top + ')';
11542 + ' wasAT(' + t.wasATX + ', ' + t.wasATY + ')';
11543 }
11544 if( e.x != 0 || e.y != 0 ){
11545     if( set == true ){
11546         //t.style.x = nx + 'px'; // not effective
11547         //t.style.y = ny + 'px'; // not effective
11548         t.style.left = nx + 'px';
11549         t.style.top = ny + 'px';
11550         log += ' Set';
11551     }else{
11552         log += ' NotSet';
11553         if( !dolog ){
11554             log = '';
11555         }
11556     }
11557 }else{
11558     log = ' What?'; // the type is event start?
11559     if( !dolog ){
11560         log = '';
11561     }
11562 }
11563 if( and(dolog, log != '' )){
11564     wlog(log);
11565 }
11566 if( true ){
11567     // should be propagated to parent in FireFox ?
11568     e.stopPropagation();
11569 }
11570 e.preventDefault();
11571 return false;
11572 }
11573 function onWinDrag(e){
11574     return onWinDragEvent('Ing',e,true,false);
11575 }
11576 function onWinDragend(e){
11577     return onWinDragEvent('End',e,false,true);
11578 }
11579 function onWinDragexit(e){
11580     return onWinDragEvent('Exit',e,false,true);
11581 }
11582 function onWinDragover(e){
11583     return onWinDragEvent('Over',e,false,true);
11584 }
11585 function onWinDragenter(e){
11586     return onWinDragEvent('Enter',e,false,true);
11587 }
11588 function onWinDragleave(e){
11589     return onWinDragEvent('Leave',e,false,true);
11590 }
11591 function onWinDragdrop(e){
11592     return onWinDragEvent('Drop',e,false,true);
11593 }
11594 function onFaviconChange(e){
11595     wlog('--Favicon #' + e.target.id + ' href='+e.details.href);
11596 }
11597 }
11598 var savedSuppressGJShell = false;
11599 function stopGShell(e){
11600     //wlog('enter Gsh STOP');
11601     savedSuppressGShell = SuppressGJShell;
11602     SuppressGJShell = true;
11603     e.stopPropagation();
11604     e.preventDefault();
11605 }
11606 function contGShell(e){
11607     //wlog('leave Gsh STOP');
11608     SuppressGJShell = savedSuppressGJShell;
11609     e.stopPropagation();
11610     e.preventDefault();
11611 }
11612 function WD_onKeydown(e){
11613     keycode = e.code;
11614     console.log('Keydown #' + e.target.id + ' '+keycode);
11615     if( keycode == 'KeyG' ){
11616         WD_setGrid1(WD_Grid_1);
11617     }else
11618     if( keycode == 'KeyI' ){
11619         WD_doZoomIN();
11620     }else
11621     if( keycode == 'KeyO' ){
11622         WD_doZoomOUT();
11623     }else
11624     if( keycode == 'KeyR' ){
11625         WD_RestoreScope(null);
11626     }else
11627     if( keycode == 'KeyS' ){
11628         WD_SaveScope(null);
11629     }
11630     e.stopPropagation();
11631     e.preventDefault();
11632 }
11633 function WD_onKeyUp(e){
11634     e.stopPropagation();
11635     e.preventDefault();
11636 }
11637 function WD_EventSetup1(){
11638     WirtualDesktop_1.addEventListener('keydown', e => { WD_onKeydown(e); });
11639     WirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeydown(e); });
11640     WD_Help_1.addEventListener('keydown', e => { WD_onKeydown(e); });
11641     WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
11642 }
11643 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
11644     function WirtualBrowserCommand(e,s,l,cmd,f){
11645         command = cmd.innerHTML;
11646         if( command == "Reload" ){
11647             href_id = e.target.href_id;
11648             d = document.getElementById(href_id);
11649             wlog('href_tag#' + href_id + '\n elem#' + href_id + '\n href='+d);
11650             url = d.innerHTML;
11651             wlog('---- Load href_tag#' + href_id + '\n elem#' + href_id + '\n href='+d
11652 + '\n url='+url);
11653             wlog('---- Load target #' + f.id + ' with url='+url;
11654             f.src = url;
11655         }else{
11656             alert('unknown command'+command+' '+e.target.id+', '+l.id+', '+f.id);
11657         }
11658     }
11659     function onKeyDown(e){
11660         if( e.code == 'Enter' ){
11661             e.stopPropagation();
11662             e.preventDefault();
11663         }
11664     }
11665     function onKeyUp(e){
11666         if( e.code == 'Enter' ){
11667             e.stopPropagation();
11668             e.preventDefault();
11669             // should reload immediately ?
11670         }
11671     }
11672 }
11673 if( false ){
11674     wlog('start settle WirtualBrowser url='+u + '\n'
11675 + ' id#' + s.id + '\n'
11676 + ' width=' + s.style.width + '\n'
11677 + ' height=' + s.style.height
11678 );
11679 }
11680 // very important for WordPress ??
11681 s.style.width = f.style.width = 501; // for WordPress ...??

```

```

11682 s.style.height = f.style.height = 271; // for WordPress ...??
11683 if( false ){
11684     wdlod('midway settle VirtualBrowser url='+u +'\'\'\'
11685         + 'id=' + s.id + '\n'
11686         + 'width=' + s.style.width + '\n'
11687         + 'height=' + s.style.height
11688     );
11689 }
11690 s.width = 502; // for WordPress ...??
11691 s.height = 272; // for WordPress ...??
11692 if( false ){
11693     wdlod('midway-2 settle VirtualBrowser url='+u +'\'\'\'
11694         + 'id=' + s.id + '\n'
11695         + 'span-width=' + s.width + '\n'
11696         + 'span-height=' + s.height
11697     );
11698 }
11699
11700 s.style.width = w + 'px';
11701 s.style.height = h + 'px';
11702 f.style.width = w + 'px';
11703 f.style.height = h + 'px';
11704 //f.style.setProperty('-webkit-transform','scale('+scale+')');
11705 f.style.setProperty('transform','scale('+scale+')');
11706
11707 //wdlod('---x1-- u='+u+' width s='+s.style.width+',f='+f.style.width);
11708 //wdlod('---x2-- u='+u+' width s='+s.style.width+',f='+f.style.width);
11709 s.setAttribute('draggable', 'true');
11710 f.setAttribute('draggable', 'false'); // why necessary?
11711 l.setAttribute('draggable', 'false'); // why necessary?
11712 cmd.setAttribute('draggable', 'false'); // why necessary?
11713 s.addEventListener('dragstart', e => { onWinDragstart(e); });
11714 s.addEventListener('drag', e => { onWinDrag(e); });
11715 s.addEventListener('exit', e => { onWinDragexit(e); });
11716 s.addEventListener('dragend', e => { onWinDragend(e); });
11717 s.addEventListener('dragexit', e => { onWinDragexit(e); });
11718 s.addEventListener('dragenter', e => { onWinDragenter(e); });
11719 s.addEventListener('dragover', e => { onWinDragover(e); });
11720 s.addEventListener('dragleave', e => { onWinDragleave(e); });
11721 s.addEventListener('drop', e => { onWinDragdrop(e); });
11722
11723 s.addEventListener('mouseenter', e => { onEnterWin(e); });
11724 s.addEventListener('mouseleave', e => { onLeaveWin(e); });
11725
11726 if( false ){
11727     s.style.position = "absolute";
11728     s.style.x = x+'px';
11729     s.style.left = x+'px';
11730     s.style.y = y+'px';
11731     s.style.top = y+'px';
11732 }else{
11733     s.style.setProperty('position','absolute','important');
11734     s.style.setProperty('x','x+'px','important');
11735     s.style.setProperty('left','x+'px','important');
11736     s.style.setProperty('y','y+'px','important');
11737     s.style.setProperty('top','y+'px','important');
11738 }
11739
11740 favicon = './favicon.ico';
11741 uv1 = s.split('/');
11742 if( 2 <= uv1.length ){
11743     uv2 = uv1[1].split('/');
11744     if( 2 <= uv2.length ){
11745         if( uv1[0] != 'file:' ){
11746             //favicon = 'file://' + uv2.slice(0,uv2.length-1).join('/');
11747             // + '/favicon.ico';
11748             favicon = './favicon.ico';
11749         }else{
11750             favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
11751         }
11752     }
11753 }
11754 //wdlod('---- favicon-url='+favicon);
11755 href_id = l.id + ' href';
11756 l.innerHTML = ''+u+'</a>';
11758 //l.addEventListener('click', e => { onClickWin(e); });
11759 l.addEventListener('mouseenter', e => { stopGShell(e); });
11760 l.addEventListener('mouseleave', e => { contGShell(e); });
11761 l.addEventListener('keydown', e => { onKeyDown(e); });
11762 l.addEventListener('keyup', e => { onKeyUp(e); });
11763
11764 cmd.href_id = href_id;
11765 wdlod(' (0)cmd#'+cmd.id);
11766 wdlod(' (1)href_id#'+href_id);
11767 wdlod(' (2)href_id#'+cmd.href_id);
11768 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
11769
11770 f.style.borderColor = c;
11771 f.src = u;
11772 //f.addEventListener('mouseenter', e => { onEnterWin(e); });
11773 //f.addEventListener('mouseleave', e => { onLeaveWin(e); });
11774
11775 //s.addEventListener('click', e => { onClickWin(e); });
11776 //f.addEventListener('click', e => { wdlod('click wbl'); });
11777 f.addEventListener('mozbrowsericonchange', onFaviconChange);
11778
11779 wdlod('done settle VirtualBrowser url='+u +'\'\'\'
11780     + 'id=' + s.id + ' '
11781     + 'width=' + s.style.width + ' '
11782     + 'height=' + s.style.height + ' '
11783     + 'cmd=' + cmd.id
11784 );
11785 }
11786
11787 function WD_EventSetup2(){
11788     dt = VirtualDesktop;
11789     dt.style.width = "800px";
11790     dt.style.height = "500px";
11791     dt.addEventListener('dragstart', e => { onWinDragstart(e); });
11792     dt.addEventListener('drag', e => { onWinDrag(e); });
11793     dt.addEventListener('exit', e => { onWinDragexit(e); });
11794 }
11795
11796 function GRonClick(){
11797     WD_SaveScope(null); // should be push
11798     t = event.target; // should be push
11799     x = t.getAttribute('data-leftx');
11800     y = t.getAttribute('data-topy');
11801     zoom = WD_Zoom_1_XY.value;
11802     x *= zoom;
11803     y *= zoom;
11804     WD_doScrollXY(event,x,y);
11805     //alert('scroll #' + t.id + ' x=' + x + ', y=' + y);
11806 }
11807
11808 function WD_setGrid(e){
11809     t = e.target;
11810     WD_setGrid1(t);
11811 }
11812 function WD_setGrid1(t){
11813     //ds = VirtualDesktop_1_GridPlane; // should be VirtualSpace_1
11814     ds = VirtualDesktop_1_GridPlane;
11815     if( t.value == 'GridOn' ){
11816         for( col = 0; col < 16; col++ ){
11817             for( row = 0; row < 16; row++ ){
11818                 gl = document.createElement('span');
11819                 gl.setAttribute('class','VirtualGrid');
11820                 leftx = col * 800;
11821                 topy = row * 500;
11822                 label = col + ' ' + row
11823                     + 'id="'+s.id+'span'+
11824                     + 'contenteditable=false onclick="GRonClick()" '
11825                     + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' '
11826                     + '>';
11827                 + gl + '</span>';
11828                 console.log('grid '+label);
11829                 gl.innerHTML = label;
11830                 gl.position = 'relative';
11831                 gl.leftx = leftx;
11832                 gl.topy = topy;
11833                 gl.style.left = gl.leftx + 'px';
11834                 gl.style.top = gl.topy + 'px';
11835                 if( col % 2 == row % 2 ){
11836                     gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
11837                 }
11838                 ds.appendChild(gl);
11839             }
11840         }
11841         t.value = 'GridOff';
11842     }else{
11843         ds.innerHTML = '';
11844         t.value = 'GridOn';
11845     }
11846 }
11847
11848 var sav_scrollLeft;
11849 var sav_scrollTop;
11850 var sav_nscale;
11851 function WD_SaveScope(e){
11852     sav_scrollLeft = WD_Left_1.value;
11853     sav_scrollTop = WD_Top_1.value;
11854     sav_nscale = WD_Zoom_1_XY.value;
11855     //console.log('Saved zoom='+sav_nscale+','+sav_nscale);
11856 }
11857 function WD_RestoreScope(e){
11858     WD_Zoom_1_XY.value = sav_nscale;

```

```

11859   WD_doZoom();
11860
11861   WD_Left_1.value = sav_scrollLeft;
11862   WD_Top_1.value = sav_scrollTop;
11863   WD_doScroll(null);
11864 }
11865 function ignoreEvent(e){
11866   e.stopPropagation();
11867   //e.preventDefault();
11868 }
11869 function zoomMag(){
11870   return WD_Zoom_1_MAG.value;
11871 }
11872 function WD_EventSetup3(){
11873   WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });
11874   WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
11875   WD_Top_1_Save.addEventListener('click', e => { WD_RestoreScope(e); });
11876   WD_Width_1.value = dt.style.width;
11877   WD_Width_1.addEventListener('keydown', ignoreEvent);
11878   WD_Width_1.addEventListener('keyup', ignoreEvent);
11879   WD_Height_1.value = dt.style.height;
11880   WD_Height_1.addEventListener('keydown', ignoreEvent);
11881   WD_Height_1.addEventListener('keyup', ignoreEvent);
11882   WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
11883   WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
11884 }
11885 function escale(e,oscale,nscale){
11886   e.style.setProperty('transform','scale('+nscale+')');
11887   nscale = oscale / nscale;
11888   w = parseInt(e.style.width);
11889   h = parseInt(e.style.height);
11890   w = w * nscale; //(oscale/nscale);
11891   h = h * nscale; //(oscale/nscale);
11892   e.style.width = w + 'px';
11893   e.style.height = h + 'px';
11894 }
11895 function scaleWD(ds,oscale,nscale){
11896   if( true ){
11897     escale(WVirtualBrowser_1,oscale,nscale);
11898     escale(WVirtualBrowser_1_Location,oscale,nscale);
11899     escale(WVirtualBrowser_1_Command,oscale,nscale);
11900     escale(WVirtualBrowser_1_Frame,oscale,nscale);
11901   }
11902   if( true ){
11903     escale(WVirtualBrowser_2,oscale,nscale);
11904     escale(WVirtualBrowser_2_Location,oscale,nscale);
11905     escale(WVirtualBrowser_2_Command,oscale,nscale);
11906     escale(WVirtualBrowser_2_Frame,oscale,nscale);
11907   }
11908   if( true ){
11909     escale(WVirtualBrowser_3,oscale,nscale);
11910     escale(WVirtualBrowser_3_Location,oscale,nscale);
11911     escale(WVirtualBrowser_3_Command,oscale,nscale);
11912     escale(WVirtualBrowser_3_Frame,oscale,nscale);
11913   }
11914 }
11915 function WD_doZoom(){
11916   ds = WVirtualDesktop_1_Content; // should be VirtualSpace_1
11917   oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
11918   nscale = WD_Zoom_1_XY.value;
11919   ds.style.zoom = nscale;
11920   WD_Zoom_1_XY.value = ds.style.zoom;
11921   WD_Zoom_1_XY.ovalue = ds.style.zoom;
11922   scaleWD(ds,oscale,nscale);
11923 }
11924 function WD_EventSetup4(){
11925   WD_Zoom_1.addEventListener('click',WD_doZoom);
11926   WD_Zoom_1_XY.addEventListener('keydown', ignoreEvent);
11927   WD_Zoom_1_XY.addEventListener('keyup', ignoreEvent);
11928 }
11929 function WD_doZoomOUT(){
11930   ds = WVirtualDesktop_1_Content; // should be VirtualSpace_1
11931   oscale = WD_Zoom_1_XY.value;
11932   if( oscale == 0 || oscale == '' ){
11933     oscale = 1;
11934   }
11935   nscale = oscale / zoomMag();
11936   ds.style.zoom = nscale;
11937   WD_Zoom_1_XY.value = ds.style.zoom;
11938   WD_Zoom_1_XY.ovalue = ds.style.zoom;
11939   scaleWD(ds,oscale,nscale);
11940 }
11941 function WD_doZoomIN(){
11942   ds = WVirtualDesktop_1_Content; // should be VirtualSpace_1
11943   oscale = WD_Zoom_1_XY.value;
11944   if( oscale == 0 || oscale == '' ){
11945     oscale = 1;
11946   }
11947   nscale = oscale * zoomMag();
11948   if( 4 < nscale ){
11949     alert('maybe too large, zoom='+nscale);
11950     return;
11951   }
11952   ds.style.zoom = nscale;
11953   WD_Zoom_1_XY.value = ds.style.zoom;
11954   WD_Zoom_1_XY.ovalue = ds.style.zoom;
11955   scaleWD(ds,oscale,nscale);
11956 }
11957 function WD_EventSetup5(){
11958   WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
11959   WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
11960 }
11961 function WD_doResize(e){
11962   dt = WVirtualDesktop_1;
11963   dt.style.width = WD_Width_1.value;
11964   dt.style.height = WD_Height_1.value;
11965   WD_Width_1.value = dt.style.width;
11966   WD_Height_1.value = dt.style.height;
11967 }
11968 WD_Resize_1.addEventListener('click',e => { WD_doResize(e); });
11969 }
11970 function WD_doRSResize(e){
11971   //alert('Resize Space');
11972   ds = WVirtualDesktop_1_Content; // should be VirtualSpace_1
11973   ds.style.width = WS_1_Width.value;
11974   ds.style.height = WS_1_Height.value;
11975   WS_1_Width.value = ds.style.width;
11976   WS_1_Height.value = ds.style.height;
11977 }
11978 function WD_EventSetup6(){
11979   ds = WVirtualDesktop_1_Content; // should be VirtualSpace_1
11980   ds.style.width = '5120px';
11981   ds.style.height = '2880px';
11982   WS_1_Width.value = ds.style.width;
11983   WS_1_Height.value = ds.style.height;
11984   WS_1_Width.addEventListener('keydown', ignoreEvent);
11985   WS_1_Height.addEventListener('keydown', ignoreEvent);
11986   WS_1_Width.addEventListener('keyup', ignoreEvent);
11987   WS_1_Height.addEventListener('keyup', ignoreEvent);
11988   WS_1_Width.addEventListener('keydown', ignoreEvent);
11989   WS_1_Height.addEventListener('keyup', ignoreEvent);
11990   WD_Resize_1.addEventListener('click',e => { WD_doRSResize(e); });
11991 }
11992 function WD_doScrollXY(e,sleft,stop){
11993   dt = WVirtualDesktop_1;
11994   dt.scrollLeft = sleft;
11995   dt.scrollTop = stop;
11996   WD_Left_1.value = dt.scrollLeft;
11997   WD_Top_1.value = dt.scrollTop;
11998   console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
11999 }
12000 function WD_doScroll(e){
12001   //dt = WVirtualDesktop_1_Content;
12002   dt = WVirtualDesktop_1;
12003   sleft = parseInt(WD_Left_1.value);
12004   stop = parseInt(WD_Top_1.value);
12005   dt.scrollLeft = sleft;
12006   dt.scrollTop = stop;
12007   WD_Left_1.value = dt.scrollLeft;
12008   WD_Top_1.value = dt.scrollTop;
12009   console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
12010 }
12011 function showScrollPosition(){
12012   if( false )
12013     console.log(
12014       'wtop=' + WVirtualDesktop_1.style.top + ', ' +
12015       'wxs=' + WVirtualDesktop_1.style.y + ', ' +
12016       'wss=' + WVirtualDesktop_1.scrollTop + ', ' +
12017       'wops=' + WVirtualDesktop_1_Content.style.top + ', ' +
12018       'wdxs=' + WVirtualDesktop_1_Content.style.y + ', ' +
12019       'wds=' + WVirtualDesktop_1_Content.scrollTop + ', ' +
12020     );
12021   WD_Left_1.value = WVirtualDesktop_1.scrollLeft;
12022   WD_Top_1.value = WVirtualDesktop_1.scrollTop;
12023 }
12024 function WD_EventSetup7(){
12025   WD_Scroll_1.addEventListener('click',e => { WD_doScroll(e); });
12026   WD_Left_1.addEventListener('keydown', ignoreEvent);
12027   WD_Left_1.addEventListener('keyup', ignoreEvent);
12028   WD_Top_1.addEventListener('keydown', ignoreEvent);
12029   WD_Top_1.addEventListener('keyup', ignoreEvent);
12030 }
12031 function WD_EventSetup8(){
12032   WVirtualDesktop_1.addEventListener('scroll',showScrollPosition);
12033   WVirtualDesktop_1_Content.addEventListener('scroll',showScrollPosition);
12034 }

```

```

12036
12037 if( false ){
12038   w = 1000 + 'px';
12039   dt.style.width = w;
12040   dt.style.height = "300px";
12041   dt.style.resize = "both";
12042   dt.style.borderWidth = 50 + 'px';
12043   dt.style.borderRadius = 25 + 'px';
12044   console.log("--2----- #'+dt.id+' style="+ dt.style);
12045   console.log("----- #'+dt.id+' width="+ dt.style.width);
12046   console.log("----- #'+dt.id+' left="+ dt.style.left);
12047   console.log("----- #'+dt.id+' border="+ dt.style.border);
12048 }
12049 function onDResize(e){
12050   dt = e.target;
12051   h = parseInt(dt.style.height);
12052   dt.style.borderWidth = (h * 0.075) + 'px';
12053   console.log("----- borderWidth="+ dt.style.borderWidth);
12054 }
12055
12056 VirtualDesktopDetails.addEventListener('toggle',VirtualDesktop_init);
12057 function VirtualDesktop_init(){
12058   if( !VirtualDesktopDetails.open ){
12059     return;
12060   }
12061   //GJ Join();
12062   VirtualDesktop_1.addEventListener('resize', e => { onDResize(e); });
12063   //console.log("----- #'+dt.id+' width="+ dt.style.width);
12064   // '+ ' borderWidth="+dt.style.getProperty('border-width');
12065
12066   settleWin(
12067     VirtualBrowser_1,
12068     VirtualBrowser_1_Location,
12069     VirtualBrowser_1_Command,
12070     VirtualBrowser_1_Frame,
12071     document.URL,
12072     500,280,50,20,'#262',1.0);
12073   settleWin(
12074     VirtualBrowser_2,
12075     VirtualBrowser_2_Location,
12076     VirtualBrowser_2_Command,
12077     VirtualBrowser_2_Frame,
12078     'https://its-morG.jp/ja_jp/',
12079     500,280,150,100,'#448',1.0);
12080   settleWin(
12081     VirtualBrowser_3,
12082     VirtualBrowser_3_Location,
12083     VirtualBrowser_3_Command,
12084     VirtualBrowser_3_Frame,
12085     '../gshell/gsh.go.html',
12086     //'http://gshell.org/gshell/gsh.go.html',
12087     'https://golang.org',
12088     500,280,250,180,'#444',1.0);
12089     //1000,720,0,0,'#444',0.125);
12090     //1200,720,-100,-50,'#444',0.4);
12091     function WD_ClockUpdate(e){
12092       VirtualDesktop_1_Clock.innerHTML = DateShort();
12093       VirtualDesktop_1_Calender.innerHTML = DateHourMin();
12094     }
12095     window.setInterval(WD_ClockUpdate,500);
12096
12097     WD_EventSetup1();
12098     WD_EventSetup2();
12099     WD_EventSetup3();
12100     WD_EventSetup4();
12101     WD_EventSetup5();
12102     WD_EventSetup6();
12103     WD_EventSetup7();
12104     WD_EventSetup8();
12105 }
12106 //VirtualDesktop_init();
12107
12108 Destroy_VirtualDesktop = function(){
12109   VirtualDesktop_1.style = "";
12110
12111   VirtualBrowser_1.removeAttribute('style');
12112   VirtualBrowser_1_Location.innerHTML = '';
12113   VirtualBrowser_1_Frame.removeAttribute('src');
12114   VirtualBrowser_1_Frame.removeAttribute('style');
12115   VirtualBrowser_1_Frame.style="";
12116
12117   VirtualBrowser_2.removeAttribute('style');
12118   VirtualBrowser_2_Location.innerHTML = '';
12119   VirtualBrowser_2_Frame.removeAttribute('src');
12120   VirtualBrowser_2_Frame.style="";
12121
12122   VirtualBrowser_3.removeAttribute('style');
12123   VirtualBrowser_3_Location.innerHTML = '';
12124   VirtualBrowser_3_Frame.removeAttribute('src');
12125   VirtualBrowser_3_Frame.style="";
12126
12127   GJFactory_1.style = "";
12128   iframe0.style = "";
12129   VirtualDesktop_1.style = "";
12130 }
12131
12132 </script>
12133 </-- VirtualDesktop } -->
12134 </details>
12135 *// /</span>
12136
12137 </-- ===== Work { ===== -->
12138 <span id="SBSidebar_WorkCodeSpan">
12139 /*
12140 <details><summary>SBSidebar</summary>
12141 </-- ===== SBSidebar // 2020-0928 SatoxITS { -->
12142 <h2>SBSidebar</h2>
12143 <input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot"
12144 <input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature"
12145 <input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode"
12146 <span id="SBSidebar_WorkCodeView"></span>
12147 <script id="SBSidebar_WorkCodeView">
12148 function SBSidebar_openWorkCodeView(){
12149   function SBSidebar_showWorkCode(){
12150     showHtmlCode(SBSidebar_WorkCodeView,SBSidebar_WorkCodeSpan);
12151   }
12152   SBSidebar_WorkCodeViewOpen.addEventListener('click',SBSidebar_showWorkCode);
12153 }
12154 SBSidebar_openWorkCodeView(); // should be invoked by an event
12155
12156 console.log("-- SbSlider // 2020-1006-01 SatoxITS --");
12157 function SetSidebar(){
12158   sidebar = document.getElementById('secondary');
12159   console.log("primary='primary'+ 'secondary'+ sidebar+' +main'+main' ");
12160   wrap = sidebar.parentNode;
12161   console.log("SbSlider parent is '+wrap+', #' +wrap.id+' .'+wrap.class);
12162   //wrap = wrap.parentNode;
12163   //console.log("SbSlider parent is '+wrap+', #' +wrap.id+' .'+wrap.class);
12164   //wrap = wrap.parentNode;
12165   //console.log("SbSlider parent is '+wrap+', #' +wrap.id+' .'+wrap.class);
12166   //nsb = sidebar.cloneNode();
12167   nsb = sidebar;
12168   nsb.style.width = '100%';
12169   slider = document.createElement('div');
12170   slider.id = 'SbSlider';
12171   slider.appendChild(nsb);
12172   slider.setAttribute('class','SbSlider');
12173   nsb.style.position = 'relative';
12174   slider.style.position = 'fixed';
12175   slider.style.display = 'block'; //inline;
12176   slider.style.zIndex = 100000;
12177   // nsb.style.zIndex = 200000;
12178   nsb.style.position = 'absolute';
12179   nsb.style.minWidth = '80px';
12180   nsb.style.left = '0px';
12181   nsb.style.top = '0px';
12182
12183   w = window.innerWidth;
12184   console.log("SliderWidth '+w+' , (w/3)+'px");
12185   if( w < 640 ){
12186     slider.style.setProperty('width',(w/3) + 'px','important');
12187   }
12188   main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
12189
12190   slider.style.resize = "both";
12191   slider.draggable = "true";
12192   wrap.appendChild(slider);
12193   console.log("-- added SbSlider");
12194   //nsb.addEventListener('scroll',SbScrolled);
12195
12196   buttons = document.createElement('div');
12197   buttons.id = 'NaviButtons';
12198   buttons.setAttribute('class','NaviButtons');
12199   buttons.align = "center";
12200   buttons.innerHTML += '<'+p><a href="#TopOfPost" draggable="true">TOP</a><'+p>';
12201   buttons.innerHTML += '<'+p><a href="#EndOfPost" draggable="true">END</a><'+p>';
12202   page.appendChild(buttons);
12203   buttons.style.position = 'fixed';
12204   buttons.style.zIndex = 30000;
12205   buttons.style.width = '180%';
12206   buttons.style.top = '320px';
12207   buttons.style.left = parseInt(w) * 1.0 + 'px';
12208   console.log("SbSlider installed (^-^)/ SatoxITS");
12209 }
12210 //window.addEventListener('load',SetSidebar); // after load
12211 DestroyNaviButtons = function(){
12212   nb = document.getElementById('NaviButtons');

```

```
12213     if( nb != null ){
12214         nb.parentNode.removeChild(NaviButtons);
12215     }
12216 }
12217 </script>
12218
12219 // 2020-1006 its-more.jp-blog-60000-style.css
12220 <!-- {
12221 <style>
12222 #NaviButtons {
12223     position:fixed;
12224     display:block;
12225     width:100%;
12226     height:100%;
12227     z-index:100px;
12228     font-size:30000;
12229     font-size:10px;
12230     color:#2ff important;
12231     text-align:center;
12232     background-color:rgba(230,230,230,0.01);
12233 }
12234
12235 #NaviButtons a {
12236     color:#2a2 important;
12237     font-size:20px;
12238     text-align:center;
12239     xtext-shadow:2px 2px #8ff;
12240     padding:0px;
12241     margin:10px;
12242     border:1px solid #288 important;
12243     border-radius:3px;
12244     background-color:rgba(160,160,160,0.05);
12245 }
12246 #SbsSlider {
12247     overflow:auto;
12248     resize:both important;
12249     xoverflow-y:hidden important;
12250     height:100px important;
12251     display:inline-block important;
12252     position:fixed important;
12253     left:0px;
12254     top:0px;
12255     width:180px;
12256     width:248;
12257     min-width:80px;
12258     height:1008 important;
12259     background-color:rgba(100,100,200,0.1);
12260 }
12261 #secondary {
12262     position:fixed;
12263     left:0px;
12264     top:0px;
12265     xxx-index:60000;
12266     scroll-behavior: overflow important;
12267     xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxwidth:188 important;
12268     padding-left:4pt;
12269     color:#fff;
12270     font-size:0.5em;
12271     background-color:rgba(64,160,64,0.6) important;
12272     white-space:nowrap;
12273 }
12274 #secondary a {
12275     color:#fff important;
12276     text-decoration:disable important;
12277 }
12278 #primary {
12279     position:relative;
12280     width:758 important;
12281     left:258 important;
12282 }
12283 #main {
12284     position:relative;
12285     width:758 important;
12286     left:258 important;
12287 }
12288 #site-navigation {
12289     position:relative;
12290     left:120px;
12291 }
12292 #adswc_countertext {
12293     color:#4b9e1;
12294     font-size:16pt important;
12295     xfont-size:108 important;
12296     font-weight:bold;
12297 }
12298 #nowTime {
12299     color:#a0ffa0;
12300     font-size:16pt important;
12301     xfont-size:108 important;
12302     font-weight:bold;
12303     text-shadow:1px 1px #fff;
12304 }
12305 #navigation-top {
12306     color:#22a important;
12307     border:0px;
12308     background-color:rgba(220,220,220,0.1);
12309 }
12310
12311 .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
12312     display: block;
12313     width: 1em;
12314     height: 1em;
12315     xheight: 1em;
12316 }
12317 .invisible-scrollbar::-webkit-scrollbar {
12318     width: 0px;
12319     height: 1px important;
12320     height: 1px important;
12321 }
12322 .mostly-customized-scrollbar::-webkit-scrollbar {
12323     width: 2px;
12324     height: 2px;
12325     xbackground-color: #aaa; xxx:or add it to the track;
12326 }
12327 .mostly-customized-scrollbar::-webkit-scrollbar-thumb {
12328     background: #000;
12329 }
12330 </style>
12331 } -->
12332
12333 </details>
12334 <!-- SBSidebar_WorkCodeSpan } -->
12335 </span>
12336 <!-- ===== Work } ===== -->
12337
12338 <!-- ===== Work { ===== -->
12339 </span id="Affiliate_WorkCodeSpan">
12340 *
12341 <details id="Affiliate_Test"><summary>Affiliate</summary>
12342 <!-- Affiliate // 2020-1010 SatoxTIS } -->
12343 <div id="AffViewDock">
12344 <div id="AffView" class="AffView" draggable="true" style="">
12345 <div id="AffItem" class="AffPlate">
12346 <iframe id="aff_0" class="AffItem"></iframe>
12347 <iframe id="aff_1" class="AffItem"></iframe>
12348 <iframe id="aff_2" class="AffItem"></iframe>
12349 <iframe id="aff_3" class="AffItem"></iframe>
12350 <iframe id="aff_4" class="AffItem"></iframe>
12351 <iframe id="aff_5" class="AffItem"></iframe>
12352 </div>
12353 </div></div>
12354 <h2>Supportive Affiliate</h2>
12355 <style>
12356 .AffView {
12357     z-index:0;
12358     overflow-x:scroll;
12359     overflow-y:scroll;
12360     position:fixed;
12361     max-width:2560px;
12362     max-height:1008;
12363     width:170px;
12364     left:758;
12365     height:958;
12366     resize:both;
12367     xleft:-108;
12368     margin-top:40px;
12369     xleft:0;
12370     xalign:right;
12371     display:block;
12372     border:4px inset rgba(255,255,255,0.1);
12373     background-color:rgba(255,255,255,0.1);
12374 }
12375
12376 .AffView:hover {
12377     z-index:1;
12378     width:300px;
12379     overflow:scroll;
12380     border:4px inset #fcc;
12381     background-color:rgba(80,80,255,0.2);
12382     background-color:#fff;
12383 }
12384
12385 .AffPlate:hover {
12386     border-left:4px dashed #888;
12387 }
12388 .AffPlate {
12389     overflow-x:visible;

```

```

12390 border-left:4px dashed rgba(255,255,255,0.1);
12391 max-width:2560px;
12392 max-height:2880px;
12393 margin-top:10px;
12394 margin-bottom:10px;
12395 margin-left:4px;
12396 width:300px;
12397 xheight:1440px;
12398 }
12399 .AffItem {
12400 overflow-x:visible;
12401 xoverflow-y:scroll;
12402 max-width:2560px;
12403 max-height:1440px;
12404 z-index:0;
12405 display:block;
12406 xposition:fixed;
12407 xposition:absolute;
12408 position:relative;
12409 //left:300px;
12410 xresize:both;
12411 padding:0px;
12412 width:600px;
12413 height:400px;
12414 max-height:800px !important;
12415 margin-top:0%;
12416 margin-left:0%;
12417 margin-right:0% !important;
12418 border:16px inset #ccc;
12419 transform:scale(0.5);
12420 background-color:rgba(255,255,255,0.2);
12421 xalign:right;
12422 }
12423 .AffItem:hover {
12424 border:16px inset #bbf;
12425 }
12426 </style>
12427 <input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12428 <input id="Affiliate_WorkCodeViewSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12429 <input id="Affiliate_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12430 <span id="Affiliate_WorkCodeView"></span>
12431 <script id="Affiliate_WorkCodeView">
12432 function Affiliate_openWorkCodeView(){
12433 function Affiliate_showWorkCode(){
12434 showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
12435 }
12436 Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
12437 }
12438 Affiliate_openWorkCodeView(); // should be invoked by an event
12439 </iframe id="aff_0" xxsrc="https://stackoverflow.com/tags" class="AffItem"></iframe>
12440 </iframe id="aff_9" xxsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffItem"></iframe>
12441 var Aff_Setup = false;
12442 Affiliate_Test.addEventListener('click',Aff_Setup);
12443 function Aff_Setup(){
12444 if( Aff_Setup { return; } Aff_Setup = true;
12445 parent = document.documentElement;
12446 parent.appendChild(AffView);
12447 AffView.style.top = '0px';
12448 AffView.style.left = (window.innerWidth - 280) + 'px';
12449 var off = 100;
12450 zoom = 0.5;
12451 ozoom = 0.3;
12452 leftx = window.innerWidth - 300;
12453 left = leftx + 'px';
12454 left = -10 + 'px';
12455 console.log('aff-init window.innerWidth='+window.innerWidth);
12456 w = 1000;
12457 h = 560;
12458 }
12459 aff_0.src="./gshell/gsh.go.html";
12460 aff_1.src="https://golang.org";
12461 aff_2.src="https://drafts.csswg.org/";
12462 aff_3.src="https://html.spec.whatwg.org/dev/";
12463 aff_4.src="https://wikipedia.org";
12464 aff_5.src="https://www.bing.com/translator";
12465 //parent.appendChild(aff_0);
12466 aff_0.style.width = zoom*w+'px';
12467 aff_0.style.height = zoom*h+'px';
12468 aff_0.style.left = left;
12469 //aff_0.style.top = off+'px'; off += ozoom*h;
12470 aff_0.draggable = 'true';
12471 //parent.appendChild(aff_1);
12472 aff_1.style.width = zoom*w+'px';
12473 aff_1.style.height = zoom*h+'px';
12474 aff_1.style.left = left;
12475 //aff_1.style.top = off+'px'; off += ozoom*h;
12476 aff_1.style.top = '-150px';
12477 aff_1.draggable = 'true';
12478 //parent.appendChild(aff_2);
12479 aff_2.style.width = zoom*w+'px';
12480 aff_2.style.height = zoom*h+'px';
12481 aff_2.style.left = left;
12482 //aff_2.style.top = off+'px'; off += ozoom*h;
12483 aff_2.style.top = '-200px';
12484 aff_2.draggable = 'true';
12485 //parent.appendChild(aff_3);
12486 aff_3.style.transform = 'scale(0.25)';
12487 aff_3.style.width = 2*zoom*w+'px';
12488 aff_3.style.height = 2*zoom*h+'px';
12489 aff_3.style.border = '32px inset #ccc';
12490 //aff_3.style.left = -390 + 'px'; //left*2;
12491 //aff_3.style.left = (leftx - 265) + 'px';
12492 aff_3.style.left = -395 + 'px';
12493 //aff_3.style.top = (-155*off)+'px'; off += ozoom*h;
12494 aff_3.style.top = '-600px';
12495 aff_3.draggable = 'true';
12496 //parent.appendChild(aff_4);
12497 aff_4.style.width = zoom*w+'px';
12498 aff_4.style.height = zoom*h+'px';
12499 aff_4.style.left = left;
12500 //aff_4.style.top = off+'px'; off += ozoom*h;
12501 aff_4.style.top = '-900px';
12502 aff_4.draggable = 'true';
12503 //parent.appendChild(aff_5);
12504 aff_5.style.transform = 'scale(0.300)';
12505 aff_5.style.width = zoom*(w*1.67)+'px';
12506 aff_5.style.height = zoom*(h*1.67)+'px';
12507 aff_5.style.border = '25px inset #ccc';
12508 //aff_5.style.left = -308+'px';
12509 //aff_5.style.left = (-175*leftx)+'px';
12510 //aff_5.style.top = (-95*off)+'px'; off += ozoom*h;
12511 aff_5.style.left = '0px';
12512 //aff_5.style.top = '0px';
12513 aff_5.style.top = '-1150px';
12514 //aff_5.style.align = 'right';
12515 aff_5.draggable = 'true';
12516 //window.addEventListener('resize',affresize);
12517 function affresize(){
12518 AffView.style.left = (window.innerWidth - 280) + 'px';
12519 leftx = window.innerWidth - 400;
12520 left = leftx + 'px';
12521 console.log('aff-resize window.innerWidth='+window.innerWidth);
12522 }
12523 //window.addEventListener('resize',affresize);
12524 //document.addEventListener('resize',affresize);
12525 //gsh.addEventListener('resize',affresize);
12526 function ResetAffView(){
12527 AffViewDoc.appendChild(AffView);
12528 AffView.removeAttribute('style');
12529 aff_0.removeAttribute('src');
12530 aff_0.removeAttribute('style'); aff_0.removeAttribute('draggable');
12531 aff_1.removeAttribute('src');
12532 aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
12533 aff_2.removeAttribute('src');
12534 aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
12535 aff_3.removeAttribute('src');
12536 aff_3.removeAttribute('style'); aff_3.removeAttribute('draggable');
12537 aff_4.removeAttribute('src');
12538 aff_4.removeAttribute('style'); aff_4.removeAttribute('draggable');
12539 aff_5.removeAttribute('src');
12540 aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
12541 }
12542 </script>
12543 </details>
12544 <div id="Affiliate_WorkCodeSpan" -->
12545 <div id="Work" -->
12546 </div>
12547 </div>
12548 <div id="TextCanvas_WorkCodeSpan" -->
12549 </div>
12550 <div id="TextCanvas_Summary" -->
12551 </div>
12552 </div>
12553 </div>
12554 </div>
12555 </div>
12556 </div>
12557 </div>
12558 </div>
12559 </div>
12560 </div>
12561 </div>
12562 </div>
12563 </div>
12564 </div>
12565 </div>
12566 </div>

```

```

1256 <details id="FontSelect_Section"><summary id="FontSelect_Summary">Font Selections/</summary>
1256 <h2>Font Selection</h2>
1256 <div>
1257 <div id="FontList"></div>
1257 </div>
1257 <style>
1257 #FontList {
1257   overflow:visible;
1257   background-color:rgba(240,245,255,1.0) !important;
1257 }
1257 #FontList td {
1257   font-size:16px;
1257   padding:0px;
1257   padding-left:2px;
1257   padding-right:2px;
1257   margin:0px;
1257   line-height:1.2;
1257   border:0px;
1257 }
1257 #FontList td:hover {
1257   background-color:#228;
1257 }
1257 #FontList tr:hover {
1257   color:#fff;
1257   background-color:#000;
1257   xborder:1px solid #000;
1257 }
1257 .xcourier { color:#000; font-size:16px; font-family:courier; }
1257 .xcursive { color:#000; font-size:16px; font-family:cursive; }
1257 .xfantasy { color:#000; font-size:16px; font-family:fantasy; }
1257 .xgeorgia { color:#000; font-size:16px; font-family:georgia; }
1257 .xmonospace { color:#000; font-size:16px; font-family:monospace; }
1257 </style>
1260 <script>
1260 function fontstr(name,text){
1260   //tr = '<'+'tr style=\'font-family:'+name+';\';>\n';
1260   tr = '<'+'tr style=\'font-family:'+name+';\';>\n';
1260   tr += '<'+'td style=\'font-family:Arial;font-size:12pt;\'>'+name+'<'+'</td>';
1260   tr += '<'+'td data-fsty="n">+text+'<'+'</td>';
1260   tr += '<'+'td data-fsty="b">'+b+'<'+'</td>';
1260   tr += '<'+'td data-fsty="i">'+i+'<'+'</td>';
1260   tr += '<'+'td data-fsty="bi">'+b+'<'+'</td>';
1260   tr += '<'+'</tr>';
1260   return tr;
1261 }
1261 function lsfont(){
1261   text = 'GShell-G0126&xlf923;';
1261   fl = '';
1261   fl += '<table>\n';
1261   fl += fontstr('Arial',text);
1261   fl += fontstr('Courier',text);
1261   fl += fontstr('Courier New',text);
1261   fl += fontstr('Georgia',text);
1261   fl += fontstr('Helvetica',text);
1261   fl += fontstr('Verdana',text);
1261   fl += fontstr('Times',text);
1261   fl += fontstr('Osaka',text);
1261   fl += fontstr('Meiryu',text);
1261   fl += fontstr('YuMincho',text);
1261   //fl += fontstr('Roman',text);
1261   //document.fonts.load('30px cursive');
1261   fl += fontstr('Serif',text);
1261   fl += fontstr('Sans-Serif',text);
1261   fl += fontstr('System-UI',text);
1261   fl += fontstr('Monospace',text);
1261   fl += fontstr('Cursive',text);
1261   fl += fontstr('Fantasy',text);
1261   fl += '</table>\n';
1261 }
1261 if( false ){
1261   fss = document.fonts.entries(); // FontFaceSet
1261   console.log('FSS='+fss);
1261   while( true ){
1261     font = fss.next();
1261     if( font.done ){
1261       break;
1261     }
1261     fl += font.value[0] + '<br>';
1261   }
1261   FontList.innerHTML = fl;
1261 }
1261 }
1261 function selectFont(e){
1261   t = e.target;
1261   let fsty = '';
1261   for( i = 0; i < 4; i++ ){
1261     //console.log('FontSelect '+t.nodeName+' #' + t.id+' '+t.style);
1261     if( t.nodeName == 'TD' ){
1261       //console.log('FontSelect '+t.outerHTML);
1261       if( t.hasAttribute('data-fsty') ){
1261         fsty = t.getAttribute('data-fsty');
1261       }
1261       //console.log('FontSelect = ' + fsty);
1261     }
1261     if( t.nodeName != 'TD' ){
1261       if( t.style != '' ){
1261         if( t.style.fontFamily != '' ){
1261           break;
1261         }
1261       }
1261     }
1261     t = t.parentNode;
1261   }
1261   if( t.style != '' ){
1261     font = t.style.fontFamily;
1261     //console.log('FontSelect: '+font);
1261     //console.log('FontSelect == '+fsty);
1261     if( font != '' ){
1261       sel = document.getElementById("TextCanvas_1_Font");
1261       if( sel != null ){
1261         if( fsty != '' ){
1261           TextCanvas_1_Bold.checked = 0 <= fsty.indexOf('b');
1261           TextCanvas_1_Italic.checked = 0 <= fsty.indexOf('i');
1261         }
1261         sel.value = font;
1261         RedrawTextCanvas();
1261       }
1261     }
1261     alert('Event: ' + e.target.nodeName + ' #' + font);
1261   }
1261 }
1261 }
1261 FontList.addEventListener('click',selectFont);
1261 document.fonts.onloadingdone = function(fsse){
1261   //alert('font-loaded '+fsse.fontfaces.length);
1261 }
1261 function FontList_Setup(){
1261   if( FontSelect_Summary.open ){
1261     lsfont();
1261   }
1261 }
1261 FontSelect_Summary.addEventListener('click',lsfont);
1261 </script>
1261 </details>
1261 </h2>Drawing Text on Canvas</h2>
1261 <div id="TextCanvas_1_Panel" class="CanvasLabel">
1261 <input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
1261 <input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
1261 <input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
1261 <input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">Bold
1261 <input id="TextCanvas_1_Italic" class="CanvasBox" type="checkbox" checked="">Italic
1261 <br>
1261 <input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
1261 <input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">
1261 </div>
1261 <div id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
1261 </div>
1261 <div id="TextCanvas_1" class="TextCanvas" width="400px" height="100px" draggable="true"></div>
1261 <div class="CanvasLabel">
1261 <input id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
1261 <input id="TextCanvas_1_ToPNG" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">PNG
1261 <input id="TextCanvas_1_ToJPEG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">JPEG
1261 <input id="TextCanvas_1_DataURL" class="CanvasBox" type="checkbox" checked="">DataURL
1261 <div id="TextCanvas_1_BgImage" class="CanvasImage"><img id="TextCanvas_1_Image" class="CanvasImage" src=""></div>
1261 <div id="TextCanvas_1_BgImage" class="CanvasImage"><br>(background image)</div>
1261 (Data URL)
1261 <div id="TextCanvas_1_DataUrlView" class="DataURLView"><span id="TextCanvas_1_DataUrlText"></span></div>
1261 </div>
1261 <style>
1261 CommandUsageText {
1261   font-family:Courier New;
1261 }
1261 TextCanvas {
1261   border:1px solid #000;
1261   resize:both;
1261   display:inline !important;
1261 }

```



```

12744.DataUrlView {
12745 width:100%;
12746 font-size:10pt;
12747 font-family:Courier New, monospace;
12748 color:#000;
12749 xbackground-color:#eee;
12750 margin-bottom:10px;
12751 xdisplay:block;
12752 xoverflow:scroll;
12753 }
12754.CanvasImage {
12755 border:1px dashed #000;
12756 }
12757.TextCanvasText {
12758 font-size:12pt;
12759 width:100%;
12760 }
12761.CanvasLabel {
12762 font-size:10pt;
12763 color:#000;
12764 }
12765.CanvasInImage {
12766 width:100%;
12767 height:160px;
12768 margin-bottom:10px;
12769 color:rgba(32,160,32,0.5);
12770 text-shadow:3px 3px #eee;
12771 background-color:#eee;
12772 xborder:1px solid #000;
12773 font-size:18pt;
12774 vertical-align:middle;
12775 }
12776.PanelRadio {
12777 font-size:12pt !important;
12778 color:#000 !important;
12779 vertical-align:middle;
12780 }
12781.CanvasBox {
12782 vertical-align:middle;
12783 margin-left:4px !important;
12784 margin-right:2px !important;
12785 }
12786.CanvasPanel {
12787 vertical-align:middle !important;
12788 height:14pt !important;
12789 width:inherit !important;
12790 padding:2px !important;
12791 margin:4px !important;
12792 margin-left:4px !important;
12793 margin-right:4px !important;
12794 font-size:10pt !important;
12795 font-family:Georgia !important;
12796 color:#000;
12797 display:inline !important;
12798 }
12799.TextCanvasPanel {
12800 vertical-align:middle;
12801 font-size:10pt !important;
12802 font-family:Georgia !important;
12803 color:#000;
12804 display:inline !important;
12805 }
12806.PanelButton {
12807 font-size:10pt !important;
12808 font-family:Georgia !important;
12809 vertical-align:middle;
12810 width:45pt !important;
12811 height:14pt !important;
12812 line-height:1.2 !important;
12813 padding:2px !important;
12814 margin:1px !important;
12815 display:inline !important;
12816 padding:1px !important;
12817 color:#fff !important;
12818 background-color:#228 !important;
12819 }
12820</style>
12821<script>
12822 function DrawTextCanvas(){
12823   ctx = TextCanvas_1.getContext('2d');
12824   var textfont = "";
12825   if( TextCanvas_1_Italic.checked ) textfont += ' italic';
12826   if( TextCanvas_1_Bold.checked ) textfont += ' bold';
12827   textfont += " " +TextCanvas_1_Size.value+'px';
12828   textfont += " " +TextCanvas_1_Font.value;
12829   //ctx.font = 'italic bold 64px Georgia';
12830   //console.log("TxFont="+textfont);
12831   ctx.fillStyle = TextCanvas_1_Color.value; //'#22a';
12832   ctx.font = textfont;
12833   ctx.fillText(TextCanvas_1_Text.value,10,80);
12834 }
12835 TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
12836 function ClearTextCanvas(){
12837   cv = TextCanvas_1;
12838   ctx = cv.getContext('2d');
12839   ctx.clearRect(0,0,cv.width,cv.height);
12840 }
12841 TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
12842 function RedrawTextCanvas(){
12843   ClearTextCanvas();
12844   DrawTextCanvas();
12845 }
12846 function ab2str(buf) {
12847   return String.fromCharCode.apply(null, new Uint16Array(buf));
12848 }
12849 }
12850 // 2020-1024, canvas to image
12851 function CanvasToImage(){
12852   canvas = TextCanvas_1;
12853   // https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL
12854   if( TextCanvas_1_ToPNG.checked ){
12855     url = canvas.toDataURL("image/png");
12856   }else{
12857     url = canvas.toDataURL("image/jpeg",1.0);
12858   }
12859   //alert('CanvasToImage: length='+url.length+'\n'+url);
12860   TextCanvas_1_Image.src = url;
12861   TextCanvas_1_BgImage.style.backgroundImage = 'url('+url+')';
12862   if( TextCanvas_1_DataURL.checked ){
12863     //TextCanvas_1_DataUrlView.innerHTML = url;
12864     txa = TextCanvas_1_DataUrlText;
12865     utxa = document.createElement('textArea');
12866     utxa.id = "TextCanvas_1_DataUrlText";
12867     utxa.style.width = "100%";
12868     utxa.style.height = "50pt";
12869     utxa.value = url;
12870     txa.parentNode.replaceChild(utxa,txa);
12871   }
12872   return TextCanvas_1_Image;
12873 }
12874 var image = new Image();
12875 image.src = url;
12876 url = str2ab(url);
12877 blob = new Blob(url,{type:'text/plain'});
12878 link = document.createElement('a');
12879 link.href = URL.createObjectURL(blob);
12880 link.download = 'character.txt';
12881 link.click();
12882 return image;
12883 }
12884 TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12885 }
12886 if( TextCanvas_Section.open ){
12887   DrawTextCanvas();
12888 }
12889 TextCanvas_Summary.addEventListener('click',DrawTextCanvas);
12890 }
12891</script>
12892<!-- -->
12893<script>
12894 //TextCanvas_1_Panel.addEventListener('mouseover',OffGJShell);
12895 //TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
12896</script>
12897<!-- Clicking the textarea is necessary to see upto the end of text. why? -->
12898<input id="TextCanvas_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12899<input id="TextCanvas_WorkCodeSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12900<input id="TextCanvas_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12901<span id="TextCanvas_WorkCodeView"></span>
12902<script id="TextCanvas_WorkScript">
12903 function TextCanvas_openWorkCodeView(){
12904   function TextCanvas_showWorkCode(){
12905     showHtmlCode(TextCanvas_WorkCodeView,TextCanvas_WorkCodeSpan);
12906   }
12907   TextCanvas_WorkCodeViewOpen.addEventListener('click',TextCanvas_showWorkCode);
12908 }
12909 TextCanvas_openWorkCodeView(); // should be invoked by an event
12910</script>
12911</details>
12912<!-- TextCanvas_WorkCodeSpan -->
12913<!-->
12914<span id="Shading_WorkCodeSpan">
12915</span>
12916</div>
12917</div>
12918<!-- Work { -->
12919<span id="Shading_WorkCodeSpan">
12920</span>

```

```

1292 <details><summary>Shading Canvas</summary>
1293 </------- Shading Canvas // 2020-1011 SatouITS { -->
1294 <Shading Canvas</h2>
1295 <note class="CommandUsageText">
1296 <b>Commands</b><br>
1297 Placement Mode<br>
1298 a ... apply (into absolute position)<br>
1299 j ... bring down (ArrowDown)<br>
1300 k ... bring up (ArrowUp)<br>
1301 h ... bring left (ArrowLeft)<br>
1302 l ... bring right (ArrowRight)<br>
1303 0 ... z-index = 0<br>
1304 + ... z-index += 1<br>
1305 - ... z-index -= 1<br>
1306 r ... return to here (relative position)<br>
1307 c ... clear the log text<br>
1308 Note: the HTML text must be contenteditable to catch Key Event.<br>
1309 </note>
1310
1311
1312 <div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
1313 <div id="Shading_1_Html" class="ShadingHtml" draggable="true"></div>
1314 <div id="Shading_1_Log" class="ShadingLog" draggable="true" style=""></div>
1315
1316
1317 <canvas id="Shading_1_Canvas" class="ShadingCanvas" width="300px" height="300px" draggable="true" style="">My Canvas.</canvas></div>
1318 <style>
1319 ShadingPlate {
1320   z-index:0;
1321   position:static;
1322   overflow:scroll;
1323   display:block;
1324   width:100%;
1325   height:400px;
1326   font-size:9pt;
1327   font-family:Courier New;
1328   border:1px dashed #000;
1329   color:#444;
1330 }
1331 ShadingLog {
1332   z-index:0;
1333   position:relative;
1334   display:block;
1335   top:0px;
1336   left:0px;
1337   overflow:scroll;
1338   width:100%;
1339   font-size:9pt;
1340   font-family:Courier New;
1341   color:#666;
1342   overflow:scroll;
1343   background:rgba(200,255,200,0.4);
1344   height:400px;
1345 }
1346 ShadingHtml {
1347   z-index:2;
1348   position:relative;
1349   display:block;
1350   top:0px;
1351   left:0px;
1352   overflow:scroll;
1353   width:100%;
1354   font-size:12pt;
1355   font-family:Courier New;
1356   color:#666;
1357   overflow:scroll;
1358   background:rgba(200,255,200,0.4);
1359   height:400px;
1360 }
1361 ShadingCanvas {
1362   z-index:1;
1363   position:relative;
1364   xdisplay:block;
1365   top:0px;
1366   left:100px;
1367   resize:both;
1368   border:1px solid #000;
1369 }
1370 </style>
1371 <input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1372 <input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1373 <input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1374 <span id="Shading_WorkCodeView" />
1375 <script id="Shading_WorkScript">
1376 function Shading_openWorkCodeView(){
1377   function Shading_showWorkCode(){
1378     showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
1379   }
1380   Shading_WorkCodeViewOpen.addEventListener('click',Shading_showWorkCode);
1381 }
1382 const BR = '<br>';
1383 Shading_openWorkCodeView(); // should be invoked by an event
1384 function sh_onClick(e){
1385   Shading_1_Log.innerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
1386     + ' offset('+e.offsetX+', '+e.offsetY+')'
1387     + client('+e.clientX+', '+e.clientY+')'
1388     + ' page(+e.pageX+', '+e.pageY+')'
1389     + ' screen('+e.screenX+', '+e.screenY+')'
1390     +BR;
1391   e.stopPropagation();
1392   e.preventDefault();
1393 }
1394 function sh_onKeyUp(e){
1395   if (Shading_1.style.zIndex == '' ){
1396     Shading_1.style.zIndex = 0;
1397   }
1398   zi = parseInt(Shading_1.style.zIndex);
1399   if( e.key.length == 1 ){
1400     Shading_1_Html.innerHTML += e.key;
1401   }
1402   if( e.key == '0' ){ zi = 0; }else
1403   if( e.key == '+' ){ zi += 1; }else
1404   if( e.key == '-' ){ zi -= 1; }else
1405   if( e.key == 'c' ){
1406     Shading_1_Log.innerHTML = '';
1407   }else
1408   if( e.key == 'r' ){
1409     Shading_1.style.position = "relative";
1410     Shading_1.style.top = '0px';
1411     Shading_1.style.left = '0px';
1412     zi = 0;
1413   }else
1414   if( e.key == 'j' || e.code == 'ArrowDown' ){
1415     topx = parseInt(Shading_1.style.top) + 50;
1416     Shading_1.style.top = topx + 'px';
1417   }else
1418   if( e.key == 'k' || e.code == 'ArrowUp' ){
1419     topx = parseInt(Shading_1.style.top) - 50;
1420     Shading_1.style.top = topx + 'px';
1421   }else
1422   if( e.key == 'l' || e.code == 'ArrowRight' ){
1423     lefty = parseInt(Shading_1.style.left) + 50;
1424     Shading_1.style.left = lefty + 'px';
1425   }else
1426   if( e.key == 'h' || e.code == 'ArrowLeft' ){
1427     lefty = parseInt(Shading_1.style.left) - 50;
1428     Shading_1.style.left = lefty + 'px';
1429   }else
1430   if( e.key == 'a' ){
1431     Shading_1.style.position = "absolute";
1432     Shading_1.style.top = '0px';
1433     Shading_1.style.left = '0px';
1434   }else{
1435     Shading_1.style.zIndex = zi;
1436     Shading_1_Log.innerHTML += 'KeyUp.'+e.target.nodeName+'#'+e.target.id
1437       + 'Up('+e.key+'/' +e.code+')'
1438       + 'z-index:'+zi+'/' +Shading_1.style.zIndex
1439       + 'top:'+Shading_1.style.top
1440       +BR;
1441     e.stopPropagation();
1442     e.preventDefault();
1443   }
1444   function sh_onKeyDown(e){
1445     Shading_1_Log.innerHTML += 'KeyDown'+e.target.nodeName+'#'+e.target.id
1446       + 'Down('+e.key+'/' +e.code+')'+BR;
1447     e.stopPropagation();
1448     e.preventDefault();
1449   }
1450 }
1451 function Shading_Setup(){
1452   Shading_1_Log.innerHTML += '<h4>Click here</h4>';
1453   Shading_1.addEventListener('keydown',sh_onKeyDown);
1454   Shading_1.addEventListener('keyup',sh_onKeyUp);
1455   Shading_1.addEventListener('click',sh_onClick);
1456   Shading_1.addEventListener('click',sh_onClick);
1457 }
1458 Shading_1_Log.style.top = "-400px";
1459 Shading_1_Log.style.left = "200px";
1460
1461 Shading_1.appendChild(Shading_1_Canvas);
1462 Shading_1_Canvas.style.width = "300px";
1463 Shading_1_Canvas.style.height = "300px";
1464 Shading_1_Canvas.style.position = "relative";
1465 Shading_1_Canvas.style.top = "-750px";
1466 Shading_1_Canvas.style.left = "100px";
1467

```

```

13098 const ctx = Shading_1_Canvas.getContext('2d');
13099 ctx.fillStyle = 'rgba(160,0,0,0.9)';
13100 ctx.fillRect(50,40,40);
13101 ctx.fillStyle = 'rgba(0,160,0,0.9)';
13102 ctx.fillRect(60,60,40,40);
13103 ctx.fillStyle = 'rgba(0,0,160,0.9)';
13104 ctx.fillRect(70,70,40,40);
13105 }
13106 function Reset_ShadingCanvas(){
13107   Shading_1_Log.removeAttribute('style');
13108   Shading_1_Log.innerHTML = '';
13109   Shading_1_Canvas.style = '';
13110   //Shading_1_Canvas.removeAttribute('style');
13111 }
13112
13113 </script>
13114 </details>
13115 <!-- Shading_WorkCodeSpan -->
13116 * //</span>
13117 <!-- ===== Work } ===== -->
13118
13119
13120 <!-- ===== Work { ===== -->
13121 * //</span id="Charmap_WorkCodeSpan">
13122 /*
13123 <details id="Charmap_Work"><summary>Character Map Mandala</summary>
13124 <!-- ===== UnicodeCharmap // 2020-1015 SatoxIFS { -->
13125 <h2>Unicode Character Map</h2>
13126 <note>code 0x0000 - 0xFFFF / 16px / 3200 x 3200 px / zoom:0.25</note>
13127 <div id="Charmap_1_Frame">
13128 <div id="Charmap_1_Text" class="Charmap">
13129 </div>
13130 </div>
13131 <br>
13132
13133 <style>
13134 #Charmap_1_Frame {
13135   overflow:scroll;
13136   height:3200px; width:3200px;
13137   xtransform:scale(0.5);
13138   zoom:0.25;
13139   resize:both;
13140   background-color:#fff;
13141 }
13142 .Charmap {
13143   zoom:0.25;
13144   font-size:16px;
13145   line-height:1.0;
13146   xfont-family:Georgia;
13147   color:#000;
13148 }
13149 </style>
13150 <script>
13151 function charmapgen(){
13152   text = '';
13153   for(cc = 0; cc < 0x10000; cc++){
13154     text += String.fromCharCode(cc);
13155   }
13156   Charmap_1_Text.innerHTML = text;
13157 }
13158 Charmap_Work.addEventListener('click', charmapgen);
13159 //charmapgen();
13160 </script>
13161
13162 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13163 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13164 <input id="Charmap_WorkCodeView" class="HtmlCodeViewButton" type="button" value="Signature">
13165 <span id="Charmap_WorkCodeView"></span>
13166 <script id="Charmap_WorkScript">
13167 function Charmap_openWorkCodeView(){
13168   function Charmap_showWorkCodeView(Charmap_WorkCodeSpan);
13169   showHtmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
13170 }
13171 Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_showWorkCodeView);
13172
13173 Charmap_openWorkCodeView(); // should be invoked by an event
13174 </script>
13175 </details>
13176 <!-- Charmap_WorkCodeSpan -->
13177 * //</span>
13178 <!-- ===== Work } ===== -->
13179
13180
13181 <!-- ===== Work { ===== -->
13182 * //</span id="Pointillism_WorkCodeSpan">
13183 /*
13184 <details><summary>Collaborated Pointillism</summary>
13185 <!-- ===== CollaboratedPointillism // 2020-1016 SatoxIFS { -->
13186 <h2><a name="Pointillism" class="Pointillism"></a><a href="#Pointillism">Collaborated Pointillism</a></h2>
13187
13188 <input id="Pointillism_1_Share" type="checkbox" class="HtmlCodeViewButton" value="Share"> Share
13189 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
13190 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
13191 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Replay">
13192 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
13193 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_SaveCanvas()" value="Save">
13194 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_LoadCanvas()" value="Load">
13195 <div id="Pointillism_1" class="Pointillism">
13196
13197 <span id="Pointillism_1_Unit_1" class="Pointillism_Unit">
13198 <span id="Pointillism_1_XY_1" class="Pointillism_XY">XY1</span>
13199 <span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY1-remote</span>
13200 <canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
13201 </span>
13202
13203 <span id="Pointillism_1_Unit_2" class="Pointillism_Unit">
13204 <span id="Pointillism_1_XY_2" class="Pointillism_XY">XY2</span>
13205 <span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
13206 <canvas id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
13207 </span>
13208 </div>
13209 <br>
13210
13211 <style>
13212 .Pointillism {
13213   display:block;
13214   resize:both;
13215   width:680px;
13216   height:380px;
13217   min-width:240px;
13218   min-height:270px;
13219   background-color:#eee;
13220   overflow:scroll;
13221   font-size:16px;
13222   font-family:Georgia;
13223   color:#000;
13224   vertical-align:middle;
13225 }
13226 .Pointillism_Unit {
13227   position:relative;
13228   top:0px;
13229   display:block;
13230   overflow:scroll;
13231   width:300px;
13232   height:350px;
13233   margin:5px;
13234   padding:10px;
13235   background-color:rgba(255,255,127,0.7);
13236 }
13237 .Pointillism_XY {
13238   display:Block;
13239   vertical-align:middle;
13240   width:290px;
13241   xheight:20px;
13242   font-size:12px;
13243   line-height:1.2;
13244   padding:5px;
13245   margin:0px;
13246   color:#fff;
13247   background-color:#44c;
13248 }
13249 .Pointillism_XY_Remote {
13250   display:Block;
13251   vertical-align:middle;
13252   width:290px;
13253   xheight:20px;
13254   font-size:12px;
13255   line-height:1.2;
13256   padding:5px;
13257   color:#fff;
13258   background-color:#4a4;
13259 }
13260 .Pointillism_Canvas {
13261   display:Block;
13262   position:relative;
13263   xpadding:20px;
13264   xieft:20px;
13265   xtop:20px;
13266   background-color:#333;
13267 }
13268 </style>
13269 <script>
13270 var points = [];
13271 var replay = [];
13272 var replayx = 0;
13273 function pClearCanvas(can){

```

```

13275 ctx = can.getContext('2d');
13276 ctx.clearRect(0,0,can.width,can.height);
13277 }
13278 function Pointillism_1_ClearCanvas(){
13279 pClearCanvas(Pointillism_1_Canvas_1);
13280 pClearCanvas(Pointillism_1_Canvas_2);
13281 }
13282 function PointsReset(){
13283 points = [];
13284 replay = [1];
13285 inRepeat = false;
13286 inReplay = false;
13287 Pointillism_1_ClearCanvas();
13288 }
13289 function Pointillism_1_ResetCanvas(){
13290 PointsReset();
13291 if( Pointillism_1_Share.checked ){
13292 //alert('---broad cast reset\n');
13293 GJ_BoastMessage('DRAW RESET');
13294 }
13295 }
13296 function Pointillism_1_ResetCanvasReceive(){
13297 //alert('---received reset\n');
13298 PointsReset();
13299 }
13300 function drawPoint(can,x,y,r,g,b){
13301 const ctx = can.getContext('2d');
13302 ctx.fillStyle = "rgba("+r+","+g+","+b+",0.7)";
13303 ctx.fillRect(x,y,8,8);
13304 }
13305 function waitMs(serno,ms){
13306 console.log("-- wait #'+serno+' '+ms+'ms');
13307 until = new Date();
13308 now = until.getTime();
13309 untilMs = now + ms;
13310 for( wi = 0; ; wi++){
13311 now = new Date();
13312 nowMs = now.getTime();
13313 remMs = untilMs - nowMs;
13314 //console.log('wait '+wi+' : '+remMs+'/'+ms);
13315 if( remMs < 0 ){
13316 break;
13317 }
13318 }
13319 }
13320 var inReplay = false;
13321 function replay1(){
13322 rx = replay;
13323 if( replay.length <= rx ){
13324 return;
13325 }
13326 replay += 1;
13327 pl = replay[rx];
13328 if( pl[1] == 1 ){
13329 can = Pointillism_1_Canvas_1;
13330 }else{
13331 can = Pointillism_1_Canvas_2;
13332 }
13333 drawPoint(can,pl[2],pl[3],pl[4],pl[5],pl[6]);
13334 if( inReplay == false ){
13335 console.log('wait '+replay+' Stopped');
13336 return;
13337 }
13338 if( rx < replay.length-1 ){
13339 prevMs = replay[rx][0].getTime();
13340 nextMs = replay[rx+1][0].getTime();
13341 delayMs = nextMs - prevMs;
13342 //console.log('wait '+replay+' '+delayMs+'ms');
13343 window.setTimeout(replay1,delayMs);
13344 }else{
13345 console.log('wait '+replay+' Finished');
13346 if( inRepeat ){
13347 window.setTimeout(repeat1,1000);
13348 }
13349 }
13350 }
13351 function Pointillism_1_ReplayCanvas(can){
13352 Pointillism_1_ClearCanvas();
13353 replay = points;
13354 replayx = 0;
13355 inReplay = true;
13356 replay1();
13357 }
13358 var inRepeat = false;
13359 function repeat1(){
13360 Pointillism_1_ClearCanvas();
13361 replay = points;
13362 replayx = 0;
13363 replay1();
13364 if( inRepeat ){
13365 //window.setTimeout(repeat1,1000);
13366 }
13367 }
13368 function Pointillism_1_RepeatCanvas(can){
13369 if( inRepeat ){
13370 inRepeat = false;
13371 inReplay = false;
13372 }else{
13373 inRepeat = true;
13374 inReplay = true;
13375 repeat1();
13376 }
13377 }
13378 function CopyLocal(){ return Pointillism_1_Share.checked == false; }
13379 function Pointillism_Setup(){
13380 var moveCount1 = 0;
13381 var moveCount2 = 0;
13382 }
13383 var gJlinked = false;
13384 function GJdraw(msg){
13385 if( gJlinked == false ){
13386 //GJLink_Section.open = true;
13387 GJ_Link();
13388 gJlinked = true;
13389 }
13390 GJ_BoastMessage('DRAW '+msg);
13391 }
13392 function showXY1(e){
13393 moveCount1 += 1;
13394 x = e.offsetX;
13395 y = e.offsetY;
13396 Pointillism_1_XY_1.innerHTML = 'XY1: '+ 'x'+x +', y'+y + ' /'+moveCount1+'/'+points.length;
13397 Pointillism_1_XY_2_Remote.innerHTML = 'XY1: '+ 'x'+x +', y'+y + ' /'+moveCount1;
13398 if( e.buttons || CopyLocal() ){
13399 points.push([new Date(),1,x,y,64,64,255]);
13400 drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
13401 if( CopyLocal() ){
13402 drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
13403 }
13404 }
13405 GJdraw('1','x+', '+y');
13406 }
13407 function showXY2(e){
13408 moveCount2 += 1;
13409 x = e.offsetX;
13410 y = e.offsetY;
13411 Pointillism_1_XY_2.innerHTML = 'XY2: '+ 'x'+x +', y'+y + ' /'+moveCount2+'/'+points.length;
13412 Pointillism_1_XY_1_Remote.innerHTML = 'XY2: '+ 'x'+x +', y'+y + ' /'+moveCount1;
13413 if( e.buttons || CopyLocal() ){
13414 points.push([new Date(),2,x,y,64,255,64]);
13415 drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
13416 if( CopyLocal() ){
13417 drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
13418 }
13419 }
13420 GJdraw('2','x+', '+y');
13421 }
13422 Pointillism_1_Canvas_1.addEventListener('mousemove',showXY1);
13423 Pointillism_1_Canvas_2.addEventListener('mousemove',showXY2);
13424 Pointillism_1_Unit_2.style.left = '340px';
13425 Pointillism_1_Unit_2.style.top = '-375px';
13426 }
13427 function Pointillism_RemoteDraw(arg){
13428 //alert('Draw at '+arg);
13429 //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
13430 if( arg == "RESET" ){
13431 Pointillism_1_ResetCanvasReceive();
13432 }else{
13433 argv = arg.split(',');
13434 x = argv[1];
13435 y = argv[2];
13436 Pointillism_1_XY_2_Remote.innerHTML = 'XYR: '+ 'x'+x +', y'+y + ' /'+points.length;
13437 drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
13438 }
13439 }
13440 </script>
13441
13442
13443
13444 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13445 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="Snapshot">
13446 <input id="Pointillism_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13447 <span id="Pointillism_WorkCodeView"></span>
13448 <script id="Pointillism_WorkScript">
13449 function Pointillism_openWorkCodeView(){
13450 function Pointillism_showWorkCode(){
13451 showHtmlCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);

```

```

13452     }
13453     Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_showWorkCode);
13454 }
13455 Pointillism_openWorkCodeView(); // should be invoked by an event
13456 </script>
13457 </details>
13458 <!-- StatCounter_WorkCodeSpan } -->
13459 * //</span>
13460 <!-- ===== Work } ===== -->
13461
13462
13463
13464 <!-- ===== Work { ===== -->
13465 <!--<span id="StatCounter_WorkCodeSpan">
13466 *
13467 <details><summary>StatCounter</summary>
13468 <!--<div class="StatCounter" /> 2020-1018 SatoxITS { -->
13469 <h2>StatCounter</h2>
13470
13471 <div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
13472 (counter as image tag)</div>
13473 <style>
13474 .statcounter {
13475     vertical-align:middle;
13476 }
13477 #sc_SatoxITS {
13478     color:#000;
13479     font-size:12pt;
13480     height:30px;
13481     width:100%;
13482     background-color:#ddd;
13483 }
13484 </style>
13485
13486 <div>
13487 <script>
13488     var sc_project=12411639;
13489     var sc_invisible=0;
13490     var sc_security="1aeb2a3a";
13491     var sc_https=1;
13492     var scJsHost = "https://";
13493 </script>
13494 <!-- script src="https://statcounter.com/counter.js" -->
13495 <!-- /script --> (counter by inline script)
13496 </div>
13497
13498 <input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13499 <input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13500 <input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13501
13502 <span id="StatCounter_WorkCodeView"></span>
13503 <script id="StatCounter_WorkScript">
13504 function StatCounter_openWorkCodeView(){
13505     function StatCounter_showWorkCode(){
13506         showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
13507     }
13508     StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
13509 }
13510 StatCounter_openWorkCodeView(); // should be invoked by an event
13511 </script>
13512
13513 </details>
13514 <!-- StatCounter_WorkCodeSpan } -->
13515 * //</span>
13516 <!-- ===== Work } ===== -->
13517
13518
13519
13520 <!-- ===== Work { ===== -->
13521 <!--<span id="CascadedCanvasBook_WorkCodeSpan">
13522 *
13523 <details id="CascadedCanvasBook_Section"><summary id="CascadedCanvasBook_Summary">CascadedCanvasBook</summary>
13524 <!--<div class="CascadedCanvasBook" /> 2020-1031 SatoxITS { -->
13525 <h2>Cascaded Canvas Book</h2>
13526
13527 <div id="CBPanel" class="CBPanel">
13528 <input id="CB_new" type="button" value="NewCanvas">
13529 </div>
13530
13531 <br>
13532
13533 <h3>Undo / Redo / Replay</h3>
13534
13535 <div id="CanvasTool_UndoRedo">
13536 <span id="DrawReplay" class="CanvasTool" draggable="true" contenteditable=""
13537 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13538 <input class="CV_Button" type="button" value="Redraw">
13539 From <input data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
13540 To <input id="DrawingSernoView" data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
13541 </span>
13542 </div>
13543
13544 <script>
13545 function childByName(node,name){
13546     for( let i = 0; i < node.children.length; i++ ){
13547         ch = node.children[i];
13548         name1 = ch.getAttribute('data-name');
13549         if( name1 == name ){
13550             return ch;
13551         }
13552     }
13553     return null;
13554 }
13555
13556 function OnWheelInt(){
13557     event.preventDefault();
13558     t = event.target;
13559     n = t.nodeName;
13560     i = t.id;
13561     p = t.parentNode;
13562     y = event.deltaY;
13563     //console.log('OnWheelInt '+y+' '+n+'#'+i+' '+t.value);
13564     if( y < 0 ){ // scroll forward (up)
13565         inc = -y;
13566     }else{
13567         inc = -y;
13568     }
13569     inc /= 6;
13570     val = parseFloat(t.value) + inc;
13571     t.value = val.toFixed(0);
13572     return val;
13573 }
13574 var DrawingSerno = 0;
13575 function saveDrawing(){
13576     DrawingSerno += 1;
13577     DrawingSernoView.value = DrawingSerno;
13578 }
13579 function to2x(x){
13580     if( x <= 0xF ){
13581         return '0'+x.toString(16);
13582     }else{
13583         return x.toString(16);
13584     }
13585 }
13586 </script>
13587
13588 <div id="InstaColorPicker" draggable="true">
13589 <h3>Color Picker</h3>
13590 <div id="CanvasColor">
13591
13592 Select value by
13593 <input id="ICPmotion" type="checkbox" checked="">Mouse Motion
13594 <input id="ICPautoAddMotion" type="checkbox" checked="">Auto. add to history
13595 <input id="ICPwheel" type="checkbox" checked="">Mouse Wheel
13596 <input id="ICPautoAddWheel" type="checkbox" checked="">Auto. add to history
13597
13598 <div data-name="Fore" id="CanvasTool_Color_Fore" class="CanvasTool" onchange="showColorSample()">
13599 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13600 <input class="CV_Button" type="button" value="Fill">
13601 <input data-name="R" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13602 <input data-name="G" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13603 <input data-name="B" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13604 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13605 RGBA<input data-name="C" class="ColorParam" type="text" value="#000000ff">
13606 <span data-name="Sample">Sample</span>
13607 </div>
13608
13609 <div data-name="Fill" id="CanvasTool_Color_Fill" class="CanvasTool" onchange="showColorSample()">
13610 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13611 <input class="CV_Button" type="button" value="Fill">
13612 <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13613 <input data-name="G" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13614 <input data-name="B" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13615 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13616 RGBA<input data-name="C" class="ColorParam" type="text" value="#000000ff">
13617 <span data-name="Sample">Sample</span>
13618 </div>
13619
13620 <div data-name="Back" id="CanvasTool_Color_Back" class="CanvasTool" onchange="showColorSample()">
13621 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13622 <input class="CV_Button" type="button" value="Back">
13623 <input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13624 <input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13625 <input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13626 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13627 RGBA<input data-name="C" class="ColorParam" type="text" value="#ffffffff">
13628 <span data-name="Sample">Sample</span>

```

```

1362</div>
1363<div data-name="Border" id="CanvasTool_Color_Border" class="CanvasTool" onchange="showColor1Sample()">
1364<input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
1365<input class="CV_Button" type="button" value="Border">
1366<input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex(">
1367<input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex(">
1368<input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex(">
1369<input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex(">
1370RGB<input data-name="C" class="ColorParam" type="text" value="#ffffff">
1371<span data-name="Sample">Sample</span>
1372</div>
1373<div id="ColorComposition" class="ColorComposition">
1374Sample
1375</div>
1376<br>
1377<input class="LargeButton CV_Button" type="button" value="Clear Color History" onclick="ClearColorHistory()">
1378<input id="LenColorHistory" class="CanvasParam" type="text" value="0">
1379/Max<input id="MaxColorHistory" class="CanvasParam" type="text" value="200">
1380<div id="Color" class="Color" onclick="SelectThisColor()"></div>
1381<div id="Color1" class="Color1" onclick="SelectThisColor()"></div>
1382</div>
1383</div>
1384</div>
1385<style>
1386.LargeButton {
1387font-size:14px !important;
1388width:160px !important;
1389color:#f00 !important;
1390}
1391.ColorComposition {
1392color:#000000ff !important;
1393font-size:16pt !important;
1394padding:4px !important;
1395border:2px solid #000000ff !important;
1396}
1397.Color1 {
1398width:100% !important;
1399height:10px !important;
1400font-family:Courier New !important;
1401font-size:10px !important;
1402background-color:#000 !important;
1403}
1404.Color1:hover {
1405border:1px solid #000;
1406}
1407.ColorHistory {
1408resize:both;
1409//overflow:scroll;
1410width:100% !important;
1411height:200px !important;
1412}
1413</style>
1414<script>
1415var ColorID = 0;
1416var LastColor = Color1;
1417var HistLength = 0;
1418function ClearColorHistory(){
1419ColorHistory.innerHTML = '';
1420cl = Color0.cloneNode();
1421cl.id = 'Color1';
1422LastColor = document.getElementById('Color1');
1423ColorHistory.appendChild(cl);
1424HistLength = 0;
1425LenColorHistory.value = '0';
1426}
1427function OnWheelHex(){
1428event.preventDefault();
1429t = event.target;
1430n = t.nodeName;
1431i = t.id;
1432p = t.parentNode;
1433y = event.deltaY;
1434inc = -y; // scroll forward (up)
1435inc /= 6;
1436val = parseFloat(t.value) + inc;
1437val = val.toFixed(0);
1438if( val < 0 ) val = 0;
1439if( 255 < val ) val = 255;
1440if( t.value != val || event.ctrlKey ){
1441t.value = val;
1442if( ICPautoAddWheel.checked ){
1443showColor1Sample();
1444}
1445}
1446return val;
1447}
1448function SelectThisColor(){
1449ok = confirm('Pick this color ? '+event.target.innerHTML);
1450if( ok ){
1451// add to Picked
1452}
1453}
1454var LastMotion = 0;
1455function motionColor(){
1456if( !ICPmotion.checked ){
1457return;
1458}
1459d = new Date();
1460if( d.getTime() - LastMotion < 100 ){
1461return;
1462}
1463LastMotion = d.getTime();
1464t = CanvasTool_Color_Fore;
1465R = childByName(t,'R');
1466G = childByName(t,'G');
1467B = childByName(t,'B');
1468A = childByName(t,'A');
1469// console.log('mouse motion '+event.x+', '+event.y+' target#'+t);
1470updated = false;
1471val = 255 * (event.x/window.innerWidth);
1472val = val.toFixed(0);
1473if( B.value != val || event.ctrlKey ){
1474B.value = val;
1475updated = true;
1476}
1477val = 255 * (event.y/window.innerHeight);
1478val = val.toFixed(0);
1479if( G.value != val || event.ctrlKey ){
1480G.value = val;
1481updated = true;
1482}
1483if( updated ){
1484showColor1Sample(t,ICPautoAddMotion.checked);
1485}
1486}
1487function scrollColor(){
1488if( !ICPmotion.checked ){
1489return;
1490}
1491d = new Date();
1492if( d.getTime() - LastMotion < 100 ){
1493return;
1494}
1495LastMotion = d.getTime();
1496t = CanvasTool_Color_Fore;
1497R = childByName(t,'R');
1498G = childByName(t,'G');
1499B = childByName(t,'B');
1500A = childByName(t,'A');
1501updated = false;
1502y = gsh.getBoundingClientRect().top.toFixed(0)
1503if( y < 0 ) y = -1;
1504size = 10000;
1505val = 255 * (y/size);
1506val = val.toFixed(0);
1507if( 255 < val ) val = 255;
1508if( R.value != val || event.ctrlKey ){
1509R.value = val;
1510updated = true;
1511}
1512if( updated ){
1513showColor1Sample(t,ICPautoAddMotion.checked);
1514}
1515}
1516function showColor1Sample(){
1517t = event.target.parentNode;
1518showColor1Sample(t,ICPautoAddWheel);
1519}
1520function showColor1Sample(t,add){
1521name = t.getAttribute('data-name');
1522R = childByName(t,'R').value;
1523G = childByName(t,'G').value;
1524B = childByName(t,'B').value;

```

```

13806 A = childByName(t,'A').value;
13807
13808 R = parseInt(R);
13809 G = parseInt(G);
13810 B = parseInt(B);
13811 A = parseInt(A);
13812
13813 R = to02x(R); //R.toString(16);
13814 G = to02x(G); //G.toString(16);
13815 B = to02x(B); //B.toString(16);
13816 A = to02x(A); //A.toString(16);
13817
13818 color = '#'+R+C+B+A;
13819 //console.log(name+color);
13820
13821 C = childByName(t,'C');
13822 C.value = color;
13823 S = childByName(t,'Sample');
13824 S.style.color = color;
13825
13826 ColorID += 1;
13827 cl = Color1;
13828 clid = cl.id;
13829 cl.id = color + "ColorID";
13830 cl.innerHTML = cl.id + ' ' + '+' + font color=black>+color+'+'//font color=white>+color+'+'//font';
13831
13832 if( name == 'Fore' ){
13833   ColorComposition.style.setProperty('color',color,'important');
13834   cl.style.setProperty('color',color,'important');
13835   cl.style.setProperty('background-color',color,'important');
13836 }
13837 if( name == 'Fill' ){
13838   ColorComposition.style.setProperty('border-color',color,'important');
13839   cl.style.setProperty('color',color,'important');
13840   cl.style.setProperty('background-color',color,'important');
13841 }
13842 if( name == 'Back' ){
13843   ColorComposition.style.setProperty('background-color',color,'important');
13844   cl.style.setProperty('background-color',color,'important');
13845 }
13846 if( name == 'Border' ){
13847   ColorComposition.style.setProperty('border-color',color,'important');
13848   cl.style.setProperty('background-color',color,'important');
13849   cl.style.setProperty('border-color',color,'important');
13850 }
13851 ccl = cl.cloneNode(true);
13852 cl.id = clid;
13853
13854 if( add ){
13855   max = parseInt(MaxColorHistory.value);
13856   if( HistLength < max ){
13857     HistLength++;
13858     LenColorHistory.value = HistLength;
13859     ColorHistory.insertBefore(ccl,LastColor);
13860     LastColor = ccl;
13861   }
13862   if( max <= HistLength ){
13863     LenColorHistory.style.setProperty('background-color','#f00','important');
13864   }else{
13865     LenColorHistory.style.setProperty('background-color','#fff','important');
13866   }
13867 }
13868 }
13869
13870 function InstaColor_Setup1(){
13871   window.addEventListener('mousemove',motionColor);
13872   window.addEventListener('scroll',scrollColor);
13873   //window.addEventListener('click',motionColor); // click is generated for animation
13874
13875   fi = document.getElementById('FeaturesView');
13876   if( fi != null ){
13877     fi.appendChild(InstaColorPicker);
13878     //cs.appendChild(InstaColorPicker);
13879     //cl.hidden = false;
13880     //cl.open = true;
13881   }
13882 }
13883
13884 function InstaColor_Setup(){
13885   if( CascadedCanvasBook Section.open ){
13886     InstaColor_Setup1();
13887   }
13888   CascadedCanvasBook_Summary.addEventListener('click',InstaColor_Setup1);
13889 }
13890 </script>
13891 <h3>Colors</h3>
13892
13893 <div id="CanvasColors">
13894 <div id="CanvasTool_Color_0" class="CanvasTool" onchange="showColorSample()">
13895 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13896 <input class="CV_Button" type="button" value="Trans">
13897 <span data-name="I" class="ColorParam" type="text">C0-0</span>
13898 <input data-name="BH" class="CanvasParam" type="text" value="0">
13899 <input data-name="FC" class="ColorParam" type="text" value="#000000">
13900 <input data-name="FO" class="CanvasParam" type="text" value="0.0">
13901 <span data-name="FCS" class="ColorSample">xxxx</span>
13902 <span data-name="BCS" class="ColorSample">xxxx</span>
13903 <input data-name="BC" class="ColorParam" type="text" value="#000000">
13904 <input data-name="BO" class="CanvasParam" type="text" value="0.0">
13905 </div>
13906 <div id="CanvasTool_Color_1" class="CanvasTool" onchange="showColorSample()">
13907 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13908 <input class="CV_Button" type="button" value="Mono">
13909 <span data-name="I" class="ColorParam" type="text">C0-1</span>
13910 <input data-name="BH" class="CanvasParam" type="text" value="1">
13911 <input data-name="FC" class="ColorParam" type="text" value="#000000">
13912 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13913 <span data-name="FCS" class="ColorSample">xxxx</span>
13914 <span data-name="BCS" class="ColorSample">xxxx</span>
13915 <input data-name="BC" class="ColorParam" type="text" value="#d0d0d0">
13916 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13917 </div>
13918 <div id="CanvasTool_Color_2" class="CanvasTool" onchange="showColorSample()">
13919 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13920 <input class="CV_Button" type="button" value="Red">
13921 <span data-name="I" class="ColorParam" type="text">C0-2</span>
13922 <input data-name="BH" class="CanvasParam" type="text" value="1">
13923 <input data-name="FC" class="ColorParam" type="text" value="#82020">
13924 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13925 <span data-name="FCS" class="ColorSample">xxxx</span>
13926 <span data-name="BCS" class="ColorSample">xxxx</span>
13927 <input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13928 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13929 </div>
13930 <div id="CanvasTool_Color_3" class="CanvasTool" onchange="showColorSample()">
13931 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13932 <input class="CV_Button" type="button" value="Green">
13933 <span data-name="I" class="ColorParam" type="text">C0-3</span>
13934 <input data-name="BH" class="CanvasParam" type="text" value="1">
13935 <input data-name="FC" class="ColorParam" type="text" value="#20c820">
13936 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13937 <span data-name="FCS" class="ColorSample">xxxx</span>
13938 <span data-name="BCS" class="ColorSample">xxxx</span>
13939 <input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13940 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13941 </div>
13942 <div id="CanvasTool_Color_4" class="CanvasTool" onchange="showColorSample()">
13943 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13944 <input class="CV_Button" type="button" value="Blue">
13945 <span data-name="I" class="ColorParam" type="text">C0-4</span>
13946 <input data-name="BH" class="CanvasParam" type="text" value="1">
13947 <input data-name="FC" class="ColorParam" type="text" value="#6080f0">
13948 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13949 <span data-name="FCS" class="ColorSample">xxxx</span>
13950 <span data-name="BCS" class="ColorSample">xxxx</span>
13951 <input data-name="BC" class="ColorParam" type="text" value="#c0c0c0">
13952 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13953 </div>
13954 <div id="CanvasTool_Color_5" class="CanvasTool" onchange="showColorSample()">
13955 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13956 <input class="CV_Button" type="button" value="Yellow">
13957 <span data-name="I" class="ColorParam" type="text">C0-5</span>
13958 <input data-name="BH" class="CanvasParam" type="text" value="1">
13959 <input data-name="FC" class="ColorParam" type="text" value="#fae600">
13960 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13961 <span data-name="FCS" class="ColorSample">xxxx</span>
13962 <span data-name="BCS" class="ColorSample">xxxx</span>
13963 <input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13964 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13965 </div>
13966 </div>
13967
13968 <script>
13969 // https://developer.mozilla.org/en-US/docs/Web/CSS/color_value
13970 function genCSSColorStyle(color,opa){
13971   opa = parseFloat(opa);
13972   opa = 0xFF; opa = opa.toFixed(0);
13973   if( 0xFF < opa ) opa = 0xFF;
13974   opa = parseInt(opa);
13975   opa = opa.toString(16);
13976   if( opa < 0x10 ) opa = '0' + opa;
13977   color += opa;
13978   return color;
13979 }
13980
13981 function CV_GenColorStyle(cole,forFill){
13982   if( forFill ){

```

```

13983     col = childByName(cole, 'FC').value;
13984     opa = childByName(cole, 'FO').value;
13985     }else{
13986     col = childByName(cole, 'BC').value;
13987     opa = childByName(cole, 'BO').value;
13988     }
13989     color = genCSSColorStyle(col,opa);
13990     return color;
13991 }
13992 function xxxshowColorSample(){
13993     t = event.target;
13994     p = t.parentNode;
13995     alert('showColorSample '+event.target.nodeName+'/#'+p.id);
13996 }
13997 function showColorSample(){
13998     var csv = CanvasColors.children;
13999     //colors.length;
14000     for( i = 0; i < csv.length; i++){
14001         fc = childByName(csv[i], 'FC').value;
14002         bc = childByName(csv[i], 'BC').value;
14003         //colors.log('colors='+csv[i].id+' fc='+fc+' bc='+bc);
14004         fcs = childByName(csv[i], 'FCS');
14005         fcs.style.color = fc;
14006         fcs.style.borderColor = fc;
14007         fcs.style.backgroundColor = bc;
14008     }
14009     bcs = childByName(csv[i], 'BCS');
14010     bcs.style.color = bc;
14011     bcs.style.borderColor = fc;
14012     bcs.style.backgroundColor = fc;
14013     }
14014     CV_redrawParts();
14015 }
14016 </script>
14017
14018 <style>
14019 ColorSample {
14020     color:#fff;
14021     background-color:#ff0;
14022     border:1px solid #000;
14023     margin:0px;
14024     padding:0px;
14025     width:12pt !important;
14026     height:12pt !important;
14027 }
14028 ColorParam {
14029     color:#000 !important;
14030     font-family:Courier New !important;
14031     font-size:9pt !important;
14032     padding:2px !important;
14033     line-height:1.1 !important;
14034     height:14pt !important;
14035     width:50pt !important;
14036     text-align:left !important;
14037     display:inline !important;
14038     vertical-align:middle !important;
14039 }
14040 CanvasParam {
14041     color:#000 !important;
14042     font-family:Courier New, Monospace !important;
14043     font-size:9pt !important;
14044     padding:2px !important;
14045     line-height:1.1 !important;
14046     height:14pt !important;
14047     width:30pt !important;
14048     text-align:right !important;
14049     display:inline !important;
14050     vertical-align:middle !important;
14051 }
14052 CV_Button {
14053     padding:2pt !important;
14054     line-height:1.1 !important;
14055     border:2px inset #bbb !important;
14056     font-size:9pt !important;
14057     font-weight:normal !important;
14058     font-family:Georgia !important;
14059     border-radius:3px !important;
14060     color:#666; background-color:#66a !important;
14061     width:50pt;
14062 }
14063 xxHtmlCodeViewText {
14064     font-size:9pt;
14065     font-family:Courier New;
14066     white-space:pre;
14067 }
14068 }
14069 xxCV_Button {
14070     font-family:Arial, Monospace, Courier New;
14071     color:#000;
14072     font-size:9pt;
14073     line-height:1.2;
14074     width:50pt;
14075 }
14076 </style>
14077
14078 <h3>Parts</h3>
14079 <div id="AppendToCanvas" class="CanvasTool" draggable="true">
14080 <input class="CanvasParam" type="text" value="100" onwheel="OnWheelResize()" %
14081 <input class="CanvasParam" type="text" value="100" onwheel="OnWheelZoom()" %
14082 <input id="RedrawImmediate" type="checkbox" value="Redraw" checked="">Redraw Immediate
14083 </div>
14084 <span id="CanvasTools" class="CanvasTools" draggable="true" contenteditable="">
14085 <div id="CanvasTool_Clear" class="CanvasTool">
14086 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14087 <input data-name="rdr" class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14088 <span data-name="cvid" class="CanvasParam" type="text" value="CL-0</span>
14089 <input type="checkbox" value="Fill" checked="">
14090 <input data-name="X" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14091 <input data-name="Y" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14092 <input data-name="Z" class="CanvasParam" type="text" value="1" onwheel="OnWheelIntRedraw()">
14093 <input data-name="H" class="CanvasParam" type="text" value="1000" onwheel="OnWheelIntRedraw()">
14094 <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14095 <input data-name="C" class="CanvasParam" type="text" value="0">
14096 </div>
14097 <div id="CanvasTool_Rect" class="CanvasTool">
14098 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14099 <input data-name="rdr" class="CV_Button" type="button" value="Rect" onclick="CV_drawRect()">
14100 <span data-name="cvid" class="CanvasParam" type="text" value="RE-0</span>
14101 <input data-name="F" type="checkbox" value="Fill">
14102 <input data-name="X" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
14103 <input data-name="Y" class="CanvasParam" type="text" value="2" onwheel="OnWheelIntRedraw()">
14104 <input data-name="Z" class="CanvasParam" type="text" value="2" onwheel="OnWheelIntRedraw()">
14105 <input data-name="H" class="CanvasParam" type="text" value="120" onwheel="OnWheelIntRedraw()">
14106 <input data-name="R" class="CanvasParam" type="text" value="80" onwheel="OnWheelIntRedraw()">
14107 <input data-name="C" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14108 <input data-name="C" class="CanvasParam" type="text" value="1">
14109 </div>
14110 <div id="CanvasTool_Circle" class="CanvasTool">
14111 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14112 <input data-name="rdr" class="CV_Button" type="button" value="Circle" onclick="CV_drawCircle()">
14113 <span data-name="cvid" class="CanvasParam" type="text" value="CI-0</span>
14114 <input data-name="F" type="checkbox" value="Fill" checked="">
14115 <input data-name="X" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
14116 <input data-name="Y" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()">
14117 <input data-name="Z" class="CanvasParam" type="text" value="3" onwheel="OnWheelIntRedraw()">
14118 <input data-name="R" class="CanvasParam" type="text" value="24" onwheel="OnWheelIntRedraw()">
14119 <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14120 <input data-name="E" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()">
14121 <input data-name="C" class="CanvasParam" type="text" value="2">
14122 </div>
14123 <div id="CanvasTool_Packman" class="CanvasTool">
14124 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14125 <input data-name="rdr" class="CV_Button" type="button" value="Packman" onclick="CV_drawPackman()">
14126 <span data-name="cvid" class="CanvasParam" type="text" value="PA-0</span>
14127 <input data-name="F" type="checkbox" value="Fill" checked="">
14128 <input data-name="X" class="CanvasParam" type="text" value="240" onwheel="OnWheelIntRedraw()">
14129 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()">
14130 <input data-name="Z" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()">
14131 <input data-name="H" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()">
14132 <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14133 <input data-name="E" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
14134 <input data-name="C" class="CanvasParam" type="text" value="5">
14135 </div>
14136 <div id="CanvasTool_Ellipse" class="CanvasTool">
14137 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14138 <input data-name="rdr" class="CV_Button" type="button" value="Ellipse" onclick="CV_drawEllipse()">
14139 <span data-name="cvid" class="CanvasParam" type="text" value="EI-0</span>
14140 <input data-name="F" type="checkbox" value="Fill" checked="">
14141 <input data-name="X" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
14142 <input data-name="Y" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
14143 <input data-name="Z" class="CanvasParam" type="text" value="4" onwheel="OnWheelIntRedraw()">
14144 <input data-name="RX" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()">
14145 <input data-name="RY" class="CanvasParam" type="text" value="20" onwheel="OnWheelIntRedraw()">
14146 <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14147 <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()">
14148 <input data-name="E" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()">
14149 <input data-name="C" class="CanvasParam" type="text" value="4">
14150 </div>
14151
14152 <div id="CanvasTool_Balloon" class="CanvasTool">
14153 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
14154 <input data-name="rdr" class="CV_Button" type="button" value="Balloon" onclick="CV_drawBalloon()">
14155 <span data-name="cvid" class="CanvasParam" type="text" value="BA-0</span>
14156 <input data-name="F" type="checkbox" value="Fill">
14157 <input data-name="X" class="CanvasParam" type="text" value="280" onwheel="OnWheelIntRedraw()">
14158 <input data-name="Y" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()">
14159 <input data-name="Z" class="CanvasParam" type="text" value="5" onwheel="OnWheelIntRedraw()">

```



```

14160 <input data-name="RX" class="CanvasParam" type="text" value="50" onwheel="OnWheelIntRedraw()>
14161 <input data-name="RY" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()>
14162 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()>
14163 <input data-name="S" class="CanvasParam" type="text" value="120" onwheel="OnWheelIntRedraw()>
14164 <input data-name="B" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()>
14165 <input data-name="C" class="CanvasParam" type="text" value="4">
14166 </div>
14167
14168 <div id="CanvasTool_Piechart" class="CanvasTool">
14169 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()>
14170 <input data-name="rdr" class="CV_Button" type="button" value="Piechart" onclick="CV_drawPiechart()>
14171 <span data-name="cvid" class="CanvasParam" type="text" value="BA-0c/span>
14172 <input data-name="F" type="checkbox" value="Fill" checked="">
14173 <input data-name="X" class="CanvasParam" type="text" value="350" onwheel="OnWheelIntRedraw()>
14174 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()>
14175 <input data-name="Z" class="CanvasParam" type="text" value="7" onwheel="OnWheelIntRedraw()>
14176 <input data-name="RW" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()>
14177 <input data-name="RV" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()>
14178 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()>
14179 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()>
14180 <input data-name="B" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()>
14181 <input data-name="C" class="CanvasParam" type="text" value="3">
14182 </div>
14183
14184 <div id="CanvasTool_XArc1" class="CanvasTool">
14185 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()>
14186 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()>
14187 <span data-name="cvid" class="CanvasParam" type="text" value="XA-0c/span>
14188 <input data-name="F" type="checkbox" value="Fill" checked="">
14189 <input data-name="X" class="CanvasParam" type="text" value="460" onwheel="OnWheelIntRedraw()>
14190 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()>
14191 <input data-name="Z" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()>
14192 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()>
14193 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()>
14194 <input data-name="R0" class="CanvasParam" type="text" value="150" onwheel="OnWheelIntRedraw()>
14195 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()>
14196 <input data-name="B" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()>
14197 <input data-name="C" class="CanvasParam" type="text" value="4">
14198 </div>
14199
14200 <div id="CanvasTool_XArc2" class="CanvasTool">
14201 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()>
14202 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()>
14203 <span data-name="cvid" class="CanvasParam" type="text" value="XA-1c/span>
14204 <input data-name="F" type="checkbox" value="Fill" checked="">
14205 <input data-name="X" class="CanvasParam" type="text" value="580" onwheel="OnWheelIntRedraw()>
14206 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()>
14207 <input data-name="Z" class="CanvasParam" type="text" value="8" onwheel="OnWheelIntRedraw()>
14208 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()>
14209 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()>
14210 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()>
14211 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()>
14212 <input data-name="B" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()>
14213 <input data-name="C" class="CanvasParam" type="text" value="2">
14214 </div>
14215
14216 <canvas id="CV_partsCanvas" data-name="canvas" class="CS_Canvas" width="740" height="200"></canvas>
14217 </span>
14218 <script>
14219 function CV_redrawParts(){
14220 return; // 2020-1120
14221 // search Z-Index and sort
14222 var parts = CanvasTools.children;
14223 //console.log("parts="+parts.length);
14224 np = [];
14225 for( i = 0; i < parts.length; i++ ){
14226     z = parts[i];
14227     z = childByName(p,'z');
14228     if( z != null ){
14229         //console.log("P#'+p.id+' z="+z+' '+z.value);
14230         np.push([z.value,p]);
14231     }
14232 }
14233 np.sort(function(np1,np2){ return np1[0] - np2[0]; });
14234 CV_clearRect();
14235 for( i = 0; i < np.length; i++ ){
14236     p = np[i][1];
14237     redraw = childByName(p,'rdr');
14238     //console.log("Redraw z="+np[i][0]+' #' +np[i][1].id+' redraw="+redraw.onclick);
14239     if( redraw != null ){
14240         redraw.click();
14241     }
14242 }
14243 }
14244
14245 function DrawingCanvas_Setup(){
14246 showColorSample();
14247 CV_redrawParts();
14248 }
14249
14250 function OnWheelZoom(){
14251 val = OnWheelInt();
14252 canvas = CV_partsCanvas;
14253 canvas.style.zoom = val + 's';
14254 CV_redrawParts();
14255 }
14256
14257 function OnWheelResize(){
14258 val = OnWheelInt();
14259 canvas = CV_partsCanvas;
14260 if( !canvas.hasAttribute('data-width') ){
14261     w = canvas.width;
14262     h = canvas.height;
14263     canvas.setAttribute('data-width',w);
14264     canvas.setAttribute('data-height',h);
14265     sw = canvas.getAttribute('data-width');
14266     sh = canvas.getAttribute('data-height');
14267     console.log("Zoom save original w="+w+',h="+h+' sw="+sw+', sh="+sh);
14268 }
14269 w = canvas.getAttribute('data-width');
14270 h = canvas.getAttribute('data-height');
14271 console.log("Zoom got original size w="+w+' h="+h);
14272 mw = w * (val/100.0);
14273 mh = h * (val/100.0);
14274 //console.log("Zoom mw="+mw+' nh="+nh);
14275 CV_partsCanvas.width = mw;
14276 CV_partsCanvas.height = mh;
14277 CV_redrawParts();
14278 }
14279
14280 function OnWheelIntRedraw(){
14281 OnWheelInt();
14282 t = event.target;
14283 n = t.nodeName;
14284 i = t.id;
14285 p = t.parentNode;
14286 Y = event.deltaY.toFixed(0);
14287 //console.log("OnWheelIntRedraw '+y+' '+n+'#'+i+' '+t.value+' #' +p.id);
14288 if( true ){
14289     CV_redrawParts();
14290 }else{
14291     if( RedrawImmediate.checked ){
14292         CV_clearRect();
14293         //If( p.id == 'CanvasTool_Circle' ){ CV_drawCircle(CanvasTool_Circle); }
14294         //If( p.id == 'CanvasTool_Rect' ){ CV_drawRect(CanvasTool_Rect); }
14295         CV_drawCircle(CanvasTool_Circle);
14296         CV_drawRect(CanvasTool_Rect);
14297     }
14298 }
14299 }
14300
14301 function CV_setCtxStyle(ctx,p){
14302 C = childByName(p,'C').value;
14303 c = document.getElementById('CanvasTool_Color'+C);
14304 ctx.fillStyle = CV_GenColorStyle(c,true);
14305 ctx.strokeStyle = CV_GenColorStyle(c,false);
14306 return ctx;
14307 }
14308
14309 function CV_clearRect(){
14310 cv = document.getElementById('CV_partsCanvas');
14311 ctx = cv.getContext('2d');
14312 ctx.clearRect(0,0,cv.width,cv.height);
14313 }
14314
14315 function CV_drawRect1(rect){
14316 canvas = document.getElementById('CV_partsCanvas');
14317 ctx = canvas.getContext('2d');
14318 ctx = CV_setCtxStyle(ctx,p);
14319
14320 p = rect;
14321 F = childByName(p,'F').checked;
14322 X = childByName(p,'X').value;
14323 Y = childByName(p,'Y').value;
14324 Z = childByName(p,'Z').value;
14325 p.style.zIndex = Z;
14326 W = childByName(p,'W').value;
14327 H = childByName(p,'H').value;
14328
14329 if( F ){
14330     ctx.fillRect(X,Y,W,H);
14331 }else{
14332     ctx.strokeRect(X,Y,W,H);
14333 }
14334 saveDrawing();
14335 }
14336
14337 function CV_drawRect(ctx){
14338 CV_drawRect1(CanvasTool_Rect);
14339 }
14340
14341 function CV_drawCircle(circle){

```

```

14337 canvas = document.getElementById('CV_partsCanvas');
14338 ctx = canvas.getContext('2d');
14339 ctx = CV_setCtxStyle(ctx,p);
14340
14341 p = circle;
14342 F = childByName(p,'F').checked;
14343 X = childByName(p,'X').value;
14344 Y = childByName(p,'Y').value;
14345 Z = childByName(p,'Z').value;
14346 p.style.zIndex = Z;
14347 R = childByName(p,'R').value;
14348 S = childByName(p,'S').value;
14349 E = childByName(p,'E').value;
14350
14351 //console.log('Circle'+X+', '+Y+' F='+F);
14352 ctx.beginPath();
14353 SA = (S / 180) * Math.PI;
14354 EA = (E / 180) * Math.PI;
14355 ctx.arc(X,Y,R,SA,EA);
14356 if( F ){
14357   ctx.fill();
14358 }else{
14359   ctx.stroke();
14360 }
14361 saveDrawing();
14362
14363 function CV_drawCircle(){
14364   CV_drawCircle1(CanvasTool_Circle);
14365 }
14366 function CV_drawEllipsel(circle){
14367   canvas = document.getElementById('CV_partsCanvas');
14368   ctx = canvas.getContext('2d');
14369   ctx = CV_setCtxStyle(ctx,p);
14370
14371   p = circle;
14372   X = childByName(p,'X').value;
14373   Y = childByName(p,'Y').value;
14374   Z = childByName(p,'Z').value;
14375   RX = childByName(p,'RX').value;
14376   RY = childByName(p,'RY').value;
14377   p.style.zIndex = Z;
14378   RO = childByName(p,'RO').value;
14379   S = childByName(p,'S').value;
14380   E = childByName(p,'E').value;
14381
14382   ctx.beginPath();
14383   SA = (S / 180) * Math.PI;
14384   EA = (E / 180) * Math.PI;
14385   ROA = (RO / 180) * Math.PI;
14386   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14387   F = childByName(p,'F').checked;
14388
14389   if( F ){
14390     ctx.fill();
14391   }
14392   // if( S ){
14393   {
14394     ctx.stroke();
14395   }
14396   saveDrawing();
14397 }
14398
14399 function CV_drawEllipse(){
14400   CV_drawEllipsel(CanvasTool_Ellipse);
14401 }
14402 function CV_drawBaloon1(baloon){
14403   canvas = document.getElementById('CV_partsCanvas');
14404   ctx = canvas.getContext('2d');
14405   ctx = CV_setCtxStyle(ctx,p);
14406
14407   p = baloon;
14408   F = childByName(p,'F').checked;
14409   X = childByName(p,'X').value;
14410   Y = childByName(p,'Y').value;
14411   Z = childByName(p,'Z').value;
14412   RX = childByName(p,'RX').value;
14413   RY = childByName(p,'RY').value;
14414   p.style.zIndex = Z;
14415   RO = childByName(p,'RO').value;
14416   S = childByName(p,'S').value;
14417   E = childByName(p,'E').value;
14418
14419   //console.log('Ellipse'+X+', '+Y+' F='+F);
14420   ctx.beginPath();
14421
14422   SA = (S / 180) * Math.PI;
14423   EA = (E / 180) * Math.PI;
14424   ROA = (RO / 180) * Math.PI;
14425   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14426
14427   PX = parseInt(X);
14428   PY = parseInt(Y);
14429   //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14430   PX -= 25;
14431   PY += 40;
14432   //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14433   ctx.lineTo(PX,PY);
14434   ctx.closePath();
14435
14436   if( F ){
14437     ctx.fill();
14438   }else{
14439     ctx.stroke();
14440   }
14441   saveDrawing();
14442 }
14443 function CV_drawBaloon(){
14444   CV_drawBaloon1(CanvasTool_Baloon);
14445 }
14446 function CV_drawPackman(circle){
14447   canvas = document.getElementById('CV_partsCanvas');
14448   ctx = canvas.getContext('2d');
14449   ctx = CV_setCtxStyle(ctx,p);
14450
14451   p = circle;
14452   F = childByName(p,'F').checked;
14453   X = childByName(p,'X').value;
14454   Y = childByName(p,'Y').value;
14455   Z = childByName(p,'Z').value;
14456   p.style.zIndex = Z;
14457   R = childByName(p,'R').value;
14458   S = childByName(p,'S').value;
14459   E = childByName(p,'E').value;
14460
14461   //console.log('Packman'+X+', '+Y+' F='+F);
14462   ctx.beginPath();
14463   SA = (S / 180) * Math.PI;
14464   //SA += 0.15 * Math.PI;
14465   EA = SA + Math.PI; //(E / 180) * Math.PI;
14466   ctx.arc(X,Y,R,SA,EA);
14467   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14468
14469   ctx.beginPath();
14470   E = 180 - E;
14471   SA += (E/180) * Math.PI;
14472   EA += (E/180) * Math.PI;
14473   ctx.arc(X,Y,R,SA,EA);
14474   if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14475
14476   saveDrawing();
14477 }
14478 function CV_drawPackman(){
14479   CV_drawPackman1(CanvasTool_Packman);
14480 }
14481 function CV_drawPiechart1(baloon){
14482   canvas = document.getElementById('CV_partsCanvas');
14483   ctx = canvas.getContext('2d');
14484   ctx = CV_setCtxStyle(ctx,p);
14485
14486   p = baloon;
14487   F = childByName(p,'F').checked;
14488   X = childByName(p,'X').value;
14489   Y = childByName(p,'Y').value;
14490   Z = childByName(p,'Z').value;
14491   RX = childByName(p,'RX').value;
14492   RY = childByName(p,'RY').value;
14493   p.style.zIndex = Z;
14494   RO = childByName(p,'RO').value;
14495   S = childByName(p,'S').value;
14496   E = childByName(p,'E').value;
14497
14498   //console.log('Ellipse'+X+', '+Y+' F='+F);
14499   ctx.beginPath();
14500
14501   SA = (S / 180) * Math.PI;
14502   EA = (E / 180) * Math.PI;
14503   ROA = (RO / 180) * Math.PI;
14504   ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14505
14506   PX = parseInt(X);
14507   PY = parseInt(Y);
14508   //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14509   //PX -= 25;
14510   //PY += 40;
14511   //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14512   ctx.lineTo(PX,PY);
14513   ctx.closePath();

```

```

14514
14515     if( F ){
14516         ctx.fill();
14517     }else{
14518         ctx.stroke();
14519     }
14520     saveDrawing();
14521 }
14522 function CV_drawPiechart(){
14523     CV_drawPiechart(CanvasTool_Piechart);
14524 }
14525 function CV_drawArcI(baloon){
14526     canvas = document.getElementById('CV_partsCanvas');
14527     ctx = canvas.getContext('2d');
14528     ctx = CV_setCtxStyle(ctx,p);
14529
14530
14531     p = baloon;
14532     F = childByName(p,'F').checked;
14533     X = childByName(p,'X').value;
14534     Y = childByName(p,'Y').value;
14535     Z = childByName(p,'Z').value;
14536     RX = childByName(p,'RX').value;
14537     RY = childByName(p,'RY').value;
14538     p.style.zIndex = Z;
14539     RO = childByName(p,'RO').value;
14540     S = childByName(p,'S').value;
14541     E = childByName(p,'E').value;
14542
14543     //console.log('Ellipse'+X+', '+Y+' F='+F);
14544     ctx.beginPath();
14545
14546     if( true ){
14547         d = new Date();
14548         ms = d.getTime();
14549         id = (ms % 300) / 10;
14550         //S = parseFloat(E) + id/2;
14551         E = parseFloat(E) - id;
14552         xd = (ms % 10000) / 5;
14553         if( 1000 < xd ){
14554             xd = 2000 - xd;
14555         }
14556         xd *= 0.8;
14557         X = parseFloat(X) + xd - 600;
14558         //console.log('Ellipse S='+S+ ', E='+E + ' id='+id);
14559
14560         SA = (S / 180) * Math.PI;
14561         EA = (E / 180) * Math.PI;
14562         ROA = (RO / 180) * Math.PI;
14563         ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14564     }
14565
14566     PX = parseInt(X);
14567     PY = parseInt(Y);
14568
14569     //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14570     //console.log('Ellipse B '+EX+', '+EY+' F='+F);
14571
14572     ctx.lineTo(PX,PY);
14573     ctx.closePath();
14574
14575     if( F ){
14576         ctx.fill();
14577     }else{
14578         ctx.stroke();
14579     }
14580     saveDrawing();
14581 }
14582 function CV_drawXArcI(){
14583     t = event.target;
14584     CV_drawXArcI(t.parentNode);
14585 }
14586 var AnimateIvl = window.setInterval(CV_redrawParts,30);
14587
14588 </script>
14589
14590 <h3>Animation</h3>
14591 <span id="Animation" class="CanvasTool" draggable="true" contenteditable="">>
14592 <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14593 <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14594 <span data-name="ovid" class="CanvasParam" type="text">ANIMA-0</span>
14595 <input data-name="R" class="ColorParam" type="text" value="Rotat">
14596 <input data-name="T" class="CanvasParam" type="text" value="30" onwheel="OnWheelInt()">ms
14597 <input data-name="T" class="CanvasParam" type="text" value="10" onwheel="OnWheelInt()">s
14598 </span>
14599
14600 <h3>Canvas</h3>
14601 <span id="AppendToCanvas" class="CanvasTool" draggable="true">
14602 <input class="CV_Button" type="button" value="Append" the above part
14603 </span>
14604 <span id="CanvasWrapTemplate" class="CanvasWrap" draggable="true">
14605 <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14606 <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14607 <span data-name="ovid" class="CanvasParam" type="text">0</span>
14608 <input data-name="width" class="CanvasParam" type="text" onchange="CS_setSize()" value="700">
14609 <input data-name="height" class="CanvasParam" type="text" onchange="CS_setSize()" value="400">
14610 <input data-name="zoom" class="CanvasParam" type="text" onchange="CS_setSize()" value="100" onwheel="OnWheelCanvasZoom()">*
14611 <input class="CV_Button" type="button" value="Remove" onclick="removeParent()"><br>
14612 <span id="DrawingCanvas" data-name="canvas" class="CS_Canvas" width="700" height="400"></canvas>
14613 </span>
14614 </script>
14615 function OnWheelCanvasZoom(){
14616     val = OnWheelInt();
14617     DrawingCanvas.style.zoom = val + '%';
14618     //CanvasWrapTemplate.style.width = (700*(val/100.0)+20) + 'px';
14619     CanvasWrapTemplate.style.height = (400*(val/100.0)+30) + 'px';
14620 }
14621 </script>
14622
14623 <h3>CanvasBook</h3>
14624 <div id="CanvasBook"></div>
14625 <br>
14626 <style>
14627 .CanvasBook {
14628     overflow:scroll;
14629 }
14630 .CBPanel {
14631 }
14632 .CanvasTool {
14633     font-size:9pt;
14634     font-family:Courier New;
14635     color:#000 !important;
14636     xborder:1px solid #aaf;
14637     background-color:rgba(127,127,0.5);
14638     width:740px;
14639     white-space:nowrap;
14640     xheight:30;
14641     margin:4px;
14642     padding-left:4px;
14643     padding-right:4px;
14644     overflow:auto;
14645     display:inline-block;
14646     resize:both;
14647     vertical-align:top;
14648     zoom:1.0;
14649 }
14650 .CanvasWrap {
14651     font-size:9pt;
14652     font-family:Courier New;
14653     color:#000 !important;
14654     border:1px solid #aaf;
14655     background-color:rgba(200,200,0.2);
14656     width:740px;
14657     height:430px;
14658     margin:4px;
14659     padding-left:4px;
14660     padding-right:4px;
14661     overflow:auto;
14662     display:inline-block;
14663     resize:both;
14664     vertical-align:top;
14665     zoom:1.0;
14666 }
14667 .CS_Panel {
14668     padding:3px;
14669     vertical-align:middle;
14670 }
14671 .CS_Canvas {
14672     border:1px dashed #fcc;
14673     resize:both;
14674     zoom:1.0;
14675 }
14676 </style>
14677 </script>
14678 var CanvasID = 0;
14679 function removeParent(){
14680     e = event.target;
14681     p = e.parentNode;
14682     if( p == CanvasWrapTemplate ){
14683         return;
14684     }
14685     pp = p.parentNode;
14686     alert('removeParent #' + pp.id + '/' + p.id + '/' + e.id);
14687     p.parentNode.removeChild(p);
14688 }
14689 function cloneParent(){
14690     b = event.target;

```

```

14691 w = b.parentNode;
14692 CanvasID += 1;
14693 //pp w.parentNode;
14694 cw = w.cloneNode(true);
14695 cw.id = "Canvas_" + CanvasID;
14696 childByName(cw, 'cvid').innerHTML = CanvasID;
14697 childByName(cw, 'cvid').value = CanvasID;
14698 CanvasBook.appendChild(cw);
14699 }
14700 function CS_resize(){
14701 e = event.target;
14702 p = e.parentNode;
14703 console.log("resize " + e.nodeName + " + p.nodeName);
14704 c = childByName(p, 'canvas');
14705 w = childByName(p, 'width').value;
14706 h = childByName(p, 'height').value;
14707 console.log("resize " + c.nodeName + " + w + " + h);
14708 c.width = w;
14709 c.height = h;
14710 p.style.width = w + 'px';
14711 p.style.height = h + 'px';
14712 console.log("c=" + c + " + w + " + h + " + c.width + " + h + " + c.height);
14713 }
14714 function CS_newFunc(){
14715 cvt = CanvasWrapTemplate;
14716 cvw = CanvasWrapTemplate.cloneNode(true); //needs an argument, otherwise 'function not found'
14717 CanvasID += 1;
14718 cw.id = "Canvas_" + CanvasID;
14719 //childByName(cw, 'cvid').contentEditable = false;
14720 childByName(cw, 'cvid').innerHTML = CanvasID;
14721 childByName(cw, 'cvid').value = CanvasID;
14722 childByName(cw, 'width').value = w = childByName(cvt, 'width').value;
14723 childByName(cw, 'height').value = h = childByName(cvt, 'height').value;
14724 cvw.style.width = w + 'px';
14725 //ncv = document.createElement('canvas');
14726 ncv = childByName(cw, 'canvas');
14727 //ncv.setAttribute('class', 'CS_Canvas');
14728 //ncv.setAttribute('data-name', 'canvas');
14729 ncv.width = w;
14730 ncv.height = h;
14731 //cvw.replaceChild(ncv, childByName(cw, 'canvas'));
14732 CanvasBook.appendChild(cw);
14733 }
14734 //CS_new.addEventListener('click', CS_newFunc);
14735 </script>
14736
14737
14738 <input id="CanvasBook_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14739 <input id="CanvasBook_WorkCodeViewOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14740 <input id="CanvasBook_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14741 <span id="CanvasBook_WorkCodeView"></span>
14742 <script id="CanvasBook_WorkCodeView">
14743 function CanvasBook_openWorkCodeView(){
14744 function CanvasBook_showWorkCode(){
14745 showHtmlCode(CanvasBook_WorkCodeView, CascadedCanvasBook_WorkCodeSpan);
14746 }
14747 CanvasBook_WorkCodeViewOpen.addEventListener('click', CanvasBook_showWorkCode);
14748 }
14749 CanvasBook_openWorkCodeView(); // should be invoked by an event
14750 </script>
14751 </details>
14752 <!-- CanvasBook_WorkCodeSpan -->
14753 </span>
14754 <!-- Work -->
14755
14756
14757 <!-- Work { -->
14758 <span id="SVG_WorkCodeSpan" open=""><a href="#SVG">SVG</a></span></a>
14759 </span>
14760 <details id="SVGfacesSection"><summary id="SVGfacesSummary">SVG Getting Started</summary>
14761 <!-- SVG // 2020-1108 SatoxITS -->
14762 <h2>Getting Started SVG</h2>
14763
14764 <div>
14765 <svg id="xSVG_01" class="SVG100">
14766 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14767 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14768 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14769 <circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14770 </svg>
14771
14772 <svg id="xSVG_02" class="SVG100" viewBox="0 0 100 100">
14773 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14774 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14775 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14776 <circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14777 </svg>
14778
14779 <svg id="xSVG_03" class="SVG100" viewBox="0 0 100 100">
14780 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14781 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14782 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14783 <path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14784 </svg>
14785
14786 <svg id="xSVG_04" class="SVG100" viewBox="0 0 100 100">
14787 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14788 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14789 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14790 <path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14791 </svg>
14792
14793 <svg id="xSVG_05" class="SVG100" viewBox="0 0 100 100">
14794 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="1" fill="#ffffff00"></circle>
14795 <circle cx="38" cy="42" r="4" stroke="black" stroke-width="1" fill="#00000000"></circle>
14796 <circle cx="62" cy="42" r="4" stroke="black" stroke-width="1" fill="#00000000"></circle>
14797 <path stroke="black" d="M 20 50 A 10 10 0 0 0 80 50" fill="#ffffff00"></path>
14798 </svg>
14799 </div>
14800
14801 <style>
14802 .SVG100 {
14803 width:100px;
14804 height:100px;
14805 }
14806 </style>
14807 <script>
14808 var svg_deg = 0;
14809 var SVGfacesInterval = 0;
14810 function rotatesVGfaces(){
14811 if( SVGfacesSection.open != true ){
14812 clearInterval(SVGfacesInterval);
14813 SVGfacesInterval = 0;
14814 return;
14815 }
14816 //ms = new Date().getTime();
14817 //de = (ms - toFixed(0)/10) % 360;
14818 svg_deg += 2;
14819 xSVG_04.style.transform = 'rotate('+svg_deg+'deg)';
14820 xSVG_05.style.transform = 'rotate('+svg_deg+'deg)';
14821 }
14822 function SVGfaces_Setup(){
14823 function startStopSVGfaces(){
14824 //if( SVGfacesSection.open == true ){ // not yet open when clicked
14825 if( SVGfacesInterval == 0 ){
14826 SVGfacesInterval = window.setInterval(rotatesVGfaces,100);
14827 }
14828 }
14829 }
14830 }
14831 SVGfacesSummary.addEventListener('click', startStopSVGfaces);
14832 document.addEventListener('load', startStopSVGfaces);
14833 }
14834 SVGfaces_Setup();
14835 </script>
14836
14837 <input id="SVG_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14838 <input id="SVG_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14839 <input id="SVG_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14840 <span id="SVG_WorkCodeView"></span>
14841 <script id="SVG_WorkCodeView">
14842 function SVG_openWorkCodeView(){
14843 function SVG_showWorkCode(){
14844 showHtmlCode(SVG_WorkCodeView, SVG_WorkCodeSpan);
14845 }
14846 SVG_WorkCodeViewOpen.addEventListener('click', SVG_showWorkCode);
14847 }
14848 SVG_openWorkCodeView(); // should be invoked by an event
14849 </script>
14850 </details>
14851 <!-- SVG_WorkCodeSpan -->
14852 </span>
14853 <!-- Work -->
14854
14855
14856 <!-- Work { -->
14857 <span id="Xyeyes_WorkCodeSpan">
14858 <details id="XyeyesSection"><summary id="XyeyesSummary">Xyeyes</summary>
14859 <a href="#Xyeyes">Xyeyes</a></span></a>
14860 <!-- Xyeyes // 2020-1117 SatoxITS -->
14861 <h2>Xyeyes</h2>
14862
14863 <style>
14864 .SVG100b { width:100px; height:100px; }
14865 </style>
14866 <div id="Xyeyes1">

```

```

1486<svg class="SVG100b" viewBox="0 0 100 100">
1487<circle cx="50" cy="50" r="40" stroke="black" stroke-width="1" fill="#ffff00ff"/></circle>
1488<circle cx="38" cy="42" r="10" stroke="black" stroke-width="1" fill="#000000ff"/></circle>
1489<circle cx="62" cy="42" r="10" stroke="black" stroke-width="1" fill="#000000ff"/></circle>
1490<path stroke="black" d="M 20 50 A 10 10 0 0 0 50" fill="#ffffff00"/></path>
1491</svg>
1492<svg class="SVG100b" viewBox="0 0 100 100">
1493<circle cx="50" cy="50" r="40" fill="#400000ff"/></circle>
1494<circle cx="50" cy="50" r="20" fill="#ffffffff"/></circle>
1495<circle cx="50" cy="50" r="10" fill="#0000ffff"/></circle>
1496</svg>
1497<svg class="SVG100b" viewBox="0 0 100 100">
1498<circle cx="50" cy="50" r="40" fill="#400000ff"/></circle>
1499<circle cx="64" cy="40" r="20" fill="#ffffffff"/></circle>
1500<circle cx="72" cy="34" r="10" fill="#0000ffff"/></circle>
1501</svg>
1502</div>
1503<div>
1504<svg id="XYeyes_0_0" class="SVG100b" viewBox="0 0 100 100">
1505<circle cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
1506<circle cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
1507<circle cx="50" cy="24" r="10" fill="#0000ffff"/></circle>
1508</svg>
1509<svg id="XYeyes_0_1" class="SVG100b" viewBox="0 0 100 100">
1510<circle cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
1511<circle cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
1512<circle cx="50" cy="24" r="10" fill="#0000ffff"/></circle>
1513</svg>
1514<div class="XYZvalues">
1515<input id="XYeyes_0_Move" class="XYZcheckbox" type="checkbox" checked/>Move
1516<input id="XYeyes_0_Cont" class="XYZcheckbox" type="checkbox" checked/>Cont
1517<input id="XYeyes_0_Degree" class="XYZvalue" type="text" value="0"/>
1518<input id="XYeyes_0_Dist0" class="XYZvalue" type="text" value="0"/>
1519<input id="XYeyes_0_Dist1" class="XYZvalue" type="text" value="0"/>
1520</div>
1521</div>
1522<script>
1523function XYeyes_Setup0(){
1524function lookat(eye,pos,x,y){
1525x0 = eye.getBoundingClientRect().left.toFixed(0);
1526y0 = eye.getBoundingClientRect().top.toFixed(0);
1527pos.value = x0 + ', ' + y0;
1528}
1529var rad = 0;
1530function rotate(){
1531if (XYeyesSection.open != true ){
1532clearInterval(XYinterval);
1533console.log('interval cleared');
1534return;
1535}
1536if ( !XYeyes_0_Move.checked ){
1537return;
1538}
1539if ( XYeyes_0_Cont.checked ){
1540rad += 2;
1541}
1542else{
1543rad += 12;
1544}
1545XYeyes_0_Degree.value = rad % 360;
1546XYeyes_0_0.style.transform = 'rotate('+rad+'deg)';
1547XYeyes_0_1.style.transform = 'rotate('+rad+'deg)';
1548lookat(XYeyes_0_0,XYeyes_0_Dist0,x,y);
1549lookat(XYeyes_0_1,XYeyes_0_Dist1,x,y);
1550}
1551function startStopXYeyes(){
1552if ( XYeyesSection.open == true ){
1553XYinterval = window.setInterval(rotate,100);
1554}
1555}
1556XYeyesSummary.addEventListener('click',startStopXYeyes);
1557document.addEventListener('load',startStopXYeyes);
1558}
1559XYeyes_Setup0();
1560</script>
1561<div id="XYeyes_1" class="XYeyes">
1562<h3>XY eyes/h3>
1563<svg id="XYeyes_1_x0" class="SVG100a" viewBox="0 0 100 100">
1564<circle cx="50" cy="50" r="40" stroke="black" stroke-width="6" fill="#ffffff00"/></circle>
1565<circle id="XYeyes_1_x0p" cx="50" cy="30" r="20" fill="#000000ff"/></circle>
1566</svg>
1567<svg id="XYeyes_1_x1" class="SVG100f" viewBox="0 0 100 100">
1568<circle cx="50" cy="50" r="40" stroke="black" stroke-width="6" fill="#ffffff00"/></circle>
1569<circle id="XYeyes_1_x1p" cx="50" cy="30" r="20" fill="#000000ff"/></circle>
1570</svg>
1571<svg id="XYeyes_1_0" class="SVG100c" viewBox="0 0 100 100">
1572<circle cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
1573<circle cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
1574<circle cx="50" cy="24" r="10" fill="#0000ffff"/></circle>
1575</svg>
1576<svg id="XYeyes_1_1" class="SVG100c" viewBox="0 0 100 100">
1577<circle data-name="ball" cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
1578<circle id="XYeyes_1_1_whyeye" data-name="whyeye" cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
1579<circle id="XYeyes_1_1_bleye" data-name="bleye" cx="50" cy="24" r="10" fill="#000000ff"/></circle>
1580</svg>
1581<svg id="XYeyes_1_2" class="SVG100c" viewBox="0 0 100 100">
1582<circle data-name="ball" cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
1583<circle id="XYeyes_1_2_whyeye" data-name="whyeye" cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
1584<circle id="XYeyes_1_2_bleye" data-name="bleye" cx="50" cy="24" r="10" fill="#000000ff"/></circle>
1585</svg>
1586<div class="XYZvalues">
1587<input id="XYeyes_1_Move" class="XYZcheckbox" type="checkbox" checked/>Move
1588<input id="XYeyes_1_MousePos" class="XYZvalue" type="text" value="0"/>
1589<input id="XYeyes_1_Distx0" class="XYZvalue" type="text" value="0"/>
1590<input id="XYeyes_1_Distx1" class="XYZvalue" type="text" value="0"/>
1591<input id="XYeyes_1_Disty0" class="XYZvalue" type="text" value="0"/>
1592<input id="XYeyes_1_Disty1" class="XYZvalue" type="text" value="0"/>
1593<input id="XYeyes_1_Dist2" class="XYZvalue" type="text" value="0"/>
1594</div>
1595</div>
1596<style>
1597.SVG100c { width:100px; height:100px; display:inline; }
1598.SVG100a { width:100px; height:100px; display:inline; }
1599.SVG100f { width:100px; height:100px; position:relative; left:-20px; display:inline; }
1600.XYeyes {
1601text-align:left;
1602}
1603.XYZcheckbox {
1604width:20px;
1605height:20px;
1606}
1607.XYZvalues {
1608z-index:2;
1609display:block;
1610background-color:#eee;
1611height:100px;
1612text-align:left;
1613}
1614.XYZvalue {
1615font-size:10pt !important;
1616width:70pt !important;
1617line-height:1.1 !important;
1618margin:1px !important;
1619display:inline !important;
1620text-align:right;
1621padding:1px !important;
1622}
1623</style>
1624<script>
1625var XYmyid = Math.random();
1626var XYsent = 0;
1627function XYeyes_recvMousePosition(msg){
1628argv = msg.split(',');
1629if( argv[0] == XYmyid ){
1630//console.log('XYeyes echo '+msg);
1631}
1632else{
1633XYrecv += 1;
1634//console.log('XYeyes recv '+XYrecv+' '+msg);
1635sx = argv[1];
1636sy = argv[2];
1637x = sx - window.screenX;
1638y = sy - window.screenY;
1639XYeyes_1_MousePos.value = 'R ' + x + ', ' + y;
1640XYlookat(XYeyes_1_x0,XYeyes_1_Distx0,x,y);
1641XYlookat(XYeyes_1_x1,XYeyes_1_Distx1,x,y);
1642}
1643}
1644var xye_gjlinked = false;
1645function XYeyes_sendMousePosition(event){
1646if( true ){
1647return; // 2020-1120
1648}
1649if( xye_gjlinked == false ){
1650GJ_Join();
1651xye_gjlinked = true;
1652}
1653}

```

```

15045 x = event.x;
15046 y = event.y;
15047 sx = window.screenX + x;
15048 sy = window.screenY + y;
15049 msg = 'MOUSEPOSITION '+XYmyid+', '+sx+', '+sy;
15050 CJ_BcastMessage(msg);
15051 XYsent += 1;
15052 //console.log('XYeyes sent '+XYsent+' '+msg);
15053 }
15054 function XYlookat(eye,pos,x,y){
15055 x0 = eye.getBoundingBoxRect().left.toFixed(0);
15056 y0 = eye.getBoundingBoxRect().top.toFixed(0);
15057 dx = x - x0;
15058 dx -= 65;
15059 dy = y - y0;
15060 dy -= 75;
15061 degx = Math.acos(dx/Math.sqrt(dx*dx+dy*dy));
15062 degx *= 180 / Math.PI;
15063 degx = degx.toFixed(0);
15064 degx = parseInt(degx);
15065 if( dx <= 0 && dy <= 0 ){
15066   degl = -45;
15067   degx = -(degx - 90);
15068 }else
15069 if( dx <= 0 && 0 < dy ){
15070   degl = -135;
15071   degx = degx + 90;
15072 }else
15073 if( 0 < dx && dy <= 0 ){
15074   degl = 45;
15075   degx = 90 - degx;
15076 }else{
15077   degl = 135;
15078   degx = degx + 90;
15079 }
15080 if( XYeyes_1.Cont.checked ){
15081   degl = degx;
15082 }
15083 eye.style.transform = 'rotate('+degl+'deg)';
15084 //pos.value = dx+', '+dy+' / '+degl+' '+degx;
15085 pos.value = dx+', '+dy+' / '+degl;
15086 }
15087 }
15088 function XYeyes_Setup1(){
15089 function motion(){
15090 XYeyes_sendMousePosition(event);
15091 if( XYeyes_1.Move.checked ){
15092   return;
15093 }
15094 x = event.x;
15095 y = event.y;
15096 XYeyes_1.MousePos.value = x + ', ' + y;
15097 XYlookat(XYeyes_1_0,XYeyes_1_dist0,x,y);
15098 XYlookat(XYeyes_1_1,XYeyes_1_dist1,x,y);
15099 XYlookat(XYeyes_1_2,XYeyes_1_dist2,x,y);
15100 XYlookat(XYeyes_1_x0,XYeyes_1_distx0,x,y);
15101 XYlookat(XYeyes_1_x1,XYeyes_1_distx1,x,y);
15102 }
15103 window.addEventListener('mousemove',motion);
15104 window.addEventListener('mouseover',motion);
15105 window.addEventListener('focusin',motion);
15106 window.addEventListener('focusout',motion);
15107 fi = document.getElementById('FeaturesView');
15108 if( fi != null ){
15109   //fi.appendChild(XYeyes_1);
15110 }
15111 XYeyes_Setup1();
15112 }
15113 }
15114 </script>
15115 </script>
15116 <input id="XYeyes_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15117 <input id="XYeyes_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15118 <input id="XYeyes_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15119 </span id="XYeyes_WorkCodeView"></span>
15120 <script id="XYeyes_WorkScript">
15121 function XYeyes_openWorkCodeView(){
15122   function XYeyes_showWorkCode(){
15123     showHtmlCode(XYeyes_WorkCodeView,XYeyes_WorkCodeSpan);
15124   }
15125   XYeyes_WorkCodeViewOpen.addEventListener('click',XYeyes_showWorkCode);
15126 }
15127 }
15128 XYeyes openWorkCodeView(); // should be invoked by an event
15129 </script>
15130 </script>
15131 <div id="XYeyes_WorkCodeSpan" -->
15132 </div>
15133 </div>
15134 <div id="XYeyes_WorkCodeSpan" -->
15135 </div>
15136 </div>
15137 </div>
15138 </div>
15139 </div>
15140 </div>
15141 </div>
15142 </div>
15143 </div>
15144 </div>
15145 </div>
15146 </div>
15147 </div>
15148 </div>
15149 </div>
15150 </div>
15151 </div>
15152 </div>
15153 </div>
15154 </div>
15155 </div>
15156 </div>
15157 </div>
15158 </div>
15159 </div>
15160 </div>
15161 </div>
15162 </div>
15163 </div>
15164 </div>
15165 </div>
15166 </div>
15167 </div>
15168 </div>
15169 </div>
15170 </div>
15171 </div>
15172 </div>
15173 </div>
15174 </div>
15175 </div>
15176 </div>
15177 </div>
15178 </div>
15179 </div>
15180 </div>
15181 </div>
15182 </div>
15183 </div>
15184 </div>
15185 </div>
15186 </div>
15187 </div>
15188 </div>
15189 </div>
15190 </div>
15191 </div>
15192 </div>
15193 </div>
15194 </div>
15195 </div>
15196 </div>
15197 </div>
15198 </div>
15199 </div>
15200 </div>
15201 </div>
15202 </div>
15203 </div>
15204 </div>
15205 </div>
15206 </div>
15207 </div>
15208 </div>
15209 </div>
15210 </div>
15211 </div>
15212 </div>
15213 </div>
15214 </div>
15215 </div>
15216 </div>
15217 </div>
15218 </div>
15219 </div>
15220 </div>
15221 </div>
15222 </div>

```

```

1522 <script>
1522 //<transform translation="0 0 0">
1522 //<transform id="Box1Box" translation="0 0 0">
1522 //</transform>
1522 function showVP(){
1522 // console.log('color0:'+Box1Material.getAttribute('diffusecolor'));
1522 m = document.getElementById('Box1Material');
1522 //console.log('color1:'+Box1Material.getFieldValue('diffusecolor'));
1522 // console.log('color1:'+m.getFieldValue('diffusecolor'));
1522 //console.log('rotation:'+Box1Box.getFieldValue('rotation'));
1522 // console.log('Box1TransRotation:'+Box1Trans.getFieldValue('rotation'));
1522
1522 e = document.getElementById('Box1View');
1522 // console.log('Box1View:'+e.runtime.properties());
1522 e = document.getElementById('X3dRofl_ViewPoint');
1522 // console.log("Box1Box:"+e.outerHTML);
1522
1522 //e.runtime.showAll();
1522 //e.runtime.resetView();
1522
1522 //console.log('Transform'+x3dom.runtime.getCurrentTransform(Box1View));
1522 //console.log('Transform'+Box1View.runtime.getCurrentTransform());
1522 //console.log('Transform'+Box1Scene.runtime.getCurrentTransform());
1522 //console.log('VPosition'+X3dRofl_ViewPoint.position);
1522 // console.log('VPosition'+X3dRofl_ViewPoint.getFieldValue('position'));
1522 // console.log('Vorientation'+X3dRofl_ViewPoint.getFieldValue('orientation'));
1522 //console.log('VPositionProp'+Box1View.runtime.properties());
1522 //console.log('VPositionProp'+X3dRofl_ViewPoint.runtime.properties());
1522 //sid = document.getElementById('Box1Scene');
1522 //console.log('Box1Scene'+sid.runtime.properties());
1522 //X3dRofl_ViewPoint.value = X3dRofl_ViewPoint.position;
1522
1522 X3dRofl_ViewPosition_Value.value = X3dRofl_ViewPoint.getFieldValue('position');
1522 X3dRofl_ViewOrientation_Value.value = X3dRofl_ViewPoint.getFieldValue('orientation');
1522 X3dRofl_BoxPosition.value = Box1Trans.getFieldValue('translation');
1522 X3dRofl_BoxRotation_Value.value = Box1Trans.getFieldValue('rotation');
1522
1522 box1 = document.getElementById('Box1');
1522 //X3dRofl_ViewOrientation_Value.value = box1.runtime.viewpoint();
1522 }
1522 //window.setInterval(showVP, 500);
1522 window.addEventListener('load', showVP);
1522
1522 function newVP(){
1522 console.log('ViewPoint changed:'+event);
1522
1522
1522 vp = document.getElementById('X3dRofl_ViewPoint');
1522 vp.addEventListener('viewpointChanged', newVP, false);
1522 x3dom.runtime.ready = function(){
1522 console.log('-- X3DOM ready ');
1522 }
1522 //x3dom.runtime.debug(true);
1522
1522 var svg_deg = 0;
1522 function rotateX3DOM(){
1522 svg_deg += 2;
1522 }
1522
1522 function X3DOM_Setup1(){
1522 X3dRofl_ViewPoint.setAttribute('position', '0.8 0.7 5');
1522 X3dRofl_ViewPoint.setAttribute('position', '0.8 0.7 4');
1522 Box1Material.setAttribute('diffusecolor', '0 1 0');
1522 Box1Trans.setAttribute('translation', '1.0 0.5 0.5');
1522 Box1Trans.setAttribute('rotation', '1 1 1');
1522 }
1522
1522 function X3DOM_Setup(){
1522 if( X3DOM_Section.open == true ){
1522 X3DOM_Setup1();
1522 }
1522 }
1522 X3DOM_Summary.addEventListener('click', X3DOM_Setup1);
1522 X3DOM_Setup();
1522 </script>
1522
1522 <h3>Canvas Smilly onto X3DOM</h3>
1522 <canvas id="Smilly_1" width="150" height="150"></canvas>
1522 <script>
1522 function draw_smile1() {
1522 var canvas = Smilly_1;
1522 if( canvas.getContext ){
1522 var ctx = canvas.getContext('2d');
1522 ctx.beginPath();
1522 ctx.arc(75, 75, 50, 0, Math.PI * 2, true); // Outer circle
1522 ctx.fillStyle = 'rgba(255,240,0,0.5)';
1522 ctx.fill();
1522
1522 ctx.beginPath();
1522 ctx.moveTo(110, 75);
1522 ctx.arc(75, 75, 35, 0, Math.PI, false); // Mouth (clockwise)
1522 ctx.moveTo(65, 65);
1522 ctx.stroke();
1522
1522 ctx.beginPath();
1522 ctx.arc(60, 65, 5, 0, Math.PI * 2, true); // Left eye
1522 ctx.moveTo(95, 65);
1522 ctx.arc(95, 65, 5, 0, Math.PI * 2, true); // Right eye
1522 ctx.stroke();
1522 ctx.fillStyle = 'rgba(0,0,0,0.8)';
1522 ctx.fill();
1522 }
1522 }
1522 function getSmillyUrl(){
1522 draw_smile1();
1522 url = Smilly_1.toDataURL("image/png");
1522 return url;
1522 }
1522 function putTextureTag(id){
1522 url = getSmillyUrl();
1522 document.write('<'+ImageTexture'
1522 + ' id'+id
1522 + ' url="'+url+'"><'+ImageTexture'');
1522 }
1522 </script>
1522
1522 <x3d width="150px" height="150px">
1522 <scene><shape>
1522 <appearance><ImageTexture url="WD-WallPaper03.png" /></appearance>
1522 <sphere DEF="obj" radius="3.3" />
1522 </shape></scene>
1522 </x3d>
1522
1522 <x3d width="150px" height="150px">
1522 <scene><shape>
1522 </
1522 <ImageTexture url="http://im3/gshell/skype-rofl-texture-02.png" />
1522 -->
1522 <script>putTextureTag("Box1Texture");</script>
1522 </appearance>
1522 <sphere DEF="obj" radius="3.3" />
1522 </shape></scene>
1522 </x3d>
1522
1522 <x3d width="150px" height="150px">
1522 <scene>
1522 <shape>
1522 <appearance><ImageTexture url="WD-WallPaper03.png" /></appearance>
1522 <sphere DEF="obj" radius="3.3" />
1522 </shape>
1522 <transform translation="1 1 0">
1522 <shape>
1522 <appearance>
1522 <script>putTextureTag("Box1Texture");</script>
1522 </
1522 <ImageTexture url="http://im3/gshell/skype-rofl-texture-02.png" />
1522 </appearance>
1522 <sphere DEF="obj" radius="3.3" />
1522 </shape>
1522 </translation>
1522 </scene>
1522 </x3d>
1522 <style>
1522 x3d {
1522 display:inline;
1522 }
1522 </style>
1522
1522 <h3>Rofl by Unicode 1F923</h3>
1522 <style>
1522 .largefont {font-size:32px !important; text-align:center !important; }
1522 .iconfont {font-size:64px !important; text-align:center !important; }
1522 .hugefont {font-size:160px !important; text-align:center !important; }
1522 </style>
1522 &#x1f923; = &#x1f923;
1522 <big>&#x1f923;</big>
1522 <span class="largefont">&#x1f923;</span>
1522 <span class="iconfont">&#x1f923;</span>
1522 <span class="hugefont">&#x1f923;</span>
1522 <br>
1522
1522 <input id="X3DOM_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1522 <input id="X3DOM_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1522 <input id="X3DOM_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1522 <span id="X3DOM_WorkCodeView"></span>
1522 <script id="X3DOM_WorkScript">
1522 function X3DOM_openWorkCodeView(){

```

```

15399 function X3DOM_showWorkCode(){
15400     showHtmlCode(X3DOM_WorkCodeView,X3DOM_WorkCodeSpan);
15401 }
15402 X3DOM_WorkCodeViewOpen.addEventListener('click',X3DOM_showWorkCode);
15403 }
15404 X3DOM_openWorkCodeView(); // should be invoked by an event
15405 </script>
15406 </details>
15407 <!--_3DRDFL_WorkCodeSpan -->
15408 *//</span>
15409 //<!-- ===== Work } ===== -->
15410
15411
15412
15413 //<!-- ===== Work { ===== -->
15414 //<span id="CloudChamber_WorkCodeSpan">
15415 /*
15416 <details id="CloudChamber Section"><summary id="CloudChamber_Summary">Cloud Chamber</summary>
15417 <!-- ----- Cloud Chamber // 2020-1117 SatoxITS { -->
15418 <h2>Cloud Chamber</h2>
15419
15420 <div id="Chamber_1">
15421 <h3>Cloud Chamber</h3>
15422 <input id="Chamber_Clear" class="HtmlCodeViewButton" type="button" value="Clear">
15423 <div id="Chamber_1_Sheet" class="CChamberSheet">
15424 <div id="Chamber_1_Box" class="CChamber"></div>
15425 </div>
15426 </div>
15427
15428 <style>
15429 @keyframes CCdot {
15430     0% {
15431         background-color:rgba(0,0,200,1.0);
15432         //background-color:rgba(255,255,255,1.0);
15433         background-opacity:1.0;
15434         xtop:0px;
15435     }
15436     20% {
15437         background-opacity:1.0;
15438         xtop:2px;
15439     }
15440     80% {
15441         background-opacity:0.0;
15442         xtop:8px;
15443     }
15444     100% {
15445         background-opacity:0.0;
15446         xtop:10px;
15447     }
15448 }
15449 .CChamberSheet {
15450     border:1px solid #000;
15451     padding:0px;
15452     overflow:visible;
15453     height:200px;
15454     xbackground-color:#202040ff;
15455 }
15456 .CChamber {
15457     xposition:absolute;
15458     xposition:relative;
15459     padding:0px;
15460 }
15461 .CCpoint {
15462     display:block;
15463     position:absolute;
15464     animation-name:CCdot;
15465     animation-duration:5s;
15466     animation-iteration-count:infinite;
15467     border:1px solid #ffffff80;
15468     width:30px;
15469     height:30px;
15470 }
15471 </style>
15472 </script>
15473 var CCpointId = 0;
15474 var CCpointsMax = 200;
15475 var CTrace = [
15476     [-20,10],
15477     [50,50],
15478     [150,70],
15479     [250,75],
15480     [300,70],
15481     [330,65],
15482     [360,60],
15483     [380,55],
15484 ];
15485 function addCCpoint(){
15486     t = event.target;
15487     //console.log('t.id='+t.id);
15488     if( t != CChamber_1_Sheet ){
15489         return;
15490     }
15491     //x0 = t.getBoudingClientRect().left.toFixed(0);
15492     //y0 = t.getBoudingClientRect().top.toFixed(0);
15493     x = event.offsetX; //event.x; // - event.pageX - event.offsetX;
15494     y = event.offsetY; //event.y; // - event.pageY - event.offsetY;
15495     CTrace[CTrace.length] = [x,y];
15496     CCadd(x,y);
15497     //console.log('points['+CTrace.length+' '+x+', '+y);
15498 }
15499 function CChamberClear(){
15500     while( 0 < CChamber_1_Box.children.length ){
15501         CChamber_1_Box.removeChild(CChamber_1_Box.children[0]);
15502     }
15503 }
15504 function CCaddlist(){
15505     for( i = 0; i < CTrace.length; i++ ){
15506         ccl = CTrace[i];
15507         CCadd(ccl[0],ccl[1]);
15508     }
15509 }
15510 function CCadd(x,y){
15511     ccp = document.createElement('span');
15512     ccp.setAttribute('class','CCpoint');
15513     CCpointId += 1;
15514     ccp.id = 'ccp_' + CCpointId;
15515     ccp.style.left = x + 'px';
15516     ccp.style.top = y + 'px';
15517     //ccp.style.backgroundColor = 'rgba(0,0,200,1.0)';
15518     CChamber_1_Box.appendChild(ccp);
15519     while( CCpointId < CChamber_1_Box.children.length ){
15520         CChamber_1_Box.removeChild(CChamber_1_Box.children[0]);
15521     }
15522 }
15523
15524 function CChamber_Setup(){
15525     CCaddlist();
15526     CChamber_1_Sheet.addEventListener('mousemove',addCCpoint);
15527     CChamber_1_Clear.addEventListener('click',CChamberClear);
15528     fi = document.getElementById('FeaturesView');
15529     if( fi != null ){
15530         //fi.appendChild(CChamber_1);
15531     }
15532 }
15533
15534 CChamber_Setup();
15535 </script>
15536
15537 <input id="CloudChamber_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15538 <input id="CloudChamber_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15539 <input id="CloudChamber_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15540 <span id="CloudChamber_WorkCodeView"></span>
15541 <script id="CloudChamber_WorkScript">
15542 function CloudChamber_openWorkCodeView(){
15543     function CloudChamber_showWorkCode(){
15544         showHtmlCode(CloudChamber_WorkCodeView,CloudChamber_WorkCodeSpan);
15545     }
15546     CloudChamber_WorkCodeViewOpen.addEventListener('click',CloudChamber_showWorkCode);
15547 }
15548 CloudChamber_openWorkCodeView(); // should be invoked by an event
15549 </script>
15550 </details>
15551 <!-- CloudChamber_WorkCodeSpan -->
15552 *//</span>
15553 //<!-- ===== Work } ===== -->
15554
15555
15556
15557
15558 //<!-- ===== Work { ===== -->
15559 //<span id="Template_WorkCodeSpan">
15560 /*
15561 <details><summary>Work Template</summary>
15562 <!-- ----- Template of Work// 2020-0928 SatoxITS { -->
15563 <h2>Template of Work</h2>
15564
15565 <style>
15566 </style>
15567 </style>
15568 </script>
15569 </script>
15570
15571 <input id="Template_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15572 <input id="Template_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15573 <input id="Template_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15574 <span id="Template_WorkCodeView"></span>
15575 <script id="Template_WorkScript">

```



```

15576 function Template_openWorkCodeView(){
15577   function Template_showWorkCode(){
15578     showHtmlCode(Template_WorkCodeView,Template_WorkCodeSpan);
15579   }
15580   Template_WorkCodeViewOpen.addEventListener('click',Template_showWorkCode);
15581 }
15582 Template_openWorkCodeView(); // should be invoked by an event
15583 </script>
15584 </details>
15585 <!-- Template_WorkCodeSpan -->
15586 * //</span>
15587 //<!-- ===== Work } ===== -->
15588
15589
15590
15591
15592 //<!-- ===== Work { ===== -->
15593 //<span id="OriginalSource WorkCodeSpan">
15594 *
15595 <details open=""><summary>Original Source</summary>
15596 <!-- OriginalSource // 2020-1009 SatonITS { -->
15597 <h2>Original Source of GShell</h2>
15598 <input id="OriginalSource_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15599 <input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15600 <input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15601 <span id="OriginalSourceTextElement"></span>
15602 <span id="OriginalSource_WorkCodeView"></span>
15603 <script id="OriginalSource_WorkScript">
15604   function OriginalSource_openWorkCodeView(){
15605     function OriginalSource_showWorkCode(){
15606       //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
15607       //OriginalSourceTextElement = OriginalSourceNode;
15608       //console.log('src='\n'+OriginalSourceNode.outerHTML);
15609       showHtmlCodeX(OriginalSource_WorkCodeView,OriginalSourceNode,
15610         '\n',
15611         '\n',true);
15612     }
15613     OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_showWorkCode);
15614   }
15615   //OriginalSourceNode = document.documentElement.cloneNode();
15616   //OriginalSourceNode = gsh.cloneNode(true); //=====
15617   //console.log('src='\n'+document.documentElement.outerHTML);
15618   //console.log('src='\n'+gsh.outerHTML);
15619   //console.log('src='\n'+OriginalSourceNode.innerHTML);
15620   OriginalSource_openWorkCodeView(); // should be invoked by an event
15621   //showNodeAsHtmlSource(OriginalSource_WorkCodeView,OriginalSourceNode);
15622   function SaveOriginalNode(){
15623     if( false ){
15624       m0 = performance.memory;
15625       mu0 = m0.usedJSHeapSize;
15626       console.log('-- heap bef clone: '
15627         +m0.usedJSHeapSize+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
15628     }
15629     OriginalSourceNode = gsh.cloneNode(true);
15630     if( false ){
15631       m1 = performance.memory;
15632       mu1 = m1.usedJSHeapSize;
15633       mu = mu1 - mu0;
15634       //alert('-- clone: used heap '+mu0+' -> '+mu1+' = '+mu+' bytes');
15635       console.log('-- heap aft clone: '
15636         +m1.usedJSHeapSize+'/'+m1.totalHeapSize+'/'+m1.jsHeapSizeLimit);
15637       //OriginalSourceNode = document.documentElement.cloneNode(true);
15638     }
15639   }
15640 }
15641 function Gsh_setupPage(){
15642   GshSetImages();
15643   //Indexer_afterLoaded();
15644   //GShell_initKeyCommands();
15645   //GJConsole_initConsole();
15646   GJConsole_initFactory();
15647   GJLink_init();
15648   InterFrameComm_init();
15649   Gshell_initTopbar();
15650   //VirtualDesktop_init();
15651   Banner_init();
15652   // AFi_Setup();
15653   Shading_Setup();
15654   window.setInterval(ShowResourceUsage,1000);
15655   //document.addEventListener('keydown',jgshCommand); // should be applied later?
15656   Pointillism_Setup();
15657   FontList_Setup();
15658   showFooter();
15659   GshInsideIconSetup();
15660   SightClass_Setup();
15661   //spawnPackmonGo();
15662   //PackmonGo_Setup(null);
15663   DrawingCanvas_Setup();
15664   InstColor_Setup();
15665   CWatch_Setup();
15666 }
15667 function OnLoad(){
15668   SaveOriginalNode();
15669   Gsh_setupPage();
15670 }
15671 document.addEventListener('load',Gsh_setupPage);
15672 </script>
15673 </details>
15674 <!-- OriginalSource_WorkCodeSpan -->
15675 * //</span>
15676 //<!-- ===== Work } ===== -->
15677
15678
15679
15680 </div>
15681 <br><script>OnLoad();</script></span>
15682

```