

```

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-0.8.2-2020-11-13--SatoxITS</span>
7 <title id="GshTitle">GShell-0.8.2 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="shiftBg();">
16 <div align="right"><note><a href="http://archive.gshell.org">GShell</a> version 0.8.2 // 2020-11-13 // SatoxITS</note></div>
17 </div>
18 <span id="FeaturesView"></span>
19 */
20
21
22
23
24
25
26 //<!-- ----- Work { ----- -->
27 //<span id="Topbar_WorkCodeSpan">
28 /*
29 <details><summary>Topbar</summary>
30 <!-- ----- Topbar // 2020-1008 SatoxITS { --->
31 <h2>Topbar</h2>
32 <input id="Topbar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
33 <input id="Topbar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
34 <input id="Topbar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
35 <span id="Topbar_WorkCodeView"></span>
36 </details>
37
38 <style>
39 #GshHeading {
40   display:inline;
41   overflow:visible;
42 }
43 .ConfigIcon {
44   position:absolute;
45   top:-6px;
46   left:92%;
47   width:32px;
48   height:32px;
49 }
50 .MetaWindow {
51   z-index:1000;
52   position:relative;
53   display:block;
54   overflow:visible !important;
55   width:99.9%;
56   height:22px;
57   top:-22px;
58   border:1px solid #22a;
59   margin:0px;
60   left:0.0%;
61   line-height:1.0;
62   font-family:Georgia;
63   color:#fff;
64   font-size:12pt;
65   text-align:center;
66   vertical-align:middle;
67   padding:4px;
68   xxbackground-color:rgba(0,8,170,0.8);
69   background-color:#3a4861;xxx-PBlue;
70   vertical-align:middle;
71 }
72 .MetaWindow:hover {
73   color:#000;
74   border:1px solid #22a;
75   background-color:rgba(255,255,255,1.0);
76 }
77 #GshBanner {
78   overflow:visible;
79   display:block;
80   width:100%;
81   height:100px;
82   left:inherit !important;
83 }
84 </style>
85 <script>
86 function Topbar_openWorkCodeView(){
87   function Topbar_showWorkCode(){
88     showHtmlCode(Topbar_WorkCodeView,Topbar_WorkCodeSpan);
89   }
90   Topbar_WorkCodeViewOpen.addEventListener('click',Topbar_showWorkCode);
91 }
92 Topbar_openWorkCodeView();
93 function ConfigClick(){
94   if (0 = AffView.style.zIndex){
95     AffView.style.saved_zIndex = AffView.style.zIndex;
96     AffView.style.zIndex = -1000;
97     GshSidebar.style.zIndex = -1;
98     GshPerfMon.style.zIndex = -1;
99   }else{
100     //AffView.style.zIndex = AffView.style.saved_zIndex;
101     AffView.style.zIndex = 1;
102     GshSidebar.style.zIndex = 1;
103     GshPerfMon.style.zIndex = 1;
104     GMenu.style.zIndex = 10000000;
105   }
106   console.log('AffZIndex='+AffView.style.zIndex);
107 }
108 function Gshell_initTopbar(){
109   GshTopbar.innerHTML = GshTitle.innerHTML;
110   //<img id="ConfigIcon" class="ConfigIcon">
111   if (true){
112     cfig = document.createElement('img');
113     cfig.id = 'ConfigIcon';
114     cfig.setAttribute('class','ConfigIcon');
115     GshTopbar.appendChild(cfig);
116     cfig.src = ConfigIcon_DATA;
117   }
118   //cfig.style.zIndex = 10000000000;
119   //cfig.addEventListener('click',ConfigClick);
120   GshTopbar.addEventListener('click',ConfigClick);
121 }
122 </script>
123 <!-- Topbar_WorkCodeSpan } -->
124 */ //</span>
125 //<!-- ----- Work } ----- -->
126
127
128 //<!-- ----- Work { ----- -->
129 //<span id="Indexer_WorkCodeSpan">
130 /*
131 <details><summary>Indexer</summary>
132 <!-- ----- Indexer // 2020-1007 SatoxITS { --->
133 <h2>Indexer</h2>
134 <input id="Indexer_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
135 <input id="Indexer_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
136 <input id="Indexer_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
137 <span id="Indexer_WorkCodeView"></span>
138 </details>
139 <style id="SidebarIndex">
140 #gsh {
141   display:block;
142   xxoverflow:scroll !important;
143 }
144 #GshMain {
145   z-index:1;
146   position:relative;
147   display:block;
148   width:80% !important;
149   left:19.5% !important;
150 }
151 #GshSidebar {
152   z-index:0;
153   position:relative !important;
154   overflow:auto;
155   resize:both !important;
156   xxoverflow-y:hidden !important;
157   xxheight:100px !important;
158   xxdisplay:inline !important;
159   left:0px;
160   top:0px;
161   width:19.5%;
162   min-width:80px;
163   xxheight:100% !important;
164   height:0px;
165   color:#f00;
166   xxbackground-color:rgba(64,64,64,0.5);
167   xxbackground-color:#DFE3EB;xxx-PBlue;
168   background-color:#e0e0e0;xxx-PBlue;
169 }
170 #GshPerfMon {
171   position:relative;
172   display:block;
173   overflow:visible;
174   z-index:0 !important;
175   xxheight:12pt;
176 }

```

```

177 font-family:monospace, Courier New !important;
178 font-size:9pt !important;
179 color:#f94;
180 top:-20px;
181 }
182 #GshPerfMon: hover {
183 z-index:3 !important;
184 }
185 #GshSidebar: hover {
186 z-index:2;
187 overflow-x:visible !important;
188 background-color:rgba(255,255,255,0.7);
189 width:50%;
190 }
191 #GshIndexer {
192 z-index:0;
193 position:relative;
194 resize:both !important;
195 height:100%;
196 left:0px;
197 top:0px;
198 scroll-behavior: overflow !important;
199 padding-left:4pt;
200 font-size:0.5em;
201 white-space:nowrap;
202 xxx-background-color:rgba(64,160,64,0.6) !important;
203 color:#7794c6;xxx-PBlue;
204 xxxbackground-color:#f9f9f9;xxx-PBlue;
205 background-color:#e9e9e9;xxx-PBlue;
206 }
207 #GshIndexer: hover {
208 z-index:1000000;
209 overflow-x:visible !important;
210 color:#000000 !important;xxx-PBlue;
211 xxxbackground-color:#ffffff;xxx-PBlue;
212 background-color:rgba(255,255,255,0.7);
213 padding-right:0px;
214 width:80%;
215 }
216 #GshIndexer:select {
217 color:#000000 !important;xxx-PBlue;
218 background-color:#ffffff;xxx-PBlue;
219 }
220 .IndexLine {
221 font-size:8pt !important;
222 font-family:Georgia;
223 display:block;
224 xxx-color:#f9f;
225 xxx-color:#f9f;xxx-PBlue;
226 xxx-color:#41516d;xxx-PBlue;
227 xxx-color:#7794c6;xxx-PBlue;
228 padding-right:4pt;
229 }
230 .IndexLine: hover {
231 font-size:10pt !important;
232 xxx-color:#f2f;
233 xxx-background-color:#fff;
234 xxxcolor:#fff;xxx-PBlue;
235 color:#516487;xxx-PBlue;
236 background-color:rgba(220,220,220,1.0);xxx-PBlue;
237 xxxtext-shadow:1px 1px #3f3;
238 text-shadow:1px 1px #eee;
239 xxxbackground-color:#516487;xxx-PBlue;
240 xxxtext-decoration:underline !important;
241 }
242 </style>
243
244 <script id="Indexer WorkScript">
245 function Indexer_openWorkCodeView(){
246 function Indexer_showWorkCode(){
247 showHtmlCode(Indexer_WorkCodeView,Indexer_WorkCodeSpan);
248 }
249 Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_showWorkCode);
250 }
251 //Indexer_WorkCodeViewOpen.addEventListener('click',Indexer_openWorkCodeView);
252 Indexer_openWorkCodeView();
253
254 var startPerfDate = new Date();
255 var prevPerfDate = startPerfDate;
256 function ShowResourceUsage(){
257 d = new Date();
258 perf = '';
259 perf += '<'+font color="gray">UA:' + window.navigator.userAgent + '<'+font><br>\n';
260 perf += DateShort0(startPerfDate) + '<br>\n';
261 perf += DateShort() + '<br>\n';
262 elps = d.getTime() - startPerfDate.getTime();
263 itvl = d.getTime() - prevPerfDate.getTime();
264 perf += 'Elapsed: ' + elps/1000 + ' s<br>\n';
265 perf += 'Skews: ' + (itvl-1000) + ' ms<br>\n';
266 prevPerfDate = d;
267
268 if( performance.memory != undefined ){
269 m0 = performance.memory;
270 mu0 = (m0.usedJSHeapSize / 1000000.0); // .toFixed(6);
271 perf += 'Memory: '+mu0+' MB<br>\n';
272 }
273 perf += '<br>\n';
274
275 //GshSidebar.innerHTML = perf;
276 GshPerfMon.innerHTML = perf;
277 //GshIndexer.innerHTML = 'Memory: '+mu0+' MB';
278 //console.log('-- PerfMon heap: '+mu0+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
279 if( true ){
280 GshSidebar.style.zIndex = 1000;
281 GshIndexer.style.zIndex = 0;
282 GshPerfMon.style.zIndex = 1;
283 //GshSidebar.appendChild(GshPerfMon);
284 if( document.getElementById('primary') == null ){ // not in WordPress
285 // GshPerfMon.style.position = 'absolute';
286 }
287 GshPerfMon.style.display = 'block';
288 GshPerfMon.style.marginLeft = '4px';
289 //GshPerfMon.style.top = '45px';
290 GshPerfMon.style.position = 'relative';
291 //GshPerfMon.style.position = 'absolute';
292 //topy = GshTopbar.getBoundingClientRect().top;
293 //topy = parseInt(topy) + 40;
294 //GshPerfMon.style.top = topy + 'px';
295 GshPerfMon.style.left = '0px';
296
297 GshMain.style.top = -GshPerfMon.getBoundingClientRect().height;
298 }
299 }
300 function ResetPerfMon(){
301 GshPerfMon.removeAttribute('style');
302 GshSidebar.removeAttribute('style');
303 }
304
305 var iserno = 0;
306 var GeneratedId = 0;
307 function generateIndex(ni,e,chkv,nch,ht){
308 // https://developer.mozilla.org/en-US/docs/Web/API/Element
309 c = '';
310 if( e.classList != null ){
311 c = e.classList.value;
312 }
313 //console.log('-- <'+e.nodeName+'> #' + e.id + ' .'+c+' '+e.attributes);
314 if( e.nodeName == '#text' ){ return ''; }
315 if( e.nodeName == '#comment' ){ return ''; }
316 if( e.nodeName == 'H2' || e.nodeName == 'H3' ){
317 id = e.innerHTML;
318 GeneratedId += 1;
319 eid = 'GeneratedId-'+GeneratedId;
320 e.id = eid;
321 }else
322 if( e.nodeName == 'SUMMARY' ){
323 id = e.innerHTML;
324 GeneratedId += 1;
325 eid = 'GeneratedId-'+GeneratedId;
326 e.id = eid;
327 }else
328 if( and(e.nodeName == 'DIV',c='xxxxxxxxxxxxxxxxentry-content') ){
329 console.log('-- DIV entry-content begin');
330 id = e.innerHTML;
331 GeneratedId += 1;
332 eid = 'GeneratedId-'+GeneratedId;
333 e.id = eid;
334 console.log('-- DIV entry-content end hash-child=',e.hasChildNodes());
335 }else
336 if( e.id == '' || e.id == 'undefined' ){
337 return '';
338 }else{
339 id = '#' + e.id;
340 eid = e.id;
341 }
342 iserno += 1;
343 ht = '<'+div id="GeneratedEref'+iserno+' class="IndexLine" href="'+eid+'">
344 + iserno+' '+ni+' '+e.nodeName + ' : ' + id;
345 if( e.id == '' || e.id == 'undefined' ){ return ht + '<'+div>; }
346 if( !e.hasChildNodes() ){ return ht + '<'+div>; }
347 chkv = e.childNodes;
348 nch = e.childNodes.length;
349 if( chkv != null ){ nch = chkv.length; }
350 ht += ' ('+nch+') '<'+div>;
351 for( let i = 0; i < chkv.length; i+ ){
352 sec = ni+'.'+i;
353 if( ni == '' ){ sec = i; }

```

```

354     ht += generateIndex(sec, chv[i], null, 0);
355 }
356 return ht;
357 }
358 function onClickIndex(e){
359     tid = e.target.id;
360     tge = document.getElementById(tid);
361     eid = tge.getAttribute('href');
362     rx = tge.getBoundingClientRect().left.toFixed(0)
363     ry = tge.getBoundingClientRect().top.toFixed(0)
364     if( false ){
365         alert('index clicked mouse(x='+e.x+', y='+e.y+')'
366             + '\ntid#'+ tid + ' rx#'+rx + ',ry#'+ry
367             + '\neid#'+ eid + '\n'
368             + '\nhtml#'+ tge.outerHTML);
369     }
370     ee = document.getElementById(eid);
371     sx = 'NaN';
372     sy = ee.getBoundingClientRect().top;
373     console.log('sx'+sx+',sy'+sy);
374     ee.scrollIntoView()
375     window.scrollTo(sx,sy)
376     //window.scrollTo({left:'NaN',top:sy,behavior:'smooth'});
377 }
378 function Indexer afterLoaded(){
379     sideindex = document.getElementById('GshIndexer');
380     ht = '<'+h3>G-index-4'/h3>';
381     ht += generateIndex("", document.getElementById('gsh'), null, 0, '');
382     if( (pri = document.getElementById('primary')) != null ){
383         ht += generateIndex("", pri, null, 0, '');
384     }
385     ht += '<'+br>';
386     ht += '<'+br>';
387     ht += '<'+br>';
388     ht += '<'+br>';
389     sideindex.innerHTML = ht;
390     sideindex.addEventListener('click', onClickIndex);
391
392     if( (pri = document.getElementById('primary')) != null ){
393         console.log('-- Seems in WordPress');
394         pri.style.zIndex = 2000;
395
396         GshSidebar.style.setProperty('position', 'relative', 'important');
397         GshSidebar.style.top = '-1400px';
398         //GshSidebar.style.setProperty('position', 'absolute', 'important');
399         //GshSidebar.style.top = '0px';
400
401         GshSidebar.style.setProperty('width', '200px', 'important');
402         GshSidebar.style.setProperty('overflow', 'scroll', 'important');
403         GshSidebar.style.resize = 'both';
404
405         GshSidebar.style.left = '-100px';
406         GshIndexer.style.left = '100px';
407         GshIndexer.style.height = '1400px';
408         gsh.appendChild(GshSidebar); // change parent
409     }else{
410         console.log('-- Seems not in WordPress');
411         GshSidebar.style.setProperty('position', 'fixed', 'important');
412     }
413 }
414 //document.addEventListener('load', Indexer_afterLoaded);
415
416 DestroyIndexer = function(){
417     sideindex = document.getElementById('GshIndexer');
418     sideindex.innerHTML = "";
419     sideindex.style = "";
420 }
421 </script>
422
423 <!-- Indexer_WorkCodeSpan -->
424 *//</span>
425 //<!-- Work -->
426
427
428
429 /*
430 <h2>GShell // A General purpose Shell built on the top of Golang</h2>
431 <p>
432 <note>
433 It is a shell for myself, by myself, of myself. --SatoxITS("-")
434 <a href="gsh-0.6.2.go.html">prev.</a>
435 <note>
436 </p>
437 <div id="GJFactory_x"></div>
438
439 <div>
440 <span id="GshMenu" class="GshMenu">
441 <span class="GshMenu" id="GshMenuEdit" onclick="html_edit();">Edit</span>
442 <span class="GshMenu" id="GshMenuSave" onclick="html_save();">Save</span>
443 <span class="GshMenu" id="GshMenuLoad" onclick="html_load();">Load</span>
444 <span class="GshMenu" id="GshMenuVers" onclick="html_ver0();">Vers</span>
445 <span id="GshMenuWinJump" onclick="win_jump('0.1');">0</span>
446 <span class="GshMenu" id="gsh-menu-exit" onclick="html_close();"></span>
447 <span class="GshMenu" id="gsh-menu-fork" onclick="html_fork();">Fork</span>
448 <span class="GshMenu" id="GshMenuStop" onclick="html_stop(this, true);">Stop</span>
449 <span class="GshMenu" id="GshMenuFold" onclick="html_fold(this);">Unfold</span>
450 <span class="GshMenu" id="gsh-menu-cksum" onclick="html_digest();">Digest</span>
451 <span class="GshMenu" id="GshMenuSign" onclick="html_sign(this);" style="">Source</span>
452 <!-- <span id="gsh-menu-pure" onclick="html_pure(this);">Pure</span -->
453 </span>
454 </div>
455 *//
456
457 /*
458 <details id="GshStatement" class="gsh-document"><summary>Statement</summary>
459 <h3>Fun to create a shell</h3>
460 <p>For a programmer, it must be far easy and fun to create his own simple shell
461 rightly fitting to his favor and necessities, than learning existing shells with
462 complex full features that he never use.
463 I, as one of programmers, am writing this tiny shell for my own real needs,
464 totally from scratch, with fun.
465 </p><p>
466 For a programmer, it is fun to learn new computer languages. For long years before
467 writing this software, I had been specialized to C and early HTML2 (-).
468 Now writing this software, I'm learning Go language, HTML5, JavaScript and CSS
469 on demand as a novice of these, with fun.
470 </p><p>
471 This single file "gsh.go", that is executable by Go, contains all of the code written
472 in Go. Also it can be displayed as "gsh.go.html" by browsers. It is a standalone
473 HTML file that works as the viewer of the code of itself, and as the "home page" of
474 this software.
475 </p><p>
476 Because this HTML file is a Go program, you may run it as a real shell program
477 on your computer.
478 But you must be aware that this program is written under situation like above.
479 Needless to say, there is no warranty for this program in any means.
480 </p>
481 <address>Aug 2020, SatoxITS (sato@its-more.jp)</address>
482 </details>
483 *//
484
485 <details id="GshFeatures" class="gsh-document"><summary>Features</summary><p>
486 <h3>Cross-browser communication</h3>
487 <p>
488 ... to be written ...
489 </p>
490 <p>
491 <h3>Vi compatible command line editor</h3>
492 <p>
493 The command line of GShell can be edited with commands compatible with
494 <a href="https://www.washington.edu/computing/unix/vi.html"><b>vi</b></a>.
495 As in vi, you can enter <b>^</b>command mode/<b>_</b> by <b>ESC</b> key,
496 then move around in the history by <b>code>j k / p a N</code>,<b>_</b>,
497 or within the current line by <b>code>l h f w b o $ %</code>,<b>_</b> or so.
498 </p>
499 </details>
500 *//
501
502 <details id="gsh-index">
503 <summary>Index</summary><div class="gsh-src">
504 Documents
505 <span class="gsh-link" onclick="jumpto_JavaScriptView();">Command summary</span>
506 Go lang part<span class="gsh-src" onclick="document.getElementById('gsh-gocode').open=true;">
507 Package structures
508 <a href="#import">import</a>
509 <a href="#struct">struct</a>
510 Main functions
511 <a href="#comexpansion">str-expansion</a> // macro processor
512 <a href="#finder">finder</a> // builtin find + du
513 <a href="#grep">grep</a> // builtin grep + wc + cksum + ...
514 <a href="#plugin">plugin</a> // plugin commands
515 <a href="#ex-commands">system</a> // external commands
516 <a href="#builtin">builtin</a> // builtin commands
517 <a href="#network">network</a> // socket handler
518 <a href="#remote-sh">remote-sh</a> // remote shell
519 <a href="#redirect">redirect</a> // Stdin/Out redirection
520 <a href="#history">history</a> // command history
521 <a href="#rusage">rusage</a> // resource usage
522 <a href="#encode">encode</a> // encode / decode
523 <a href="#lme">lme</a> // command line lme
524 <a href="#getline">getline</a> // line editor
525 <a href="#scanf">scanf</a> // string decomposer
526 <a href="#interpreter">interpreter</a> // command interpreter
527 <a href="#main">main</a>
528 </span>
529 JavaScript part
530 <a href="#script-src-view" class="gsh-link" onclick="jumpto_JavaScriptView();">Source</a>

```

```

531 <a href="#gsh-data-frame" class="gsh-link" onclick="jumpto_DataView();">Builtin data</a>
532 CSS part
533 <a href="#style-src-view" class="gsh-link" onclick="jumpto_StyleView();">Source</a>
534 References
535 <a href="#" class="gsh-link" onclick="jumpto_WholeView();">Internal</a>
536 <a href="#gsh-reference" class="gsh-link" onclick="jumpto_ReferenceView();">External</a>
537 Whole parts
538 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Source</a>
539 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Download</a>
540 <a href="#whole-src-view" class="gsh-link" onclick="jumpto_WholeView();">Dump</a>
541
542 </div>
543 </details>
544 */
545 </details id="gsh-gocode">
546 <<summary>Go Source</summary><div class="gsh-src" onclick="document.getElementById('gsh-gocode').open=false;">
547 // gsh - Go lang based shell
548 // (c) 2020 ITS more Co., Ltd.
549 // 2020-0807 created by SatoxITS (sato@its-more.jp)
550
551 package main // gsh main
552
553 // <a name="import">Imported packages</a> // <a href="https://golang.org/pkg/">Packages</a>
554 import (
555     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
556     "errors"
557     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
558     "strconv" // <a href="https://golang.org/pkg/strconv/">strconv</a>
559     "sort" // <a href="https://golang.org/pkg/sort/">sort</a>
560     "time" // <a href="https://golang.org/pkg/time/">time</a>
561     "bufio" // <a href="https://golang.org/pkg/bufio/">bufio</a>
562     "io/ioutil" // <a href="https://golang.org/pkg/io/ioutil/">ioutil</a>
563     "os" // <a href="https://golang.org/pkg/os/">os</a>
564     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
565     "plugin" // <a href="https://golang.org/pkg/plugin/">plugin</a>
566     "net" // <a href="https://golang.org/pkg/net/">net</a>
567     "net/http" // <a href="https://golang.org/pkg/net/http/">http</a>
568     "html" // <a href="https://golang.org/pkg/html/">html</a>
569     "path/filepath" // <a href="https://golang.org/pkg/path/filepath/">filepath</a>
570     "go/types" // <a href="https://golang.org/pkg/go/types/">types</a>
571     "go/token" // <a href="https://golang.org/pkg/go/token/">token</a>
572     "encoding/base64" // <a href="https://golang.org/pkg/encoding/base64/">base64</a>
573     "unicode/utf8" // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
574     // "gshdata" // gshell's logo and source code
575     "hash/crc32" // <a href="https://golang.org/pkg/hash/crc32/">crc32</a>
576     "golang.org/x/net/websocket"
577     "runtime"
578 )
579
580
581 /*
582 #include <stdio.h> // </stdio.h> to be closed as HTML tag :-p
583 #ifdef _WIN32
584 #include <windows.h> // </windows.h>
585 // 2020-1022 added -- terminal mode on Windows
586 // https://docs.microsoft.com/en-us/windows/console/setconsolemode
587 // https://docs.microsoft.com/en-us/windows/win32/inputdev/using-keyboard-input
588 int setTermRaw(){
589     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
590     DWORD tmode = 0;
591     if( GetConsoleMode(hStdin,&tmode) ){
592         DWORD xmode = tmode;
593         xmode &= ~ENABLE_ECHO_INPUT;
594         xmode &= ~ENABLE_LINE_INPUT;
595         xmode |= ENABLE_PROCESSED_INPUT; // Control+C for SIGINT
596         if( SetConsoleMode(hStdin,xmode) ){
597             return tmode;
598         }
599     }
600     return 0;
601 }
602 int setTermMode(int tmode){
603     HANDLE hStdin = GetStdHandle(STD_INPUT_HANDLE);
604     SetConsoleMode(hStdin,tmode);
605     return 0;
606 }
607 #else
608 int setTermRaw(){
609     return -1;
610 }
611 int setTermMode(int tmode){
612     return 0;
613 }
614 #endif
615 */
616 import "C"
617
618 //
619 // // 2020-0906 added,
620 // // <a href="https://golang.org/cmd/cgo/">CGo</a>
621 // #include "poll.h" // "poll.h" // </poll.h> to be closed as HTML tag :-p
622 // typedef struct { struct pollfd fdv[8]; } pollFdv;
623 // int pollx(pollFdv *fdv, int nfds, int timeout){
624 //     return poll(fdv->fdv,nfds,timeout);
625 // }
626 // import "C"
627
628 // 2020-1021 replaced poll() with channel/select
629 // // 2020-0906 added,
630 // func CpollInl(fp *os.File, timeoutUs int)(ready uintptr){
631 //     var fdv = C.pollFdv{}
632 //     var nfds = 1
633 //     var timeout = timeoutUs/1000
634 //     fdv.fdv[0].fd = C.int(fp.Fd())
635 //     fdv.fdv[0].events = C.POLLIN
636 //     if( 0 < EventRecvFd {
637 //         fdv.fdv[1].fd = C.int(EventRecvFd)
638 //         fdv.fdv[1].events = C.POLLIN
639 //         nfds += 1
640 //     }
641 //     r := C.pollx(&fdv,C.int(nfds),C.int(timeout))
642 //     if( r <= 0 ){
643 //         return 0
644 //     }
645 //     if (int(fdv.fdv[1].revents) & int(C.POLLIN)) != 0 {
646 //         //printf(stderr,"-D- got Event\n");
647 //         return uintptr(EventPdoffset + fdv.fdv[1].fd)
648 //     }
649 //     if (int(fdv.fdv[0].revents) & int(C.POLLIN)) != 0 {
650 //         return uintptr(NormalPdoffset + fdv.fdv[0].fd)
651 //     }
652 //     return 0
653 // }
654 // */
655
656 const (
657     NAME = "gsh"
658     VERSION = "0.8.2"
659     DATE = "2020-11-13"
660     AUTHOR = "SatoxITS(-)://"
661 )
662
663 var (
664     GSH_HOME = ".gsh" // under home directory
665     GSH_PORT = 9999
666     MaxStreamsSize = int64(128*1024*1024) // 128GiB is too large?
667     PROMPT = ">"
668     LINESIZE = (8*1024)
669     PATHSEP = "/" // should be ";" in Windows
670     DIRSEP = "/" // canbe \ in Windows
671     OnWindows = false;
672 )
673
674 func initGshEnv(){
675     if( runtime.GOOS == "windows" ){
676         PATHSEP = ";";
677         DIRSEP = "\\";
678         OnWindows = true;
679     }else{
680     }
681 }
682
683 // -x logging control
684 // -- all
685 // --l- info.
686 // --D- debug
687 // --T- time and resource usage
688 // --W- warning
689 // --E- error
690 // --F- fatal error
691 // --N- network
692
693 // <a name="struct">Structures</a>
694
695 // 2020-1022 Unix/Windows
696 // -----
697 //type aStat_t syscall.Stat_t;
698 //type aStat_t struct { syscall.Stat_t }
699 type aStat_t struct {
700     Size int64
701     Mode os.FileMode
702     Rdev int64
703     Blocks int64
704     Nlink int64
705 }
706
707 func aLStat(path string, astat *aStat_t)(error){
708     /*
709     sstat := syscall.Stat_t{};

```

```

708     err := syscall.Lstat(path,&stat);
709     *astat = aStat_t(stat);
710     */
711     fi,err := os.Stat(path);
712     if( err == nil ){
713         astat.Mode = fi.Mode();
714         astat.Size = fi.Size();
715     }
716     return err;
717 }
718
719 func aFstat(fd int, astat *aStat_t)(error){
720     /*
721     sstat := syscall.Stat_t{};
722     err := syscall.Fstat(fd,&sstat);
723     *astat = aStat_t(sstat);
724     */
725     err := errors.New("NotImplemented-Fstat");
726     //fmt.Printf("---E-- fstat(%v,%v)\n",fd,err);
727     return err;
728 }
729
730 func aAccess(path string, mode uint32)(error){
731     //err := syscall.Access(path,mode);
732     //err := errors.New("NotImplemented-Access");
733     fi,err := os.Stat(path);
734     //fmt.Printf("-- Access(%v,%v)\n(%v)\n",path,mode,err);
735     if( err == nil ){
736         fmode := fi.Mode();
737         if( fmode.IsRegular() ){
738             perm := fmode.Perm();
739             if( (uint32(perm) & mode) != 0 ){
740                 return nil;
741             }
742             return errors.New("NotAccessible");
743         }
744         return errors.New("NotRegularFile");
745     }
746     return err;
747 }
748 // 2020-1022 Unix/Windows
749 // -----
750 type aRusage struct {
751     syscall.Rusage
752     Utime         time.Duration
753     Stime         time.Duration
754     //Sys         interface{}
755 }
756 /*
757 */
758 const aRUSAGE_SELF = syscall.RUSAGE_SELF
759 const aRUSAGE_CHILDREN = syscall.RUSAGE_CHILDREN
760 /*
761 */
762 const aRUSAGE_SELF = 0
763 const aRUSAGE_CHILDREN = 1
764 func aGetRusage(sel int, ru *aRusage){
765     /*
766     sysru := syscall.Rusage{};
767     syscall.GetRusage(sel,&sysru);
768     ru.Utime = time.Duration(int64(sysru.Utime.Sec)*1000000000+int64(sysru.Utime.Usec)*1000);
769     ru.Stime = time.Duration(int64(sysru.Stime.Sec)*1000000000+int64(sysru.Stime.Usec)*1000);
770     */
771 }
772 func aSetRusage(ru *aRusage, ps *os.ProcessState){
773     ru.Utime = ps.UserTime();
774     ru.Stime = ps.SystemTime();
775 }
776
777 func showRusage(what string,argv []string, ru *aRusage){
778     fmt.Printf("%s: ",what);
779     //fmt.Printf("User=%d.%06ds",ru.Utime.Sec,ru.Utime.Usec)
780     //fmt.Printf("Sys=%d.%06ds",ru.Stime.Sec,ru.Stime.Usec)
781     //fmt.Printf("User=%d.%06ds ",ru.Utime/1000000000,(ru.Utime/1000)%1000000);
782     //fmt.Printf(" Sys=%d.%06ds ",ru.Stime/1000000000,(ru.Stime/1000)%1000000);
783     /*
784     */
785     fmt.Printf(" Rss=%vB",ru.Maxrss)
786     if isin("-l",argv) {
787         fmt.Printf(" MinFlt=%v",ru.Minflt)
788         fmt.Printf(" MajFlt=%v",ru.Majflt)
789         fmt.Printf(" IxRSS=%vB",ru.Ixrss)
790         fmt.Printf(" IdRSS=%vB",ru.Idrss)
791         fmt.Printf(" Nswap=%vB",ru.Nswap)
792         fmt.Printf(" Head=%v",ru.Inblock)
793         fmt.Printf(" Write=%v",ru.Obblock)
794     }
795     fmt.Printf(" Snd=%v",ru.Msgsnd)
796     fmt.Printf(" Rcv=%v",ru.Msgrcv)
797     //if isin("-l",argv) {
798     //    fmt.Printf(" Sig=%v",ru.Nsignals)
799     //}
800     /*
801     */
802     fmt.Printf("\n");
803 }
804
805 type GCommandHistory struct {
806     StartAt    time.Time // command line execution started at
807     EndAt      time.Time // command line execution ended at
808     ResCode    int // exit code of (external command)
809     CmdError   error // error string
810     OutData    *os.File // output of the command
811     FoundFile  []string // output - result of ufind
812     Rusageev   [2]aRusage // Resource consumption, CPU time or so
813     Cmdid      int // maybe with identified with arguments or impact
814     // recursion commands should not be the Cmdid
815     WorkDir    string // working directory at start
816     WorkDirX   int // index in ChdirHistory
817     CmdLine    string // command line
818 }
819
820 type GChdirHistory struct {
821     Dir         string
822     MovedAt    time.Time
823     CmdIndex    int
824 }
825
826 type CmdMode struct {
827     BackGround bool
828 }
829
830 type Event struct {
831     When        time.Time
832     event       int
833     evarg       int64
834     CmdIndex    int
835 }
836
837 var CmdIndex int
838 var Events []Event
839
840 type PluginInfo struct {
841     Spec        *plugin.Plugin
842     Addr        plugin.Symbol
843     Name        string // maybe relative
844     Path        string // this is in Plugin but hidden
845 }
846
847 type GServer struct {
848     host        string
849     port        string
850 }
851
852 // <a href="https://tools.ietf.org/html/rfc3230">Digest</a>
853 const ( // SumType
854     SUM_ITEMS = 0x000001 // items count
855     SUM_SIZE = 0x000002 // data length (simply added)
856     SUM_SIZEHASH = 0x000004 // data length (hashed sequence)
857     SUM_DATEHASH = 0x000008 // date of data (hashed sequence)
858     // also envelope attributes like time stamp can be a part of digest
859     // hashed value of sizes or mod-date of files will be useful to detect changes
860
861     SUM_WORDS = 0x000010 // word count is a kind of digest
862     SUM_LINES = 0x000020 // line count is a kind of digest
863     SUM_SUM64 = 0x000040 // simple add of bytes, useful for human too
864
865     SUM_SUM32_BITS = 0x0000100 // the number of true bits
866     SUM_SUM32_2BYTE = 0x0000200 // 16bits words
867     SUM_SUM32_4BYTE = 0x0000400 // 32bits words
868     SUM_SUM32_8BYTE = 0x0000800 // 64bits words
869
870     SUM_SUM16_BSD = 0x0010000 // UNIXsum -sum -bsd
871     SUM_SUM16_SYSV = 0x0020000 // UNIXsum -sum -sysv
872     SUM_UNIXFILE = 0x0040000
873     SUM_CRCIEEE = 0x0080000
874 )
875
876 type CheckSum struct {
877     Files        int64 // the number of files (or data)
878     Size         int64 // content size
879     Words        int64 // word count
880     Lines        int64 // line count
881     SumType      int
882     Sum64        uint64
883     Crc32Table   crc32.Table
884     Crc32Val     uint32
885     Sum16        int
886     Ctime        time.Time
887     Atime        time.Time
888     Mtime        time.Time
889     Start        time.Time
890     Done         time.Time
891     RusageStart [2]aRusage
892     RusageEnd   [2]aRusage
893 }
894
895 type ValueStack [][]string
896 type GshContext struct {

```

```

885 StartDir string // the current directory at the start
886 GetLine string // gsh-getline command as a input line editor
887 CmdrHistory []|GCmdrHistory // the 1st entry is wd at the start
888 //gshPA syscall.ProcAttr
889 gshPA os.ProcAttr
890 CommandHistory []|GCommandHistory
891 Cmdcurrent GCommandHistory
892 Background bool
893 BackgroundJobs []|os.ProcessState; //[]|int
894 LastRunage aRunage
895 GshHomeDir string
896 TerminalId int
897 CmdTrace bool // should be [map]
898 CmdTime bool // should be [map]
899 PluginFuncs []|PluginInfo
900 IValues []|string
901 IDelimiter string // field separator of print out
902 IFormat string // default print format (of integer)
903 IValStack ValueStack
904 LastServer GServer
905 RSRV string // [gsh://host[:port]
906 RWD string // remote (target, there) working directory
907 lastChecksum CheckSum
908 }
909 }
910 func nsleep(ns time.Duration){
911     time.Sleep(ns)
912 }
913 func usleep(ns time.Duration){
914     nsleep(ns*1000)
915 }
916 func msleep(ns time.Duration){
917     nsleep(ns*1000000)
918 }
919 func sleep(ns time.Duration){
920     nsleep(ns*1000000000)
921 }
922 }
923 func strBegins(str, pat string)(bool){
924     if len(pat) <= len(str){
925         yes := str[0:len(pat)] == pat
926         //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,yes)
927         return yes
928     }
929     //fmt.Printf("--D-- strBegins(%v,%v)=%v\n",str,pat,false)
930     return false
931 }
932 func isin(what string, list []|string) bool {
933     for v := range list {
934         if v == what {
935             return true
936         }
937     }
938     return false
939 }
940 func isinX(what string, list[]|string)(int){
941     for i,v := range list {
942         if v == what {
943             return i
944         }
945     }
946     return -1
947 }
948 }
949 func env(opts []|string) {
950     env := os.Environ()
951     if isin("-s", opts){
952         sort.Slice(env, func(i,j int) bool {
953             return env[i] < env[j]
954         })
955     }
956     for _, v := range env {
957         fmt.Printf("%v\n",v)
958     }
959 }
960 }
961 // - rewriting should be context dependent
962 // - should postpone until the real point of evaluation
963 // - should rewrite only known notation of symbol
964 func scanInt(str string)(val int, leng int){
965     leng = -1
966     for i, ch := range str {
967         if '0' <= ch && ch <= '9' {
968             leng = i+1
969         }else{
970             break
971         }
972     }
973     if 0 < leng {
974         ival, _ := strconv.Atoi(str[0:leng])
975         return ival, leng
976     }else{
977         return 0, 0
978     }
979 }
980 func substHistory(gshCtx *GshContext, str string, i int, rstr string)(leng int, rst string){
981     if len(str[i+1:]) == 0 {
982         return 0, rstr
983     }
984     hi := 0
985     histlen := len(gshCtx.CommandHistory)
986     if str[i+1] == '1' {
987         hi = histlen - 1
988         leng = 1
989     }else{
990         hi, leng = scanInt(str[i+1:])
991         if leng == 0 {
992             return 0, rstr
993         }
994         if hi < 0 {
995             hi = histlen + hi
996         }
997     }
998     if 0 <= hi && hi < histlen {
999         var ext byte
1000         if 1 < len(str[i+leng:]){
1001             ext = str[i+leng:][1]
1002         }
1003         //fmt.Printf("--D-- %v(%c)\n",str[i+leng:],str[i+leng])
1004         if ext == 'f' {
1005             leng = 1
1006             xlist := []|string{}
1007             list := gshCtx.CommandHistory[hi].FoundFile
1008             for v := range list {
1009                 //list[i] = escapeWhiteSP(v)
1010                 xlist = append(xlist, escapeWhiteSP(v))
1011             }
1012             rstr += strings.Join(list, " ")
1013             rstr += strings.Join(xlist, " ")
1014         }else
1015         if ext == 'd' || ext == 'd' {
1016             // IN0 .. workdir at the start of the command
1017             leng = 1
1018             rstr += gshCtx.CommandHistory[hi].WorkDir
1019         }else{
1020             rstr += gshCtx.CommandHistory[hi].CmdLine
1021         }
1022     }else{
1023         leng = 0
1024     }
1025     return leng, rstr
1026 }
1027 func escapeWhiteSP(str string)(string){
1028     if len(str) == 0 {
1029         return "\\z" // empty, to be ignored
1030     }
1031     rstr := ""
1032     for _,ch := range str {
1033         switch ch {
1034             case '\\': rstr += "\\\\"
1035             case ' ': rstr += "\\s"
1036             case '\t': rstr += "\\t"
1037             case '\r': rstr += "\\r"
1038             case '\n': rstr += "\\n"
1039             default: rstr += string(ch)
1040         }
1041     }
1042     return rstr
1043 }
1044 func unescapeWhiteSP(str string)(string){ // strip original escapes
1045     rstr := ""
1046     for i := 0; i < len(str); i++ {
1047         ch := str[i]
1048         if ch == '\\ {
1049             if i+1 < len(str) {
1050                 switch str[i+1] {
1051                     case '2':
1052                         continue;
1053                 }
1054             }
1055         }
1056         rstr += string(ch)
1057     }
1058     return rstr
1059 }
1060 func unescapeWhiteSPV(strv []|string)([]|string){ // strip original escapes
1061     ustrv := []|string{}

```

```

1062 for _,v := range strv {
1063     ustrv = append(ustrv,unescapeWhiteSP(v))
1064 }
1065 return ustrv
1066 }
1067
1068 // <a name="comexpansion">str-expansion</a>
1069 // - this should be a macro processor
1070 func strsubst(gshCtx *GshContext,str string,histonly bool) string {
1071     rbuff := []byte{}
1072     if false {
1073         //@@U Unicode should be cared as a character
1074         return str
1075     }
1076     //rstr := ""
1077     inEsc := 0 // escape character mode
1078     for i := 0; i < len(str); i++ {
1079         //fmt.Printf("--D--Subst %v:%v\n",i,str[i:])
1080         ch := str[i]
1081         if inEsc == 0 {
1082             if ch == '\\' {
1083                 //leng,xrstr := substHistory(gshCtx,str,i,rstr)
1084                 leng,rs := substHistory(gshCtx,str,i,"")
1085                 if 0 < leng {
1086                     //leng,xrstr := substHistory(gshCtx,str,i,rstr)
1087                     rbuff = append(rbuff,[]byte(rs)...)
1088                     i += leng
1089                     //rstr = xrstr
1090                     continue
1091                 }
1092             }
1093             switch ch {
1094                 case '\\': inEsc = '\\'; continue
1095                 case '$': inEsc = '$'; continue
1096                 case '$':
1097             }
1098         }
1099         switch inEsc {
1100             case '\\':
1101                 switch ch {
1102                     case '\\': ch = '\\'
1103                     case '$': ch = '$'
1104                     case 't': ch = '\t'
1105                     case 'r': ch = '\r'
1106                     case 'n': ch = '\n'
1107                     case 'z': inEsc = 0; continue // empty, to be ignored
1108                 }
1109             case '$':
1110                 inEsc = 0
1111                 switch {
1112                     case ch == '$': ch = '$'
1113                     case ch == 'T':
1114                         //rstr = rstr + time.Now().Format(time.Stamp)
1115                         rs := time.Now().Format(time.Stamp)
1116                         rbuff = append(rbuff,[]byte(rs)...)
1117                         inEsc = 0
1118                         continue;
1119                     default:
1120                         // postpone the interpretation
1121                         //rstr = rstr + "%" + string(ch)
1122                         rbuff = append(rbuff,ch)
1123                         inEsc = 0
1124                         continue;
1125                 }
1126             case 0:
1127                 //rstr = rstr + string(ch)
1128                 rbuff = append(rbuff,ch)
1129             }
1130         }
1131         //fmt.Printf("--D--subst(%s)(%s)\n",str,string(rbuff))
1132         return string(rbuff)
1133     }
1134     //return rstr
1135 }
1136
1137 func showFileInfo(path string, opts []string) {
1138     if !isIn("-l",opts) || !isIn("-ls",opts) {
1139         fi, err := os.Stat(path)
1140         if err != nil {
1141             fmt.Printf("----- (%s)",err)
1142         }else{
1143             mod := fi.ModTime()
1144             date := mod.Format(time.Stamp)
1145             fmt.Printf("%v %v %s ",fi.Mode(),fi.Size(),date)
1146         }
1147     }
1148     fmt.Printf("%s",path)
1149     if !isIn("-sp",opts) {
1150         fmt.Printf(" ")
1151     }else{
1152         if !isIn("-n",opts) {
1153             fmt.Printf("\n")
1154         }
1155     }
1156 }
1157
1158 func userHomeDir()(string,bool){
1159     /*
1160     homedir,_ = os.UserHomeDir() // not implemented in older Golang
1161     */
1162     homedir,found := os.LookupEnv("HOME")
1163     //fmt.Printf("--I-- HOME=%v\n",homedir,found)
1164     if !found {
1165         return "/tmp",found
1166     }
1167     return homedir,found
1168 }
1169
1170 func toFullpath(path string) (fullpath string) {
1171     if path[0] == '/' {
1172         return path
1173     }
1174     pathv := strings.Split(path,DIRSEP)
1175     switch {
1176         case pathv[0] == ".":
1177             pathv[0],_ = os.Getwd()
1178         case pathv[0] == "..": // all ones should be interpreted
1179             cwd,_ = os.Getwd()
1180             ppathv := strings.Split(cwd,DIRSEP)
1181             pathv[0] = strings.Join(ppathv,DIRSEP)
1182         case pathv[0] == "~":
1183             pathv[0],_ = userHomeDir()
1184         default:
1185             cwd,_ = os.Getwd()
1186             pathv[0] = cwd + DIRSEP + pathv[0]
1187     }
1188     return strings.Join(pathv,DIRSEP)
1189 }
1190
1191 func IsRegFile(path string)(bool){
1192     fi, err := os.Stat(path)
1193     if err == nil {
1194         fm := fi.Mode()
1195         return fm.IsRegular();
1196     }
1197     return false
1198 }
1199
1200 // <a name="encode">Encode / Decode</a>
1201 // <a href="https://golang.org/pkg/encoding/base64/#example_NewEncoder">Encoder</a>
1202 func (gshCtx *GshContext)Enc(argv[]string){
1203     file := os.Stdin
1204     buff := make([]byte,LINESIZE)
1205     li := 0
1206     encoder := base64.NewEncoder(base64.StdEncoding,os.Stdout)
1207     for li = 0; ; li++ {
1208         count, err := file.Read(buff)
1209         if count <= 0 {
1210             break
1211         }
1212         if err != nil {
1213             break
1214         }
1215         encoder.Write(buff[0:count])
1216     }
1217     encoder.Close()
1218 }
1219
1220 func (gshCtx *GshContext)Dec(argv[]string){
1221     decoder := base64.NewDecoder(base64.StdEncoding,os.Stdin)
1222     li := 0
1223     buff := make([]byte,LINESIZE)
1224     for li = 0; ; li++ {
1225         count, err := decoder.Read(buff)
1226         if count <= 0 {
1227             break
1228         }
1229         if err != nil {
1230             break
1231         }
1232         os.Stdout.Write(buff[0:count])
1233     }
1234 }
1235
1236 // Insp [N] [-crlf][-C \ \ ]
1237 func (gshCtx *GshContext)SplitLine(argv[]string){
1238     strRep := isin("-str",argv) // "...n"
1239     reader := bufio.NewReaderSize(os.Stdin,64*1024)
1240     ni := 0
1241     toi := 0
1242     for ni = 0; ; ni++ {
1243         line, err := reader.ReadString('\n')
1244         if len(line) <= 0 {

```

```

1239     if err != nil {
1240         fmt.Fprintf(os.Stderr, "--I-- lnspp %d to %d (%v)\n", ni, toi, err)
1241         break
1242     }
1243 }
1244 off := 0
1245 linen := len(line)
1246 remlen := len(line)
1247 if strRep { os.Stdout.Write([]byte("\n")) }
1248 for oi := 0; 0 < remlen; oi++ {
1249     olen := remlen
1250     addnl := false
1251     if 72 < olen {
1252         olen = 72
1253         addnl = true
1254     }
1255     fmt.Fprintf(os.Stderr, "--D-- write %d [%d.%d] %d %d/%d/%d\n",
1256         toi, ni, oi, off, olen, remlen, linen)
1257     toi += 1
1258     os.Stdout.Write([]byte(line[0:olen]))
1259     if addnl {
1260         if strRep {
1261             os.Stdout.Write([]byte("\n\n"))
1262         } else {
1263             //os.Stdout.Write([]byte("\r\n"))
1264             os.Stdout.Write([]byte("\n"))
1265             os.Stdout.Write([]byte("\n"))
1266         }
1267     }
1268     line = line[olen:]
1269     off += olen
1270     remlen -= olen
1271 }
1272 if strRep { os.Stdout.Write([]byte("\n")) }
1273 }
1274 fmt.Fprintf(os.Stderr, "--I-- lnspp %d to %d\n", ni, toi)
1275 }
1276
1277 // CRC32 <a href="http://qolang.jp/pkg/hash-crc32">crc32</a>
1278 // 1 0000 0100 1100 0001 0001 1101 1011 0111
1279 var CRC32UNIX uint32 = uint32(0x04c11d97) // Unix cksum
1280 var CRC32IEEE uint32 = uint32(0x2db88320)
1281 func byteCRC32add(crc uint32, str []byte, len uint64) (uint32) {
1282     var oi uint64
1283     for oi = 0; oi < len; oi++ {
1284         var oct = str[oi]
1285         for bi := 0; bi < 8; bi++ {
1286             //fmt.Printf(stderr, "--CRC32 %d %X (%d.%d)\n", crc, oct, oi, bi)
1287             ovf1 := (crc & 0x80000000) != 0
1288             ovf2 := (oct & 0x80) != 0
1289             ovf := (ovf1 && !ovf2) || (!ovf1 && ovf2)
1290             oct <<= 1
1291             crc <<= 1
1292             if ovf { crc ^= CRC32UNIX }
1293         }
1294     }
1295     //fmt.Printf(stderr, "--CRC32 return %d %d\n", crc, len)
1296     return crc;
1297 }
1298 func byteCRC32end(crc uint32, len uint64) (uint32) {
1299     var slen = make([]byte, 4)
1300     var li = 0
1301     for li = 0; li < 4; {
1302         slen[li] = byte(len)
1303         li += 1
1304         len >>= 8
1305         if (len == 0) {
1306             break
1307         }
1308     }
1309     crc = byteCRC32add(crc, slen, uint64(li))
1310     crc ^= 0xFFFFFFFF
1311     return crc
1312 }
1313 func strCRC32(str string, len uint64) (crc uint32) {
1314     crc = byteCRC32add(0, []byte(str), len)
1315     crc = byteCRC32end(crc, len)
1316     //fmt.Printf(stderr, "--CRC32 %d %d\n", crc, len)
1317     return crc
1318 }
1319 func CRC32Finish(crc uint32, table *crc32.Table, len uint64) (uint32) {
1320     var slen = make([]byte, 4)
1321     var li = 0
1322     for li = 0; li < 4; {
1323         slen[li] = byte(len & 0xFF)
1324         li += 1
1325         len >>= 8
1326         if (len == 0) {
1327             break
1328         }
1329     }
1330     crc = crc32.Update(crc, table, slen)
1331     crc ^= 0xFFFFFFFF
1332     return crc
1333 }
1334
1335 func (gsh*GshContext)xChecksum(path string, argv []string, sum*Checksum) (int64) {
1336     if !isIn("-type/t", argv) && !IsRegFile(path) {
1337         return 0
1338     }
1339     if !isIn("-type/d", argv) && !IsRegFile(path) {
1340         return 0
1341     }
1342     file, err := os.OpenFile(path, os.O_RDONLY, 0)
1343     if err != nil {
1344         fmt.Printf("--E-- cksum %v (%v)\n", path, err)
1345         return -1
1346     }
1347     defer file.Close()
1348     if gsh.CmdTrace { fmt.Printf("--I-- cksum %v %v\n", path, argv) }
1349
1350     bi := 0
1351     var buff = make([]byte, 32*1024)
1352     var total int64 = 0
1353     var initTime = time.Time{}
1354     if sum.Start == initTime {
1355         sum.Start = time.Now()
1356     }
1357     for bi = 0; ; bi++ {
1358         count, err := file.Read(buff)
1359         if count <= 0 || err != nil {
1360             break
1361         }
1362         if (sum.SumType & SUM_SUM64) != 0 {
1363             s := sum.Sum64
1364             for _, c := range buff[0:count] {
1365                 s += uint64(c)
1366             }
1367             sum.Sum64 = s
1368         }
1369         if (sum.SumType & SUM_UNIXFILE) != 0 {
1370             sum.Crc32Val = byteCRC32add(sum.Crc32Val, buff, uint64(count))
1371         }
1372         if (sum.SumType & SUM_CRCIEEE) != 0 {
1373             sum.Crc32Val = crc32.Update(sum.Crc32Val, &sum.Crc32Table, buff[0:count])
1374         }
1375         // <a href="https://en.wikipedia.org/wiki/BSD_checksum">BSD checksum</a>
1376         if (sum.SumType & SUM_SUM16_BSD) != 0 {
1377             s := sum.Sum16
1378             for _, c := range buff[0:count] {
1379                 s = (s >> 1) + ((s & 1) << 15)
1380                 s += int(c)
1381                 s &= 0xFFFF
1382                 //fmt.Printf("BSDsum: %d[%d] %d\n", sum.Size+int64(i), i, s)
1383             }
1384             sum.Sum16 = s
1385         }
1386         if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1387             for bj := 0; bj < count; bj++ {
1388                 sum.Sum16 += int(buff[bj])
1389             }
1390         }
1391         total += int64(count)
1392     }
1393     sum.Done = time.Now()
1394     sum.Files += 1
1395     sum.Size += total
1396     if !isIn("-s", argv) {
1397         fmt.Printf("%v ", total)
1398     }
1399     return 0
1400 }
1401
1402 // <a name="grep">grep</a>
1403 // "lines", "lin" or "lnp" for "(text) line processor" or "scanner"
1404 // a*, lab, c, ... sequential combination of patterns
1405 // what "LINE" is should be definable
1406 // generic line-by-line processing
1407 // grep [-v]
1408 // cat -n -v
1409 // uniq [-c]
1410 // tail -f
1411 // sed s/x/y/ or awk
1412 // grep with line count like wc
1413 // rewrite contents if specified
1414 func (gsh*GshContext)xGrep(path string, rexp []string) (int) {
1415     file, err := os.OpenFile(path, os.O_RDONLY, 0)

```



```

1416 if err != nil {
1417     fmt.Printf("--E-- grep %v (%v)\n",path,err)
1418     return -1
1419 }
1420 defer file.Close()
1421 if gsh.CmdTrace { fmt.Printf("--I-- grep %v %v\n",path,rxpv) }
1422 //reader := bufio.NewReaderSize(file,LINESIZE)
1423 reader := bufio.NewReaderSize(file,80)
1424 li := 0
1425 found := 0
1426 for li = 0 ; li++ {
1427     line, err := reader.ReadString('\n')
1428     if len(line) <= 0 {
1429         break
1430     }
1431     if 150 < len(line) {
1432         // maybe binary
1433         break;
1434     }
1435     if err != nil {
1436         break
1437     }
1438     if 0 < strings.Index(string(line),rxpv[0]) {
1439         found += 1
1440         fmt.Printf("%s%d: %s",path,li,line)
1441     }
1442 }
1443 //fmt.Printf("total %d lines %s\n",li,path)
1444 //if( 0 < found ){ fmt.Printf("(found %d lines %s)\n",found,path); }
1445 return found
1446 }
1447
1448 // <a name="finder">Finder</a>
1449 // finding files with it name and contents
1450 // file names are OKed
1451 // show the content with %x fmt list
1452 // ls -R
1453 // tar command by adding output
1454 type fileSum struct {
1455     Err int64 // access error or so
1456     Size int64 // content size
1457     DupSize int64 // content size from hard links
1458     Blocks int64 // number of blocks (of 512 bytes)
1459     DupBlocks int64 // Blocks pointed from hard links
1460     HLinks int64 // hard links
1461     Words int64
1462     Lines int64
1463     Files int64
1464     Dirs int64 // the num. of directories
1465     SymLink int64
1466     Flats int64 // the num. of flat files
1467     MaxDepth int64
1468     MaxNameLen int64 // max. name length
1469     nextRepo time.Time
1470 }
1471 func showFusage(dir string,fusage *fileSum){
1472     bsume := float64(((fusage.Blocks-fusage.DupBlocks)/2)*1024)/1000000.0
1473     //bsumdup := float64((fusage.Blocks/2)*1024)/1000000.0
1474     fmt.Printf("%v: %v files (%vd %vs %vh) %.6f MB (%.2f MBK)\n",
1475         dir,
1476         fusage.Files,
1477         fusage.Dirs,
1478         fusage.SymLink,
1479         fusage.HLinks,
1480         float64(fusage.Size)/1000000.0,bsume);
1481 }
1482
1483 const (
1484     S_IPMT = 0170000
1485     S_IPCHR = 0020000
1486     S_IPDIR = 0040000
1487     S_IFREG = 0100000
1488     S_IFLNK = 0120000
1489     S_IFSOCK = 0140000
1490 )
1491 func cumPinfo(fsum *fileSum, path string, stater error, fstat aStat_t, argv[string,verb bool])(*fileSum){
1492     now := time.Now()
1493     if time.Second <= now.Sub(fsum.nextRepo) {
1494         if fsum.nextRepo.IsZero(){
1495             tstamp := now.Format(time.Stamp)
1496             showFusage(tstamp,fsum)
1497         }
1498         fsum.nextRepo = now.Add(time.Second)
1499     }
1500     if stater != nil {
1501         fsum.Err += 1
1502         return fsum
1503     }
1504     fsum.Files += 1
1505     if l < fstat.Nlink {
1506         // must count only once...
1507         // at least ignore ones in the same directory
1508         //if finfo.Mode().IsRegular() {
1509             if (fstat.Mode & S_IPMT) == S_IFREG {
1510                 fsum.HLinks += 1
1511                 fsum.DupBlocks += int64(fstat.Blocks)
1512                 //fmt.Printf("----Dup HardLink %v %s\n",fstat.Nlink,path)
1513             }
1514         }
1515         //fsum.Size += finfo.Size()
1516         fsum.Size += fstat.Size
1517         fsum.Blocks += int64(fstat.Blocks)
1518         //if verb { fmt.Printf("%8dBlk %s",fstat.Blocks/2,path) }
1519         if isin("-ls",argv){
1520             //if verb { fmt.Printf("%4d %8d ",fstat.Blksize,fstat.Blocks) }
1521             // fmt.Printf("%d\t",fstat.Blocks/2)
1522         }
1523         //if finfo.IsDir()
1524         if (fstat.Mode & S_IPMT) == S_IFDIR {
1525             fsum.Dirs += 1
1526         }
1527         //if (finfo.Mode() & os.ModeSymLink) != 0
1528         if (fstat.Mode & S_IPMT) == S_IFLNK {
1529             //if verb { fmt.Printf("symlink(%v,%s)\n",fstat.Mode,finfo.Name()) }
1530             //{ fmt.Printf("symlink(%o,%s)\n",fstat.Mode,finfo.Name()) }
1531             fsum.SymLink += 1
1532         }
1533     }
1534     return fsum
1535 }
1536 func (gsh*GshContext)xxFindEntv(depth int,total *fileSum,dir string, dstat aStat_t, ei int, envt []string,npatv[]string,argv[string])(*fileSum){
1537     nols := isin("-grep",argv)
1538     // sort envt
1539     /*
1540     if isin("-t",argv){
1541         sort.Slice(filev, func(i,j int) bool {
1542             return 0 < filev[i].ModTime().Sub(filev[j].ModTime())
1543         })
1544     }
1545     */
1546     /*
1547     if isin("-u",argv){
1548         sort.Slice(filev, func(i,j int) bool {
1549             return 0 < filev[i].AccTime().Sub(filev[j].AccTime())
1550         })
1551     }
1552     if isin("-U",argv){
1553         sort.Slice(filev, func(i,j int) bool {
1554             return 0 < filev[i].CreateTime().Sub(filev[j].CreateTime())
1555         })
1556     }
1557     */
1558     if isin("-s",argv){
1559         sort.Slice(filev, func(i,j int) bool {
1560             return filev[j].Size() < filev[i].Size()
1561         })
1562     }
1563     /*
1564     for _,filename := range envt {
1565         For _,npat := range npatv {
1566             match := true
1567             if npat == "*" {
1568                 match = true
1569             }else{
1570                 match, _ = filepath.Match(npat,filename)
1571             }
1572             path := dir + DIRSEP + filename
1573             if !match {
1574                 continue
1575             }
1576             var fstat aStat_t
1577             stater := alstat(path,&fstat)
1578             if stater != nil {
1579                 if !isin("-w",argv){fmt.Printf("ufind: %v\n",stater) }
1580                 continue;
1581             }
1582             if isin("-du",argv) && (fstat.Mode & S_IPMT) == S_IFDIR {
1583                 // should not show size of directory in "-du" mode ...
1584             }else
1585             if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1586                 if isin("-du",argv) {
1587                     fmt.Printf("%d\t",fstat.Blocks/2)
1588                 }
1589                 showFileInfo(path,argv)
1590             }
1591             if true { // && isin("-du",argv)
1592                 total = cumPinfo(total,path,stater,fstat,argv,false)

```

```

1593     }
1594     /*
1595     if isin("-wc",argv) {
1596     }
1597     */
1598     if gsh.lastCheckSum.SumType != 0 {
1599     gsh.xCksum(path,argv,gsh.lastCheckSum);
1600     }
1601     x := isinX("-grep",argv); // -grep will be convenient like -ls
1602     if 0 <= x && x+1 <= len(argv) { // -grep will be convenient like -ls
1603     if isRegFile(path){
1604     found := gsh.xGrep(path,argv[x+1]);
1605     if 0 < found {
1606     foundv := gsh.CmdCurrent.FoundFile
1607     if len(foundv) < 10 {
1608     gsh.CmdCurrent.FoundFile =
1609     append(gsh.CmdCurrent.FoundFile,path)
1610     }
1611     }
1612     }
1613     }
1614     if !isin("-r0",argv) { // -d 0 in du, -depth n in find
1615     //total.Depth += 1
1616     if (fstat.Mode & S_IFMT) == S_IFLNK {
1617     continue
1618     }
1619     if dstat.Rdev != fstat.Rdev {
1620     fmt.Printf("--I-- don't follow different device %v(%v) %v(%v)\n",
1621     dir,dstat.Rdev,path,fstat.Rdev)
1622     }
1623     if (fstat.Mode & S_IFMT) == S_IFDIR {
1624     total = gsh.xxFind(depth+1,total,path,npatv,argv)
1625     }
1626     }
1627     }
1628     }
1629     return total
1630 }
1631 func (gsh*GshContext)xxFind(depth int,total *fileSum,dir string,npatv[]string,argv[]string)(*fileSum){
1632     nols := isin("-grep",argv)
1633     dirfile,oerr := os.OpenFile(dir,os.O_RDONLY,0)
1634     if oerr == nil {
1635     //fmt.Printf("--I-- %v(%v)[%d]\n",dir,dirfile,dirfile.Fd())
1636     defer dirfile.Close()
1637     }else{
1638     }
1639     }
1640     prev := *total
1641     var dstat aStat_t
1642     staterr = aStat(dir,dstat) // should be flstat
1643     }
1644     if staterr != nil {
1645     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",staterr) }
1646     return total
1647     }
1648     //filev,err := ioutil.ReadDir(dir)
1649     //_,err := ioutil.ReadDir(dir) // ReadDir() heavy and bad for huge directory
1650     /*
1651     if err != nil {
1652     if !isin("-w",argv){ fmt.Printf("ufind: %v\n",err) }
1653     return total
1654     }
1655     */
1656     if depth == 0 {
1657     total = cumFinfo(total,dir,staterr,dstat,argv,true)
1658     if !nols && !isin("-s",argv) && (!isin("-du",argv) || isin("-a",argv)) {
1659     showFileinfo(dir,argv)
1660     }
1661     }
1662     // it is not a directory, just scan it and finish
1663     }
1664     for ei := 0; ; ei++ {
1665     entv,rderr := dirfile.Readdirnames(8*1024)
1666     if len(entv) == 0 || rderr != nil {
1667     //if rderr != nil { fmt.Printf("[%d] len=%d (%v)\n",ei,len(entv),rderr) }
1668     break
1669     }
1670     if 0 < ei {
1671     fmt.Printf("--I-- xxFind[%d] %d large-dir: %s\n",ei,len(entv),dir)
1672     }
1673     total = gsh.xxFindEntv(depth,total,dir,dstat,ei,entv,npatv,argv)
1674     }
1675     if isin("-du",argv) {
1676     // if in "du" mode
1677     fmt.Printf("%d\t%v\n",total.Blocks-prev.Blocks)/2,dir)
1678     }
1679     return total
1680 }
1681 }
1682 // {ufind|fu|ls} [Files] [-- Names] [-- Expressions]
1683 // Files is "." by default
1684 // Names is "*" by default
1685 // Expressions is "-print" by default for "ufind", or -du for "fu" command
1686 func (gsh*GshContext)xXFind(argv[]string){
1687     if 0 < len(argv) && strBegins(argv[0],"?"){
1688     showFound(gsh,argv)
1689     return
1690     }
1691     if isin("-cksum",argv) || isin("-sum",argv) {
1692     gsh.lastCheckSum = CheckSum{}
1693     if isin("-sum",argv) && isin("-add",argv) {
1694     gsh.lastCheckSum.SumType = SUM_SUM64
1695     }else
1696     if isin("-sum",argv) && isin("-size",argv) {
1697     gsh.lastCheckSum.SumType = SUM_SIZE
1698     }else
1699     if isin("-sum",argv) && isin("-bsd",argv) {
1700     gsh.lastCheckSum.SumType = SUM_SUM16_BSD
1701     }else
1702     if isin("-sum",argv) && isin("-sysv",argv) {
1703     gsh.lastCheckSum.SumType = SUM_SUM16_SYSV
1704     }else
1705     if isin("-sum",argv) {
1706     gsh.lastCheckSum.SumType = SUM_SUM64
1707     }
1708     if isin("-unix",argv) {
1709     gsh.lastCheckSum.SumType = SUM_UNIXFILE
1710     gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32UNIX)
1711     }
1712     if isin("-leee",argv){
1713     gsh.lastCheckSum.SumType = SUM_CRCIEEE
1714     gsh.lastCheckSum.Crc32Table = *crc32.MakeTable(CRC32IEEB)
1715     }
1716     gsh.lastCheckSum.RusgAtStart = Getrusagev()
1717     }
1718     var total = fileSum{}
1719     npats = []string{}
1720     for v := range argv {
1721     if 0 < len(v) && v[0] != '-' {
1722     npats = append(npats,v)
1723     }
1724     if v == "/" { break }
1725     if v == "--" { break }
1726     if v == "-grep" { break }
1727     if v == "-ls" { break }
1728     }
1729     if len(npats) == 0 {
1730     npats = []string{"*"}
1731     }
1732     cwd := "."
1733     // if to be fullpath :: cwd, _ := os.Getwd()
1734     if len(npats) == 0 { npats = []string{"*"} }
1735     fusage := gsh.xxFind(0,&total,cwd,npats,argv)
1736     if gsh.lastCheckSum.SumType != 0 {
1737     var sumi uint64 = 0
1738     sum := gsh.lastCheckSum
1739     if (sum.SumType & SUM_SIZE) != 0 {
1740     sumi = uint64(sum.Size)
1741     }
1742     if (sum.SumType & SUM_SUM64) != 0 {
1743     sumi = sum.Sum64
1744     }
1745     if (sum.SumType & SUM_SUM16_SYSV) != 0 {
1746     s := uint32(sum.Sum16)
1747     r := (s & 0xFFFF) + ((s & 0xFFFFFFF) >> 16)
1748     s = (r & 0xFFFF) + (r >> 16)
1749     sum.Crc32Val = uint32(s)
1750     sumi = uint64(s)
1751     }
1752     if (sum.SumType & SUM_SUM16_BSD) != 0 {
1753     sum.Crc32Val = uint32(sum.Sum16)
1754     sumi = uint64(sum.Sum16)
1755     }
1756     if (sum.SumType & SUM_UNIXFILE) != 0 {
1757     sum.Crc32Val = byteCRC32end(sum.Crc32Val,uint64(sum.Size))
1758     sumi = uint64(byteCRC32end(sum.Crc32Val,uint64(sum.Size)))
1759     }
1760     if 1 < sum.Files {
1761     fmt.Printf("%v %v // %v / %v files, %v/file\r\n",
1762     sumi,sum.Size,
1763     absSize(sum.Size),sum.Files,
1764     absSize(sum.Size/sum.Files))
1765     }else{
1766     fmt.Printf("%v %v %v\n",
1767     sumi,sum.Size,npats[0])
1768     }
1769     }

```

```

1770 if !isin("-grep",argv) {
1771     showFusage("total",fusage)
1772 }
1773 if !isin("-s",argv){
1774     hits := len(gsh.CmdCurrent.FoundFile)
1775     if 0 < hits {
1776         fmt.Printf("--!-  %d files hits // can be refered with !&df\n",
1777             hits,len(gsh.CommandHistory))
1778     }
1779 }
1780 if gsh.lastCheckSum.SumType != 0 {
1781     if !isin("-ru",argv) {
1782         sum := gsh.lastCheckSum
1783         sum.Done = time.Now()
1784         gsh.lastCheckSum.RusgAtEnd = Getrusagev()
1785         elps := sum.Done.Sub(sum.Start)
1786         fmt.Printf("--cksum-size: %v (%v) / %v files, %v/file\n",
1787             sum.Size,abssize(sum.Size),sum.Files,abssize(sum.Size/sum.Files))
1788         nanos := int64(elps)
1789         dnanos := time.Duration(nanos)
1790         fmt.Printf("--cksum-time: %v/total, %v/file, %i if files/s, %v\n",
1791             abftime(dnanos),
1792             abftime(time.Duration(nanos/sum.Files)),
1793             (float64(sum.Files)*1000000000.0)/float64(nanos),
1794             abbsped(sum.Size,nanos))
1795         diff := RusageSubv(sum.RusgAtEnd,sum.RusgAtStart)
1796         fmt.Printf("--cksum-rusg: %v\n",sRusagef("",argv,diff))
1797     }
1798 }
1799 return
1800 }
1801
1802 func showFiles(files []string){
1803     sp := ""
1804     for i, file := range files {
1805         if 0 < i { sp = " " } else { sp = "" }
1806         fmt.Printf(sp+"%s",escapeWhiteSP(file))
1807     }
1808 }
1809
1810 func showFound(gshCtx *GshContext, argv []string){
1811     for i,v := range gshCtx.CommandHistory {
1812         if 0 < len(v.FoundFile) {
1813             fmt.Printf("%d (%d)",i,len(v.FoundFile))
1814             if !isin("-ls",argv){
1815                 fmt.Printf("\n")
1816                 for _,file := range v.FoundFile {
1817                     fmt.Printf("") //sub number?
1818                     showFileInfo(file,argv)
1819                 }
1820             }else{
1821                 showFiles(v.FoundFile)
1822                 fmt.Printf("\n")
1823             }
1824         }
1825     }
1826 }
1827
1828 func showMatchFile(filev []os.FileInfo, npat,dir string, argv []string)(string,bool){
1829     fname := ""
1830     found := false
1831     for _,v := range filev {
1832         match, _ := filepath.Match(npat,(v.Name()))
1833         if match {
1834             fname = v.Name()
1835             found = true
1836             //fmt.Printf("[%d] %s\n",i,v.Name())
1837             showIfExecutable(fname,dir,argv)
1838         }
1839     }
1840     return fname,found
1841 }
1842
1843 func showIfExecutable(name,dir string,argv []string)(ffullpath string,ffound bool){
1844     var fullpath string
1845     if strBegins(name,DIRSEP){
1846         fullpath = name
1847     }else
1848     if( len(dir) == 0 ){
1849         fullpath = name;
1850     }else{
1851         fullpath = dir + DIRSEP + name
1852     }
1853     fi, err := os.Stat(fullpath)
1854     //fmt.Printf("--Dp-- %v\n-- %v\n",fullpath,err);
1855     if err != nil {
1856         fullpath = ".exe";
1857         fi, err = os.Stat(fullpath)
1858     }
1859     if err != nil {
1860         fullpath = dir + DIRSEP + name + ".go"
1861         fi, err = os.Stat(fullpath)
1862     }
1863     if err == nil {
1864         fm := fi.Mode()
1865         if fm.IsRegular() {
1866             // R_OK=4, W_OK=2, X_OK=1, F_OK=0
1867             if aAccess(fullpath,5) == nil {
1868                 ffullpath = fullpath
1869                 ffound = true
1870                 if !isin("-s", argv) {
1871                     showFileInfo(ffullpath,argv)
1872                 }
1873             }
1874         }
1875     }
1876     return ffullpath, ffound
1877 }
1878
1879 func which(list string, argv []string) (fullpath []string, itis bool){
1880     if len(argv) <= 1 {
1881         fmt.Printf("Usage: which comand [-s] [-a] [-ls]\n")
1882         return []string(""), false
1883     }
1884     path := argv[1]
1885     if strBegins(path,"/") {
1886         // should check if executable?
1887         ,exOK := showIfExecutable(path,"",argv)
1888         fmt.Printf("--D- %v exOK=%v\n",path,exOK)
1889         return []string(path),exOK
1890     }
1891     pathenv, efound := os.LookupEnv(list)
1892     if !efound {
1893         fmt.Printf("--E- which: no \"%s\" environment\n",list)
1894         return []string(""), false
1895     }
1896     //fmt.Printf("PATH=%v\n",pathenv);
1897     showall := !isin("-a",argv) || 0 < strings.Index(path,"*")
1898     dirv := strings.Split(pathenv,PATHEP)
1899     ffound := false
1900     ffullpath := path
1901     for dir := range dirv {
1902         if 0 <= strings.Index(path,"*") { // by wild-card
1903             list, _ := ioutil.ReadDir(dir)
1904             ffullpath, ffound = showMatchFile(list,path,dir,argv)
1905         }else{
1906             fullpath, ffound = showIfExecutable(path,dir,argv)
1907         }
1908         //if ffound && !isin("-a", argv) {
1909             if ffound && !showall {
1910                 break;
1911             }
1912     }
1913     return []string(ffullpath), ffound
1914 }
1915
1916 func stripLeadingWSParg(argv []string) ([]string){
1917     for ; 0 < len(argv); {
1918         if len(argv[0]) == 0 {
1919             argv = argv[1:]
1920         }else{
1921             break
1922         }
1923     }
1924     return argv
1925 }
1926
1927 func xEval(argv []string, nlend bool){
1928     argv = stripLeadingWSParg(argv)
1929     if len(argv) == 0 {
1930         fmt.Printf("eval [%format] [Go-expression]\n")
1931         return
1932     }
1933     pfmt := "%v"
1934     if argv[0][0] == '$' {
1935         pfmt = argv[0]
1936         argv = argv[1:]
1937     }
1938     if len(argv) == 0 {
1939         return
1940     }
1941     gocode := strings.Join(argv," ");
1942     //fmt.Printf("eval [%v] [%v]\n",pfmt,gocode)
1943     fset := token.NewFileSet()
1944     rval := types.Eval(fset,nil,token.NoPos,gocode)
1945     fmt.PrIntf(pfmt,rval.Value)
1946     if nlend { fmt.Printf("\n") }
1947 }
1948
1949 func getval(name string) (found bool, val int) {
1950     /* should expand the name here */

```

```

1947 if name == "gsh.pid" {
1948     return true, os.Getpid()
1949 }else
1950 if name == "gsh.ppid" {
1951     return true, os.Getppid()
1952 }
1953 return false, 0
1954 }
1955
1956 func echo(argv []string, nlen bool){
1957     for ai := 1; ai < len(argv); ai++ {
1958         if 1 < ai {
1959             fmt.Printf(" ");
1960         }
1961         arg := argv[ai]
1962         found, val := getval(arg)
1963         if found {
1964             fmt.Printf("%d",val)
1965         }else{
1966             fmt.Printf("%s",arg)
1967         }
1968     }
1969     if nlen {
1970         fmt.Printf("\n");
1971     }
1972 }
1973
1974 func resfile() string {
1975     return "gsh.tmp"
1976 }
1977 //var resF *File
1978 func resmap() {
1979     //_, err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, os.ModeAppend)
1980     //https://developpaper.com/solution-to-golang-bad-file-descriptor-problem/
1981     err := os.OpenFile(resfile(), os.O_RDWR|os.O_CREATE, 0600)
1982     if err != nil {
1983         fmt.Printf("refF could not open: %s\n",err)
1984     }else{
1985         fmt.Printf("refF opened\n")
1986     }
1987 }
1988
1989 // @@2020-0821
1990 func gshScanArg(str string,strip int)(argv []string){
1991     var si = 0
1992     var sb = 0
1993     var inBracket = 0
1994     var argl = make([]byte,LINESIZE)
1995     var ax = 0
1996     debug := false
1997
1998     for ; si < len(str); si++ {
1999         if str[si] != ' ' {
2000             break
2001         }
2002     }
2003     sb = si
2004     for ; si < len(str); si++ {
2005         if sb <= si {
2006             if debug {
2007                 fmt.Printf("--Da- +%d %d-%d %s ... %s\n",
2008                     inBracket,sb,si,argl[0:ax],str[si:])
2009             }
2010             ch := str[si]
2011             if ch == '{' {
2012                 inBracket++
2013                 if 0 < strip && inBracket <= strip {
2014                     //fmt.Printf("stripLEV %d <= %d?\n",inBracket,strip)
2015                     continue
2016                 }
2017             }
2018             if 0 < inBracket {
2019                 if ch == '}' {
2020                     inBracket--
2021                     if 0 < strip && inBracket < strip {
2022                         //fmt.Printf("stripLEV %d < %d?\n",inBracket,strip)
2023                         continue
2024                     }
2025                 }
2026                 argl[ax] = ch
2027                 ax++
2028                 continue
2029             }
2030             if str[si] == ' ' {
2031                 argv = append(argv,string(argl[0:ax]))
2032                 if debug {
2033                     fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2034                         -1*len(argv),sb,si,str[sb:si],string(str[si:]))
2035                 }
2036                 sb = si+1
2037                 ax = 0
2038                 continue
2039             }
2040             argl[ax] = ch
2041             ax++
2042         }
2043     }
2044     if sb < si {
2045         argv = append(argv,string(argl[0:ax]))
2046         if debug {
2047             fmt.Printf("--Da- [%v][%v-%v] %s ... %s\n",
2048                 -1*len(argv),sb,si,string(argl[0:ax]),string(str[si:]))
2049         }
2050     }
2051     if debug {
2052         fmt.Printf("--Da- %d [%s] => [%d]%v\n",strip,str,len(argv),argv)
2053     }
2054     return argv
2055 }
2056
2057 // should get stderr (into tmpfile ?) and return
2058 func (gsh*GshContext)Popen(name,mode string)(pin*os.File,pout*os.File,err bool){
2059     //var pv = []int{-1,-1}
2060     //syscall.Pipe(pv)
2061
2062     xarg := gshScanArg(name,1)
2063     name = strings.Join(xarg, " ")
2064
2065     //pin = os.NewFile(uintptr(pv[0]),"StdoutOf-"+name+"")
2066     //pout = os.NewFile(uintptr(pv[1]),"StdinOf-"+name+"")
2067     pin,pout,_ = os.Pipe()
2068
2069     fdix := 0
2070     dir := ""
2071     if mode == "r" {
2072         dir = "<"
2073         fdix = 1 // read from the stdout of the process
2074     }else{
2075         dir = ">"
2076         fdix = 0 // write to the stdin of the process
2077     }
2078     gshPA := gsh.gshPA
2079     savfd := gshPA.Files[fdix]
2080
2081     var fd uintptr = 0
2082     if mode == "r" {
2083         //fd = pout.Fd()
2084         //gshPA.Files[fdix] = pout.Fd()
2085         gshPA.Files[fdix] = pout;
2086     }else{
2087         //fd = pin.Fd()
2088         //gshPA.Files[fdix] = pin.Fd()
2089         gshPA.Files[fdix] = pin;
2090     }
2091     // should do this by Goroutine?
2092     if false {
2093         fmt.Printf("--Ip- Opened fd[%v] %s %v\n",fd,dir,name)
2094         fmt.Printf("--RED1 [%d,%d,%d]->[%d,%d,%d]\n",
2095             os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd(),
2096             pin.Fd(),pout.Fd(),pout.Fd())
2097     }
2098     savi := os.Stdin
2099     savo := os.Stdout
2100     save := os.Stderr
2101     os.Stdin = pin
2102     os.Stdout = pout
2103     os.Stderr = pout
2104     gsh.BackGround = true
2105     gsh.gshellh(name)
2106     gsh.BackGround = false
2107     os.Stdin = savi
2108     os.Stdout = savo
2109     os.Stderr = save
2110
2111     gshPA.Files[fdix] = savfd
2112     return pin,pout,false
2113 }
2114
2115 // <a name="ex-commands">External commands</a>
2116 func (gsh*GshContext)excommand(exec bool, argv []string) (notf bool,exit bool) {
2117     if gsh.CmdTrace { fmt.Printf("--I- excommand[%v](%v)\n",exec,argv) }
2118
2119     gshPA := gsh.gshPA
2120     fullpathv, itis := which("PATH",[]string{"which",argv[0],"-s"})
2121     if itis == false {
2122         return true,false
2123     }
2124 }

```

```

2124 fullpath := fullpathv[0]
2125 argv = unescapeWhiteSPV(argv)
2126 if 0 < strings.Index(fullpath, ".go") {
2127     nargv := argv // []string{}
2128     gofullpathv, itis := which("PATH", []string{"which", "go", "-s"})
2129     if itis == false {
2130         fmt.Printf("--Go not found\n")
2131         return false, true
2132     }
2133     gofullpath := gofullpathv[0]
2134     nargv = []string{ gofullpath, "run", fullpath }
2135     fmt.Printf("--I- %s (%s %s %s)\n", gofullpath,
2136         nargv[0], nargv[1], nargv[2])
2137     if exec {
2138         syscall.Exec(gofullpath, nargv, os.Environ())
2139     } else {
2140         // /pid, _ := syscall.ForkExec(gofullpath, nargv, &gshPA)
2141         proc, _ := os.StartProcess(gofullpath, nargv, &gshPA);
2142         pstat, _ := proc.Wait();
2143         pid := pstat.Pid();
2144         if gsh.Background {
2145             fmt.Fprintf(stderr, "--Ip- in Background pid(%d)&($v)\n", pid, len(argv), nargv)
2146             //gsh.BackgroundJobs = append(gsh.BackgroundJobs, pid)
2147             gsh.BackgroundJobs = append(gsh.BackgroundJobs, *pstat)
2148         } else {
2149             /*
2150             rusage := aRusage {
2151             syscall.Wait4(pid, nil, 0, &rusage)
2152             gsh.LastRusage = rusage
2153             gsh.CmdCurrent.Rusageev[1] = rusage
2154             */
2155         }
2156         /*
2157         gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2158         gsh.CmdCurrent.Rusageev[1] = *pstat.SysUsage().(*aRusage);
2159         */
2160         aSetRusage(&gsh.LastRusage, pstat);
2161         gsh.CmdCurrent.Rusageev[1] = gsh.LastRusage;
2162     }
2163 }
2164 } else {
2165     if exec {
2166         syscall.Exec(fullpath, argv, os.Environ())
2167     } else {
2168         // /pid, _ := syscall.ForkExec(fullpath, argv, &gshPA)
2169         proc, _ := os.StartProcess(fullpath, argv, &gshPA);
2170         pstat, _ := proc.Wait();
2171         pid := pstat.Pid();
2172         // /fmt.Printf("%d\n", pid); // 's' to be background
2173         if false {
2174             // /fmt.Printf("Sys=%v\n", gshPA.Sys);
2175             if gshPA.Sys != nil {
2176                 // /fmt.Printf("inPG=%v\n", gshPA.Sys.Foreground);
2177             }
2178         }
2179         if gsh.Background {
2180             fmt.Fprintf(stderr, "--Ip- in Background pid(%d)&($v)\n", pid, len(argv), argv)
2181             //gsh.BackgroundJobs = append(gsh.BackgroundJobs, pid)
2182             gsh.BackgroundJobs = append(gsh.BackgroundJobs, *pstat)
2183         } else {
2184             /*
2185             rusage := aRusage {
2186             syscall.Wait4(pid, nil, 0, &rusage);
2187             gsh.LastRusage = rusage
2188             gsh.CmdCurrent.Rusageev[1] = rusage
2189             */
2190         }
2191         /*
2192         gsh.LastRusage = *pstat.SysUsage().(*aRusage);
2193         gsh.CmdCurrent.Rusageev[1] = *pstat.SysUsage().(*aRusage);
2194         */
2195         aSetRusage(&gsh.LastRusage, pstat);
2196         gsh.CmdCurrent.Rusageev[1] = gsh.LastRusage;
2197     }
2198 }
2199 return false, false
2200 }
2201 // <a name="builtin">Builtin Commands</a>
2202 func (gshCtx *GshContext) sleep(argv []string) {
2203     if len(argv) < 2 {
2204         fmt.Printf("Sleep 100ms, 100us, 100ns, ... \n")
2205         return
2206     }
2207     duration := argv[1];
2208     d, err := time.ParseDuration(duration)
2209     if err != nil {
2210         d, err = time.ParseDuration(duration+"s")
2211         if err != nil {
2212             fmt.Printf("duration ? %s (%s)\n", duration, err)
2213             return
2214         }
2215     }
2216     // /fmt.Printf("Sleep %v\n", duration)
2217     time.Sleep(d)
2218     if 0 < len(argv[2:]) {
2219         gshCtx.gshellv(argv[2:])
2220     }
2221 }
2222 func (gshCtx *GshContext) repeat(argv []string) {
2223     if len(argv) < 2 {
2224         return
2225     }
2226     start0 := time.Now()
2227     for ri, _ := strconv.Atoi(argv[1]); 0 < ri; ri-- {
2228         if 0 < len(argv[2:]) {
2229             // /start := time.Now()
2230             gshCtx.gshellv(argv[2:])
2231             end := time.Now()
2232             elps := end.Sub(start0);
2233             if (1000000000 < elps) {
2234                 fmt.Printf("repeat#%d %v\n", ri, elps);
2235             }
2236         }
2237     }
2238 }
2239 }
2240 func (gshCtx *GshContext) gen(argv []string) {
2241     gshPA := gshCtx.gshPA
2242     if len(argv) < 2 {
2243         fmt.Printf("Usage: %s N\n", argv[0])
2244         return
2245     }
2246     // should be repeated by "repeat" command
2247     count, _ := strconv.Atoi(argv[1])
2248     // /fd := gshPA.Files[1] // Stdout
2249     // /file := os.NewFile(fd, "internalStdout")
2250     file := gshPA.Files[1] // Stdout
2251     fmt.Printf("--I- Gen. Count=%d to (%d)\n", count, file.Fd())
2252     // /buf := []byte{}
2253     outdata := "0123 5678 0123 5678 0123 5678 0123 5678\r"
2254     for gi := 0; gi < count; gi++ {
2255         file.WriteString(outdata)
2256     }
2257     // /file.WriteString("\n")
2258     fmt.Printf("\n(%d B)\n", count*len(outdata));
2259     // /file.Close()
2260 }
2261 }
2262 // <a name="rexec">Remote Execution</a> // 2020-0820
2263 func Elapsed(from time.Time)(string){
2264     elps := time.Now().Sub(from)
2265     if 1000000000 < elps {
2266         return fmt.Sprintf("%5d.%02ds", elps/1000000000, (elps%1000000000)/10000000)
2267     } else {
2268         if 1000000 < elps {
2269             return fmt.Sprintf("%3d.%03dms", elps/1000000, (elps%1000000)/1000)
2270         } else {
2271             return fmt.Sprintf("%3d.%03dus", elps/1000, (elps%1000))
2272         }
2273     }
2274 }
2275 // /func abftime(nanos int64)(string){
2276 func abftime(nanos time.Duration)(string){
2277     if 1000000000 < nanos {
2278         return fmt.Sprintf("%d.%02ds", nanos/1000000000, (nanos%1000000000)/10000000)
2279     } else {
2280         if 1000000 < nanos {
2281             return fmt.Sprintf("%d.%03dms", nanos/1000000, (nanos%1000000)/1000)
2282         } else {
2283             return fmt.Sprintf("%d.%03dus", nanos/1000, (nanos%1000))
2284         }
2285     }
2286 }
2287 func absfsize(size int64)(string){
2288     fsize := float64(size)
2289     if 1024*1024*1024 < size {
2290         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))
2291     } else {
2292         if 1024*1024 < size {
2293             return fmt.Sprintf("%.3fMiB", fsize/(1024*1024))
2294         } else {
2295             return fmt.Sprintf("%.3fKiB", fsize/1024)
2296         }
2297     }
2298 }
2299 func absfsize(size int64)(string){
2300     fsize := float64(size)
2301     if 1024*1024*1024 < size {
2302         return fmt.Sprintf("%.2fGiB", fsize/(1024*1024*1024))

```

```

2301 }else
2302 if 1024*1024 < size {
2303 return fmt.Sprintf("%8.3fMiB",fsize/(1024*1024))
2304 }else{
2305 return fmt.Sprintf("%8.3fKiB",fsize/1024)
2306 }
2307 }
2308 func abspspeed(totalB int64,ns int64)(string){
2309 MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2310 if 1000 <= MBs {
2311 return fmt.Sprintf("%6.3fGB/s",MBs/1000)
2312 }
2313 if 1 <= MBs {
2314 return fmt.Sprintf("%6.3fMB/s",MBs)
2315 }else{
2316 return fmt.Sprintf("%6.3fKB/s",MBs*1000)
2317 }
2318 }
2319 func abspspeed(totalB int64,ns time.Duration)(string){
2320 MBs := (float64(totalB)/1000000) / (float64(ns)/1000000000)
2321 if 1000 <= MBs {
2322 return fmt.Sprintf("%6.3fGBps",MBs/1000)
2323 }
2324 if 1 <= MBs {
2325 return fmt.Sprintf("%6.3fMbps",MBs)
2326 }else{
2327 return fmt.Sprintf("%6.3fKBps",MBs*1000)
2328 }
2329 }
2330 func fileRelay(what string,in*os.File,out*os.File,size int64,bsiz int)(wcount int64){
2331 Start := time.Now()
2332 buff := make([]byte,bsiz)
2333 var total int64 = 0
2334 var rem int64 = size
2335 nio := 0
2336 Prev := time.Now()
2337 var PrevSize int64 = 0
2338
2339 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) START\n",
2340 what,absize(total),size,nio)
2341
2342 for i:= 0; ; i++ {
2343 var len = bsiz
2344 if int(rem) < len {
2345 len = int(rem)
2346 }
2347 Now := time.Now()
2348 Elps := Now.Sub(Prev);
2349 if 1000000000 < Now.Sub(Prev) {
2350 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %s\n",
2351 what,absize(total),size,nio,
2352 abspspeed((total-PrevSize),Elps))
2353 Prev = Now;
2354 PrevSize = total
2355 }
2356 rlen := len
2357 if in != nil {
2358 // should watch the disconnection of out
2359 rcc,err := in.Read(buff[0:rlen])
2360 if err != nil {
2361 fmt.Printf(Elapsed(Start)+"--En- X: %s read(%v,%v)<%v\n",
2362 what,rcc,err,in.Name())
2363 break
2364 }
2365 rlen = rcc
2366 if string(buff[0:rlen]) == "(SoftEOF " {
2367 var ecc int64 = 0
2368 fmt.Sscanf(string(buff),"(SoftEOF %v",&ecc)
2369 fmt.Printf(Elapsed(Start)+"--En- X: %s Recv ((SoftEOF %v))/%v\n",
2370 what,ecc,total)
2371 if ecc == total {
2372 break
2373 }
2374 }
2375 }
2376 wlen := rlen
2377 if out != nil {
2378 wcc,err := out.Write(buff[0:rlen])
2379 if err != nil {
2380 fmt.Printf(Elapsed(Start)+"--En- X: %s write(%v,%v)>%v\n",
2381 what,wcc,err,out.Name())
2382 break
2383 }
2384 wlen = wcc
2385 }
2386 if wlen < rlen {
2387 fmt.Printf(Elapsed(Start)+"--En- X: %s incomplete write (%v/%v)\n",
2388 what,wlen,rlen)
2389 break;
2390 }
2391 }
2392 nio += 1
2393 total += int64(rlen)
2394 rem -= int64(rlen)
2395 if rem <= 0 {
2396 break
2397 }
2398 }
2399 }
2400 Done := time.Now()
2401 Elps := float64(Done.Sub(Start))/1000000000 //Seconds
2402 TotalMB := float64(total)/1000000 //MB
2403 MBps := totalMB / Elps
2404 fmt.Printf(Elapsed(Start)+"--In- X: %s (%v/%v) %v % 3fMB/s\n",
2405 what,total,size,nio,absize(total),MBps)
2406 return total
2407 }
2408 func tcpPush(clnt *os.File){
2409 // shrink socket buffer and recover
2410 usleep(100);
2411 }
2412 func (gsh*GshContext)RexecServer(argv[]string){
2413 debug := true
2414 Start0 := time.Now()
2415 Start := Start0
2416 // if local == ":" { local = "0.0.0.0:9999" }
2417 local := "0.0.0.0:9999"
2418
2419 if 0 < len(argv) {
2420 if argv[0] == "-" {
2421 debug = false
2422 argv = argv[1:]
2423 }
2424 }
2425 if 0 < len(argv) {
2426 argv = argv[1:]
2427 }
2428 port, err := net.ResolveTCPAddr("tcp",local);
2429 if err != nil {
2430 fmt.Printf("--En- S: Address error: %s (%s)\n",local,err)
2431 return
2432 }
2433 fmt.Printf(Elapsed(Start)+"--In- S: Listening at %s...\n",local);
2434 sconn, err := net.ListenTCP("tcp", port)
2435 if err != nil {
2436 fmt.Printf(Elapsed(Start)+"--En- S: Listen error: %s (%s)\n",local,err)
2437 return
2438 }
2439
2440 reqbuf := make([]byte,LINESIZE)
2441 res := ""
2442 for {
2443 fmt.Printf(Elapsed(Start0)+"--In- S: Listening at %s...\n",local);
2444 aconn, err := sconn.AcceptTCP()
2445 Start = time.Now()
2446 if err != nil {
2447 fmt.Printf(Elapsed(Start)+"--En- S: Accept error: %s (%s)\n",local,err)
2448 return
2449 }
2450 clnt, _ := aconn.File()
2451 fd := Clnt.Fd()
2452 ar := aconn.RemoteAddr()
2453 if debug { fmt.Printf(Elapsed(Start0)+"--In- S: Accepted TCP at %s [%d] <- %v\n",
2454 local,fd,ar) }
2455 res = fmt.Sprintf("220 GShell/%s Server\r\n",VERSION)
2456 fmt.Fprintln(clnt, res)
2457 if debug { fmt.Printf(Elapsed(Start)+"--In- S: %s",res) }
2458 count, err := clnt.Read(reqbuf)
2459 if err != nil {
2460 fmt.Printf(Elapsed(Start)+"--En- C: (%v %v) %v",
2461 count,err,string(reqbuf))
2462 }
2463 req := string(reqbuf[:count])
2464 if debug { fmt.Printf(Elapsed(Start)+"--In- C: %v",string(req)) }
2465 reqv := strings.Split(string(req),"\r")
2466 cmdv := gshScanArg(reqv[0],0)
2467 //cmdv := strings.Split(reqv[0]," ")
2468 switch cmdv[0] {
2469 case "HELO":
2470 res = fmt.Sprintf("250 %v",req)
2471 case "GRR":
2472 // download [remotefile]-zN [localfile]
2473 var dszize int64 = 32*1024*1024
2474 var bsiz int = 64*1024
2475 var fname string = ""
2476 var in *os.File = nil
2477 var pseudoEOF = false

```

```

2478     if l < len(cmdv) {
2479         fname = cmdv[l]
2480         if strbegins(fname, "z") {
2481             fmt.Sscanf(fname[2:], "%d", &dsize)
2482         } else
2483         if strbegins(fname, "(") {
2484             xin, xout, err := gsh.Popen(fname, "r")
2485             if err {
2486                 } else {
2487                 xout.Close()
2488                 defer xin.Close()
2489                 in = xin
2490                 dsize = MaxStreamSize
2491                 pseudoEOF = true
2492             }
2493         } else {
2494             xin, err := os.Open(fname)
2495             if err != nil {
2496                 fmt.Printf("--En- GET (%v)\n", err)
2497             } else {
2498                 defer xin.Close()
2499                 in = xin
2500                 fi, _ := xin.Stat()
2501                 dsize = fi.Size()
2502             }
2503         }
2504     }
2505     //fmt.Printf(Elapsed(Start)+"--In- GET %v:%v\n", dsize, bsize)
2506     res = fmt.Sprintf("200 %v\n", dsize)
2507     fmt.Fprintf(clnt, "%v", res)
2508     tcpPush(clnt); // should be separated as line in receiver
2509     fmt.Printf(Elapsed(Start)+"--In- S: %v", res)
2510     wcount := fileRelay("SendGET", in, clnt, dsize, bsize)
2511     if pseudoEOF {
2512         in.Close() // pipe from the command
2513         // show end of stream data (its size) by OOB?
2514         SoftEOF := fmt.Sprintf("(SoftEOF %v)", wcount)
2515         fmt.Printf(Elapsed(Start)+"--In- S: Send %v\n", SoftEOF)
2516     }
2517     tcpPush(clnt); // to let SoftEOF data appear at the top of received data
2518     fmt.Fprintf(clnt, "%v\n", SoftEOF)
2519     tcpPush(clnt); // to let SoftEOF alone in a packet (separate with 200 OK)
2520     // with client generated random?
2521     //fmt.Printf("--In- L: close %v (%v)\n", in.Fd(), in.Name())
2522     }
2523     res = fmt.Sprintf("200 GET done\n")
2524     case "PUT":
2525         // upload (srcfile|-zN) [dstfile]
2526         var dsize int64 = 32*1024*1024
2527         var bsize int = 64*1024
2528         var fname string = ""
2529         var out *os.File = nil
2530         if l < len(cmdv) { // Localfile
2531             fmt.Sscanf(cmdv[l], "%d", &dsize)
2532         }
2533         if 2 < len(cmdv) {
2534             fname = cmdv[2]
2535             if fname == "-" {
2536                 // nul dev
2537             } else
2538             if strbegins(fname, "(") {
2539                 xin, xout, err := gsh.Popen(fname, "w")
2540                 if err {
2541                     } else {
2542                     xin.Close()
2543                     defer xout.Close()
2544                     out = xout
2545                 }
2546             } else {
2547                 // should write to temporary file
2548                 // should suppress "C on tty
2549                 xout, err := os.OpenFile(fname, os.O_CREATE|os.O_RDWR|os.O_TRUNC, 0600)
2550                 //fmt.Printf("--In- S: open(%v) out(%v) err(%v)\n", fname, xout, err)
2551                 if err != nil {
2552                     fmt.Printf("--En- PUT (%v)\n", err)
2553                 } else {
2554                     out = xout
2555                 }
2556             }
2557             fmt.Printf(Elapsed(Start)+"--In- L: open(%v,w) %v (%v)\n",
2558                 fname, Local, err)
2559         }
2560         fmt.Printf(Elapsed(Start)+"--In- PUT %v (%v)\n", dsize, bsize)
2561         fmt.Printf(Elapsed(Start)+"--In- S: 200 %v OK\n", dsize)
2562         fmt.Fprintf(clnt, "200 %v OK\n", dsize)
2563         fileRelay("RecvPUT", clnt, out, dsize, bsize)
2564         res = fmt.Sprintf("200 PUT done\n")
2565         default:
2566             res = fmt.Sprintf("400 What? %v", req)
2567     }
2568     swcc, serr := clnt.Write([]byte(res))
2569     if serr != nil {
2570         fmt.Printf(Elapsed(Start)+"--In- S: (wc=%v er=%v) %v", swcc, serr, res)
2571     } else {
2572         fmt.Printf(Elapsed(Start)+"--In- S: %v", res)
2573     }
2574     aconn.Close();
2575     clnt.Close();
2576     }
2577     sconn.Close();
2578 }
2579 func (gsh*GshContext)RexecClient(argv []string)(int, string){
2580     debug = true
2581     Start := time.Now()
2582     if len(argv) == 1 {
2583         return -1, "EmptyARG"
2584     }
2585     argv = argv[1:]
2586     if argv[0] == "-serv" {
2587         gsh.RexecServer(argv[1:])
2588         return 0, "Server"
2589     }
2590     remote := "0.0.0.0:9999"
2591     if argv[0][0] == '#' {
2592         remote = argv[0][1:]
2593         argv = argv[1:]
2594     }
2595     if argv[0] == "-s" {
2596         debug = false
2597         argv = argv[1:]
2598     }
2599     dport, err := net.ResolveTCPAddr("tcp", remote);
2600     if err != nil {
2601         fmt.Printf(Elapsed(Start)+"Address error: %s (%s)\n", remote, err)
2602         return -1, "AddressError"
2603     }
2604     fmt.Printf(Elapsed(Start)+"--In- C: Connecting to %s\n", remote)
2605     serv, err := net.DialTCP("tcp", nil, dport)
2606     if err != nil {
2607         fmt.Printf(Elapsed(Start)+"Connection error: %s (%s)\n", remote, err)
2608         return -1, "CannotConnect"
2609     }
2610     if debug {
2611         al := serv.LocalAddr()
2612         fmt.Printf(Elapsed(Start)+"--In- C: Connected to %v <- %v\n", remote, al)
2613     }
2614     req := ""
2615     res = make([]byte, LINESIZE)
2616     count, err := serv.Read(res)
2617     if err != nil {
2618         }
2619         fmt.Printf("--En- S: (%3d,%v) %v", count, err, string(res))
2620     }
2621     if debug { fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res)) }
2622 }
2623 if argv[0] == "GET" {
2624     savPA := gsh.gshPA
2625     var bsize int = 64*1024
2626     req = fmt.Sprintf("%v\n", strings.Join(argv, " "))
2627     fmt.Printf(Elapsed(Start)+"--In- C: %v", req)
2628     fmt.Fprintf(serv, req)
2629     count, err = serv.Read(res)
2630     if err != nil {
2631         } else {
2632         var dsize int64 = 0
2633         var out *os.File = nil
2634         var out_tobeclosed *os.File = nil
2635         var fname string = ""
2636         var rcode int = 0
2637         var pid int = -1
2638         fmt.Sscanf(string(res), "%d %d", &rcode, &dsize)
2639         fmt.Printf(Elapsed(Start)+"--In- S: %v", string(res[0:count]))
2640         if 3 <= len(argv) {
2641             fname = argv[2]
2642             if strbegins(fname, "(") {
2643                 xin, xout, err := gsh.Popen(fname, "w")
2644                 if err {
2645                     } else {
2646                     xin.Close()
2647                     defer xout.Close()
2648                     out = xout
2649                     out_tobeclosed = xout
2650                     pid = 0 // should be its pid
2651                 }
2652             } else {
2653                 // should write to temporary file
2654                 // should suppress "C on tty

```

```

2655     xout,err := os.OpenFile(fname,os.O_CREATE|os.O_RDWR|os.O_TRUNC,0600)
2656     if err != nil {
2657         fmt.Printf("--En- %v\n",err)
2658     }
2659     out = xout
2660     //fmt.Printf("--In-- %d > %s\n",out.Fd(),fname)
2661 }
2662 }
2663 in, := serv.File()
2664 fileRelay("RecvGET",in,out,dsize,bsize)
2665 if 0 <= pid {
2666     gsh.gshPA = savPA // recovery of Fd(), and more?
2667     fmt.Printf(Elapsed(Start)+"--In- L: close Pipe > %v\n",fname)
2668     out.tobeclosed.Close()
2669     //syscall.Wait4(pid,nil,0,nil) //##
2670 }
2671 }
2672 }else
2673 if argv[0] == "PUT" {
2674     remote, := serv.File()
2675     var local *os.File = nil
2676     var dsize int64 = 32*1024*1024
2677     var bsize int = 64*1024
2678     var ofile string = ""
2679     //fmt.Printf("--In- Rex %v\n",argv)
2680     if 1 < len(argv) {
2681         fname := argv[1]
2682         if strBegins(fname,"-") {
2683             fmt.Sscanf(fname[2:],"%d",&dsize)
2684         }else
2685         if strBegins(fname,"{") {
2686             xin,xout,err := gsh.Popen(fname,"r")
2687             if err {
2688                 }else{
2689                 xout.Close()
2690                 defer xin.Close()
2691                 //in = xin
2692                 local = xin
2693                 fmt.Printf("--In- [%d] < Upload output of %v\n",
2694                     local.Fd(),fname)
2695                 ofile = "-from."+fname
2696                 dsize = MaxStreamSize
2697             }
2698         }else{
2699             xlocal,err := os.Open(fname)
2700             if err != nil {
2701                 fmt.Printf("--En- (%s)\n",err)
2702                 local = nil
2703             }else{
2704                 local = xlocal
2705                 fi, _ := local.Stat()
2706                 dsize = fi.Size()
2707                 defer local.Close()
2708                 //fmt.Printf("--In- Rex in(%v / %v)\n",ofile,dsize)
2709             }
2710             ofile = fname
2711             fmt.Printf(Elapsed(Start)+"--In- L: open(%v,r)=%v %v (%v)\n",
2712                 fname,dsize,local,err)
2713         }
2714     }
2715     if 2 < len(argv) && argv[2] != "" {
2716         ofile = argv[2]
2717         //fmt.Printf("(%d)%v B.ofile=%v\n",len(argv),argv,ofile)
2718     }
2719     //fmt.Printf(Elapsed(Start)+"--In- Rex out(%v)\n",ofile)
2720     fmt.Printf(Elapsed(Start)+"--In- PUT %v (/%v)\n",dsize,bsize)
2721     req = fmt.Sprintf("PUT %v %v \n",dsize,ofile)
2722     if debug {fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2723     fmt.Fprintf(serv,"%v",req)
2724     count,err = serv.Read(res)
2725     if debug {fmt.Printf(Elapsed(Start)+"--In- S: %v",string(res[0:count])) }
2726     fileRelay("SendPUT",local,remote,dsize,bsize)
2727 }else{
2728     req = fmt.Sprintf("%v\n",strings.Join(argv," "))
2729     if debug {fmt.Printf(Elapsed(Start)+"--In- C: %v",req) }
2730     fmt.Fprintf(serv,"%v",req)
2731     //fmt.Printf("--In- sending RexRequest(%v)\n",len(req))
2732 }
2733 //fmt.Printf(Elapsed(Start)+"--In- waiting RexResponse...\n")
2734 count,err = serv.Read(res)
2735 res = ""
2736 if count == 0 {
2737     res = "(nil)\r\n"
2738 }else{
2739     res = string(res[:count])
2740 }
2741 if err != nil {
2742     fmt.Printf(Elapsed(Start)+"--En- S: (%d,%v) %v",count,err,res)
2743 }else{
2744     fmt.Printf(Elapsed(Start)+"--In- S: %v",res)
2745 }
2746 serv.Close()
2747 //conn.Close()
2748 }
2749 var stat string
2750 var rcode int
2751 fmt.Sscanf(res,"%d %s",&rcode,&stat)
2752 //fmt.Printf("--D-- Client: %v (%v)",rcode,stat)
2753 return rcode,res
2754 }
2755 }
2756 // <a name="remote-sh">Remote Shell</a>
2757 // gcp file [ . | . ] [host]:[port:] [dir] | dir // -p | -no-p
2758 func (gsh+GshContext)FileCopy(argv[]string){
2759     var host = ""
2760     var port = ""
2761     var upload = false
2762     var download = false
2763     var xargv = []string{"rex-gcp"}
2764     var srcv = []string{}
2765     var dstv = []string{}
2766     argv = argv[1:]
2767 }
2768 for _,v := range argv {
2769     /*
2770     if v[0] == '-' { // might be a pseudo file (generated date)
2771         continue
2772     }
2773     */
2774     obj := strings.Split(v,":")
2775     //fmt.Printf("%d %v %v\n",len(obj),v,obj)
2776     if 1 < len(obj) {
2777         host = obj[0]
2778         file = ""
2779         if 0 < len(host) {
2780             gsh.LastServer.host = host
2781         }else{
2782             host = gsh.LastServer.host
2783             port = gsh.LastServer.port
2784         }
2785     }
2786     if 2 < len(obj) {
2787         port = obj[1]
2788         if 0 < len(port) {
2789             gsh.LastServer.port = port
2790         }else{
2791             port = gsh.LastServer.port
2792         }
2793     }
2794     file = obj[2]
2795     }else{
2796         file = obj[1]
2797     }
2798     if len(srcv) == 0 {
2799         download = true
2800         srcv = append(srcv,file)
2801         continue
2802     }
2803     upload = true
2804     dstv = append(dstv,file)
2805     continue
2806 }
2807 /*
2808 idx := strings.Index(v,":")
2809 if 0 <= idx {
2810     remote = v[0:idx]
2811     if len(srcv) == 0 {
2812         download = true
2813         srcv = append(srcv,v[idx+1:])
2814     }
2815     continue
2816 }
2817 upload = true
2818 dstv = append(dstv,v[idx+1:])
2819 continue
2820 }
2821 if download {
2822     dstv = append(dstv,v)
2823 }else{
2824     srcv = append(srcv,v)
2825 }
2826 }
2827 }
2828 hostport := "@" + host + ":" + port
2829 if upload {
2830     if host != "" { xargv = append(xargv,hostport) }
2831     xargv = append(xargv,"PUT")
2832     xargv = append(xargv,srcv[0:]...)
2833     xargv = append(xargv,dstv[0:]...)
2834 }
2835 //fmt.Printf("--In- FileCopy PUT gsh://%s/%v < %v // %v\n",hostport,dstv,srcv,xargv)

```



```

2832 fmt.Printf("--I- FileCopy PUT gsh://%s/%v < %v\n",hostport,dstv,srcv)
2833 gsh.RexecClient(xargv)
2834 }else
2835 if download {
2836 if host != "" { xargv = append(xargv,hostport) }
2837 xargv = append(xargv,"GET")
2838 xargv = append(xargv,srcv[0:]...)
2839 xargv = append(xargv,dstv[0:]...)
2840 //fmt.Printf("--I- FileCopy GET gsh://%v/%v > %v // %v\n",hostport,srcv,dstv,xargv)
2841 fmt.Printf("--I- FileCopy GET gsh://%v/%v > %v\n",hostport,srcv,dstv)
2842 gsh.RexecClient(xargv)
2843 }else{
2844 }
2845 }
2846
2847 // target
2848 func (gsh*GshContext)Trelpath(rloc string)(string){
2849 cwd, _ := os.Getwd()
2850 os.Chdir(gsh.RWD)
2851 os.Chdir(rloc)
2852 twd, _ := os.Getwd()
2853 os.Chdir(cwd)
2854
2855 tpath := twd + "/" + rloc
2856 return tpath
2857 }
2858 // join to rremote GShell - [user@]host[:port] or cd host[:port]:path
2859 func (gsh*GshContext)Rjoin(argv[]string){
2860 if len(argv) <= 1 {
2861 fmt.Printf("--I- current server = %v\n",gsh.RSERV)
2862 return
2863 }
2864 serv := argv[1]
2865 servv := strings.Split(serv,":")
2866 if 1 <= len(servv) {
2867 if servv[0] == "lo" {
2868 servv[0] = "localhost"
2869 }
2870 }
2871 switch len(servv) {
2872 case 1:
2873 //if strings.Index(serv,":") < 0 {
2874 serv = servv[0] + ":" + fmt.Sprintf("%d",GSH_PORT)
2875 //}
2876 case 2: // host:port
2877 serv = strings.Join(servv,":")
2878 }
2879 xargv := []string{"rex-join", "@"+serv,"HELLO"}
2880 rcode,stat := gsh.RexecClient(xargv)
2881 if (rcode / 100) == 2 {
2882 fmt.Printf("--I- OK Joined (%v) %v\n",rcode,stat)
2883 gsh.RSERV = serv
2884 }else{
2885 fmt.Printf("--I- NG, could not joined (%v) %v\n",rcode,stat)
2886 }
2887 }
2888 func (gsh*GshContext)Rexec(argv[]string){
2889 if len(argv) <= 1 {
2890 fmt.Printf("--I- rexec command [ | file | | command ]\n",gsh.RSERV)
2891 return
2892 }
2893
2894 /*
2895 nargv := gsh.ScanArg(strings.Join(argv," "),0)
2896 fmt.Printf("--D- nargc=%d (%v)\n",len(nargv),nargv)
2897 if nargv[1][0] != '{' {
2898 nargv[1] = "{" + nargv[1] + "}"
2899 fmt.Printf("--D- nargc=%d (%v)\n",len(nargv),nargv)
2900 }
2901 argv = nargv
2902 */
2903 nargv := []string{}
2904 nargv = append(nargv,"{"+strings.Join(argv[1:], " ")+"}")
2905 fmt.Printf("--D- nargc=%d %v\n",len(nargv),nargv)
2906 argv = nargv
2907
2908 xargv := []string{"rex-exec", "@"+gsh.RSERV,"GET"}
2909 xargv = append(xargv,argv...)
2910 xargv = append(xargv,"/dev/tty")
2911 rcode,stat := gsh.RexecClient(xargv)
2912 if (rcode / 100) == 2 {
2913 fmt.Printf("--I- OK Rexec (%v) %v\n",rcode,stat)
2914 }else{
2915 fmt.Printf("--I- NG Rexec (%v) %v\n",rcode,stat)
2916 }
2917 }
2918 func (gsh*GshContext)Rchdir(argv[]string){
2919 if len(argv) <= 1 {
2920 return
2921 }
2922 cwd, _ := os.Getwd()
2923 os.Chdir(gsh.RWD)
2924 os.Chdir(argv[1])
2925 twd, _ := os.Getwd()
2926 gsh.RWD = twd
2927 fmt.Printf("--I- JWD=%v\n",twd)
2928 os.Chdir(cwd)
2929 }
2930 func (gsh*GshContext)Rpwd(argv[]string){
2931 fmt.Printf("%v\n",gsh.RWD)
2932 }
2933 func (gsh*GshContext)Rls(argv[]string){
2934 cwd, _ := os.Getwd()
2935 os.Chdir(gsh.RWD)
2936 argv[0] = "-ls"
2937 gsh.xFind(argv)
2938 os.Chdir(cwd)
2939 }
2940 func (gsh*GshContext)Rput(argv[]string){
2941 var local string = ""
2942 var remote string = ""
2943 if 1 < len(argv) {
2944 local = argv[1]
2945 remote = local // base name
2946 }
2947 if 2 < len(argv) {
2948 remote = argv[2]
2949 }
2950 fmt.Printf("--I- jput from=%v to=%v\n",local,gsh.Trelpath(remote))
2951 }
2952 func (gsh*GshContext)Rget(argv[]string){
2953 var remote string = ""
2954 var local string = ""
2955 if 1 < len(argv) {
2956 remote = argv[1]
2957 local = remote // base name
2958 }
2959 if 2 < len(argv) {
2960 local = argv[2]
2961 }
2962 fmt.Printf("--I- jget from=%v to=%v\n",gsh.Trelpath(remote),local)
2963 }
2964
2965 // <a name="network">network</a>
2966 // -s, -si, -so // bi-directional, source, sync (maybe socket)
2967 func (gshCtx*GshContext)sconnect(inTCP bool, argv []string) {
2968 gshPA := gshCtx.gshPA
2969 if len(argv) < 2 {
2970 fmt.Printf("Usage: -s [host]:[port[:udp]]\n")
2971 return
2972 }
2973 remote := argv[1]
2974 if remote == "" { remote = "0.0.0.0:9999" }
2975
2976 if inTCP { // TCP
2977 dport, err := net.ResolveTCPAddr("tcp",remote);
2978 if err != nil {
2979 fmt.Printf("Address error: %s (%s)\n",remote,err)
2980 return
2981 }
2982 conn, err := net.DialTCP("tcp",nil,dport)
2983 if err != nil {
2984 fmt.Printf("Connection error: %s (%s)\n",remote,err)
2985 return
2986 }
2987 file, _ := conn.File();
2988 //fd := file.Fd();
2989 //fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,fd)
2990 fmt.Printf("Socket: connected to %s, socket[%d]\n",remote,file.Fd())
2991
2992 savfd := gshPA.Files[1]
2993 //gshPA.Files[1] = fd;
2994 gshPA.Files[1] = file;
2995 gshCtx.gshellv(argv[2:])
2996 gshPA.Files[1] = savfd
2997 file.Close()
2998 conn.Close()
2999 }else{
3000 //dport, err := net.ResolveUDPAddr("udp4",remote);
3001 dport, err := net.ResolveUDPAddr("udp",remote);
3002 if err != nil {
3003 fmt.Printf("Address error: %s (%s)\n",remote,err)
3004 return
3005 }
3006 //conn, err := net.DialUDP("udp4",nil,dport)
3007 conn, err := net.DialUDP("udp",nil,dport)
3008 if err != nil {

```

```

3009         fmt.Printf("Connection error: %s (%s)\n", remote, err)
3010         return
3011     }
3012     file, _ := conn.File();
3013     //fd := file.Fd()
3014
3015     ar := conn.RemoteAddr()
3016     //al := conn.LocalAddr()
3017     fmt.Printf("Socket: connected to %s [%s], socket[%d]\n",
3018 //remote, ar.String(), fd)
3019     remote, ar.String(), file.Fd())
3020
3021     savfd := gshPA.Files[1]
3022     //gshPA.Files[1] = fd;
3023     gshPA.Files[1] = file;
3024     gshCtx.gsheliv(argv[2:])
3025     gshPA.Files[1] = savfd
3026     file.Close()
3027     conn.Close()
3028 }
3029 }
3030 func (gshCtx*GshContext)saccept(intTCP bool, argv []string) {
3031     gshPA := gshCtx.gshPA
3032     if len(argv) < 2 {
3033         fmt.Printf("Usage: -ac [host]:[port.[udp]]\n")
3034         return
3035     }
3036     local := argv[1]
3037     if local == "" { local = "0.0.0.0:9999" }
3038     if intTCP { // TCP
3039         port, err := net.ResolveTCPAddr("tcp", local);
3040         if err != nil {
3041             fmt.Printf("Address error: %s (%s)\n", local, err)
3042             return
3043         }
3044         //fmt.Printf("Listen at %s...\n", local);
3045         sconn, err := net.ListenTCP("tcp", port)
3046         if err != nil {
3047             fmt.Printf("Listen error: %s (%s)\n", local, err)
3048             return
3049         }
3050         //fmt.Printf("Accepting at %s...\n", local);
3051         aconn, err := sconn.AcceptTCP()
3052         if err != nil {
3053             fmt.Printf("Accept error: %s (%s)\n", local, err)
3054             return
3055         }
3056         file, _ := aconn.File()
3057         fd := file.Fd()
3058         //fmt.Printf("Accepted TCP at %s [%d]\n", local, fd)
3059         fmt.Printf("Accepted TCP at %s [%d]\n", local, file.Fd())
3060
3061         savfd := gshPA.Files[0]
3062         //gshPA.Files[0] = fd;
3063         gshPA.Files[0] = file;
3064         gshCtx.gsheliv(argv[2:])
3065         gshPA.Files[0] = savfd
3066
3067         sconn.Close();
3068         aconn.Close();
3069         file.Close();
3070     } else {
3071         //port, err := net.ResolveUDPAddr("udp4", local);
3072         port, err := net.ResolveUDPAddr("udp", local);
3073         if err != nil {
3074             fmt.Printf("Address error: %s (%s)\n", local, err)
3075             return
3076         }
3077         fmt.Printf("Listen UDP at %s...\n", local);
3078         //uconn, err := net.ListenUDP("udp4", port)
3079         uconn, err := net.ListenUDP("udp", port)
3080         if err != nil {
3081             fmt.Printf("Listen error: %s (%s)\n", local, err)
3082             return
3083         }
3084         file, _ := uconn.File()
3085         //fd := file.Fd()
3086         ar := uconn.RemoteAddr()
3087         remote := ""
3088         if ar != nil { remote = ar.String() }
3089         if remote == "" { remote = "?" }
3090
3091         // not yet received
3092         //fmt.Printf("Accepted at %s [%d] <- %s\n", local, fd, "")
3093
3094         savfd := gshPA.Files[0]
3095         //gshPA.Files[0] = fd;
3096         gshPA.Files[0] = file;
3097         savenv := gshPA.Env
3098         gshPA.Env = append(savenv, "REMOTE_HOST="+remote)
3099         gshCtx.gsheliv(argv[2:])
3100         gshPA.Env = savenv
3101         gshPA.Files[0] = savfd
3102
3103         uconn.Close();
3104         file.Close();
3105     }
3106 }
3107 }
3108 // empty line command
3109 func (gshCtx*GshContext)xPwv(argv []string) {
3110     // execute context command, pwd + date
3111     // context notation, representation scheme, to be resumed at re-login
3112     cwd, _ := os.Getwd()
3113     switch {
3114     case isin("-a", argv):
3115         gshCtx.ShowChdirHistory(argv)
3116     case isin("-ls", argv):
3117         showFileInfo(cwd, argv)
3118     default:
3119         fmt.Printf("%s\n", cwd)
3120     case isin("-v", argv): // obsolete empty command
3121         t := time.Now()
3122         date := t.Format(time.UnixDate)
3123         exe, _ := os.Executable()
3124         host, _ := os.Hostname()
3125         fmt.Printf("PWD=\"%s\", cwd\n", cwd)
3126         fmt.Printf(" HOST=\"%s\", host\n", host)
3127         fmt.Printf(" DATE=\"%s\", date\n", date)
3128         fmt.Printf(" TIME=\"%s\", t.String()\n", t.String())
3129         fmt.Printf(" PID=\"%d\", os.Getpid()\n", os.Getpid())
3130         fmt.Printf(" EXE=\"%s\", exe\n", exe)
3131         fmt.Printf("\n")
3132     }
3133 }
3134 }
3135 // <a name="history">History</a>
3136 // these should be browsed and edited by HTTP browser
3137 // show the time of command with -t and direcotry with -ls
3138 // openfile-history, sort by -a -m -c
3139 // sort by elapsed time by -t -s
3140 // search by "more" like interface
3141 // edit history
3142 // sort history, and we or uniq
3143 // CPU and other resource consumptions
3144 // limit showing range (by time or so)
3145 // export / import history
3146 func (gshCtx *GshContext)xHistory(argv []string){
3147     atWorkDirX := -1
3148     if 1 < len(argv) && strBegins(argv[1], "0") {
3149         atWorkDirX, _ = strconv.Atoi(argv[1][1:])
3150     }
3151     //fmt.Printf("--D-- showHistory(%v)\n", argv)
3152     for i, v := range gshCtx.CommandHistory {
3153         // exclude commands not to be listed by default
3154         // internal commands may be suppressed by default
3155         if v.CmdLine == "" && !isin("-a", argv) {
3156             continue;
3157         }
3158         if 0 <= atWorkDirX {
3159             if v.WorkDirX != atWorkDirX {
3160                 continue
3161             }
3162         }
3163         if !isin("-n", argv) { // like "fc"
3164             fmt.Printf("%8-2d ", i)
3165         }
3166         if isin("-v", argv) {
3167             fmt.Println(v) // should be with it date
3168         } else {
3169             if isin("-l", argv) || isin("-10", argv) {
3170                 elps := v.EndAt.Sub(v.StartAt);
3171                 start := v.StartAt.Format(time.Stamp)
3172                 fmt.Printf("%8d ", v.WorkDirX)
3173                 fmt.Printf("[%v] %11v/t ", start, elps)
3174             }
3175             if isin("-l", argv) && !isin("-10", argv) {
3176                 fmt.Printf("%v", Rusagef("%t %u/t// %s", argv, v.Rusagev))
3177             }
3178             if isin("-at", argv) { // isin("-ls", argv) {
3179                 dhi := v.WorkDirX // workdir history index
3180                 fmt.Printf("%8d %s\t", dhi, v.WorkDir)
3181                 // show the FileInfo of the output command??
3182             }
3183             fmt.Printf("%s", v.CmdLine)
3184             fmt.Printf("\n")
3185         }
3186     }

```

```

3186 }
3187 }
3188 // in - history index
3189 func searchHistory(gshCtx GshContext, gline string) (string, bool, bool){
3190 if gline[0] == 'l' {
3191 hix, err := strconv.Atoi(gline[1:])
3192 if err != nil {
3193 fmt.Printf("--E-- (%s : range)\n",hix)
3194 return "", false, true
3195 }
3196 if hix < 0 || len(gshCtx.CommandHistory) <= hix {
3197 fmt.Printf("--E-- (%d : out of range)\n",hix)
3198 return "", false, true
3199 }
3200 return gshCtx.CommandHistory[hix].CmdLine, false, false
3201 }
3202 // search
3203 //for i, v := range gshCtx.CommandHistory {
3204 //}
3205 return gline, false, false
3206 }
3207 func (gsh*GshContext)cmdStringInHistory(hix int)(cmd string, ok bool){
3208 if 0 <= hix && hix < len(gsh.CommandHistory) {
3209 return gsh.CommandHistory[hix].CmdLine,true
3210 }
3211 return "",false
3212 }
3213 }
3214 // temporary adding to PATH environment
3215 // cd name -lib for LD_LIBRARY_PATH
3216 // chdir with directory history (date + full-path)
3217 // =s for sort option (by visit date or so)
3218 func (gsh*GshContext)ShowChdirHistory1(i int,v GChdirHistory, argv []string){
3219 fmt.Printf("%18-2d ",v.CmdIndex) // the first command at this WorkDir
3220 fmt.Printf("%18s ",v.CmdLine)
3221 fmt.Printf("[%v] ",v.MovedAt.Format(time.Stamp))
3222 showFileInfo(v.Dir,argv)
3223 }
3224 func (gsh*GshContext)ShowChdirHistory(argv []string){
3225 for i, v := range gsh.ChdirHistory {
3226 gsh.ShowChdirHistory1(i,v,argv)
3227 }
3228 }
3229 func skipOpts(argv[]string)(int){
3230 for i,v := range argv {
3231 if strBegins(v, "-") {
3232 }else{
3233 return i
3234 }
3235 }
3236 return -1
3237 }
3238 func (gshCtx*GshContext)xChdir(argv []string){
3239 cdhist := gshCtx.ChdirHistory
3240 if isin("?",argv) || isin("-",argv) || isin("-",argv) {
3241 gshCtx.ShowChdirHistory(argv)
3242 return
3243 }
3244 pwd, _ := os.Getwd()
3245 dir := ""
3246 if len(argv) <= 1 {
3247 dir = toFullpath("-")
3248 }else{
3249 i := skipOpts(argv[1:])
3250 if i < 0 {
3251 dir = toFullpath("-")
3252 }else{
3253 dir = argv[i+1]
3254 }
3255 }
3256 if strBegins(dir,"0") {
3257 if dir == "00" { // obsolete
3258 dir = gshCtx.StartDir
3259 }else
3260 if dir == "0:" {
3261 index := len(cdhist) - 1
3262 if 0 < index { index -= 1 }
3263 dir = cdhist[index].Dir
3264 }else{
3265 index, err := strconv.Atoi(dir[1:])
3266 if err != nil {
3267 fmt.Printf("--E-- xChdir(%v)\n",err)
3268 dir = "?"
3269 }else
3270 if len(gshCtx.ChdirHistory) <= index {
3271 fmt.Printf("--E-- xChdir(history range error)\n")
3272 dir = "?"
3273 }else{
3274 dir = cdhist[index].Dir
3275 }
3276 }
3277 }
3278 if dir != "?" {
3279 err := os.Chdir(dir)
3280 if err != nil {
3281 fmt.Printf("--E-- xChdir(%s)(%v)\n",argv[1],err)
3282 }else{
3283 cwd, _ := os.Getwd()
3284 if cwd != pwd {
3285 hist1 := GChdirHistory { }
3286 hist1.Dir = cwd
3287 hist1.MovedAt = time.Now()
3288 hist1.CmdIndex = len(gshCtx.CommandHistory)+1
3289 gshCtx.ChdirHistory = append(cdhist,hist1)
3290 if !isin("-",argv){
3291 //cwd, _ := os.Getwd()
3292 //fmt.Printf("%s\n",cwd)
3293 ix := len(gshCtx.ChdirHistory)-1
3294 gshCtx.ShowChdirHistory1(ix,hist1,argv)
3295 }
3296 }
3297 }
3298 }
3299 if isin("-ls",argv){
3300 cwd, _ := os.Getwd()
3301 showFileInfo(cwd,argv);
3302 }
3303 }
3304 func TimeValSub(tv1 *time.Duration, tv2 *time.Duration){
3305 //tv1 := syscall.NsecToTimeval(tv1.Nano()) - tv2.Nano()
3306 *tv1 -= *tv2;
3307 }
3308 func RusageSubv(ru1, ru2 [2]aRusage){[2]aRusage){
3309 TimeValSub(&ru1[0].Utime,&ru2[0].Utime)
3310 TimeValSub(&ru1[0].Stime,&ru2[0].Stime)
3311 TimeValSub(&ru1[1].Utime,&ru2[1].Utime)
3312 TimeValSub(&ru1[1].Stime,&ru2[1].Stime)
3313 return ru1
3314 }
3315 func TimeValAdd(tv1 time.Duration, tv2 time.Duration)(time.Duration){
3316 //tv1 := syscall.NsecToTimeval(tv1.Nano()) + tv2.Nano()
3317 tv1 += tv2;
3318 return tv1;
3319 }
3320 /*
3321 func RusageAddv(ru1, ru2 [2]aRusage){[2]aRusage){
3322 TimeValAdd(ru1[0].Utime,ru2[0].Utime)
3323 TimeValAdd(ru1[0].Stime,ru2[0].Stime)
3324 TimeValAdd(ru1[1].Utime,ru2[1].Utime)
3325 TimeValAdd(ru1[1].Stime,ru2[1].Stime)
3326 return ru1
3327 }
3328 */
3329 }
3330 // <a name="rusage">Resource Usage</a>
3331 func rUsageOf(fmtspec string, argv []string, ru [2]aRusage)(string){
3332 // ru[0] self , ru[1] children
3333 ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3334 st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3335 //uu := (int64(ut.Sec)*1000000 + int64(ut.Usec)) * 1000
3336 //su := (int64(st.Sec)*1000000 + int64(st.Usec)) * 1000
3337 uu := ut; // in nano sec
3338 su := st; // in nano sec
3339 tu := uu + su
3340 ret := fmt.Sprintf("%v/sum",abftime(tu))
3341 ret += fmt.Sprintf(", %v/ur",abftime(uu))
3342 ret += fmt.Sprintf(", %v/sy",abftime(su))
3343 return ret
3344 }
3345 func RusageOf(fmtspec string, argv []string, ru [2]aRusage)(string){
3346 ut := TimeValAdd(ru[0].Utime,ru[1].Utime)
3347 st := TimeValAdd(ru[0].Stime,ru[1].Stime)
3348 //fmt.Printf("%d.%06ds/u ",ut.Sec,ut.Usec) //ru[1].Utime.Sec,ru[1].Utime.Usec)
3349 //fmt.Printf("%d.%06ds/s ",st.Sec,st.Usec) //ru[1].Stime.Sec,ru[1].Stime.Usec)
3350 fmt.Printf("%d.%06ds/u ",ut/1000000000,(ut/1000)%1000000);
3351 fmt.Printf("%d.%06ds/s ",st/1000000000,(st/1000)%1000000);
3352 return ""
3353 }
3354 }
3355 func Getrusagev(){[2]aRusage){
3356 var ruv = [2]aRusage{}
3357 aGetrusage(aRUSAGE_SELF,&ruv[0])
3358 aGetrusage(aRUSAGE_CHILDREN,&ruv[1])
3359 return ruv
3360 }
3361 }
3362 func (gshCtx *GshContext)xTime(argv[]string)(bool){
3363 if 2 <= len(argv){

```

```

3363     gshCtx.LastRusage = aRusage{}
3364     rusagev1 := GetRusagev()
3365     fin := gshCtx.gshelly(argv[1:])
3366     rusagev2 := GetRusagev()
3367     showRusage(argv[1], argv, &gshCtx.LastRusage)
3368     rusagev := RusageSubv(rusagev2, rusagev1)
3369     showRusage("self", argv, &rusagev(0))
3370     showRusage("chld", argv, &rusagev(1))
3371     return fin
3372 }else{
3373     rusage:= aRusage {}
3374     aGetRusage(aRUSAGE_SELF, &rusage)
3375     showRusage("self", argv, &rusage)
3376     aGetRusage(aRUSAGE_CHILDREN, &rusage)
3377     showRusage("chld", argv, &rusage)
3378     return false
3379 }
3380 }
3381 func (gshCtx *GshContext)xJobs(argv []string){
3382     fmt.Printf("%d Jobs\n", len(gshCtx.BackGroundJobs))
3383     for ji, pid := range gshCtx.BackGroundJobs {
3384         //wstat := syscall.WaitStatus(0)
3385         rusage := aRusage {}
3386         //wpid, err := syscall.Wait4(pid, &wstat, syscall.WNOHANG, &rusage);
3387         //wpid, err := syscall.Wait4(pid, nil, syscall.WNOHANG, &rusage);
3388
3389         wpid := pid.Pid();
3390         err := errors.New("stab_NoError");
3391
3392         if err != nil {
3393             fmt.Printf("--E-- %%%d [%d] (%v)\n", ji, wpid, err)
3394         }else{
3395             fmt.Printf("%%d[%d]\n", ji, wpid)
3396             showRusage("chld", argv, &rusage)
3397         }
3398     }
3399 }
3400 func (gsh *GshContext)inBackground(argv []string)(bool){
3401     if gsh.CmdTrace {
3402         if gsh.CmdTrace {
3403             gsh.Background = true // set background option
3404             xfin := false
3405             xfin = gsh.gshelly(argv)
3406             gsh.Background = false
3407             return xfin
3408         }
3409         // -o file without command means just opening it and refer by #N
3410         // should be listed by "files" command
3411         func (gshCtx *GshContext)xOpen(argv []string){
3412             //var pv = [int(-1), -1]
3413             //err := syscall.Pipe(pv)
3414             //fmt.Printf("--I-- pipe()=[#d, #d](%v)\n", pv[0], pv[1], err)
3415             pin, pout, err := os.Pipe();
3416             fmt.Printf("--I-- pipe()=[#d, #d](%v)\n", pin.Fd(), pout.Fd(), err)
3417         }
3418         func (gshCtx *GshContext)fromPipe(argv []string){
3419             //
3420         }
3421         func (gshCtx *GshContext)xClose(argv []string){
3422             //
3423         }
3424         // <a name="redirect">redirect</a>
3425         func (gshCtx *GshContext)redirect(argv []string)(bool){
3426             if len(argv) < 2 {
3427                 return false
3428             }
3429             cmd := argv[0]
3430             fname := argv[1]
3431             var file *os.File = nil
3432             fdix := 0
3433             mode := os.O_RDONLY
3434
3435             switch {
3436             case cmd == "-i" || cmd == "<":
3437                 fdix = 0
3438                 mode = os.O_RDONLY
3439             case cmd == "-o" || cmd == ">":
3440                 fdix = 1
3441                 mode = os.O_RDWR | os.O_CREATE
3442             case cmd == "-a" || cmd == ">>":
3443                 fdix = 1
3444                 mode = os.O_RDWR | os.O_CREATE | os.O_APPEND
3445             }
3446             if fname[0] == '#' {
3447                 fd, err := strconv.Atoi(fname[1:])
3448                 if err != nil {
3449                     fmt.Printf("--E-- (%v)\n", err)
3450                     return false
3451                 }
3452                 file = os.NewFile(uintptr(fd), "MaybePipe")
3453             }else{
3454                 file, err := os.OpenFile(argv[1], mode, 0660)
3455                 if err != nil {
3456                     fmt.Printf("--E-- (%s)\n", err)
3457                     return false
3458                 }
3459                 file = xfile
3460             }
3461             gshPA := gshCtx.gshPA
3462             savfd := gshPA.Files[fdix]
3463             //gshPA.Files[fdix] = file.Fd()
3464             gshPA.Files[fdix] = file;
3465             fmt.Printf("--I-- Opened [%d] %s\n", file.Fd(), argv[1])
3466             gshCtx.gshelly(argv[2:])
3467             gshPA.Files[fdix] = savfd
3468             return false
3469         }
3470     }
3471     //fmt.Fprintf(res, "GShell Status: %q", html.EscapeString(req.URL.Path))
3472     func httpHandler(res http.ResponseWriter, req *http.Request){
3473         path := req.URL.Path
3474         fmt.Printf("--I-- Got HTTP Request(%s)\n", path)
3475         {
3476             gshCtxBuf, _ := setupGshContext()
3477             gshCtx := &gshCtxBuf
3478             fmt.Printf("--I-- %s\n", path[1:])
3479             gshCtx.tgshelly(path[1:])
3480         }
3481         fmt.Fprintf(res, "Hello(~/)\n\n%s\n", path)
3482     }
3483     func (gshCtx *GshContext) httpServer(argv []string){
3484         http.HandleFunc("/", httpHandler)
3485         accport := "localhost:9999"
3486         fmt.Printf("--I-- HTTP Server Start at [%s]\n", accport)
3487         http.ListenAndServe(accport, nil)
3488     }
3489     func (gshCtx *GshContext)xGo(argv []string){
3490         go gshCtx.gshelly(argv[1:]);
3491     }
3492     func (gshCtx *GshContext) xPs(argv []string){}
3493 }
3494 }
3495 // <a name="plugin">Plugin</a>
3496 // plugin [-s [names]] to list plugins
3497 // Reference: <a href="https://golang.org/src/plugin/">plugin</a> source code
3498 func (gshCtx *GshContext) whichPlugin(name string, argv []string)(pi *PluginInfo){
3499     pi = nil
3500     for , p := range gshCtx.PluginFncs {
3501         if p.Name == name && pi == nil {
3502             pi = &p
3503         }
3504         if !isin("-s", argv){
3505             //fmt.Printf("%v %v ", i, p)
3506             if !isin("-ls", argv){
3507                 showFileInfo(p.Path, argv)
3508             }else{
3509                 fmt.Printf("%s\n", p.Name)
3510             }
3511         }
3512     }
3513     return pi
3514 }
3515 func (gshCtx *GshContext) xPlugin(argv []string) (error) {
3516     if len(argv) == 0 || argv[0] == "-ls" {
3517         gshCtx.whichPlugin("", argv)
3518         return nil
3519     }
3520     name := argv[0]
3521     pin := gshCtx.whichPlugin(name, []string{"-s"})
3522     if pin != nil {
3523         os.Args = argv // should be recovered?
3524         pin.Addr.(func)()
3525         return nil
3526     }
3527     sofile := toFullpath(argv[0] + ".so") // or find it by which($PATH)
3528     p, err := plugin.Open(sofile)
3529     if err != nil {
3530         fmt.Printf("--E-- plugin.Open(%s)(%v)\n", sofile, err)
3531         return err
3532     }
3533     fname := "Main"
3534     f, err := p.Lookup(fname)
3535     if (err != nil) {
3536         fmt.Printf("--E-- plugin.Lookup(%s)(%v)\n", fname, err)
3537         return err
3538     }
3539 }

```

```

3540 }
3541 pin := PluginInfo {p,f,name,sofile}
3542 gshCtx.PluginFuncs = append(gshCtx.PluginFuncs,pin)
3543 fmt.Printf("--I-- added (%d)\n",len(gshCtx.PluginFuncs))
3544
3545 //fmt.Printf("--I-- first call(%s:%s)\v\n",sofile,fname,argv)
3546 os.Args = argv
3547 f.(func())()
3548 return err
3549 }
3550 func (gshCtx*GshContext)Args(argv[]string){
3551 for i,v := range os.Args {
3552     fmt.Printf("[%v] %v\n",i,v)
3553 }
3554 }
3555 func (gshCtx *GshContext) showVersion(argv[]string){
3556 if isin("-i",argv) {
3557     fmt.Printf("%v/%v (%v)",NAME,VERSION,DATE);
3558 }else{
3559     fmt.Printf("%v",VERSION);
3560 }
3561 if isin("-a",argv) {
3562     fmt.Printf(" %s",AUTHOR)
3563 }
3564 if !isin("-n",argv) {
3565     fmt.Printf("\n")
3566 }
3567 }
3568
3569 // <a name="scanf">Scanf</a> // string decomposer
3570 // scanf [format] [input]
3571 func scanf(str string)(strv[]string){
3572     strv = strings.Split(str, " ")
3573     return strv
3574 }
3575 func scanfUntil(src,end string)(rstr string, leng int){
3576     idx := strings.Index(src,end)
3577     if 0 <= idx {
3578         rstr = src[0:idx]
3579         return rstr,idx+leng(end)
3580     }
3581     return src,0
3582 }
3583
3584 // -bn -- display base-name part only // can be in some %fmt, for sed rewriting
3585 func (gsh*GshContext)printVal(fmts string, vstr string, optv[]string){
3586     //vint,err := strconv.Atoi(vstr)
3587     var ival int64 = 0
3588     n := 0
3589     err := error(nil)
3590     if strBegins(vstr, "-") {
3591         vx_ := strconv.Atoi(vstr[1:])
3592         if vx < len(gsh.iValues) {
3593             vstr = gsh.iValues[vx]
3594         }else{
3595             }
3596     }
3597     // should use Eval()
3598     if strBegins(vstr, "0x") {
3599         n,err = fmt.Sscanf(vstr[2:], "%x", &ival)
3600     }else{
3601         n,err = fmt.Sscanf(vstr, "%d", &ival)
3602     }
3603     //fmt.Printf("--D-- n=%d err=%v) (%s)=%v\n",n,err,vstr, ival)
3604 }
3605 if n == 1 && err == nil {
3606     //fmt.Printf("--D-- format(%v) ival(%v)\n",fmts,ival)
3607     fmt.Printf("%s"+fmts,ival)
3608 }else{
3609     if isin("-bn",optv){
3610         fmt.Printf("%s"+fmts,filepath.Base(vstr))
3611     }else{
3612         fmt.Printf("%s"+fmts,vstr)
3613     }
3614 }
3615 }
3616 func (gsh*GshContext)printfv(fmts,div string,argv[]string,optv[]string,list[]string){
3617     //fmt.Printf("%d",len(list))
3618     //curfmt := ""
3619     outlen := 0
3620     curfmt := gsh.iFormat
3621     if 0 < len(fmts) {
3622         for xi := 0; xi < len(fmts); xi++ {
3623             fch := fmts[xi]
3624             if fch == '%' {
3625                 if xi+1 < len(fmts) {
3626                     curfmt = string(fmts[xi+1])
3627                 }
3628                 gsh.iFormat = curfmt
3629                 xi += 1
3630             }
3631             if xi+1 < len(fmts) && fmts[xi+1] == '(' {
3632                 vals,leng := scanfUntil(fmts[xi+2:],")")
3633                 //fmt.Printf("--D-- show fmt(%v) val(%v) next(%v)\n",curfmt,vals,leng)
3634                 gsh.printVal(curfmt,vals,optv)
3635                 xi += 2+leng-1
3636                 outlen += 1
3637             }
3638             continue
3639         }
3640     }
3641     if fch == '.' {
3642         hi,leng := scanfUntil(fmts[xi+1:],")")
3643         if 0 < leng {
3644             if hi < len(gsh.iValues) {
3645                 gsh.printVal(curfmt,gsh.iValues[hi],optv)
3646                 outlen += 1 // should be the real length
3647             }else{
3648                 fmt.Printf("((out-range))")
3649             }
3650         }
3651         xi += leng
3652         continue;
3653     }
3654     }
3655     }
3656     }
3657     }
3658     }
3659     }
3660     }
3661     }
3662     }
3663     }
3664     }
3665     }
3666     }
3667     }
3668     }
3669     }
3670     }
3671     }
3672     }
3673     }
3674     }
3675     }
3676     }
3677     }
3678     }
3679     }
3680     }
3681     }
3682     }
3683     }
3684     }
3685     }
3686     }
3687     }
3688     }
3689     }
3690     }
3691     }
3692     }
3693     }
3694     }
3695     }
3696     }
3697     }
3698     }
3699     }
3700     }
3701     }
3702     }
3703     }
3704     }
3705     }
3706     }
3707     }
3708     }
3709     }
3710     }
3711     }
3712     }
3713     }
3714     }
3715     }
3716     }

```

```

3717     }
3718     }
3719     if 0 < len(argv) {
3720         fmts = strings.Join(argv, " ")
3721     }
3722     gsh.printf(fmts,div,argv,optv,gsh.iValues)
3723 }
3724 func (gsh*GshContext)Basename(argv[]string){
3725     for i,v := range gsh.iValues {
3726         gsh.iValues[i] = filepath.Base(v)
3727     }
3728 }
3729 func (gsh*GshContext)Sortv(argv[]string){
3730     sv := gsh.iValues
3731     sort.Slice(sv, func(i,j int) bool {
3732         return sv[i] < sv[j]
3733     })
3734 }
3735 func (gsh*GshContext)Shiftv(argv[]string){
3736     vl := len(gsh.iValues)
3737     if 0 < vl {
3738         if isin("-r",argv) {
3739             top := gsh.iValues[0]
3740             gsh.iValues = append(gsh.iValues[1:],top)
3741         }else{
3742             gsh.iValues = gsh.iValues[1:]
3743         }
3744     }
3745 }
3746 func (gsh*GshContext)Eng(argv[]string){
3747 }
3748 func (gsh*GshContext)Deq(argv[]string){
3749 }
3750 func (gsh*GshContext)Push(argv[]string){
3751     gsh.iValStack = append(gsh.iValStack,argv[1:])
3752     fmt.Printf("depth=%d\n",len(gsh.iValStack))
3753 }
3754 func (gsh*GshContext)Dump(argv[]string){
3755     for i,v := range gsh.iValStack {
3756         fmt.Printf("%d %v\n",i,v)
3757     }
3758 }
3759 }
3760 func (gsh*GshContext)Pop(argv[]string){
3761     depth := len(gsh.iValStack)
3762     if 0 < depth {
3763         v := gsh.iValStack[depth-1]
3764         if isin("-cat",argv){
3765             gsh.iValues = append(gsh.iValues,v...)
3766         }else{
3767             gsh.iValues = v
3768         }
3769         gsh.iValStack = gsh.iValStack[0:depth-1]
3770         fmt.Printf("depth=%d %s\n",len(gsh.iValStack),gsh.iValues)
3771     }else{
3772         fmt.Printf("depth=%d\n",depth)
3773     }
3774 }
3775 }
3776 // <a name="interpreter">Command Interpreter</a>
3777 func (gshCtx*GshContext)gshellv(argv []string) (fin bool) {
3778     fin = false
3779     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\n",len(argv)) }
3780     if len(argv) <= 0 {
3781         return false
3782     }
3783     xargv := []string{}
3784     for ai := 0; ai < len(argv); ai++ {
3785         xargv = append(xargv,subst(gshCtx,argv[ai],false))
3786     }
3787     argv = xargv
3788     if false {
3789         for ai := 0; ai < len(argv); ai++ {
3790             fmt.Printf("%d %s [%d]!\n",
3791                 ai,argv[ai],len(argv[ai]),argv[ai])
3792         }
3793     }
3794     cmd := argv[0]
3795     if gshCtx.CmdTrace { fmt.Fprintf(os.Stderr,"--I-- gshellv(%d)\v\n",len(argv),argv) }
3796     switch { // https://tour.golang.org/flowcontrol/11
3797     case cmd == "":
3798         gshCtx.xPwd([]string{}); // empty command
3799     case cmd == "-x":
3800         gshCtx.CmdTrace = 1 gshCtx.CmdTrace
3801     case cmd == "-xt":
3802         gshCtx.CmdTime = 1 gshCtx.CmdTime
3803     case cmd == "-ot":
3804         gshCtx.sconnect(true, argv)
3805     case cmd == "-ou":
3806         gshCtx.sconnect(false, argv)
3807     case cmd == "-it":
3808         gshCtx.saccept(true, argv)
3809     case cmd == "-iu":
3810         gshCtx.saccept(false, argv)
3811     case cmd == "-i" || cmd == "<" || cmd == ">" || cmd == "-" || cmd == ">>" || cmd == "-g" || cmd == "<":
3812         gshCtx.redirect(argv)
3813     case cmd == "|":
3814         gshCtx.fromPipe(argv)
3815     case cmd == "args":
3816         gshCtx.Args(argv)
3817     case cmd == "bg" || cmd == "-bg":
3818         rfin := gshCtx.inbackground(argv[1:])
3819         return rfin
3820     case cmd == "-bn":
3821         gshCtx.Basename(argv)
3822     case cmd == "cal":
3823         _ = gshCtx.excommand(false,argv[1:])
3824     case cmd == "cd" || cmd == "chdir":
3825         gshCtx.xChdir(argv);
3826     case cmd == "-cksum":
3827         gshCtx.xFind(argv)
3828     case cmd == "-sum":
3829         gshCtx.xFind(argv)
3830     case cmd == "-sumtest":
3831         str := ""
3832         if 1 < len(argv) { str = argv[1] }
3833         crc := strCRC32(str,uint64(len(str)))
3834         fprintf(stderr,"%v %v\n",crc,len(str))
3835     case cmd == "close":
3836         gshCtx.xClose(argv)
3837     case cmd == "gcp":
3838         gshCtx.FileCopy(argv)
3839     case cmd == "dec" || cmd == "decode":
3840         gshCtx.Dec(argv)
3841     case cmd == "#define":
3842         case cmd == "dic" || cmd == "d":
3843             xDic(argv)
3844         case cmd == "dump":
3845             gshCtx.Dump(argv)
3846         case cmd == "echo" || cmd == "e":
3847             echo(argv,true)
3848         case cmd == "enc" || cmd == "encode":
3849             gshCtx.Enc(argv)
3850         case cmd == "env":
3851             env(argv)
3852         case cmd == "eval":
3853             xEval(argv[1:],true)
3854         case cmd == "ev" || cmd == "events":
3855             dumpEvents(argv)
3856         case cmd == "exec":
3857             _ = gshCtx.excommand(true,argv[1:])
3858             // should not return here
3859         case cmd == "exit" || cmd == "quit":
3860             // write Result code EXIT to 3>
3861             return true
3862         case cmd == "fdls":
3863             // dump the attributes of fds (of other process)
3864         case cmd == "-find" || cmd == "fin" || cmd == "ufind" || cmd == "uf":
3865             gshCtx.xFind(argv[1:])
3866         case cmd == "fu":
3867             gshCtx.xFind(argv[1:])
3868         case cmd == "fork":
3869             // mainly for a server
3870         case cmd == "-gen":
3871             gshCtx.gen(argv)
3872         case cmd == "-go":
3873             gshCtx.xGo(argv)
3874         case cmd == "-grep":
3875             gshCtx.xFind(argv)
3876         case cmd == "gdeg":
3877             gshCtx.Deg(argv)
3878         case cmd == "genq":
3879             gshCtx.Enc(argv)
3880         case cmd == "gpop":
3881             gshCtx.Pop(argv)
3882         case cmd == "gpush":
3883             gshCtx.Push(argv)
3884         case cmd == "history" || cmd == "hi": // hi should be alias
3885             gshCtx.xHistory(argv)
3886         case cmd == "jobs":
3887             gshCtx.xJobs(argv)
3888         case cmd == "insp" || cmd == "nls":
3889             gshCtx.SplitLine(argv)
3890         case cmd == "ig":
3891             gshCtx.xFind(argv)
3892         case cmd == "nop":
3893     }

```

```

3894 // do nothing
3895 case cmd == "pipe":
3896     gshCtx.xOpen(argv)
3897 case cmd == "plug" || cmd == "plugin" || cmd == "pin":
3898     gshCtx.xPlugin(argv[1:])
3899 case cmd == "print" || cmd == "-pr":
3900     // output internal slice // also sprintf should be
3901     gshCtx.Printv(argv)
3902 case cmd == "ps":
3903     gshCtx.xPs(argv)
3904 case cmd == "pstable":
3905     // to be gsh.title
3906 case cmd == "rexecc" || cmd == "rexd":
3907     gshCtx.RexeccServer(argv)
3908 case cmd == "rexecc" || cmd == "rex":
3909     gshCtx.RexeccClient(argv)
3910 case cmd == "repeat" || cmd == "rep": // repeat cond command
3911     gshCtx.repeat(argv)
3912 case cmd == "replay":
3913     gshCtx.xReplay(argv)
3914 case cmd == "scan":
3915     // scan input (or so in fscanf) to internal slice (like Files or map)
3916     gshCtx.Scanv(argv)
3917 case cmd == "set":
3918     // set name ..
3919 case cmd == "serv":
3920     gshCtx.xHttpServer(argv)
3921 case cmd == "shift":
3922     gshCtx.Shiftv(argv)
3923 case cmd == "sleep":
3924     gshCtx.xSleep(argv)
3925 case cmd == "sort":
3926     gshCtx.Sortv(argv)
3927
3928 case cmd == "j" || cmd == "join":
3929     gshCtx.Rjoin(argv)
3930 case cmd == "a" || cmd == "alpa":
3931     gshCtx.Rexecc(argv)
3932 case cmd == "jed" || cmd == "jchdir":
3933     gshCtx.Rchdir(argv)
3934 case cmd == "jget":
3935     gshCtx.Rget(argv)
3936 case cmd == "jls":
3937     gshCtx.Rls(argv)
3938 case cmd == "jput":
3939     gshCtx.Rput(argv)
3940 case cmd == "jpwd":
3941     gshCtx.Rpwd(argv)
3942
3943 case cmd == "time":
3944     fin = gshCtx.xTime(argv)
3945 case cmd == "ungets":
3946     if 1 < len(argv) {
3947         ungets(argv[1]+"\\n")
3948     } else {
3949     }
3950 case cmd == "pwd":
3951     gshCtx.xPwd(argv)
3952 case cmd == "ver" || cmd == "-ver" || cmd == "version":
3953     gshCtx.showVersion(argv)
3954 case cmd == "where":
3955     // data file or so?
3956 case cmd == "which":
3957     which("PATH", argv)
3958 case cmd == "gj" && 1 < len(argv) && argv[1] == "listen":
3959     go gj_server(argv[1:])
3960 case cmd == "gj" && 1 < len(argv) && argv[1] == "serve":
3961     go gj_server(argv[1:])
3962 case cmd == "gj" && 1 < len(argv) && argv[1] == "join":
3963     go gj_client(argv[1:])
3964 case cmd == "gj":
3965     jsend(argv)
3966 case cmd == "jsend":
3967     jsend(argv)
3968 default:
3969     if gshCtx.whichPlugin(cmd, []string{"-s"}) != nil {
3970         gshCtx.xPlugin(argv)
3971     } else {
3972         notfound, := gshCtx.excommand(false, argv)
3973         if notfound {
3974             fmt.Printf("--E-- command not found (%v)\\n", cmd)
3975         }
3976     }
3977 }
3978 return fin
3979 }
3980
3981 func (gsh *GshContext) gshell(gline string) (rfin bool) {
3982     argv := strings.Split(string(gline), " ")
3983     fin := gsh.gshellv(argv)
3984     return fin
3985 }
3986 func (gsh *GshContext) tgshell(gline string) (xfin bool) {
3987     start := time.Now()
3988     fin := gsh.gshell(gline)
3989     end := time.Now()
3990     elps := end.Sub(start)
3991     if gsh.CmdTime {
3992         fmt.Printf("--T-- " + time.Now().Format(time.Stamp) + " (%.9ds)\\n",
3993             elps*100000000, elps*100000000)
3994     }
3995     return fin
3996 }
3997 func Ttyid() (int) {
3998     fi, err := os.Stdin.Stat()
3999     if err != nil {
4000         return 0;
4001     }
4002     //fmt.Printf("Stdin: %v Dev=%d\\n",
4003     //fi.Mode(), fi.Mode() & os.ModeDevice)
4004     if (fi.Mode() & os.ModeDevice) != 0 {
4005         stat := aStat_t{}
4006         err := aStat_t(&stat)
4007         if err != nil {
4008             //fmt.Printf("--I-- Stdin: (%v)\\n", err)
4009         } else {
4010             //fmt.Printf("--I-- Stdin: rdev=%d %d\\n",
4011             // stat.Rdev & 0xFF, stat.Rdev);
4012             //fmt.Printf("--I-- Stdin: tty=%d\\n", stat.Rdev & 0xFF);
4013             return int(stat.Rdev & 0xFF);
4014         }
4015     }
4016     return 0
4017 }
4018 func (gshCtx *GshContext) ttyfile() string {
4019     //fmt.Printf("--I-- GSH_HOME=%s\\n", gshCtx.GshHomeDir)
4020     ttyfile := gshCtx.GshHomeDir + "/" + "gsh-tty" +
4021         fmt.Sprintf("%02d", gshCtx.TerminalId)
4022     //strconv.Itoa(gshCtx.TerminalId)
4023     //fmt.Printf("--I-- ttyfile=%s\\n", ttyfile)
4024     return ttyfile
4025 }
4026 func (gshCtx *GshContext) ttyline() (*os.File) {
4027     file, err := os.OpenFile(gshCtx.ttyfile(), os.O_RDWR|os.O_CREATE|os.O_TRUNC, 0600)
4028     if err != nil {
4029         fmt.Printf("--F-- cannot open %s (%s)\\n", gshCtx.ttyfile(), err)
4030         return file;
4031     }
4032     return file
4033 }
4034 func (gshCtx *GshContext) getline(hix int, skipping bool, prevline string) (string) {
4035     if skipping {
4036         reader := bufio.NewReaderSize(os.Stdin, LINESIZE)
4037         line, _, _ := reader.ReadLine()
4038         return string(line)
4039     } else {
4040         if true {
4041             return xgetline(hix, prevline, gshCtx)
4042         }
4043         /*
4044         else
4045         if( with_exgetline && gshCtx.GetLine != "" ){
4046             //var xhix int64 = int64(hix); // cast
4047             newenv := os.Environ()
4048             newenv = append(newenv, "GSH_LINENO="+strconv.FormatInt(int64(hix), 10) )
4049             tty := gshCtx.ttyline()
4050             tty.WriteString(prevline)
4051             Pa := os.ProcAttr{
4052                 " ", // start dir
4053                 newenv, //os.Environ(),
4054                 []os.File{os.Stdin, os.Stdout, os.Stderr, tty},
4055                 nil,
4056             }
4057             //fmt.Printf("--I-- getline=%s // %s\\n", gsh_getline[0], gshCtx.GetLine)
4058             proc, err := os.StartProcess(gsh_getline[0], []string{"getline", "gPa")
4059             if err != nil {
4060                 fmt.Printf("--F-- getline process error (%v)\\n", err)
4061                 // for ; ; {
4062                 return "exit (getline program failed)"
4063             }
4064             //stat, err := proc.Wait()
4065             proc.Wait()
4066             buff := make([]byte, LINESIZE)
4067             count, err := tty.Read(buff)
4068             //_, err = tty.Read(buff)
4069             //fmt.Printf("--D-- getline (%d)\\n", count)

```

```

4071     if err != nil {
4072         if i (count == 0) { // && err.String() == "EOF" } {
4073             fmt.Printf("--E-- getline error (%s)\n",err)
4074         }
4075     }else{
4076         //fmt.Printf("--I-- getline OK \"%s\"\n",buff)
4077     }
4078     tty.Close()
4079     gline := string(buff[0:count])
4080     return gline
4081 }else
4082 /*
4083 {
4084     // if isatty {
4085         fmt.Printf("%d",hix)
4086         fmt.Print(PROMPT)
4087     }
4088     reader := bufio.NewReaderSize(os.Stdin,LINESIZE)
4089     line, _, _ := reader.ReadLine()
4090     return string(line)
4091 }
4092 }
4093
4094 //== begin ===== getline
4095 /*
4096 * getline.c
4097 * 2020-0819 extracted from dog.c
4098 * getline.go
4099 * 2020-0822 ported to Go
4100 */
4101 /*
4102 package main // getline main
4103 import (
4104     "fmt" // <a href="https://golang.org/pkg/fmt/">fmt</a>
4105     "strings" // <a href="https://golang.org/pkg/strings/">strings</a>
4106     "os" // <a href="https://golang.org/pkg/os/">os</a>
4107     "syscall" // <a href="https://golang.org/pkg/syscall/">syscall</a>
4108     "bytes" // <a href="https://golang.org/pkg/bytes/">bytes</a>
4109     "os/exec" // <a href="https://golang.org/pkg/os/exec/">os/exec</a>
4110 )
4111 */
4112 // C language compatibility functions
4113 var errno = 0
4114 var stdin *os.File = os.Stdin
4115 var stdout *os.File = os.Stdout
4116 var stderr *os.File = os.Stderr
4117 var EOF = -1
4118 var NULL = 0
4119 type FILE os.File
4120 type StrBuff []byte
4121 var NULL_FP *os.File = nil
4122 var NULL_SP = 0
4123 //var LINESIZE = 1024
4124
4125 func system(cmdstr string)(int){
4126     //PA := syscall.ProcAttr {
4127     PA := os.ProcAttr {
4128         "", // the starting directory
4129         os.Environ(),
4130         //[]uintptr(os.Stdin.Fd(),os.Stdout.Fd(),os.Stderr.Fd()),
4131         []os.File{os.Stdin,os.Stdout,os.Stderr},
4132         nil,
4133     }
4134     argv := strings.Split(cmdstr, " ")
4135     //pid,err := syscall.ForkExec(argv[0],argv,&PA)
4136     proc_err := os.StartProcess(argv[0],argv,&PA);
4137     if (err != nil){
4138         //fmt.Printf("--Es-- system(%v)\n(%v)\n",cmdstr,err);
4139         return -1;
4140     }
4141     pstat, _ := proc.Wait();
4142     pid := pstat.Pid();
4143     if (err != nil){
4144         fmt.Printf("--E-- pid=%v syscall(%v) err(%v)\n",pid,cmdstr,err)
4145     }
4146     //syscall.Wait4(pid,nil,0,nil)
4147     //fmt.Printf("===E=== pid=%d exit=%v stat=%v\n",pid,pstat.Exited(),pstat.ExitCode());
4148 }
4149
4150 /*
4151 argv := strings.Split(cmdstr, " ")
4152 fmt.Printf(os.Stderr, "--I-- system(%v)\n", argv)
4153 //cmd := exec.Command(argv[0],...)
4154 cmd := exec.Command(argv[0],argv[1],argv[2])
4155 cmd.Stdin = strings.NewReader("output of system")
4156 var out bytes.Buffer
4157 cmd.Stdout = &out
4158 var serr bytes.Buffer
4159 cmd.Stderr = &serr
4160 err := cmd.Run()
4161 if err != nil {
4162     fmt.Fprintf(os.Stderr, "--E-- system(%v)err(%v)\n",argv,err)
4163     fmt.Printf("ERR:%s\n",serr.String())
4164 }else{
4165     fmt.Printf("%s",out.String())
4166 }
4167 */
4168 return 0
4169 }
4170 func atoi(str string)(ret int){
4171     ret,err := fmt.Sscanf(str,"%d",&ret)
4172     if err == nil {
4173         return ret
4174     }else{
4175         // should set errno
4176         return 0
4177     }
4178 }
4179 func getenv(name string)(string){
4180     val, got := os.LookupEnv(name)
4181     if got {
4182         return val
4183     }else{
4184         return "?"
4185     }
4186 }
4187 func strcpy(dst StrBuff, src string){
4188     var i int
4189     srcb := []byte(src)
4190     for i = 0; i < len(src) && srcb[i] != 0; i++ {
4191         dst[i] = srcb[i]
4192     }
4193     dst[i] = 0
4194 }
4195 func xstrcpy(dst StrBuff, src StrBuff){
4196     dst = src
4197 }
4198 func strcat(dst StrBuff, src StrBuff){
4199     dst = append(dst,src...)
4200 }
4201 func strdup(str StrBuff)(string){
4202     return string(str[0:strlen(str)])
4203 }
4204 func strlen(str string)(int){
4205     return len(str)
4206 }
4207 func strlen(str StrBuff)(int){
4208     var i int
4209     for i = 0; i < len(str) && str[i] != 0; i++ {
4210     }
4211     return i
4212 }
4213 func sizeof(data StrBuff)(int){
4214     return len(data)
4215 }
4216 func isatty(fd int)(ret int){
4217     return 1
4218 }
4219
4220 func fopen(file string,mode string)(fp*os.File){
4221     if mode == "r" {
4222         fp,err := os.Open(file)
4223         if (err != nil){
4224             fmt.Printf("--E-- fopen(%s,%s)=(%v)\n",file,mode,err)
4225             return NULL_FP;
4226         }
4227         return fp;
4228     }else{
4229         fp,err := os.OpenFile(file,os.O_RDWR|os.O_CREATE|os.O_TRUNC,0600)
4230         if (err != nil){
4231             return NULL_FP;
4232         }
4233         return fp;
4234     }
4235 }
4236 func fclose(fp*os.File){
4237     fp.Close()
4238 }
4239 func fflush(fp *os.File)(int){
4240     return 0
4241 }
4242 func fgetc(fp*os.File)(int){
4243     var buf [1]byte
4244     _,err := fp.Read(buf[0:1])
4245     if (err != nil){
4246         return EOF;
4247     }else{

```



```

4248     return int(buf[0])
4249 }
4250 }
4251 func sfgets(str*string, size int, fp*os.File)(int){
4252     buf := make(StrBuff,size)
4253     var ch int
4254     var i int
4255     for i = 0; i < len(buf)-1; i++ {
4256         ch = fgetc(fp)
4257         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
4258         if (ch == EOF ){
4259             break;
4260         }
4261         buf[i] = byte(ch);
4262         if( ch == '\n' ){
4263             break;
4264         }
4265     }
4266     buf[i] = 0
4267     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4268     return i
4269 }
4270 func fgetc(buf StrBuff, size int, fp*os.File)(int){
4271     var ch int
4272     var i int
4273     for i = 0; i < len(buf)-1; i++ {
4274         ch = fgetc(fp)
4275         //fprintf(stderr,"--fgets %d/%d %X\n",i,len(buf),ch)
4276         if( ch == EOF ){
4277             break;
4278         }
4279         buf[i] = byte(ch);
4280         if( ch == '\n' ){
4281             break;
4282         }
4283     }
4284     buf[i] = 0
4285     //fprintf(stderr,"--fgets %d/%d (%s)\n",i,len(buf),buf[0:i])
4286     return i
4287 }
4288 func fputc(ch int , fp*os.File)(int){
4289     var buf []byte
4290     buf[0] = byte(ch)
4291     fp.Write(buf[0:1])
4292     return 0
4293 }
4294 func fputs(buf StrBuff, fp*os.File)(int){
4295     fp.Write(buf)
4296     return 0
4297 }
4298 func xfputs(str string, fp*os.File)(int){
4299     return fputs([]byte(str),fp)
4300 }
4301 func sscanf(str StrBuff,fmts string, params ...interface{})(int){
4302     fmt.Sscanf(string(str[0:strlen(str)]),fmts,params...)
4303     return 0
4304 }
4305 func fprintf(fp*os.File,fmts string, params ...interface{})(int){
4306     fmt.Fprintf(fp,fmts,params...)
4307     return 0
4308 }
4309 }
4310 // <a name="IME">Command Line IME</a>
4311 //----- MyIME
4312 var MyIMEVER = "MyIME/0.0.2";
4313 type RomKana struct {
4314     dic string // dictionary ID
4315     pat string // input pattern
4316     out string // output pattern
4317     hit int64 // count of hit and used
4318 }
4319 var dicents = 0
4320 var romkana [1024]RomKana
4321 var Romkan []RomKana
4322 }
4323 func isinDic(str string)(int){
4324     for i,v := range Romkan {
4325         if v.pat == str {
4326             return i
4327         }
4328     }
4329     return -1
4330 }
4331 const (
4332     DIC_COM_LOAD = "im"
4333     DIC_COM_DUMP = "s"
4334     DIC_COM_LIST = "ls"
4335     DIC_COM_ENA = "en"
4336     DIC_COM_DIS = "di"
4337 )
4338 func helpDic(argv []string){
4339     out := stderr
4340     cmd :=
4341     if 0 < len(argv) { cmd = argv[0] }
4342     fprintf(out,"-- %v Usage\n",cmd)
4343     fprintf(out,"  Commands\n")
4344     fprintf(out,"... %v %3v [dicName] [dicURL] -- Import dictionary\n",cmd,DIC_COM_LOAD)
4345     fprintf(out,"... %v %3v [pattern] -- Search in dictionary\n",cmd,DIC_COM_DUMP)
4346     fprintf(out,"... %v %3v [dicName] -- List dictionaries\n",cmd,DIC_COM_LIST)
4347     fprintf(out,"... %v %3v [dicName] -- Disable dictionaries\n",cmd,DIC_COM_DIS)
4348     fprintf(out,"... %v %3v [dicName] -- Enable dictionaries\n",cmd,DIC_COM_ENA)
4349     fprintf(out,"... Keys ... %v\n", "ESC can be used for '\n'")
4350     fprintf(out,"... \xc -- Reverse the case of the last character\n")
4351     fprintf(out,"... \\\i -- Replace input with translated text\n")
4352     fprintf(out,"... \\\j -- On/Off translation mode\n")
4353     fprintf(out,"... \\\l -- Force Lower Case\n")
4354     fprintf(out,"... \\\u -- Force Upper Case (software CapsLock)\n")
4355     fprintf(out,"... \\\v -- Show translation actions\n")
4356     fprintf(out,"... \\\x -- Replace the last input character with it Hexa-Decimal\n")
4357 }
4358 func xDic(argv[]string){
4359     if len(argv) <= 1 {
4360         helpDic(argv)
4361         return
4362     }
4363     argv = argv[1:]
4364     var debug = false
4365     var info = false
4366     var silent = false
4367     var dump = false
4368     var builtin = false
4369     cmd := argv[0]
4370     argv = argv[1:]
4371     opt := ""
4372     arg := ""
4373 }
4374 if 0 < len(argv) {
4375     arg1 := argv[0]
4376     if arg1[0] == '-' {
4377         switch arg1 {
4378             default: fmt.Printf("--Ed-- Unknown option(%v)\n",arg1)
4379             return
4380             case "-b": builtin = true
4381             case "-d": debug = true
4382             case "-s": silent = true
4383             case "-v": info = true
4384         }
4385         opt = arg1
4386         argv = argv[1:]
4387     }
4388 }
4389 }
4390 dicName := ""
4391 dicURL := ""
4392 if 0 < len(argv) {
4393     arg = argv[0]
4394     dicName = arg
4395     argv = argv[1:]
4396 }
4397 if 0 < len(argv) {
4398     dicURL = argv[0]
4399     argv = argv[1:]
4400 }
4401 if false {
4402     fprintf(stderr,"--Dd-- com(%v) opt(%v) arg(%v)\n",cmd,opt,arg)
4403 }
4404 if cmd == DIC_COM_LOAD {
4405     //dicType := ""
4406     dicBody := ""
4407     if !builtin && dicName != "" && dicURL == "" {
4408         f,err := os.Open(dicName)
4409         if err == nil {
4410             dicURL = dicName
4411         }else{
4412             f,err = os.Open(dicName+".html")
4413             if err == nil {
4414                 dicURL = dicName+".html"
4415             }else{
4416                 f,err = os.Open("gshdic-"+dicName+".html")
4417                 if err == nil {
4418                     dicURL = "gshdic-"+dicName+".html"
4419                 }
4420             }
4421         }
4422     }
4423     if err == nil {
4424         var buf = make([]byte,128*1024)
4425         count,err := f.Read(buf)
4426     }

```

```

4425     f.Close()
4426     if info {
4427         fprintf(stderr, "--Id-- ReadDic(%v,%v)\n", count, err)
4428     }
4429     dicBody = string(buf[0:count])
4430 }
4431 }
4432 if dicBody == "" {
4433     switch arg {
4434     default:
4435         dicName = "WorldDic"
4436         dicURL = WorldDic
4437         if info {
4438             fprintf(stderr, "--Id-- default dictionary \"%v\"\n",
4439                 dicName);
4440         }
4441     case "wnn":
4442         dicName = "WnnDic"
4443         dicURL = WnnDic
4444     case "sumomo":
4445         dicName = "SumomoDic"
4446         dicURL = SumomoDic
4447     case "sijimi":
4448         dicName = "SijimiDic"
4449         dicURL = SijimiDic
4450     case "jkl":
4451         dicName = "JKLJaDic"
4452         dicURL = JA_JKLJaDic
4453     }
4454     if debug {
4455         fprintf(stderr, "--Id-- %v URL=%v\n\n", dicName, dicURL);
4456     }
4457     dicv := strings.Split(dicURL, ",")
4458     if debug {
4459         fprintf(stderr, "--Id-- %v encoded data...\n", dicName)
4460         fprintf(stderr, "Type: %v\n", dicv[0])
4461         fprintf(stderr, "Body: %v\n", dicv[1])
4462         fprintf(stderr, "\n")
4463     }
4464     body, _ := base64.StdEncoding.DecodeString(dicv[1])
4465     dicBody = string(body)
4466 }
4467 if info {
4468     fmt.Printf("--Id-- %v %v\n", dicName, dicURL)
4469     fmt.Printf("%s\n", dicBody)
4470 }
4471 if debug {
4472     fprintf(stderr, "--Id-- dicName %v text...\n", dicName)
4473     fprintf(stderr, "%v\n", string(dicBody))
4474 }
4475 envv := strings.Split(dicBody, "\n");
4476 if info {
4477     fprintf(stderr, "--Id-- %v scan...\n", dicName);
4478 }
4479 var added int = 0
4480 var dup int = 0
4481 for _, v := range envv {
4482     var pat string
4483     var out string
4484     fmt.Sscanf(v, "%s %s", &pat, &out)
4485     if len(pat) <= 0 {
4486     } else {
4487         if 0 <= isinDic(pat) {
4488             dup += 1
4489             continue
4490         }
4491         romkana[dicents] = RomKana{dicName, pat, out, 0}
4492         dicents += 1
4493         added += 1
4494         Romkan = append(Romkan, RomKana{dicName, pat, out, 0})
4495         if debug {
4496             fmt.Printf("[%3v]:[%2v]%-8v [%2v] %v\n",
4497                 i, len(pat), pat, len(out), out)
4498         }
4499     }
4500 }
4501 if !silent {
4502     url := dicURL
4503     if strBegins(url, "data:") {
4504         url = "builtin"
4505     }
4506     fprintf(stderr, "--Id-- %v scan... %v added, %v dup. / %v total (%v)\n",
4507         dicName, added, dup, len(Romkan), url);
4508 }
4509 // should sort by pattern length for complete match, for performance
4510 if debug {
4511     arg = "" // search pattern
4512     dump = true
4513 }
4514 if cmd == DIC_COM_DUMP || dump {
4515     fprintf(stderr, "--Id-- %v dump... %v entries:\n", dicName, len(Romkan));
4516     var match = 0
4517     for i := 0; i < len(Romkan); i++ {
4518         dic := Romkan[i].dic
4519         pat := Romkan[i].pat
4520         out := Romkan[i].out
4521         if arg == "" || 0 <= strings.Index(pat, arg) || 0 <= strings.Index(out, arg) {
4522             fmt.Printf("\n\\%v\\t%v [%2v]%-8v [%2v] %v\n",
4523                 i, dic, len(pat), pat, len(out), out)
4524             match += 1
4525         }
4526     }
4527     fprintf(stderr, "--Id-- %v matched %v / %v entries:\n", arg, match, len(Romkan));
4528 }
4529 }
4530 func loadDefaultDic(dic int) {
4531     if (0 < len(Romkan)) {
4532         return
4533     }
4534     //fprintf(stderr, "\r\n")
4535     xdic([]string{"dic", DIC_COM_LOAD});
4536 }
4537 var info = false
4538 if info {
4539     fprintf(stderr, "--Id-- Conguraturations!! WorldDic is now activated.\r\n")
4540     fprintf(stderr, "--Id-- enter \"dic\" command for help.\r\n")
4541 }
4542 }
4543 }
4544 func readDic()(int) {
4545     /*
4546     var rk *os.File;
4547     var dic = "MyIME-dic.txt";
4548     //rk = fopen("romkana.txt", "r");
4549     //rk = fopen("JK-JA-morse-dic.txt", "r");
4550     rk = fopen(dic, "r");
4551     if (rk == NULL_FP) {
4552         if true {
4553             fprintf(stderr, "--s-- Could not load %s\n", MyIMEVER, dic);
4554         }
4555         return -1;
4556     }
4557     if true {
4558         var di int;
4559         var line = make(StrBuff, 1024);
4560         var pat string
4561         var out string
4562         for di = 0; di < 1024; di++ {
4563             if !fgets(line, sizeof(line), rk) == NULLSP {
4564                 break;
4565             }
4566             fmt.Sscanf(string(line[0:strlen(line)]), "%s %s", &pat, &out);
4567             //sscanf(line, "%s %s", &pat, &out);
4568             romkana[di].pat = pat;
4569             romkana[di].out = out;
4570             //fprintf(stderr, "--bd- %s-%s %s\n", pat, out)
4571         }
4572         dicents += di
4573         if false {
4574             fprintf(stderr, "--s-- loaded romkana.txt [%d]\n", MyIMEVER, di);
4575             for di = 0; di < dicents; di++ {
4576                 fprintf(stderr,
4577                     "%s %v\n", romkana[di].pat, romkana[di].out);
4578             }
4579         }
4580     }
4581     fclose(rk);
4582 }
4583 //romkana[dicents].pat = "//ddump"
4584 //romkana[dicents].pat = "//ddump" // dump the dic. and clean the command input
4585 }
4586 return 0;
4587 }
4588 func matchlen(stri string, pati string)(int) {
4589     if strBegins(stri, pati) {
4590         return len(pati)
4591     } else {
4592         return 0
4593     }
4594 }
4595 func convs(src string)(string) {
4596     var si int;
4597     var sx = len(src);
4598     var di int;
4599     var mi int;
4600     var dstb []byte
4601 }

```

```

4602 for si = 0; si < sx; { // search max. match from the position
4603   if strBegins(src[si:], "%x/") {
4604     // \x/integer/ // s/a/b/
4605     ix := strings.Index(src[si+3:], "/")
4606     if 0 < ix {
4607       var iv int = 0
4608       //fmt.Sscanf(src[si+3:si+3+ix], "%d", &iv)
4609       fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4610       sval := fmt.Sprintf("%x", iv)
4611       bval := []byte(sval)
4612       dstb = append(dstb, bval...)
4613       si = si+3+ix+1
4614       continue
4615     }
4616   }
4617   if strBegins(src[si:], "%d/") {
4618     // \d/integer/ // s/a/b/
4619     ix := strings.Index(src[si+3:], "/")
4620     if 0 < ix {
4621       var iv int = 0
4622       fmt.Sscanf(src[si+3:si+3+ix], "%v", &iv)
4623       sval := fmt.Sprintf("%d", iv)
4624       bval := []byte(sval)
4625       dstb = append(dstb, bval...)
4626       si = si+3+ix+1
4627       continue
4628     }
4629   }
4630   if strBegins(src[si:], "%t") {
4631     now := time.Now()
4632     if true {
4633       date := now.Format(time.Stamp)
4634       dstb = append(dstb, []byte(date)...)
4635       si = si+3
4636     }
4637     continue
4638   }
4639   var maxlen int = 0;
4640   var len int;
4641   mi = -1;
4642   for di = 0; di < dicents; di++ {
4643     len = matchlen(src[si:], romkana[di].pat);
4644     if (maxlen < len) {
4645       maxlen = len;
4646       mi = di;
4647     }
4648   }
4649   if (0 < maxlen) {
4650     out := romkana[mi].out;
4651     dstb = append(dstb, []byte(out)...);
4652     si += maxlen;
4653   } else {
4654     dstb = append(dstb, src[si])
4655     si += 1;
4656   }
4657 }
4658 return string(dstb)
4659 }
4660 func trans(src string) int {
4661   dst := convs(src); // s/a/b/
4662   xfprintf(dst, stderr);
4663   return 0;
4664 }
4665 }
4666 //----- LINEEDIT
4667 // "?" at the top of the line means searching history
4668
4669 // should be compatible with Telnet
4670 const (
4671   EV_MODE      = 255
4672   EV_IDLE      = 254
4673   EV_TIMEOUT   = 253
4674
4675   GO_UP        = 252 // k
4676   GO_DOWN      = 251 // j
4677   GO_RIGHT     = 250 // l
4678   GO_LEFT      = 249 // h
4679   DEL_RIGHT    = 248 // x
4680   GO_TOP      = 'A'-0x40 // 0
4681   GO_ENDL     = 'E'-0x40 // $
4682
4683   GO_TOPW     = 239 // b
4684   GO_ENDW     = 238 // e
4685   GO_NEXTW    = 237 // w
4686
4687   GO_FORWCH   = 229 // f
4688   GO_PAIRCH   = 228 // %
4689
4690   GO_DEL      = 219 // d
4691
4692   HI_SRCH_FW  = 209 // /
4693   HI_SRCH_BK  = 208 // ?
4694   HI_SRCH_RFW = 207 // n
4695   HI_SRCH_RBK = 206 // N
4696 )
4697
4698 // should return number of octets ready to be read immediately
4699 //fprintf(stderr, "\n--Select(%v %v)\n", err, r.Bits[0])
4700
4701
4702 var EventRecvFd = -1 // file descriptor
4703 var EventSendFd = -1
4704 const EventFdOffset = 1000000
4705 const NormalFdOffset = 100
4706
4707 /* 2020-1021 replaced poll() with channel/select
4708 func putKeyEvent(event int, evarg int) {
4709   if true {
4710     if EventRecvFd < 0 {
4711       var pv = [int(-1), -1]
4712       syscall.Pipe(pv)
4713       EventRecvFd = pv[0]
4714       EventSendFd = pv[1]
4715       //fmt.Printf("--De-- EventPipe created(%v,%v)\n", EventRecvFd, EventSendFd)
4716     }
4717   } else {
4718     if EventRecvFd < 0 {
4719       // the document differs from this spec
4720       // https://golang.org/src/syscall/syscall_unix.go?s=8096:8158#L340
4721       sv, err := syscall.Socketpair(syscall.AF_UNIX, syscall.SOCK_STREAM, 0)
4722       EventRecvFd = sv[0]
4723       EventSendFd = sv[1]
4724       if err != nil {
4725         fmt.Printf("--De-- EventSock created(%v,%v)(%v)\n",
4726           EventRecvFd, EventSendFd, err)
4727       }
4728     }
4729   }
4730   var buf = []byte{ byte(event) }
4731   n, err := syscall.Write(EventSendFd, buf)
4732   if err != nil {
4733     fmt.Printf("--De-- putEvent(%v) (%3v) (%v %v)\n", EventSendFd, event, n, err)
4734   }
4735 }
4736 */
4737 func ungets(str string) {
4738   for _, ch := range str {
4739     putKeyEvent(int(ch), 0)
4740   }
4741 }
4742 func (gsh*GshContext)xReplay(argv []string) {
4743   hix := 0
4744   tempo := 1.0
4745   xtempo := 1.0
4746   repeat := 1
4747
4748   for _, a := range argv { // tempo
4749     if strBegins(a, "x") {
4750       fmt.Sscanf(a[1:], "%f", &xtempo)
4751       tempo = 1 / xtempo
4752       //fprintf(stderr, "--Dr-- tempo=[%v]\n", a[2:], tempo);
4753     } else {
4754       if strBegins(a, "r") { // repeat
4755         fmt.Sscanf(a[1:], "%v", &repeat)
4756       } else {
4757         if strBegins(a, "l") {
4758           fmt.Sscanf(a[1:], "%d", &hix)
4759         } else {
4760           fmt.Sscanf(a, "%d", &hix)
4761         }
4762       }
4763     }
4764     if hix == 0 || len(argv) <= 1 {
4765       hix = len(gsh.CommandHistory) - 1
4766     }
4767     fmt.Printf("--Ir-- Replay(!%v x%v r%v)\n", hix, xtempo, repeat)
4768     //gsh.xScanReplay(hix, false, repeat, tempo, argv)
4769     go gsh.xScanReplay(hix, true, repeat, tempo, argv)
4770
4771     runtime.Gosched(); // wait xScanReplay is launched
4772     //fmt.Printf("--Ir-- Replay set\n");
4773   }
4774
4775   // <a href="https://golang.org/pkg/syscall/#FdSet">syscall.Select</a>
4776   // 2020-0827, GShell-0.2.3
4777   /*
4778   func FpollInl(fp *os.File, usec int) (uintptr) {

```

```

4779 nfd := 1
4780
4781 rdv := syscall.FdSet {}
4782 fd1 := fp.Fd()
4783 bank1 := fd1/32
4784 mask1 := int32(1 << fd1)
4785 rdv.Bits[bank1] = mask1
4786
4787 fd2 := -1
4788 bank2 := -1
4789 var mask2 int32 = 0
4790
4791 if 0 <= EventRecvFd {
4792     fd2 = EventRecvFd
4793     nfd = fd2 + 1
4794     bank2 = fd2/32
4795     mask2 = int32(1 << fd2)
4796     rdv.Bits[bank2] |= mask2
4797     //fmt.Printf("--De-- EventPoll mask added [%d][%v][%v]\n",fd2,bank2,mask2)
4798 }
4799
4800 tout := syscall.NsecToTimeval(int64(usec*1000))
4801 //n,err := syscall.Select(nfd,&rdv,nil,nil,&tout) // spec. mismatch
4802 err := syscall.Select(nfd,&rdv,nil,nil,&tout)
4803 if err != nil {
4804     //fmt.Printf("--De-- select() err(%v)\n",err)
4805 }
4806 if err == nil {
4807     if 0 <= fd2 && (rdv.Bits[bank2] & mask2) != 0 {
4808         if false {
4809             fmt.Printf("--De-- got Event\n")
4810         }
4811         return uintptr(EventFdOffset + fd2)
4812     }else{
4813         if (rdv.Bits[bank1] & mask1) != 0 {
4814             return uintptr(NormalFdOffset + fd1)
4815         }else{
4816             return 1
4817         }
4818     }else{
4819         return 0
4820     }
4821 }
4822 /*
4823 */
4824 func fgetcTimeout1(fp *os.File,usec int)(int){
4825     READ1:
4826     //readyFd := FpollIn1(fp,usec)
4827     readyFd := CpollIn1(fp,usec)
4828     if readyFd < 100 {
4829         return EV_TIMEOUT
4830     }
4831
4832     var buf [1]byte
4833
4834     if EventFdOffset <= readyFd {
4835         fd := int(readyFd-EventFdOffset)
4836         _,err := syscall.Read(fd,buf[0:1])
4837         if (err != nil) {
4838             return EOF;
4839         }else{
4840             if buf[0] == EV_MODE {
4841                 recvKeyEvent(fd)
4842                 goto READ1
4843             }
4844             return int(buf[0])
4845         }
4846     }
4847     _,err := fp.Read(buf[0:1])
4848     if( err != nil ){
4849         return EOF;
4850     }else{
4851         return int(buf[0])
4852     }
4853 }
4854 /*
4855 */
4856 func visibleChar(ch int)(string){
4857     switch {
4858     case ' ' <= ch && ch <= '-':
4859         return string(ch)
4860     }
4861     switch ch {
4862     case '\a': return "\a"
4863     case '\n': return "\n"
4864     case '\r': return "\r"
4865     case '\t': return "\t"
4866     }
4867     switch ch {
4868     case 0x00: return "NUL"
4869     case 0x07: return "BEL"
4870     case 0x08: return "BS"
4871     case 0x0E: return "SO"
4872     case 0x0F: return "SI"
4873     case 0x1B: return "ESC"
4874     case 0x7F: return "DEL"
4875     }
4876     switch ch {
4877     case EV_IDLE: return fmt.Sprintf("IDLE")
4878     case EV_MODE: return fmt.Sprintf("MODE")
4879     }
4880     return fmt.Sprintf("%X",ch)
4881 }
4882 /*
4883 */
4884 func recvKeyEvent(fd int){
4885     var buf = make([]byte,1)
4886     _,err := syscall.Read(fd,buf[0:1])
4887     if( buf[0] != 0 ){
4888         romkanmode = true
4889     }else{
4890         romkanmode = false
4891     }
4892 }
4893 /*
4894 */
4895 func (gsh*GshContext)xScanReplay(hix int,replay bool,repeat int,tempo float64,argv[]string){
4896     var Start time.Time
4897     var events = []Event{}
4898     for _,e := range Events {
4899         if hix == 0 || e.CmdIndex == hix {
4900             events = append(events,e)
4901         }
4902     }
4903     elen := len(events)
4904     if 0 < elen {
4905         if events[elen-1].event == EV_IDLE {
4906             events = events[0:elen-1]
4907         }
4908     }
4909     for r := 0; r < repeat; r++ {
4910         for i,e := range events {
4911             nano := e.when.Nanosecond()
4912             micro := nano / 1000
4913             if Start.Second() == 0 {
4914                 Start = time.Now()
4915             }
4916             diff := time.Now().Sub(Start)
4917             if replay {
4918                 if e.event != EV_IDLE {
4919                     //fmt.Printf("--replay %v / %v event=%X\n",i,len(events),e.event);
4920                     putKeyEvent(e.event,0)
4921                     if e.event == EV_MODE { // event with arg
4922                         putKeyEvent(int(e.evarg),0)
4923                     }
4924                 }else{
4925                     //fmt.Printf("--replay %v / %v idle=%X\n",i,len(events),e.event);
4926                 }
4927             }else{
4928                 fmt.Printf("#%3fms #%-3v !%-3v [%v.%06d] %3v %02X %%-4v %10.3fms\n",
4929                     float64(diff)/1000000.0,
4930                     i,
4931                     e.CmdIndex,
4932                     e.when.Format(time.Stamp),micro,
4933                     e.event,e.event,visibleChar(e.event),
4934                     float64(e.evarg)/1000000.0)
4935             }
4936             if e.event == EV_IDLE {
4937                 //fmt.Printf("--replay %v / %v delay\n",i,len(events));
4938                 d := time.Duration(float64(time.Duration(e.evarg)) * tempo)
4939                 //nleeeep(time.Duration(e.evarg))
4940                 nleeeep(d)
4941             }
4942         }
4943     }
4944     func dumpEvents(argv[]string){
4945         hix = 0
4946         if 1 < len(argv) {
4947             fmt.Sscanf(argv[1],"%d",&hix)
4948         }
4949         for i,e := range Events {
4950             nano := e.when.Nanosecond()
4951             micro := nano / 1000
4952             //if e.event != EV_TIMEOUT {
4953             if hix == 0 || e.CmdIndex == hix {
4954                 fmt.Printf("#%-3v !%-3v [%v.%06d] %3v %02X %%-4v %10.3fms\n",i,
4955                     e.CmdIndex,

```

```

4956         e.when.Format(time.Stamp),micro,
4957         e.event,e.event,visibleChar(e.event),float64(e.evarg)/1000000.0)
4958     }
4959     //}
4960 }
4961 }
4962 /*
4963 func fgetcTimeout(fp *os.File,usec int)(int){
4964     ch := fgetcTimeout(fp,usec)
4965     if ch == EV_TIMEOUT {
4966         now := Time.Now()
4967         if 0 < len(Events) {
4968             last := Events[len(Events)-1]
4969             dura := int64(now.Sub(last.when))
4970             Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
4971         }
4972         Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
4973     }
4974     return ch
4975 }
4976 */
4977
4978 // 2020-1021 replaced poll() with channel/select
4979 var Kbd = make(chan int);
4980 var Kbinet = false;
4981 var evQ = make(chan int);
4982 /*
4983 func keyInput(kbd chan int, fp *os.File){
4984     for {
4985         ch := C.getc(C.stdin);
4986         if (ch == C.EOF) {
4987             break;
4988         }
4989         kbd <- int(ch);
4990     }
4991 }
4992 */
4993 // https://godoc.org/golang.org/x/crypto/ssh/terminal
4994 // https://stackoverflow.com/questions/14094190/function-similar-to-getchar
4995 func keyInput(kbd chan int, tty *os.File){
4996     tmode := C.setTermRaw();
4997     defer func(){ C.setTermMode(tmode); }();
4998     if( !OnWindows ){
4999         system("/bin/stty -echo -icanon");
5000         defer func(){ system("/bin/stty echo sane"); }();
5001     }
5002     for {
5003         var rbuf []byte = make([]byte,1);
5004         if( OnWindows ){
5005             C.setTermRaw();
5006         }
5007         _,rerr := tty.Read(rbuf);
5008         if( rerr != nil ){
5009             break;
5010         }
5011         //fmt.Printf("++KBD[%X]\n",rbuf[0]);
5012         kbd <- int(rbuf[0]);
5013     }
5014     if( !OnWindows ){ system("/bin/stty echo sane"); }
5015 }
5016 func fgetcTimeout(fp *os.File,usec int)(int){
5017     if( !Kbinet ){
5018         Kbinet = true;
5019         go keyInput(Kbd,fp);
5020     }
5021     for {
5022         select {
5023             case <- time.After(time.Duration(usec*1000)):
5024                 //fmt.Printf("--Timeout %v us\n",usec);
5025                 return EV_TIMEOUT;
5026             case ch := <- Kbd:
5027                 //fmt.Printf("--KBD[%X]\n",ch);
5028                 // record a KeyIn(ch) Event
5029                 {
5030                     now := time.Now()
5031                     if 0 < len(Events) {
5032                         last := Events[len(Events)-1]
5033                         dura := int64(now.Sub(last.when))
5034                         Events = append(Events,Event{last.when,EV_IDLE,dura,last.CmdIndex})
5035                     }
5036                     Events = append(Events,Event{time.Now(),ch,0,CmdIndex})
5037                 }
5038                 return ch;
5039             case ch := <- evQ:
5040                 if( ch == EV_MODE ){
5041                     recvKeyEvent()
5042                 }else{
5043                     return ch;
5044                 }
5045         }
5046     }
5047 }
5048 func putKeyInEvent(event int, evarg int){
5049     evQ <- event;
5050 }
5051 func recvKeyEvent(){
5052     ch := <- evQ;
5053     if( ch != 0 ){
5054         romkanmode = true
5055     }else{
5056         romkanmode = false
5057     }
5058 }
5059
5060 var AtConsoleLineTop = true
5061 var TtyMaxCol = 72 // to be obtained by ioctl?
5062 var EscTimeout = (100*100)
5063 var (
5064     MODE_VicMode    bool // vi compatible command mode
5065     MODE_ShowMode   bool // shown translation mode, the mode to be retained
5066     MODE_Recursive  bool // recursive translation
5067     MODE_CapsLock   bool // software CapsLock
5068     MODE_LowerLock  bool // force lower-case character lock
5069     MODE_VInsert    int // visible insert mode, should be like "I" icon in X Window
5070     MODE_ViTrace    bool // output newline before translation
5071 )
5072
5073 type IInput struct {
5074     lno int
5075     lastlno int
5076     pch []int // input queue
5077     prompt string
5078     line string
5079     right string
5080     inmode bool
5081     pinmode bool
5082     waitingMeta string // waiting meta character
5083     LastCmd string
5084 }
5085 func (iin*IInput)Getc(timeoutUs int)(int){
5086     ch1 := EOF
5087     ch2 := EOF
5088     ch3 := EOF
5089     if( 0 < len(iin.pch) ){ // deQ
5090         ch1 = iin.pch[0]
5091         iin.pch = iin.pch[1:]
5092     }else{
5093         ch1 = fgetcTimeout(stdin,timeoutUs);
5094     }
5095     if( ch1 == 033 ){ // escape sequence
5096         ch2 = fgetcTimeout(stdin,EscTimeout);
5097         if( ch2 == EV_TIMEOUT ){
5098             ch3 = fgetcTimeout(stdin,EscTimeout);
5099             if( ch3 == EV_TIMEOUT ){
5100                 iin.pch = append(iin.pch,ch2) // enQ
5101             }else{
5102                 switch( ch2 ){
5103                     default:
5104                         iin.pch = append(iin.pch,ch2) // enQ
5105                         iin.pch = append(iin.pch,ch3) // enQ
5106                     case '[':
5107                         switch( ch3 ){
5108                             case 'A': ch1 = GO_UP; // ^
5109                             case 'B': ch1 = GO_DOWN; // v
5110                             case 'C': ch1 = GO_RIGHT; // >
5111                             case 'D': ch1 = GO_LEFT; // <
5112                             case '3':
5113                                 ch4 := fgetcTimeout(stdin,EscTimeout);
5114                                 if( ch4 == '-' ){
5115                                     //fprintf(stderr,"X[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
5116                                     ch1 = DEL_RIGHT
5117                                 }
5118                             case '\\':
5119                                 //ch4 := fgetcTimeout(stdin,EscTimeout);
5120                                 //fprintf(stderr,"Y[%02X %02X %02X %02X]\n",ch1,ch2,ch3,ch4);
5121                                 switch( ch3 ){
5122                                     case '-': ch1 = DEL_RIGHT
5123                                 }
5124                             }
5125                         }
5126                     }
5127             }
5128         }
5129     }
5130     return ch1
5131 }
5132 func (iin*IInput)clearline(){

```

```

5133 var i int
5134 fprintf(stderr, "\r");
5135 // should be ANSI ESC sequence
5136 for i = 0; i < TtyMaxCol; i++ { // to the max. position in this input action
5137     fputc(' ', os.Stderr);
5138 }
5139 fprintf(stderr, "\r");
5140 }
5141 func (iin*IInput)Redraw(){
5142     redraw(iin, iin.lno, iin.line, iin.right)
5143 }
5144 func redraw(iin *IInput, lno int, line string, right string){
5145     inMeta := false
5146     showMode := ""
5147     showMeta := "" // visible Meta_mode on the cursor position
5148     showLine := fmt.Sprintf("%d", lno)
5149     InsertMark := "" // in visible insert mode
5150
5151     if MODE_VicMode {
5152     }else{
5153         if 0 < len(iin.right) {
5154             InsertMark = " "
5155         }
5156
5157         if( 0 < len(iin.waitingMeta) ){
5158             inMeta = true
5159             if iin.waitingMeta[0] != 033 {
5160                 showMeta = iin.waitingMeta
5161             }
5162         }
5163         if( romkanmode ){
5164             //romkanmark = " *";
5165         }else{
5166             //romkanmark = "";
5167         }
5168         if MODE_ShowMode {
5169             romkan := "--"
5170             inmeta := "--"
5171             inveri := "--"
5172             if MODE_CapsLock {
5173                 inmeta = "A"
5174             }
5175             if MODE_LowerLock {
5176                 inmeta = "a"
5177             }
5178             if MODE_ViTrace {
5179                 inveri = "v"
5180             }
5181             if MODE_VicMode {
5182                 inveri = ":"
5183             }
5184             if romkanmode {
5185                 romkan = "343201202"
5186                 if MODE_CapsLock {
5187                     inmeta = "R"
5188                 }else{
5189                     inmeta = "r"
5190                 }
5191             }
5192             if inMeta {
5193                 inmeta = "\\ "
5194             }
5195             showMode = "["+romkan+inmeta+inveri+"]";
5196         }
5197         Pre := "\r" + showMode + showLine
5198         Output := ""
5199         Left := ""
5200         Right := ""
5201         if romkanmode {
5202             Left = convs(line)
5203             Right = InsertMark+convs(right)
5204         }else{
5205             Left = line
5206             Right = InsertMark+right
5207         }
5208         Output = Pre+Left
5209         if MODE_ViTrace {
5210             Output += iin.LastCmd
5211         }
5212         Output += showMeta+Right
5213         for len(Output) < TtyMaxCol { // to the max. position that may be dirty
5214             Output += " "
5215             // should be ANSI ESC sequence
5216             // not necessary just after newline
5217         }
5218         Output += Pre+Left+showMeta // to set the cursor to the current input position
5219         fprintf(stderr, "%s", Output)
5220
5221         if MODE_ViTrace {
5222             if 0 < len(iin.LastCmd) {
5223                 iin.LastCmd = ""
5224                 fprintf(stderr, "\r\n")
5225             }
5226         }
5227         AtConsoleLineTop = false
5228         //fmt.Printf("(Redraw(%v)(%v)\n", len(line), len(right));
5229     }
5230     // <a href="https://golang.org/pkg/unicode/utf8/">utf8</a>
5231     func delHeadChar(str string)(rline string, head string){
5232         clen := utf8.DecodeRune([]byte(str))
5233         head = string(str[0:clen])
5234         return str[clen:], head
5235     }
5236     func delTailChar(str string)(rline string, last string){
5237         var i = 0
5238         var clen = 0
5239         for {
5240             _, siz := utf8.DecodeRune([]byte(str)[i:])
5241             if siz == 0 { break }
5242             clen = siz
5243             i += siz
5244         }
5245         last = str[len(str)-clen:]
5246         return str[0:len(str)-clen:], last
5247     }
5248
5249     // > for output and history
5250     // < for keylog?
5251     // <a name="getline">Command Line Editor</a>
5252     func xgetline(lno int, prevline string, gsh*GshContext)(string){
5253         var iin IInput
5254         iin.lno = lno
5255         iin.lno = lno
5256
5257         CmdIndex = len(gsh.CommandHistory)
5258         if( isatty(0) == 0 ){
5259             if( sifgets(&iin.line, LINESIZE, stdin) == NULL ){
5260                 iin.line = "exit\n";
5261             }else{
5262             }
5263             return iin.line
5264         }
5265         if( true ){
5266             //var pts string;
5267             //pts = ptsname(0);
5268             //pts = ttyname(0);
5269             //fprintf(stderr, "--pts[0] = %s\n, pts?pts: "?");
5270         }
5271         if( false ){
5272             fprintf(stderr, "! ");
5273             fflush(stderr);
5274             sifgets(&iin.line, LINESIZE, stdin);
5275             return iin.line
5276         }
5277         if( !onWindows ){ system("/bin/stty -echo -icanon"); }
5278         xline := iin.xgetline(prevline, gsh)
5279         if( !onWindows ){ system("/bin/stty echo sane"); }
5280         return xline
5281     }
5282     func (iin*IInput)Translate(cmdch int){
5283         romkanmode = !romkanmode;
5284         if MODE_ViTrace {
5285             fprintf(stderr, "%v\r\n", string(cmdch));
5286         }else{
5287             if( cmdch == 'J' ){
5288                 fprintf(stderr, "J\r\n");
5289                 iin.injmode = true
5290             }
5291             iin.Redraw();
5292             loadDefaultDic(cmdch);
5293             iin.Redraw();
5294         }
5295     }
5296     func (iin*IInput)Replace(cmdch int){
5297         iin.LastCmd = fmt.Sprintf("%v", string(cmdch))
5298         iin.Redraw();
5299         loadDefaultDic(cmdch);
5300         dst := convs(iin.line+iin.right);
5301         iin.line = dst
5302         iin.right = ""
5303         if( cmdch == '+' ){
5304             fprintf(stderr, "\r\n");
5305             iin.injmode = true
5306         }
5307         iin.Redraw();
5308     }
5309     // aa 12 alal
5310     func isAlpha(ch rune)(bool){

```

```

5310 if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5311     return true
5312 }
5313 return false
5314 }
5315 func isAlnum(ch rune)(bool){
5316     if 'a' <= ch && ch <= 'z' || 'A' <= ch && ch <= 'Z' {
5317         return true
5318     }
5319     if '0' <= ch && ch <= '9' {
5320         return true
5321     }
5322     return false
5323 }
5324 }
5325 // 0.2.8 2020-0901 created
5326 // <a href="https://golang.org/pkg/unicode/utf8/#DecodeRuneInString">DecodeRuneInString</a>
5327 func (iin*Input)GotoTOPW(){
5328     str := iin.line
5329     i := len(str)
5330     if i <= 0 {
5331         return
5332     }
5333     //i0 := i
5334     i -= 1
5335     lastSize := 0
5336     var lastRune rune
5337     var found = -1
5338     for 0 < i { // skip preamble spaces
5339         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5340         if !isAlnum(lastRune) { // character, type, or string to be searched
5341             i = lastSize
5342             continue
5343         }
5344         break
5345     }
5346     for 0 < i {
5347         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5348         if lastSize <= 0 { continue } // not the character top
5349         if !isAlnum(lastRune) { // character, type, or string to be searched
5350             found = i
5351             break
5352         }
5353         i -= lastSize
5354     }
5355     if found < 0 && i == 0 {
5356         found = 0
5357     }
5358     if 0 <= found {
5359         if isAlnum(lastRune) { // or non-kana character
5360             }else{ // when positioning to the top o the word
5361                 i += lastSize
5362             }
5363             iin.right = str[i:] + iin.right
5364             if 0 < i {
5365                 iin.line = str[0:i]
5366             }else{
5367                 iin.line = ""
5368             }
5369         }
5370         //fmt.Printf("\n(%d,%d,%d)[%s][%s]\n",i0,i,found,iin.line,iin.right)
5371         //fmt.Printf("") // set debug messae at the end of line
5372     }
5373 // 0.2.8 2020-0901 created
5374 func (iin*Input)GotoENDW(){
5375     str := iin.right
5376     if len(str) <= 0 {
5377         return
5378     }
5379     lastSize := 0
5380     var lastRune rune
5381     var lastW = 0
5382     i := 0
5383     inWord := false
5384     lastRune, lastSize = utf8.DecodeRuneInString(str[0:])
5385     if isAlnum(lastRune) {
5386         r, r := utf8.DecodeRuneInString(str[lastSize:])
5387         if 0 < z && isAlnum(r) {
5388             inWord = true
5389         }
5390     }
5391     for i < len(str) {
5392         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5393         if lastSize <= 0 { break } // broken data?
5394         if !isAlnum(lastRune) { // character, type, or string to be searched
5395             break
5396         }
5397         lastW = i // the last alnum if in alnum word
5398         i += lastSize
5399     }
5400     if inWord {
5401         goto DISP
5402     }
5403     for i < len(str) {
5404         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5405         if lastSize <= 0 { break } // broken data?
5406         if !isAlnum(lastRune) { // character, type, or string to be searched
5407             break
5408         }
5409         i += lastSize
5410     }
5411     for i < len(str) {
5412         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5413         if lastSize <= 0 { break } // broken data?
5414         if !isAlnum(lastRune) { // character, type, or string to be searched
5415             break
5416         }
5417         lastW = i
5418         i += lastSize
5419     }
5420     DISP:
5421     if 0 < lastW {
5422         iin.line = iin.line + str[0:lastW]
5423         iin.right = str[lastW:]
5424     }
5425     //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
5426     //fmt.Printf("") // set debug messae at the end of line
5427 }
5428 // 0.2.8 2020-0901 created
5429 func (iin*Input)GotoNEXTW(){
5430     str := iin.right
5431     if len(str) <= 0 {
5432         return
5433     }
5434     lastSize := 0
5435     var lastRune rune
5436     var found = -1
5437     i := 1
5438     for i < len(str) {
5439         lastRune, lastSize = utf8.DecodeRuneInString(str[i:])
5440         if lastSize <= 0 { break } // broken data?
5441         if !isAlnum(lastRune) { // character, type, or string to be searched
5442             found = i
5443             break
5444         }
5445         i += lastSize
5446     }
5447     if 0 < found {
5448         if isAlnum(lastRune) { // or non-kana character
5449             }else{ // when positioning to the top o the word
5450                 found += lastSize
5451             }
5452             iin.line = iin.line + str[0:found]
5453             if 0 < found {
5454                 iin.right = str[found:]
5455             }else{
5456                 iin.right = ""
5457             }
5458         }
5459         //fmt.Printf("\n(%d)[%s][%s]\n",i,iin.line,iin.right)
5460         //fmt.Printf("") // set debug messae at the end of line
5461     }
5462 // 0.2.8 2020-0902 created
5463 func (iin*Input)GotoPAIRCH(){
5464     str := iin.right
5465     if len(str) <= 0 {
5466         return
5467     }
5468     lastRune, lastSize := utf8.DecodeRuneInString(str[0:])
5469     if lastSize <= 0 {
5470         return
5471     }
5472     forw := false
5473     back := false
5474     pair := ""
5475     switch string(lastRune){
5476     case "(": pair = ")"; forw = true
5477     case ")": pair = "("; back = true
5478     case "[": pair = "]"; forw = true
5479     case "]": pair = "["; back = true
5480     case "<": pair = ">"; forw = true
5481     case ">": pair = "<"; back = true
5482     case "`": pair = "`"; // context depednet, can be f" or back-double quote
5483     case "": pair = ""; // context depednet, can be f' or back-quote
5484     }

```

```

5487 // case Japanese Kakkos
5488 }
5489 if forw {
5490 iin.SearchForward(pair)
5491 }
5492 if back {
5493 iin.SearchBackward(pair)
5494 }
5495 }
5496 // 0.2.8 2020-0902 created
5497 func (iin*Input)SearchForward(pat string)(bool){
5498 right := iin.right
5499 found := -1
5500 i := 0
5501 if strBegins(right,pat) {
5502 _,z := utf8.DecodeRuneInString(right[i:])
5503 if 0 < z {
5504 i += z
5505 }
5506 }
5507 for i < len(right) {
5508 if strBegins(right[i:],pat) {
5509 found = i
5510 break
5511 }
5512 _,z := utf8.DecodeRuneInString(right[i:])
5513 if z <= 0 { break }
5514 i += z
5515 }
5516 if 0 <= found {
5517 iin.line = iin.line + right[0:found]
5518 iin.right = iin.right[found:]
5519 return true
5520 }else{
5521 return false
5522 }
5523 }
5524 // 0.2.8 2020-0902 created
5525 func (iin*Input)SearchBackward(pat string)(bool){
5526 line := iin.line
5527 found := -1
5528 i := len(line)-1
5529 for i = i; 0 <= i; i-- {
5530 _,z := utf8.DecodeRuneInString(line[i:])
5531 if z <= 0 {
5532 continue
5533 }
5534 //fprintf(stderr,"-- %v %v\n",pat,line[i:])
5535 if strBegins(line[i:],pat) {
5536 found = i
5537 break
5538 }
5539 }
5540 //fprintf(stderr,"--%d\n",found)
5541 if 0 <= found {
5542 iin.right = line[found:] + iin.right
5543 iin.line = line[0:found]
5544 return true
5545 }else{
5546 return false
5547 }
5548 }
5549 // 0.2.8 2020-0902 created
5550 // search from top, end, or current position
5551 func (gsh*GshContext)SearchHistory(pat string, forw bool)(bool,string){
5552 if forw {
5553 v := range gsh.CommandHistory {
5554 if 0 <= strings.Index(v.CmdLine,pat) {
5555 //fprintf(stderr,"n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
5556 return true,v.CmdLine
5557 }
5558 }
5559 }else{
5560 hlen := len(gsh.CommandHistory)
5561 for i := hlen-1; 0 < i; i-- {
5562 v := gsh.CommandHistory[i]
5563 if 0 <= strings.Index(v.CmdLine,pat) {
5564 //fprintf(stderr,"n--De-- found !%v [%v]%v\n",i,pat,v.CmdLine)
5565 return true,v.CmdLine
5566 }
5567 }
5568 }
5569 //fprintf(stderr,"n--De-- not-found(%v)\n",pat)
5570 return false,"(Not Found in History)"
5571 }
5572 // 0.2.8 2020-0902 created
5573 func (iin*Input)GotoFORWSTR(pat string,gsh*GshContext){
5574 found := false
5575 if 0 < len(iin.right) {
5576 found = iin.SearchForward(pat)
5577 }
5578 if !found {
5579 found,line := gsh.SearchHistory(pat,true)
5580 if found {
5581 iin.line = line
5582 iin.right = ""
5583 }
5584 }
5585 }
5586 func (iin*Input)GotoBACKSTR(pat string, gsh*GshContext){
5587 found := false
5588 if 0 < len(iin.line) {
5589 found = iin.SearchBackward(pat)
5590 }
5591 if !found {
5592 found,line := gsh.SearchHistory(pat,false)
5593 if found {
5594 iin.line = line
5595 iin.right = ""
5596 }
5597 }
5598 }
5599 func (iin*Input)getStringl(prompt string)(string){ // should be editable
5600 iin.clearline();
5601 fprintf(stderr,"\r%v",prompt)
5602 str := ""
5603 for {
5604 ch := iin.Getc(10*1000*1000)
5605 if ch == '\n' || ch == '\r' {
5606 break
5607 }
5608 sch := string(ch)
5609 str += sch
5610 fprintf(stderr,"%s",sch)
5611 }
5612 return str
5613 }
5614 }
5615 // search pattern must be an array and selectable with 'N/'P
5616 var SearchPat = ""
5617 var SearchForw = true
5618 }
5619 func (iin*Input)xgetline(prevline string, gsh*GshContext)(string){
5620 var ch int;
5621 }
5622 MODE_ShowMode = false
5623 MODE_VicMode = false
5624 iin.Redraw();
5625 first := true
5626 }
5627 for cix := 0; ; cix++ {
5628 iin.pinJmode = iin.inJmode
5629 iin.inJmode = false
5630 }
5631 ch = iin.Getc(1000*1000)
5632 }
5633 if ch != EV_TIMEOUT && first {
5634 first = false
5635 mode := 0
5636 if romkanmode {
5637 mode = 1
5638 }
5639 now := time.Now()
5640 Events = append(Events,Event(now,EV_MODE,int64(mode),CmdIndex))
5641 }
5642 if ch == 033 {
5643 MODE_ShowMode = true
5644 MODE_VicMode = IMODE_VicMode
5645 iin.Redraw();
5646 continue
5647 }
5648 if MODE_VicMode {
5649 swiTch ch {
5650 case 'o': ch = GO_TOPL
5651 case '$': ch = GO_ENDL
5652 case 'b': ch = GO_TOPW
5653 case 'e': ch = GO_ENDW
5654 case 'w': ch = GO_NEXTW
5655 case '$': ch = GO_PAIRCH
5656 }
5657 case 'j': ch = GO_DOWN
5658 case 'k': ch = GO_UP
5659 case 'h': ch = GO_LEFT
5660 case 'l': ch = GO_RIGHT
5661 case 'x': ch = DEF_RIGHT
5662 case 'a': MODE_VicMode = IMODE_VicMode
5663 ch = GO_RIGHT

```



```

5664     case 'i': MODE_VicMode = IMODE_VicMode
5665     iin.Redraw();
5666     continue
5667     case '-':
5668     right,head := delHeadChar(iin.right)
5669     if len([]byte(head)) == 1 {
5670     ch = int(head[0])
5671     if 'a' <= ch && ch <= 'z' {
5672     ch = ch + 'A'-'a'
5673     }else
5674     if 'A' <= ch && ch <= 'Z' {
5675     ch = ch + 'a'-'A'
5676     }
5677     iin.right = string(ch) + right
5678     }
5679     iin.Redraw();
5680     continue
5681     case 'f': // GO_FORWARD
5682     iin.Redraw();
5683     ch = iin.Getc(3*1000+1000)
5684     if ch == EV_TIMEOUT {
5685     iin.Redraw();
5686     continue
5687     }
5688     SearchPat = string(ch)
5689     SearchForw = true
5690     iin.GotoFORWSTR(SearchPat,gsh)
5691     iin.Redraw();
5692     continue
5693     case '/':
5694     SearchPat = iin.getstringl("/") // should be editable
5695     SearchForw = true
5696     iin.GotoFORWSTR(SearchPat,gsh)
5697     iin.Redraw();
5698     continue
5699     case '?':
5700     SearchPat = iin.getstringl("?") // should be editable
5701     SearchForw = false
5702     iin.GotoBACKSTR(SearchPat,gsh)
5703     iin.Redraw();
5704     continue
5705     case 'n':
5706     if SearchForw {
5707     iin.GotoFORWSTR(SearchPat,gsh)
5708     }else{
5709     iin.GotoBACKSTR(SearchPat,gsh)
5710     }
5711     iin.Redraw();
5712     continue
5713     case 'N':
5714     if !SearchForw {
5715     iin.GotoFORWSTR(SearchPat,gsh)
5716     }else{
5717     iin.GotoBACKSTR(SearchPat,gsh)
5718     }
5719     iin.Redraw();
5720     continue
5721     }
5722     }
5723     switch ch {
5724     case GO_TOPW:
5725     iin.GotoTOPW()
5726     iin.Redraw();
5727     continue
5728     case GO_ENDW:
5729     iin.GotoENDW()
5730     iin.Redraw();
5731     continue
5732     case GO_NEXTW:
5733     // To next space then
5734     iin.GotoNEXTW()
5735     iin.Redraw();
5736     continue
5737     case GO_PAIRCH:
5738     iin.GotoPAIRCH()
5739     iin.Redraw();
5740     continue
5741     }
5742     //fprintf(stderr,"A[%02X]\n",ch);
5743     if (ch == '\\') { ch = 033 }
5744     MODE_ShowMode = true
5745     metach := ch
5746     iin.waitingMeta = string(ch)
5747     iin.Redraw();
5748     // set cursor //fprintf(stderr,"???b\b\b")
5749     ch = fgetcTimeout(stdin,2000*1000)
5750     // reset cursor
5751     iin.waitingMeta = ""
5752     cmdch := ch
5753     if (ch == EV_TIMEOUT) {
5754     if metach == 033 {
5755     continue
5756     }
5757     ch = metach
5758     }else
5759     /*
5760     if (ch == 'm' || ch == 'M') {
5761     mch := fgetcTimeout(stdin,1000*1000)
5762     if mch == 'r' {
5763     romkanmode = true
5764     }else{
5765     romkanmode = false
5766     }
5767     }
5768     continue
5769     */
5770     if (ch == 'k' || ch == 'K') {
5771     MODE_Recursive = !MODE_Recursive
5772     iin.Translate(cmdch);
5773     continue
5774     }else
5775     if (ch == 'j' || ch == 'J') {
5776     iin.Translate(cmdch);
5777     continue
5778     }else
5779     if (ch == 'i' || ch == 'I') {
5780     iin.Replace(cmdch);
5781     continue
5782     }else
5783     if (ch == 'l' || ch == 'L') {
5784     MODE_LowerLock = !MODE_LowerLock
5785     MODE_CapsLock = false
5786     if MODE_ViTrace {
5787     fprintf(stderr,"%v\n",string(cmdch));
5788     }
5789     iin.Redraw();
5790     continue
5791     }else
5792     if (ch == 'u' || ch == 'U') {
5793     MODE_CapsLock = !MODE_CapsLock
5794     MODE_LowerLock = false
5795     if MODE_ViTrace {
5796     fprintf(stderr,"%v\n",string(cmdch));
5797     }
5798     iin.Redraw();
5799     continue
5800     }else
5801     if (ch == 'v' || ch == 'V') {
5802     MODE_ViTrace = !MODE_ViTrace
5803     if MODE_ViTrace {
5804     fprintf(stderr,"%v\n",string(cmdch));
5805     }
5806     iin.Redraw();
5807     continue
5808     }else
5809     if (ch == 'c' || ch == 'C') {
5810     if 0 < len(iin.line) {
5811     xline,tail := delTailChar(iin.line)
5812     if len([]byte(tail)) == 1 {
5813     ch = int(tail[0])
5814     if 'a' <= ch && ch <= 'z' {
5815     ch = ch + 'A'-'a'
5816     }else
5817     if 'A' <= ch && ch <= 'Z' {
5818     ch = ch + 'a'-'A'
5819     }
5820     }
5821     iin.line = xline + string(ch)
5822     }
5823     }
5824     if MODE_ViTrace {
5825     fprintf(stderr,"%v\n",string(cmdch));
5826     }
5827     iin.Redraw();
5828     continue
5829     }else{
5830     iin.pch = append(iin.pch,ch) // push
5831     ch = '\\'
5832     }
5833     }
5834     }
5835     switch (ch) {
5836     case 'P'-0x40: ch = GO_UP
5837     case 'N'-0x40: ch = GO_DOWN
5838     case 'B'-0x40: ch = GO_LEFT
5839     case 'F'-0x40: ch = GO_RIGHT
5840     }

```

```

5841 //fprintf(stderr, "%02X\n", ch);
5842 switch (ch) {
5843     case 0:
5844         continue;
5845
5846     case '\t':
5847         iin.Replace('j');
5848         continue;
5849     case 'X'-0x40:
5850         iin.Replace('j');
5851         continue;
5852
5853     case EV_TIMEOUT:
5854         iin.Redraw();
5855         if iin.pinMode {
5856             fprintf(stderr, "\\r\n")
5857             iin.inMode = true
5858         }
5859         continue;
5860     case GO_UP:
5861         if iin.lno == 1 {
5862             continue;
5863         }
5864         cmd, ok := gsh.cmdStringInHistory(iin.lno-1)
5865         if ok {
5866             iin.line = cmd
5867             iin.right = ""
5868             iin.lno = iin.lno - 1
5869         }
5870         iin.Redraw();
5871         continue;
5872     case GO_DOWN:
5873         cmd, ok := gsh.cmdStringInHistory(iin.lno+1)
5874         if ok {
5875             iin.line = cmd
5876             iin.right = ""
5877             iin.lno = iin.lno + 1
5878         } else {
5879             iin.line = ""
5880             iin.right = ""
5881             if iin.lno == iin.lastlno-1 {
5882                 iin.lno = iin.lno + 1
5883             }
5884         }
5885         iin.Redraw();
5886         continue;
5887     case GO_LEFT:
5888         if 0 < len(iin.line) {
5889             xline, tail := delTailChar(iin.line)
5890             iin.line = xline
5891             iin.right = tail + iin.right
5892         }
5893         iin.Redraw();
5894         continue;
5895     case GO_RIGHT:
5896         if 0 < len(iin.right) && iin.right[0] != 0 {
5897             xright, head := delHeadChar(iin.right)
5898             iin.right = xright
5899             iin.line += head
5900         }
5901         iin.Redraw();
5902         continue;
5903     case EOF:
5904         goto EXIT;
5905     case 'R'-0x40: // replace
5906         dst := convs(iin.line+iin.right);
5907         iin.line = dst
5908         iin.right = ""
5909         iin.Redraw();
5910         continue;
5911     case 'T'-0x40: // just show the result
5912         readDic();
5913         romkanmode = !romkanmode;
5914         iin.Redraw();
5915         continue;
5916     case 'L'-0x40:
5917         iin.Redraw();
5918         continue;
5919     case 'K'-0x40:
5920         iin.right = ""
5921         iin.Redraw();
5922         continue;
5923     case 'E'-0x40:
5924         iin.line += iin.right
5925         iin.right = ""
5926         iin.Redraw();
5927         continue;
5928     case 'A'-0x40:
5929         iin.right = iin.line + iin.right
5930         iin.line = ""
5931         iin.Redraw();
5932         continue;
5933     case 'U'-0x40:
5934         iin.line = ""
5935         iin.right = ""
5936         iin.clearline();
5937         iin.Redraw();
5938         continue;
5939     case DEL_RIGHT:
5940         if 0 < len(iin.right) {
5941             iin.right, _ = delHeadChar(iin.right)
5942             iin.Redraw();
5943         }
5944         continue;
5945     case 0x7F: // BS? not DEL
5946         if 0 < len(iin.line) {
5947             iin.line, _ = delTailChar(iin.line)
5948             iin.Redraw();
5949         }
5950         /*
5951         else
5952             if 0 < len(iin.right) {
5953                 iin.right, _ = delHeadChar(iin.right)
5954                 iin.Redraw();
5955             }
5956         */
5957         continue;
5958     case 'H'-0x40:
5959         if 0 < len(iin.line) {
5960             iin.line, _ = delTailChar(iin.line)
5961             iin.Redraw();
5962         }
5963         continue;
5964     }
5965     if (OnWindows && ch == '\n') {
5966         continue;
5967     }
5968     if (ch == '\n' || ch == '\r') {
5969         iin.line += iin.right;
5970         iin.right = ""
5971         iin.Redraw();
5972         /*putc(ch, stderr);
5973         fprintf(stderr, "\r\n"); // NL on Unix, CR on Windows
5974         AtConsoleLineTop = true
5975         break;
5976     }
5977     if MODE_CapsLock {
5978         if 'a' <= ch && ch <= 'z' {
5979             ch = ch+'A'-'a'
5980         }
5981     }
5982     if MODE_LowerLock {
5983         if 'A' <= ch && ch <= 'Z' {
5984             ch = ch+'a'-'A'
5985         }
5986     }
5987     iin.line += string(ch);
5988     iin.Redraw();
5989 }
5990 EXIT:
5991 return iin.line + iin.right;
5992 }
5993
5994 func getline main(){
5995     line := xgetline(0, "", nil)
5996     fprintf(stderr, "%s\n", line);
5997     /*
5998     dp = strpbk(line, "\r\n");
5999     if ( dp != NULL ) {
6000         *dp = 0;
6001     }
6002
6003     if ( 0 ) {
6004         fprintf(stderr, "\n(%d)\n", int(strlen(line)));
6005     }
6006     if ( lseek(3, 0, 0) == 0 ) {
6007         if ( romkanmode ) {
6008             var buf [8*1024]byte;
6009             convs(line, buf);
6010             strcpy(line, buf);
6011         }
6012         write(3, line, strlen(line));
6013         truncate(3, lseek(3, 0, SEEK_CUR));
6014         //fprintf(stderr, "outsize=%d\n", (int)lseek(3, 0, SEEK_END));
6015         lseek(3, 0, SEEK_SET);
6016         close(3);
6017     } else {

```

```

6018     fprintf(stderr, "\r\n gotline: ");
6019     trans(line);
6020     //printf("%s\n", line);
6021     printf("\n");
6022 }
6023 */
6024 }
6025 //== end ===== getline
6026
6027 //
6028 // $USERHOME/.gsh/
6029 // gsh-rc.txt, or gsh-configure.txt
6030 // gsh-history.txt
6031 // gsh-aliases.txt // should be conditional?
6032 //
6033 func (gshCtx *GshContext)gshSetupHomedir() (bool) {
6034     homedir, found := userHomeDir()
6035     if !found {
6036         fmt.Printf("--E-- You have no UserHomeDir\n")
6037         return true
6038     }
6039     gshhome := homedir + "/" + GSH_HOME
6040     _, err2 := os.Stat(gshhome)
6041     if err2 != nil {
6042         if err3 := os.Mkdir(gshhome, 0700)
6043             if err3 != nil {
6044                 fmt.Printf("--E-- Could not Create %s (%s)\n",
6045                     gshhome, err3)
6046                 return true
6047             }
6048             fmt.Printf("--I-- Created %s\n", gshhome)
6049         }
6050         gshCtx.GshHomeDir = gshhome
6051         return false
6052     }
6053 func setupGshContext()(GshContext, bool){
6054     //gshPA := syscall.ProcAttr {
6055     gshPA := os.ProcAttr {
6056         // the starting directory
6057         os.Environ(), // environ[]
6058         //[]uintptr(os.Stdin.Fd(), os.Stdout.Fd(), os.Stderr.Fd()),
6059         //[]os.File(os.Stdin, os.Stdout, os.Stderr),
6060         nil, // OS specific
6061     }
6062     cwd, _ := os.Getwd()
6063     gshCtx := gshContext {
6064         cwd, // StartDir
6065         //[]GchdirHistory { {cwd, time.Now(), 0} }, // ChdirHistory
6066         gshPA,
6067         []GCommandHistory {}, // something for invocation?
6068         GCommandHistory {}, // CmdCurrent
6069         false,
6070         []os.ProcessState {}, //[]int {},
6071         aRusage {},
6072         // // GshHomeDir
6073         Ttyid {},
6074         false,
6075         false,
6076         //[]PluginInfo {},
6077         []string {},
6078         "",
6079         "",
6080         ValueStack {},
6081         GServer {"", ""}, // LastServer
6082         // // RSDRV
6083         cwd, // RMD
6084         CheckSum {},
6085     }
6086     err := gshCtx.gshSetupHomedir()
6087     return gshCtx, err
6088 }
6089 }
6090 func (gsh *GshContext)gshellh(gline string) (bool) {
6091     ghist := gsh.CmdCurrent
6092     ghist.WorkDir_ = os.Getwd()
6093     ghist.WorkDirX = len(gsh.ChdirHistory) - 1
6094     //fmt.Printf("--D-- ChdirHistory(%d)\n", len(gsh.ChdirHistory))
6095     ghist.StartAt = time.Now()
6096     rusagev1 := Getrusagev()
6097     gsh.CmdCurrent.FoundFile = []string {}
6098     fin := gsh.gshellh(gline)
6099     rusagev2 := Getrusagev()
6100     ghist.Rusagev = RusageSubv(rusagev2, rusagev1)
6101     ghist.EndAt = time.Now()
6102     ghist.CmdLine = gline
6103     ghist.FoundFile = gsh.CmdCurrent.FoundFile
6104
6105     /* record it but not show in list by default
6106     if len(gline) == 0 {
6107         continue
6108     }
6109     if gline == "hi" || gline == "history" { // don't record it
6110         continue
6111     }
6112     */
6113     gsh.CommandHistory = append(gsh.CommandHistory, ghist)
6114     return fin
6115 }
6116 // <a name="main">Main loop</a>
6117 func script(gshCtxGiven *GshContext) (_ GshContext) {
6118     gshCtxBuf, err0 := setupGshContext()
6119     if err0 {
6120         return gshCtxBuf;
6121     }
6122     gshCtx := *gshCtxBuf
6123     //fmt.Printf("--I-- GSH_HOME=%s\n", gshCtx.GshHomeDir)
6124     //resmap()
6125
6126     /*
6127     if false {
6128         gsh_getline, with exgetline :=
6129             which("PATH", []string{"which", "gsh-getline", "-s"})
6130         if with exgetline {
6131             gsh_getline[0] = toFullpath(gsh_getline[0])
6132             gshCtx.GetLine = toFullpath(gsh_getline[0])
6133         } else {
6134             fmt.Printf("--W-- No gsh-getline found. Using internal getline.\n");
6135         }
6136     }
6137     */
6138 }
6139
6140 ghist0 := gshCtx.CmdCurrent // something special, or gshrc script, or permanent history
6141 gshCtx.CommandHistory = append(gshCtx.CommandHistory, ghist0)
6142
6143 prevline := ""
6144 skipping := false
6145 for hix := len(gshCtx.CommandHistory); ; {
6146     gline := gshCtx.getline(hix, skipping, prevline)
6147     if skipping {
6148         if strings.Index(gline, "fi") == 0 {
6149             fmt.Printf("fi\n");
6150             skipping = false;
6151         } else {
6152             //fmt.Printf("%s\n", gline);
6153         }
6154         continue
6155     }
6156     if strings.Index(gline, "if") == 0 {
6157         //fmt.Printf("--D-- if start: %s\n", gline);
6158         skipping = true;
6159         continue
6160     }
6161     if false {
6162         os.Stdout.Write([]byte("gotline:"))
6163         os.Stdout.Write([]byte(gline))
6164         os.Stdout.Write([]byte("\n"))
6165     }
6166     gline = strsubst(gshCtx, gline, true)
6167     if false {
6168         fmt.Printf("fmt.Printf %v - %v\n", gline)
6169         fmt.Printf("fmt.Printf %s - %s\n", gline)
6170         fmt.Printf("fmt.Printf %x - %s\n", gline)
6171         fmt.Printf("fmt.Printf %U - %s\n", gline)
6172         fmt.Printf("Stout.Write -")
6173         os.Stdout.Write([]byte(gline))
6174         fmt.Printf("\n")
6175     }
6176     /*
6177     // should be cared in substitution ?
6178     if 0 < len(gline) && gline[0] == 'l' {
6179         xgline, set, err := searchHistory(gshCtx, gline)
6180         if err {
6181             continue
6182         }
6183         if set {
6184             // set the line in command line editor
6185         }
6186         gline = xgline
6187     }
6188     */
6189     fin := gshCtx.gshellh(gline)
6190     if fin {
6191         break;
6192     }
6193     prevline = gline;
6194     hix++;

```

```

6195 }
6196 return *gshCtx
6197 }
6198 func ftest(where, path string){
6199 //fi, err := os.Stat(path);
6200 //fmt.Printf("-- %v os.Stat(%v)=(%v)%v\n", where, path, err, fi);
6201 }
6202 func main() {
6203 initGshEnv();
6204 ftest("gsh-main", "-.");
6205 ftest("gsh-main", "gsh.go");
6206 ftest("gsh-main", "gsh.exe");
6207 ftest("gsh-main", "gsh");
6208
6209 gshCtxBuf := GshContext{}
6210 gsh := *gshCtxBuf
6211 argv := os.Args
6212
6213 if( isin("wss", argv) ){
6214   gj_server(argv[1:]);
6215   return;
6216 }
6217 if( isin("wsc", argv) ){
6218   gj_client(argv[1:]);
6219   return;
6220 }
6221 if 1 < len(argv) {
6222   if isin("version", argv){
6223     gsh.showVersion(argv)
6224     return
6225   }
6226   if argv[1] == "gj" {
6227     if argv[2] == "listen" { go gj_server(argv[2:]); }
6228     if argv[2] == "server" { go gj_server(argv[2:]); }
6229     if argv[2] == "serve" { go gj_server(argv[2:]); }
6230     if argv[2] == "client" { go gj_client(argv[2:]); }
6231     if argv[2] == "join" { go gj_client(argv[2:]); }
6232   }
6233   comx := isinX("-c", argv)
6234   if 0 < comx {
6235     gshCtxBuf, err := setupGshContext()
6236     gsh := *gshCtxBuf
6237     if !err {
6238       gsh.gshellv(argv[comx+1:])
6239     }
6240     return
6241   }
6242   if 1 < len(argv) && isin("-s", argv) {
6243     gsh.showVersion(append(argv, []string{"-l", "-a"}...))
6244   } else {
6245     script(nil)
6246     //gshCtx := script(nil)
6247     //gshell(gshCtx, "time")
6248   }
6249 }
6250 }
6251 }
6252 }
6253 }
6254 }
6255 }
6256 }
6257 }
6258 }
6259 }
6260 }
6261 }
6262 }
6263 }
6264 }
6265 }
6266 }
6267 }
6268 }
6269 }
6270 }
6271 }
6272 }
6273 }
6274 }
6275 }
6276 }
6277 }
6278 }
6279 }
6280 }
6281 }
6282 }
6283 }
6284 }
6285 }
6286 }
6287 }
6288 }
6289 }
6290 }
6291 }
6292 }
6293 }
6294 }
6295 }
6296 }
6297 }
6298 }
6299 }
6300 }
6301 }
6302 }
6303 }
6304 }
6305 }
6306 }
6307 }
6308 }
6309 }
6310 }
6311 }
6312 }
6313 }
6314 }
6315 }
6316 }
6317 }
6318 }
6319 }
6320 }
6321 }
6322 }
6323 }
6324 }
6325 }
6326 }
6327 }
6328 }
6329 }
6330 }
6331 }
6332 }
6333 }
6334 }
6335 }
6336 }
6337 }
6338 }
6339 }
6340 }
6341 }
6342 }
6343 }
6344 }
6345 }
6346 }
6347 }
6348 }
6349 }
6350 }
6351 }
6352 }
6353 }
6354 }
6355 }
6356 }
6357 }
6358 }
6359 }
6360 }
6361 }
6362 }
6363 }
6364 }
6365 }
6366 }
6367 }
6368 }
6369 }
6370 }
6371 }

```

```

6372 <a href="https://developer.mozilla.org/en-US/docs/Web/CSS/background-repeat">repeat</a></span>
6373 <a href="https://developer.mozilla.org/en-US/docs/Web/JavaScript">JavaScript</a>
6374 <a href="https://developer.mozilla.org/en-US/docs/Web/HTTP">HTTP</a>
6375 <a href="https://mdn-web-dna.s3-us-west-2.amazonaws.com/MDN-Browser-Compatibility-Report-2020.pdf">MDN Browser Compatibility Report (2020)</a>
6376
6377 Go language (August 2020 / Go 1.15)
6378 <a href="https://golang.org">The Go Programming Language</a>
6379 <a href="https://golang.org/pkg/">Packages</a>
6380 <a href="https://godoc.org/golang.org/x/net/websocket">WebSocket</a>
6381
6382 <a href="https://stackoverflow.com/">Stackoverflow</a>
6383 <!--
6384 <iframe loading="lazy" src="https://golang.org" width="100%" height="300"></iframe>
6385 -->
6386 </div></details>
6387 */
6388 /*
6389 <details id="html-src" onclick="frame_open();"><summary>Raw Source</summary><div>
6390
6391 <!-- h2>The full of this HTML including the Go code is here.</h2 -->
6392 <details id="gsh-whole-view"><summary>Whole file</summary>
6393 <a name="whole-src-view"></a>
6394 <span id="src-frame"></span><!-- a window to show source code -->
6395 </details>
6396
6397 <details id="gsh-style-frame" onclick="fill_CSSView()"><summary>CSS part</summary>
6398 <a name="style-src-view"></a>
6399 <span id="gsh-style-view"></span>
6400 </details>
6401
6402 <details id="gsh-script-frame" onclick="fill_JavaScriptView()"><summary>JavaScript part</summary>
6403 <a name="script-src-view"></a>
6404 <span id="gsh-script-view"></span>
6405 </details>
6406
6407 <details id="gsh-data-frame" onclick="fill_DataView()"><summary>Builtin data part</summary>
6408 <a name="gsh-data-frame"></a>
6409 <span id="gsh-data-view"></span>
6410 </details>
6411
6412 </div></details>
6413 */
6414 /*
6415 <div id="GshFooter0"></div>
6416 <!-- 2020-09-17 started, visible script { -->
6417 <details><summary>GJScript</summary>
6418 <style>.gjscript { font-family:Georgia; }</style>
6419 <pre id="gjscript_1" class="gjscript"> function gjtest1(){ alert('Hello GJScript!'); }
6420 gjtest1()
6421 </pre>
6422 <script>
6423 gjs = document.getElementById('gjscript_1');
6424 //eval(gjs.innerHTML);
6425 //gjs.outerHTML = ""
6426 </script>
6427 </details><!-- END-OF-VISIBLE-PART ----- -->
6428
6429 <!--
6430 // 2020-0906 added,
6431 https://developer.mozilla.org/en-US/docs/Web/CSS/z-index
6432 https://developer.mozilla.org/en-US/docs/Web/CSS/position
6433 -->
6434 <span id="GshGrid">(^_^)</small><small>[Hit j k l h]</small></span>
6435
6436 <span id="GStat"><br>
6437 </span>
6438 <span id="GMenu" onclick="GShellMenu(this)" draggable="true"></span>
6439 <span id="GTop"></span>
6440 <div id="GShellPlane" onclick="showGShellPlane();"></div>
6441 <div id="RawTextViewer"></div>
6442 <div id="RawTextViewerClose" onclick="hideRawTextViewer()"> CLOSE </div>
6443
6444 <style id="GshStyleDef">
6445 #LineNumbered table,tr,td {
6446 margin:0;
6447 padding:4px;
6448 spacing:0;
6449 border:1px;
6450 }
6451
6452 textarea.LineNumber {
6453 font-size:12px;
6454 font-family:monospace,Courier New;
6455 color:#282;
6456 padding:4px;
6457 text-align:right;
6458 }
6459
6460 textarea.LineNumbered {
6461 font-size:12px;
6462 font-family:monospace,Courier New;
6463 padding:4px;
6464 wrap:off;
6465 }
6466 #RawTextViewer{
6467 z-index:0;
6468 position:fixed; top:0px; left:0px;
6469 width:100%; height:50px;
6470 overflow:auto;
6471 color:#fff; background-color:rgba(128,128,256,0.2);
6472 font-size:12px;
6473 spellcheck:false;
6474 }
6475 #RawTextViewerClose{
6476 z-index:0;
6477 position:fixed; top:-100px; left:-100px;
6478 color:#fff; background-color:rgba(128,128,256,0.2);
6479 font-size:20px; font-family:Georgia;
6480 white-space:pre;
6481 }
6482 #xxxGShellPlane{
6483 z-index:0;
6484 position:fixed; top:0px; left:0px;
6485 width:100%; height:50px;
6486 overflow:auto;
6487 color:#fff; background-color:rgba(128,128,256,0.3);
6488 font-size:12px;
6489 }
6490 #xxxGTop{
6491 z-index:9;
6492 opacity:1.0;
6493 position:fixed; top:0px; left:0px;
6494 width:320px; height:20px;
6495 color:#fff; background-color:rgba(32,32,160,0.15);
6496 }
6497 #xxxGPos{
6498 z-index:12;
6499 position:fixed; top:0px; left:0px;
6500 opacity:1.0;
6501 width:640px; height:30px;
6502 color:#fff; background-color:rgba(0,0,0,0.2);
6503 font-size:12px;
6504 }
6505 #GMenu{
6506 z-index:100000000;
6507 position:fixed; top:250px; left:0px;
6508 opacity:1.0;
6509 width:100px; height:100px;
6510 color:#fff;
6511 color:#fff; background-color:rgba(0,0,0,0.0);
6512 color:#fff; font-size:16px; font-family:Georgia;
6513 background-repeat:no-repeat;
6514 }
6515 #xxxGStat{
6516 z-index:8;
6517 opacity:0.0;
6518 position:fixed; top:20px; left:0px;
6519 width:640px;
6520 width:100%; height:90px;
6521 color:#fff; background-color:rgba(0,0,128,0.04);
6522 font-size:20px; font-family:Georgia;
6523 }
6524 #GLog{
6525 z-index:10;
6526 position:fixed; top:50px; left:0px;
6527 opacity:1.0;
6528 width:640px; height:60px;
6529 color:#fff; background-color:rgba(0,0,128,0.10);
6530 font-size:12px;
6531 }
6532 #GshGrid {
6533 z-index:11;
6534 opacity:0.0;
6535 position:fixed; top:0px; left:0px;
6536 width:320px; height:30px;
6537 color:#9f9; font-size:16px;
6538 }
6539 xbody {display:none;}
6540 .gsh-link{color:green;}
6541 #gsh {border-width:1px;margin:0;padding:0;}
6542 #gsh {font-family:monospace,Courier New;color:#ddd;font-size:8px;}
6543 #gsh header{height:100px;}
6544 #xgsh header{height:100px;background-image:url(GShell-Logo00.png);}
6545 #GshMenu{font-size:14pt;color:#c44;}
6546 .GshMenu{font-size:14pt;color:#c44;}
6547 .GshMenu{
6548 font-size:14pt;color:#2a2;padding:4px;text-align:right;

```



```

7080 width:166px; height:20px;
7081 font-family:monospace;
7082 font-size:9pt;
7083 line-height:1.0;
7084 color:#fff; background-color:rgba(0,0,64,0.2);
7085 text-align:center;
7086 vertical-align:middle;
7087 }
7088 .GJIcon{
7089 display:inline;
7090 position:relative;
7091 top:0px; left:1px;
7092 border:2px solid #444;
7093 margin:0px; padding:1px;
7094 width:13.2; height:13.2px;
7095 border-radius:2px;
7096 font-family:Georgia;
7097 font-size:13.2px;
7098 line-height:1.0;
7099 white-space:nowrap;
7100 color:#fff; background-color:rgba(32,32,160,0.8);
7101 text-align:center;
7102 vertical-align:middle;
7103 text-shadow:0px 0px;
7104 }
7105 .GJText:focus{
7106 color:#fff !important;
7107 background-color:rgba(32,32,160,0.8) !important;
7108 line-height:1.0;
7109 }
7110 .GJText{
7111 display:inline;
7112 position:relative;
7113 top:0px; left:0px;
7114 border:0px solid #000; margin:0px; padding:0px;
7115 width:280px; height:160px;
7116 border:0px;
7117 font-family:Courier New,monospace !important;
7118 font-size:8pt;
7119 line-height:1.0;
7120 white-space:pre;
7121 color:#fff; xbackground-color:rgba(0,0,64,0.5);
7122 background-color:rgba(32,32,128,0.8) !important;
7123 }
7124 .GJMode{
7125 display:inline;
7126 position:relative;
7127 top:0px; left:0px;
7128 border:0px solid #000; border-radius:0px;
7129 margin:0px; padding:0px;
7130 width:280px; height:20px;
7131 font-size:9pt;
7132 line-height:1.0;
7133 white-space:nowrap;
7134 color:#fff; background-color:rgba(0,0,64,0.7);
7135 text-align:left;
7136 vertical-align:middle;
7137 }
7138 </style>
7139
7140 <script id="gsh-script">
7141 // 2020-0909 added, permanet local storage
7142 // https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
7143 var MyHistory
7144 Permanent = localStorage;
7145 MyHistory = Permanent.getItem('MyHistory')
7146 if( MyHistory == null ){ MyHistory = "" }
7147 d = new Date()
7148 MyHistory = d.getTime()/1000+ " "+document.URL+"\n" + MyHistory
7149 Permanent.setItem('MyHistory',MyHistory)
7150 //Permanent.setItem('MyWindow',window)
7151
7152 var GJLog_Win = null
7153 var GJLog_Tab = null
7154 var GJLog_Stat = null
7155 var GJLog_Text = null
7156 var GJWin_Mode = null
7157 var FProductInterval = 0
7158
7159 var GJ_FactoryID = -1
7160 var GJFactory = null
7161 if( e = document.getElementById('GJFactory_0') ){
7162 GJFactory_1.height = 0
7163 GJFactory = e
7164 e.setAttribute('class','GJFactory')
7165 var GJ_FactoryID = 0
7166 }else{
7167 GJFactory = GJFactory_1
7168 var GJ_FactoryID = 1
7169 }
7170
7171 function GJFactory_Destroy(){
7172 gjf = GJFactory
7173 //gjf = document.getElementById('GJFactory')
7174 //alert('gjf='+gjf)
7175 if( gjf != null ){
7176 if( gjf.childNodes != null ){
7177 for( i = 0; i < gjf.childNodes.length; i++ ){
7178 } gjf.removeChild(gjf.childNodes[i])
7179 }
7180 }
7181 gjf.innerHTML = ''
7182 gjf.style.width = 0
7183 gjf.style.height = 0
7184 gjf.removeAttribute('style')
7185 GJLog_Win = GJLog_Tab = GJLog_Stat = GJLog_Text = GJWin_Mode = null
7186 window.clearInterval(FProductInterval)
7187 return '-- Destroy: work product destroyed'
7188 }else{
7189 return '-- Destroy: work product not exist'
7190 }
7191 }
7192
7193 var TransMode = false
7194 var onKeyControl = false
7195 var onKeyShift = false
7196 var onKeyAlt = false
7197 var onKeyJ = false
7198 var onKeyK = false
7199 var onKeyL = false
7200
7201 function GJWin_OnKeyUp(ev){
7202 keycode = ev.code;
7203 if( keycode == 'ShiftLeft' ){
7204 onKeyShift = false
7205 }else
7206 if( keycode == 'ControlLeft' ){
7207 onKeyControl = false
7208 }else
7209 if( keycode == 'AltLeft' ){
7210 onKeyAlt = false
7211 }else
7212 if( keycode == 'KeyJ' ){ onKeyJ = false }else
7213 if( keycode == 'KeyK' ){ onKeyK = false }else
7214 if( keycode == 'KeyL' ){ onKeyL = false }else
7215 {
7216 }
7217 ev.preventDefault()
7218 }
7219 function and(a,b){ if(a){ if(b){ return true; } return false; } }
7220 function GJWin_OnKeyDown(ev){
7221 keycode = ev.code;
7222 mode = ''
7223 key = ''
7224 if( keycode == 'ControlLeft' ){
7225 onKeyControl = true
7226 ev.preventDefault()
7227 return;
7228 }else
7229 if( keycode == 'ShiftLeft' ){
7230 onKeyShift = true
7231 ev.preventDefault()
7232 return;
7233 }else
7234 if( keycode == 'AltLeft' ){
7235 ev.preventDefault()
7236 onKeyAlt = true
7237 return;
7238 }else
7239 if( keycode == 'Backquote' ){
7240 TransMode = !TransMode
7241 ev.preventDefault()
7242 }else
7243 if( and(keycode == 'Space', onKeyShift) ){
7244 TransMode = !TransMode
7245 ev.preventDefault()
7246 }else
7247 if( keycode == 'ShiftRight' ){
7248 TransMode = !TransMode
7249 }else
7250 if( keycode == 'Escape' ){
7251 TransMode = true
7252 ev.preventDefault()
7253 }else
7254 if( keycode == 'Enter' ){
7255 TransMode = false
7256 //ev.preventDefault()

```

```

7257 }
7258 if( keycode == 'KeyJ' ){ OnKeyJ = true }else
7259 if( keycode == 'KeyK' ){ OnKeyK = true }else
7260 if( keycode == 'KeyL' ){ OnKeyL = true }else
7261 {
7262 }
7263 }
7264 if( ev.altKey ){ key += 'Alt+' }
7265 if( onKeyControl ){ key += 'Ctrl+' }
7266 if( onKeyShift ){ key += 'Shift+' }
7267 if( and(keycode != 'KeyJ', OnKeyJ) ){ key += 'J+' }
7268 if( and(keycode != 'KeyK', OnKeyK) ){ key += 'K+' }
7269 if( and(keycode != 'KeyL', OnKeyL) ){ key += 'L+' }
7270 key = keycode
7271 }
7272 if( TransMode ){
7273 //mode = "[343]201|202;"
7274 JaUtf8 = new Uint8Array([0343,0201,0202]);
7275 utf8dec = new TextDecoder();
7276 Ja = utf8dec.decode(JaUtf8);
7277 mode = '[' + Ja + 'r';
7278 }
7279 }else{
7280 mode = '[-]'
7281 }
7282 }
7283 //[[[ /gmode.innerHTML = "[-]"
7284 GJWin_Mode.innerHTML = mode + ' ' + key
7285 //alert( 'Key: '+keycode)
7286 ev.stopPropagation()
7287 //ev.preventDefault()
7288 }
7289 function GJWin_OnScroll(ev){
7290 x = DragStartX = gsh.getBoundingClientRect().left.toFixed(0)
7291 y = DragStartY = gsh.getBoundingClientRect().top.toFixed(0)
7292 GJLog_append( 'OnScroll: x='+x+',y='+y)
7293 }
7294 document.addEventListener( 'scroll',GJWin_OnScroll)
7295 function GJWin_OnResize(ev){
7296 w = window.innerWidth
7297 h = window.innerHeight
7298 GJLog_append( 'OnResize: w='+w+',h='+h)
7299 }
7300 window.addEventListener( 'resize',GJWin_OnResize)
7301 }
7302 var DragStartX = 0
7303 var DragStartY = 0
7304 function GJWin_DragStart(ev){
7305 // maybe this is the grabbing position
7306 this.style.position = 'fixed'
7307 x = DragStartX = this.getBoundingClientRect().left.toFixed(0)
7308 y = DragStartY = this.getBoundingClientRect().top.toFixed(0)
7309 GJLog_Stat.value = 'DragStart: x='+x+',y='+y
7310 }
7311 function GJWin_Drag(ev){
7312 x = ev.clientX; y = ev.clientY // x = ev.pageX; y = ev.pageY
7313 this.style.left = x - DragStartX
7314 this.style.top = y - DragStartY
7315 this.style.zIndex = '30000'
7316 this.style.position = 'fixed'
7317 x = this.getBoundingClientRect().left.toFixed(0)
7318 y = this.getBoundingClientRect().top.toFixed(0)
7319 GJLog_Stat.value = 'x='+x+',y'+y
7320 ev.preventDefault()
7321 ev.stopPropagation()
7322 }
7323 function GJWin_DragEnd(ev){
7324 x = ev.clientX; y = ev.clientY
7325 //x = ev.pageX; y = ev.pageY
7326 this.style.left = x - DragStartX
7327 this.style.top = y - DragStartY
7328 this.style.zIndex = '30000'
7329 this.style.position = 'fixed'
7330 if( true ){
7331 console.log( "Dropped: "+this.nodeName+'#'+this.id+' x='+x+' y'+y
7332 +', parent'+this.parentNode.id)
7333 }
7334 x = this.getBoundingClientRect().left.toFixed(0)
7335 y = this.getBoundingClientRect().top.toFixed(0)
7336 GJLog_Stat.value = 'x='+x+',y'+y
7337 ev.preventDefault()
7338 ev.stopPropagation()
7339 }
7340 function GJWin_DragIgnore(ev){
7341 ev.preventDefault()
7342 ev.stopPropagation()
7343 }
7344 // 2020-09-15 let every object have console view
7345 var GJ_ConsoleID = 0
7346 var PrevReport = new Date()
7347 function GJLog_StatUpdate(){
7348 txa = GJLog_Stat;
7349 if( txa == null ){
7350 return;
7351 }
7352 tmlap0 = new Date();
7353 p = txa.parentNode;
7354 pw = txa.getBoundingClientRect().width;
7355 ph = txa.getBoundingClientRect().height;
7356 //txa.value += '#'+p.id+' pw='+pw+' ph='+ph+'\n';
7357 txl = '#'+p.id+' pw'+pw+' ph'+ph'\n';
7358 w = txa.getBoundingClientRect().width;
7359 h = txa.getBoundingClientRect().height;
7360 //txa.value += 'w'+w+', h'+h+'\n';
7361 txl += 'w'+w+', h'+h+'\n';
7362 //txa.value += '\n';
7363 //txa.value += DateShort() + '\n';
7364 txl += '\n';
7365 txl += DateShort() + '\n';
7366 tmlap1 = new Date();
7367 tmlap1 = new Date();
7368 txa.value += txl;
7369 tmlap2 = new Date();
7370 }
7371 }
7372 // vertical centering of the last line
7373 sHeight = txa.scrollHeight - 30; // depends on the font-size
7374 tmlap3 = new Date();
7375 }
7376 txa.scrollTop = sHeight; // depends on the font-size
7377 tmlap4 = new Date();
7378 }
7379 now = tmlap0.getTime();
7380 if( PrevReport == 0 || 10000 <= now-PrevReport ){
7381 PrevReport = now;
7382 console.log( "StatBarUpdate: "
7383 + ' leng=' + txa.value.length + ' byte, '
7384 + ' times' + (tmlap1 - tmlap0) + ' ms {
7385 + ' tadd=' + (tmlap2 - tmlap1) + ', '
7386 + ' hcal=' + (tmlap3 - tmlap2) + ', '
7387 + ' scri=' + (tmlap4 - tmlap3) + ' }'
7388 );
7389 }
7390 }
7391 GJWin_StatUpdate = GJLog_StatUpdate;
7392 function GJ_showFimeI(wid){
7393 //e = document.getElementById(wid);
7394 //console.log(wid.id+'.value.length'+wid.value.length)
7395 if( e != null ){
7396 //e.value = DateShort();
7397 }
7398 }
7399 // should remove the Listener
7400 }
7401 }
7402 function GJWin_OnResizeTextarea(ev){
7403 this.value += 'resized: ' + '\n'
7404 }
7405 }
7406 function GJ_NewConsole(wname){
7407 wid = wname + ' ' + GJ_ConsoleID
7408 GJ_ConsoleID += 1
7409 }
7410 GJFactory.style.setProperty( 'width',360+'px' ); //GJFsize
7411 GJFactory.style.setProperty( 'height',320+'px' )
7412 e = GJFactory;
7413 console.log( 'GJFA #'+e.id+' from w='+e.style.width+', h'+e.style.height)
7414 }
7415 if( GJFactory.innerHTML == "" ){
7416 GJFactory.innerHTML = '<'+H3>GJ_Factory_'+ GJ_FactoryID +'<'+H3><'+hr>\n'
7417 }
7418 }else{
7419 GJFactory.innerHTML += '<'+hr>\n'
7420 }
7421 }
7422 gJwin = GJLog_Min = document.createElement( 'span' )
7423 gJwin.id = wid
7424 gJwin.setAttribute( 'class', 'GJWin' )
7425 gJwin.setAttribute( 'draggable', 'true' )
7426 gJwin.addEventListener( 'dragstart', GJWin_DragStart )
7427 gJwin.addEventListener( 'drag', GJWin_Drag )
7428 gJwin.addEventListener( 'dragend', GJWin_Drag )
7429 gJwin.addEventListener( 'dragover', GJWin_DragIgnore )
7430 gJwin.addEventListener( 'dragenter', GJWin_DragIgnore )
7431 gJwin.addEventListener( 'dragleave', GJWin_DragIgnore )
7432 gJwin.addEventListener( 'dragexit', GJWin_DragIgnore )
7433 gJwin.addEventListener( 'drop', GJWin_DragIgnore )
7434 gJwin.addEventListener( 'keydown', GJWin_OnKeyDown )
7435 }
7436 }
7437 gJtab = GJLog_Tab = document.createElement( 'textarea' )

```

```

7434 gjtab.addEventListener('keydown',GJWin_OnKeyDown)
7435 gjtab.style.readonly = true
7436 gjtab.contenteditable = false
7437 gjtab.value = wid
7438 gjtab.id = wid + ' Tab'
7439 gjtab.setAttribute('class','GJTab')
7440 gjtab.setAttribute('spellcheck','false')
7441 gjwin.appendChild(gjtab)
7442
7443 gjstat = GJLog_Stat = document.createElement('textarea')
7444 gjstat.addEventListener('keydown',GJWin_OnKeyDown)
7445 gjstat.id = wid + ' Stat'
7446 gjstat.value = DateShort()
7447 gjstat.setAttribute('class','GJStat')
7448 gjstat.setAttribute('spellcheck','false')
7449 gjwin.appendChild(gjstat)
7450
7451 gjicon = document.createElement('span')
7452 gjicon.addEventListener('keydown',GJWin_OnKeyDown)
7453 gjicon.id = wid + ' Icon'
7454 gjicon.innerHTML = '<font color="#f4">J</font>'
7455 gjicon.setAttribute('class','GJIcon')
7456 gjicon.setAttribute('spellcheck','false')
7457 gjwin.appendChild(gjicon)
7458
7459 gjtext = GJLog_Text = document.createElement('textarea')
7460 gjtext.addEventListener('keydown',GJWin_OnKeyDown)
7461 gjtext.addEventListener('keyup',GJWin_OnKeyUp)
7462 gjtext.addEventListener('resize',GJWin_OnResizeTextarea)
7463 gjtext.id = wid + ' Text'
7464 gjtext.setAttribute('class','GJText')
7465 gjtext.setAttribute('spellcheck','false')
7466 gjwin.appendChild(gjtext)
7467
7468 // user's mode as of IME
7469 gjmode = GJWin_Mode = document.createElement('textarea')
7470 gjmode.addEventListener('keydown',GJWin_OnKeyDown)
7471 gjmode.innerHTML = '<div style="border: 1px solid black; padding: 2px; width: 100px; height: 1em;">
7472 gjmode.id = wid + ' Mode'
7473 gjmode.setAttribute('class','GJMode')
7474 gjmode.setAttribute('spellcheck','false')
7475 gjmode.innerHTML = '---'
7476 gjwin.appendChild(gjmode)
7477
7478 gjwin.zIndex = 30000
7479 GJFactory.appendChild(gjwin)
7480
7481 gjtab.scrollTop = 0
7482 gjstat.scrollTop = 0
7483
7484 //x = gjwin.getBoundingClientRect().left.toFixed(0)
7485 //y = gjwin.getBoundingClientRect().top.toFixed(0)
7486 //gjwin.style.position = 'static'
7487 //gjwin.style.left = 0
7488 //gjwin.style.top = 0
7489
7490 //update = '{wid+'.value=DateShort()}',
7491 update = '{GJ_showTime("wid")}',
7492 // 2020-09-19 this causes memory leaks
7493 //ProductInterval = window.setInterval(update,200)
7494 //ProductInterval = window.setInterval(GJWin_StatUpdate,200)
7495 //ProductInterval = window.setInterval(GJ_showTime,200,wid);
7496 //ProductInterval = window.setInterval(GJ_showTime,200,gjstat);
7497 return update
7498 }
7499
7500 function xxxGJF_StripClass(){
7501 GJLog_Win.style.removeProperty('width')
7502 GJLog_Tab.style.removeProperty('width')
7503 GJLog_Stat.style.removeProperty('width')
7504 GJLog_Text.style.removeProperty('width')
7505 return "Stripped classes"
7506 }
7507 function isElem(id){
7508 return document.getElementById(id) != null
7509 }
7510 function GJLog_append(...args){
7511 txt = GJLog_Text;
7512 if( txt == null ){
7513 return; // maybe GJLog element is removed
7514 }
7515 logs = args.join(' ');
7516 txt.value += logs + '\n'
7517 txt.scrollTop = txt.scrollHeight
7518 //GJLog_Stat.value = DateShort()
7519 }
7520 //window.addEventListener('time',GJLog_StatUpdate)
7521 function test_GJ_Console(){
7522 window.setInterval(GJLog_StatUpdate,1000);
7523 GJ_NewConsole('GJ_Console')
7524 e = GJFactory;
7525 console.log('GJF0 #' + e.id + ' from w=' + e.style.width + ', h=' + e.style.height)
7526 e.style.width = 360; //GJFsize
7527 e.style.height = 320;
7528 console.log('GJF0 #' + e.id + ' to w=' + e.style.width + ', h=' + e.style.height)
7529 }
7530 // test_GJ_Console();
7531
7532 var StopConsoleLog = true
7533 // 2020-09-15 added,
7534 // log should be saved to permanent memory
7535 // const px = new Proxy(console.log,{ alert() })
7536 __console_log = console.log
7537 __console_info = console.info
7538 __console_warn = console.warn
7539 __console_error = console.error
7540 console.exception = console.exception
7541 // should pop callstack info.
7542 console.exception = function(...args){
7543 console.exception(...args)
7544 alert('-- got console.exception("+args+")')
7545 }
7546 console.error = function(...args){
7547 console.error(...args)
7548 alert('-- got console.error("+args+")')
7549 }
7550 console.warn = function(...args){
7551 console.warn(...args)
7552 alert('-- got console.warn("+args+")')
7553 }
7554 console.info = function(...args){
7555 console.info(...args)
7556 alert('-- got console.info("+args+")')
7557 }
7558 __console_info(...args)
7559 console.xxxlog = function(...args){ // rewrite xxxlog to log to enable it
7560 console_log(...args)
7561 if( StopConsoleLog ){
7562 return;
7563 }
7564 if( 0 <= args[0].indexOf('!') ){
7565 //alert('-- got console.log("+args+")')
7566 }
7567 GJLog_append(...args)
7568 }
7569
7570 //document.getElementById('GshFaviconURL').href = GShellFavicon
7571 //document.getElementById('GshFaviconURL').href = GShellInsideIcon
7572 //document.getElementById('GshFaviconURL').href = ITSmoreQR
7573 //document.getElementById('GshFaviconURL').href = GShellLogo
7574
7575 // id of GShell HTML elements
7576 var E_BANNER = "GshBanner" // banner element in HTML
7577 var E_FOOTER = "GshFooter" // footer element in HTML
7578 var E_GINDEX = "gsh-gindex" // index of Golang code of GShell
7579 var E_GOCODE = "gsh-gocode" // Golang code of GShell
7580 var E_TODO = "gsh-todo" // TODO of GShell
7581 var E_DICT = "gsh-dict" // Dictionary of GShell
7582
7583 function bannerElem(){ return document.getElementById(E_BANNER); }
7584 function bannerStyleFunc(){ return bannerElem().style; }
7585
7586 function GshSetImages(){
7587 document.getElementById('GshFaviconURL').href = GShellInsideIcon
7588 bannerStyle.backgroundImage = "url("+GShellLogo+")";
7589 //bannerStyle.backgroundImage = "url("+GShellInsideIcon+")";
7590 //bannerStyle.backgroundImage = "url("+GShellFavicon+")";
7591 //GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7592 //showFooter();
7593 }
7594
7595 function GshInsideIconSetup(){
7596 GMenu.style.backgroundImage = "url("+GShellInsideIcon+")";
7597 GMenu.style.zIndex = 10000000;
7598 //GMenu.style.left = window.innerWidth - 100
7599 GMenu.style.left = 0;
7600 GMenu.style.top = window.innerHeight - 90; // - 200
7601 window.addEventListener('resize',GshInsideIconSetup);
7602 }
7603
7604 function footerElem(){ return document.getElementById(E_FOOTER); }
7605 function footerStyle(){ return footerElem().style; }
7606 //footerElem().style.backgroundImage="url("+ITSmoreQR+")";
7607 //footerStyle().backgroundImage = "url("+ITSmoreQR+")";
7608
7609 function html_fold(e){
7610 if( e.innerHTML == "Fold" ){
7611 e.innerHTML = "Unfold"
7612 document.getElementById('gsh-menu-exit').innerHTML=""
7613 }
7614 }

```

```

7611 document.getElementById('GshStatement').open=false
7612 GshFeatures.open = false
7613 document.getElementById('html-src').open=false
7614 document.getElementById(E_GINDEX).open=false
7615 document.getElementById(E_GOCODE).open=false
7616 document.getElementById(E_TODO).open=false
7617 document.getElementById('References').open=false
7618 }else{
7619 e.innerHTML = "Fold"
7620 document.getElementById('GshStatement').open=true
7621 GshFeatures.open = true
7622 document.getElementById(E_GINDEX).open=true
7623 document.getElementById(E_GOCODE).open=true
7624 document.getElementById(E_TODO).open=true
7625 document.getElementById('References').open=true
7626 }
7627 }
7628 function html_pure(e){
7629 if( e.innerHTML == "Pure" ){
7630 document.getElementById('gsh').style.display=true
7631 //document.style.display = false
7632 e.innerHTML = "Unpure"
7633 }else{
7634 document.getElementById('gsh').style.display=false
7635 //document.style.display = true
7636 e.innerHTML = "Pure"
7637 }
7638 }
7639 }
7640 var bannerIsStopping = false
7641 //NOTE: .com/jsref/prop_style_backgroundposition.asp
7642 function shiftBG(){
7643 bannerIsStopping = !bannerIsStopping
7644 bannerStyle.backgroundPosition = "0 0";
7645 }
7646 // status should be inherited on Window Fork(), so use the status in DOM
7647 function html_stop(e,toggle){
7648 if( toggle ){
7649 if( e.innerHTML == "Stop" ){
7650 bannerIsStopping = true
7651 e.innerHTML = "Start"
7652 }else{
7653 bannerIsStopping = false
7654 e.innerHTML = "Stop"
7655 }
7656 }else{
7657 // update JavaScript variable from DOM status
7658 if( e.innerHTML == "Stop" ){ // shown if it's running
7659 bannerIsStopping = false
7660 }else{
7661 bannerIsStopping = true
7662 }
7663 }
7664 }
7665 html_stop(document.getElementById('GshMenuStop'),false) // onInit.
7666 //html_stop(bannerElem(),false) // onInit.
7667 }
7668 //https://www.w3schools.com/jsref/met_win_setinterval.asp
7669 var banNshift = 0;
7670 function console_log(str){
7671 //console.log(str);
7672 }
7673 function shiftBanner(){
7674 var now = new Date().getTime();
7675 bpos = ((now/10)\10000).toFixed(0)+"px" + " 0px";
7676 if( !bannerIsStopping ){
7677 bannerStyle.backgroundPosition = bpos;
7678 //GshBanner.style.setProperty('background-position',bpos,'important');
7679 banNshift += 1;
7680 console.log('shiftBanner <'+GshBanner.nodeName+'> '+banNshift
7681 + " now="+now+";
7682 //+ ' stop='+bannerIsStopping
7683 + ' pos='+bpos
7684 + ' -> '+bannerStyle.backgroundPosition);
7685 }
7686 }
7687 function Banner_init(){
7688 console.log("-- Banner Shift init.");
7689 window.setInterval(shiftBanner,10); // onInit.
7690 }
7691 }
7692 // <a href="https://developer.mozilla.org/ja/docs/Web/API/Window/open">window.open()</a>
7693 // from embedded html to standalone page
7694 var MyChildren = 0
7695 function html_fork(){
7696 ResetPerfMon();
7697 ResetAffView();
7698 Reset_ShadingCanvas();
7699 GIFactory_Destroy();
7700 MyChildren += 1
7701 WinId = document.getElementById('gsh-WinId').innerHTML + "." + MyChildren;
7702 newwin = window.open("",WinId,"");
7703 src = document.getElementById("gsh");
7704 srctml = src.outerHTML
7705 newwin.document.write("/<"+html>\n");
7706 newwin.document.write(srctml);
7707 newwin.document.write("<"+html>\n");
7708 newwin.document.getElementById('gsh-menu-exit').innerHTML = "Close";
7709 newwin.document.getElementById('gsh-WinId').innerHTML = WinId;
7710 newwin.document.close();
7711 newwin.focus();
7712 }
7713 function html_close(){
7714 window.close()
7715 }
7716 function win_jump(win){
7717 //win = window;
7718 win = window.opener; // https://developer.mozilla.org/ja/docs/Web/API/window/opener
7719 if( win == null ){
7720 console.log("jump to window.opener("+win+") (Error)\n")
7721 }else{
7722 console.log("jump to window.opener("+win+")\n")
7723 win.focus();
7724 }
7725 }
7726 }
7727 // 0.2.9 2020-0902 created checksum of HTML
7728 CRC32UNIX = 0x04c1db7 // Unix cksum
7729 function byteCRC32add(bigcrc,octstr,octlen){
7730 var crc = new Int32Array(1)
7731 crc[0] = bigcrc
7732 }
7733 let oi = 0
7734 for( ; oi < octlen; oi++){
7735 var oct = new Int8Array(1)
7736 oct[0] = octstr[oi]
7737 for( bi = 0; bi < 8; bi++){
7738 //console.log("----CRC32 "+crc[0]+" "+oct[0].toString(16)+" "+oi+"."+bi+"\n")
7739 ovf1 = crc[0] < 0 ? 1 : 0
7740 ovf2 = oct[0] < 0 ? 1 : 0
7741 ovf = ovf1 ^ ovf2
7742 oct[0] <<= 1
7743 crc[0] <<= 1
7744 if( ovf ){ crc[0] ^= CRC32UNIX }
7745 }
7746 }
7747 //console.log("----CRC32 byteAdd return crc="+crc[0]+" "+oi+"/"+octlen+"\n")
7748 return crc[0];
7749 }
7750 function strCRC32add(bigcrc,stri,strlen){
7751 var crc = new Uint32Array(1)
7752 crc[0] = bigcrc
7753 var code = new Uint8Array(strlen);
7754 for( i = 0; i < strlen; i++){
7755 code[i] = stri.charCodeAt(i) // not charAt() !!!!
7756 //console.log("== "+code[i].toString(16)+" <== "+stri[i]+" \n")
7757 }
7758 crc[0] = byteCRC32add(crc,code,strlen)
7759 //console.log("----CRC32 strAdd return crc="+crc[0]+" \n")
7760 return crc[0]
7761 }
7762 function byteCRC32end(bigcrc,len){
7763 var crc = new Uint32Array(1)
7764 crc[0] = bigcrc
7765 var slen = new Uint8Array(4)
7766 let li = 0
7767 for( ; li < 4; ){
7768 slen[li] = len
7769 li += 1
7770 len >>= 8
7771 if( len == 0 ){
7772 break
7773 }
7774 }
7775 crc[0] = byteCRC32add(crc[0],slen,li)
7776 crc[0] = 0xFFFFFFFF
7777 return crc[0]
7778 }
7779 function strCRC32(stri,len){
7780 var crc = new Uint32Array(1)
7781 crc[0] = 0
7782 crc[0] = strCRC32add(0,stri,len)
7783 crc[0] = byteCRC32end(crc[0],len)
7784 //console.log("----CRC32 "+crc[0]+" "+len+"\n")
7785 return crc[0]
7786 }
7787 }

```

```

7788 DestroyGJLink = null; // to be replaced
7789 DestroyFooter = null; // to be defined
7790 DestroyEventSharingCodeview = function dummy(){}
7791 Destroy_VirtualDesktop = function(){}
7792 DestroyNaviButtons = function(){}
7793
7794 function getSourceText(){
7795     if( DestroyFooter != null ) DestroyFooter();
7796     version = document.getElementById('GshVersion').innerHTML
7797     sfavicon = document.getElementById('GshFaviconURL').href;
7798     sbanner = document.getElementById('GshBanner').style.backgroundImage;
7799     spositi = document.getElementById('GshBanner').style.backgroundPosition;
7800
7801     if( document.getElementById('GJC_1') != null ){ GJC_1.remove() }
7802     if( DestroyGJLink != null ) DestroyGJLink();
7803     DestroyEventSharingCodeview();
7804     Destroy_VirtualDesktop();
7805     GshTopbar.innerHTML = "";
7806     DestroyIndexBar();
7807     DestroyNaviButtons();
7808     ResetPerMon();
7809     ResetAffView();
7810     Reset_ShadingCanvas();
7811
7812     // these should be removed by CSS selector or class, after seaved to non-printed attribute
7813     GshBanner.removeAttribute('style');
7814     document.getElementById('GshMenuSign').removeAttribute("style");
7815     styleMenu = GMenu.getAttribute("style");
7816     GMenu.removeAttribute("style");
7817     styleGStat = GStat.getAttribute("style");
7818     GStat.removeAttribute("style");
7819     styleGTop = GTop.getAttribute("style");
7820     GTop.removeAttribute("style");
7821     styleGshGrid = GshGrid.getAttribute("style");
7822     GshGrid.removeAttribute("style");
7823     //styleGPos = GPos.getAttribute("style");
7824     //GPos.removeAttribute("style");
7825     //GPos.innerHTML = "";
7826     //styleGLog = GLog.getAttribute("style");
7827     //GLog.removeAttribute("style");
7828     //GLog.innerHTML = "";
7829     styleShellPlane = GShellPlane.getAttribute("style");
7830     GShellPlane.removeAttribute("style");
7831     styleRawTextViewer = RawTextViewer.getAttribute("style");
7832     RawTextViewer.removeAttribute("style");
7833     styleRawTextViewClose = RawTextViewClose.getAttribute("style");
7834     RawTextViewClose.removeAttribute("style");
7835
7836     GshFaviconURL.href = "";
7837     if( !selem('ConfigIcon') ) ConfigIcon.src = "";
7838
7839     //it seems that interHTML and outerHTML generate style="" for these (??)
7840     //GshBanner.removeAttribute("style");
7841     //GshFooter.removeAttribute("style");
7842     //GshMenuSign.removeAttribute("style");
7843     GshBanner.style=""
7844     GshMenuSign.style=""
7845
7846     textarea = document.createElement("textarea")
7847     srchtml = document.getElementById("gsh").outerHTML;
7848     //textarea = document.createElement("textarea")
7849     // 2020-0910 ?? ... this causes inserting style="" to Banner and Footer,
7850     // with Chromium/ after reloading from file:///
7851     textarea.innerHTML = srchtml
7852     // <a href="https://stackoverflow.com/questions/5796718/html-entity-decode">Thanks</a>
7853     var rawtext = textarea.value
7854     //textarea.destroy()
7855     //rawtext = gsh.textContent // this removes #include <FILENAME> too
7856     var orgtext = ""
7857     + /<.*html>\n" // lost preamble text
7858     + rawtext
7859     + "<.*html>\n" // lost trail text
7860     ;
7861
7862     tlen = orgtext.length
7863     //console.log("getSourceText: length="+tlen+"\n")
7864     document.getElementById('GshFaviconURL').href = sfavicon;
7865
7866     document.getElementById('GshBanner').style.backgroundImage = sbanner;
7867     document.getElementById('GshBanner').style.backgroundPosition = spositi;
7868
7869     GStat.setAttribute("style",styleGStat)
7870     GMenu.setAttribute("style",styleGMenu)
7871     GTop.setAttribute("style",styleGTop)
7872     //GLog.setAttribute("style",styleGLog)
7873     //GPos.setAttribute("style",styleGPos)
7874     GshGrid.setAttribute("style",styleGshGrid)
7875     GShellPlane.setAttribute("style",styleGShellPlane)
7876     RawTextViewer.setAttribute("style",styleRawTextViewer)
7877     RawTextViewClose.setAttribute("style",styleRawTextViewClose)
7878     canontext = orgtext.replace(' style=""','')
7879     // open" too
7880     return canontext
7881 }
7882 function getDigest(){
7883     var text = ""
7884     text = getSourceText()
7885     var digest = ""
7886     tlen = text.length
7887     digest = strCRC32(text,tlen) + " " + tlen
7888     return { text, digest }
7889 }
7890 function html_digest(){
7891     version = document.getElementById('GshVersion').innerHTML
7892     let {text, digest} = getDigest()
7893     alert("cksum: " + digest + " " + version)
7894 }
7895 function charsin(str, char){
7896     ln = 0;
7897     for( i = 0; i < str.length; i++ ){
7898         if( str.charCodeAt(i) == char.charCodeAt(0) )
7899             ln++;
7900     }
7901     return ln;
7902 }
7903
7904 //class digestElement extends HTMLElement {
7905 //< script>customElements.define('digest',digestElement)< /script>
7906 function showDigest(e){
7907     result = 'version=' + GshVersion.innerHTML + '\n'
7908     result += 'lines=' + e.dataset.lines + '\n'
7909     result += 'length=' + e.dataset.length + '\n'
7910     result += 'crc32u=' + e.dataset.crc32u + '\n'
7911     result += 'time=' + e.dataset.time + '\n';
7912
7913     alert(result)
7914 }
7915
7916 function html_sign(e){
7917     if( RawTextViewer.style.zIndex == 1000 ){
7918         hideRawTextViewer()
7919         return
7920     }
7921     GshTopbar.innerHTML = "";
7922     ResetPerMon();
7923     ResetAffView();
7924     Reset_ShadingCanvas();
7925     DestroyIndexBar();
7926     DestroyNaviButtons();
7927     DestroyEventSharingCodeview();
7928     Destroy_VirtualDesktop();
7929     GJFactory.Destroy();
7930     if( DestroyGJLink != null ) DestroyGJLink();
7931     //gsh_digest.innerHTML = "";
7932     text = getSourceText() // the original text
7933     tlen = text.length
7934     digest = strCRC32(text,tlen)
7935     //gsh_digest.innerHTML = digest + " " + tlen
7936     //text = getSourceText() // the text with its digest
7937     Lines = charsin(text, '\n')
7938
7939     name = "gsh"
7940     sid = name + "-" + digest
7941     d = new Date()
7942     signedAt = d.getTime()
7943
7944     sign = '/' + '<' + 'span'\n'
7945     + ' id="" + sid + '\n'
7946     + ' class="digest"\n'
7947     + ' data-target-id="" + name + '\n'
7948     + ' data-crc32u="" + digest + '\n'
7949     + ' data-length="" + tlen + '\n'
7950     + ' data-lines="" + Lines + '\n'
7951     + ' data-time="" + signedAt + '\n'
7952     + '>' + '/span>\n' + '\n'
7953
7954     text = sign + text
7955
7956     txhtml = '<' + 'table id="LineNumbered"><' + 'tr><' + 'td'
7957     + '<' + 'textarea cols=5 rows= ' + Lines + ' class="LineNumber">'
7958     for( i = 1; i <= Lines; i++ ){
7959         txhtml += i.toString() + '\n'
7960     }
7961     txhtml += ""
7962     + '<' + '/textarea>'
7963     + '<' + 'td><' + 'td'>'
7964     + '<' + 'textarea cols=150 rows= ' + Lines + ' spellcheck="false"

```

```

7965     + ' class="LineNumbered">'
7966     + text + '<+/'textarea>'
7967     + '<+ /td><+ /tr><+ /table>'
7968
7969     for( i = 1; i <= 30; i++ ){
7970         txhtml += '<br>\n'
7971     }
7972     RawTextViewer.innerHTML = txhtml
7973     RawTextViewer.spellcheck = false // (spellcheck above seems ineffective)
7974
7975     btn = e
7976     e.style.color = "rgba(128,128,255,0.9)";
7977     y = e.getBoundingClientRect().top.toFixed(0)
7978     //h = e.getBoundingClientRect().height.toFixed(0)
7979     RawTextViewer.style.top = Number(y) + 30
7980     RawTextViewer.style.left = 100;
7981     RawTextViewer.style.height = window.innerHeight - 20;
7982     //RawTextViewer.style.Opacity = 1.0;
7983     //RawTextViewer.style.backgroundColor = "rgba(0,0,0,0.0)";
7984     RawTextViewer.style.backgroundColor = "rgba(255,255,255,0.8)";
7985     RawTextViewer.style.zIndex = 1000;
7986     RawTextViewer.style.display = true;
7987
7988     if( RawTextViewerClose.style == null ){
7989         RawTextViewerClose.style = "";
7990     }
7991     RawTextViewerClose.style.top = Number(y) + 10
7992     RawTextViewerClose.style.left = 100;
7993     RawTextViewerClose.style.zIndex = 1001;
7994
7995     ScrollToElement(CurElement,RawTextViewerClose)
7996 }
7997
7998 function hideRawTextViewer(){
7999     RawTextViewer.style.left = 10000;
8000     RawTextViewer.style.zIndex = -100;
8001     RawTextViewer.style.Opacity = 0.0;
8002     RawTextViewer.style = null
8003     RawTextViewer.innerHTML = "";
8004
8005     GshMenuSign.style.color = "rgba(255,128,128,1.0)";
8006     RawTextViewerClose.style.top = 0;
8007     RawTextViewerClose.style = null
8008 }
8009
8010 // source code view
8011 function frame_close(){
8012     srcframe.innerHTML = "";
8013     //srcframe.style.cols = 1;
8014     srcframe.style.rows = 1;
8015     srcframe.style.height = 0;
8016     srcframe.style.display = false;
8017     src = document.getElementById("SrcTextarea");
8018     src.innerHTML = "";
8019     //src.cols = 0
8020     src.rows = 0
8021     src.display = false
8022     //alert("--closed--")
8023 }
8024
8025 <!-- | <span onclick="html_view();">Source</span> -->
8026 <!-- | <span onclick="frame_close();">SourceClose</span> -->
8027 <!-- | <span>Download</span> -->
8028
8029 function frame_open(){
8030     GshTopbar.innerHTML = "";
8031     ResetPerfMon();
8032     ResetAffView();
8033     Reset_ShadingCanvas();
8034     DestroyIndexBar();
8035     DestroyNavButtons();
8036     if( DestroyFooter != null ) DestroyFooter();
8037     document.getElementById("GshFaviconURL").href = "";
8038     oldsrc = document.getElementById("GENSRC");
8039     if( oldsrc != null ){
8040         //alert("--I--(erasing old text)")
8041         oldsrc.innerHTML = "";
8042         return
8043     }else{
8044         //alert("--I--(no old text)")
8045     }
8046     styleBanner = GshBanner.getAttribute("style")
8047     GshBanner.removeAttribute("style")
8048     if( document.getElementById('GJC_1') ){ GJC_1.remove() }
8049
8050     GshFaviconURL.href = "";
8051     if( iselem('ConfigIcon') ) ConfigIcon.src = "";
8052     GStat.removeAttribute('style')
8053     GshGrid.removeAttribute('style')
8054     GshMenuSign.removeAttribute('style')
8055     //GPos.removeAttribute('style')
8056     //GPos.innerHTML = "";
8057     //GLog.removeAttribute('style')
8058     //GLog.innerHTML = "";
8059     GMenu.removeAttribute('style')
8060     GTop.removeAttribute('style')
8061     GShellPlane.removeAttribute('style')
8062     RawTextViewer.removeAttribute('style')
8063     RawTextViewerClose.removeAttribute('style')
8064
8065     if( DestroyGJLink != null ) DestroyGJLink();
8066     GJFactory_Destroy()
8067     Destroy_VirtualDesktop();
8068     DestroyEventSharingCodeview();
8069
8070     src = document.getElementById("gsh");
8071     srhtml = src.outerHTML
8072     srcframe.innerHTML =
8073     + "<+<cite id='GENSRC'>\n"
8074     + "<+<style>\n"
8075     + "#GENSRC textarea{tab-size:4;}\n"
8076     + "#GENSRC textarea{-o-tab-size:4;}\n"
8077     + "#GENSRC textarea{-moz-tab-size:4;}\n"
8078     + "#GENSRC textarea{spellcheck:false;}\n"
8079     + "<+</style>\n"
8080     + "<+<textarea id='SrcTextarea' cols=100 rows=20 class='gsh-code' spellcheck='false'>"
8081     + "<+<html>\n" // lost preamble text
8082     + "<+</html>\n" // lost trail text
8083     + "<+</textarea>\n"
8084     + "<+</cite><!-- GENSRC -->\n";
8085
8086     //srcframe.style.cols = 80;
8087     //srcframe.style.rows = 80;
8088
8089     GshBanner.setAttribute('style',styleBanner)
8090 }
8091
8092 function fill_CSSView(){
8093     part = document.getElementById('GshStyleDef')
8094     view = document.getElementById('gsh-style-view')
8095     view.innerHTML = ""
8096     + "<+<textarea cols=100 rows=20 class='gsh-code'>"
8097     + part.innerHTML
8098     + "<+</textarea>"
8099 }
8100
8101 function fill_JavaScriptView(){
8102     jspart = document.getElementById('gsh-script')
8103     view = document.getElementById('gsh-script-view')
8104     view.innerHTML = ""
8105     + "<+<textarea cols=100 rows=20 class='gsh-code'>"
8106     + jspart.innerHTML
8107     + "<+</textarea>"
8108 }
8109
8110 function fill_DataView(){
8111     part = document.getElementById('gsh-data')
8112     view = document.getElementById('gsh-data-view')
8113     view.innerHTML = ""
8114     + "<+<textarea cols=100 rows=20 class='gsh-code'>"
8115     + part.innerHTML
8116     + "<+</textarea>"
8117 }
8118
8119 function jumpto_StyleView(){
8120     jstyleview = document.getElementById('html-src')
8121     jstyleview.open = true
8122     jstyleview = document.getElementById('gsh-style-frame')
8123     jstyleview.open = true
8124     fill_CSSView()
8125 }
8126
8127 function jumpto_JavaScriptView(){
8128     jscriptview = document.getElementById('html-src')
8129     jscriptview.open = true
8130     jscriptview = document.getElementById('gsh-script-frame')
8131     jscriptview.open = true
8132     fill_JavaScriptView()
8133 }
8134
8135 function jumpto_DataView(){
8136     jdataview = document.getElementById('html-src')
8137     jdataview.open = true
8138     jdataview = document.getElementById('gsh-data-frame')
8139     jdataview.open = true
8140     fill_DataView()
8141 }
8142
8143 function jumpto_WholeView(){
8144     jwholeview = document.getElementById('html-src')
8145     jwholeview.open = true
8146     jwholeview = document.getElementById('gsh-whole-view')
8147     jwholeview.open = true
8148     frame_open()

```

```

8142 }
8143 function html_view(){
8144     html_stop();
8145
8146     banner = document.getElementById('GshBanner').style.backgroundImage;
8147     footer = document.getElementById('GshFooter').style.backgroundImage;
8148     document.getElementById('GshBanner').style.backgroundImage = "";
8149     document.getElementById('GshBanner').style.backgroundPosition = "";
8150     document.getElementById('GshFooter').style.backgroundImage = "";
8151
8152     //srcwin = window.open("", "CodeView2", "");
8153     srcwin = window.open("", "", "");
8154     srcwin.document.write("<span id='gsh'\>\n");
8155
8156     src = document.getElementById("gsh");
8157     srcwin.document.write("<"+style>\n");
8158     srcwin.document.write("textare{tab-size:4;\n");
8159     srcwin.document.write("textare{-tab-size:4;\n");
8160     srcwin.document.write("textare{-moz-tab-size:4;\n");
8161     srcwin.document.write("</style>\n");
8162     srcwin.document.write("<+>\n");
8163     srcwin.document.write("<"+span onclick='window.close();>Close</span> | \n");
8164     //srcwin.document.write("<"+span onclick='html_stop();>Run</span>\n");
8165     srcwin.document.write("</+>\n");
8166     srcwin.document.write("<+textare id='gsh-src-src' cols=100 rows=60>");
8167     srcwin.document.write("<+>\n");
8168     srcwin.document.write("<"+span id='gsh'>");
8169     srcwin.document.write(src.innerHTML);
8170     srcwin.document.write("</+>\n");
8171     srcwin.document.write("</+>");
8172
8173     document.getElementById('GshBanner').style.backgroundImage = banner;
8174     document.getElementById('GshFooter').style.backgroundImage = footer;
8175
8176     sty = document.getElementById("GshStyleDef");
8177     srcwin.document.write("<+>\n");
8178     srcwin.document.write(sty.innerHTML);
8179     srcwin.document.write("<+>\n");
8180
8181     run = document.getElementById("gsh-script");
8182     srcwin.document.write("<+>\n");
8183     srcwin.document.write(run.innerHTML);
8184     srcwin.document.write("<+>\n");
8185
8186     srcwin.document.write("<+>\n"); // gsh span
8187     srcwin.document.close();
8188     srcwin.focus();
8189 }
8190 GSH = document.getElementById("gsh")
8191
8192 //GSH.onclick = "alert('Ouch!')"
8193 //GSH.css = "{background-color:#eef;}"
8194 //GSH.style = "background-color:#eef;"
8195 //GSH.style.display = false;
8196 //alert('Ouch0!')
8197 //GSH.style.display = true;
8198
8199 // 2020-0904 created, tentative
8200 //document.addEventListener('keydown', jgshCommand);
8201 //CurElement = GshStatement
8202 CurElement = GshMenu
8203 MemElement = GshMenu
8204
8205 function nextSib(e){
8206     n = e.nextSibling;
8207     for( i = 0; i < 100; i++){
8208         if( n == null ){
8209             break;
8210         }
8211         if( n.nodeName == "DETAILS" ){
8212             return n;
8213         }
8214         n = n.nextSibling;
8215     }
8216     return null;
8217 }
8218 function prevSib(e){
8219     n = e.previousSibling;
8220     for( i = 0; i < 100; i++){
8221         if( n == null ){
8222             break;
8223         }
8224         if( n.nodeName == "DETAILS" ){
8225             return n;
8226         }
8227         n = n.previousSibling;
8228     }
8229     return null;
8230 }
8231 function setColor(e,eName,eColor){
8232     if( e.hasChildNodes() ){
8233         e = e.childNodes;
8234         if( e != null ){
8235             for( ci = 0; ci < s.length; ci++){
8236                 if( s[ci].nodeName == eName ){
8237                     s[ci].style.color = eColor;
8238                     //s[ci].style.backgroundColor = eColor;
8239                     break;
8240                 }
8241             }
8242         }
8243     }
8244 }
8245
8246 // https://docs.microsoft.com/en-us/previous-versions//hh781509(v=vs.85)
8247 function showCurElementPosition(ev){
8248     // if( document.getElementById("GPos") == null ){
8249     //     return;
8250     // }
8251     // if( GPos == null ){
8252     //     return;
8253     // }
8254     e = CurElement
8255     y = e.getBoundingClientRect().top.toFixed(0)
8256     x = e.getBoundingClientRect().left.toFixed(0)
8257
8258     h = ev + " "
8259     h += 'y='+y+", " + 'x'+x+" -- "
8260     h += "w=" + window.innerWidth + ", h=" + window.innerHeight + " -- "
8261     //GPos.test = h
8262     //GPos.innerHTML = h
8263     // GPos.innerHTML = h
8264 }
8265
8266 function zero2(n){
8267     if( n < 10 ){
8268         return '0' + n;
8269     }else{
8270         return n;
8271     }
8272 }
8273 function DateHourMin(){
8274     d = new Date();
8275     //return '%02d:%02d'.sprintf(d.getHours(),d.getMinutes())
8276     return zero2(d.getHours()) + ":" + zero2(d.getMinutes());
8277 }
8278 function DateShort0(d){
8279     return 'd.getFullYear()
8280         + '/' + zero2(d.getMonth())
8281         + '/' + zero2(d.getDate())
8282         + ' ' + zero2(d.getHours())
8283         + ':' + zero2(d.getMinutes())
8284         + '.' + zero2(d.getSeconds())
8285 }
8286 function DateShort(){
8287     return DateShort0(new Date());
8288 }
8289 function DateLong0(ms){
8290     d = new Date();
8291     d.setTime(ms);
8292     return DateShort0(d)
8293         + '.' + d.getMilliseconds()
8294         + " + " + d.getTimezoneOffset()/60
8295         + " + " + d.getTime() + '.' + d.getMilliseconds()
8296 }
8297 function DateLong(){
8298     return DateLong0(new Date());
8299 }
8300 function GShellMenu(e){
8301     //GLog.innerHTML = "Hello, World! (" + DateLong() + ")"
8302     //showGShellPlane()
8303     ConfigClick();
8304 }
8305 // placements of planes
8306 function GShellResize(ev){
8307     //if( document.getElementById("GMenu") != null ){
8308     //GShellInsideIconSetup();
8309     //GMenu.style.left = window.innerWidth - 100
8310     //GMenu.style.top = window.innerHeight - 90 - 200
8311     //console.log("place GMENU "+GMenu.style.left+" "+GMenu.style.top)
8312
8313     //}
8314     GStat.style.width = window.innerWidth
8315     //if( document.getElementById("GPos") != null ){
8316     //GPos.style.width = window.innerWidth
8317     //GPos.style.top = window.innerHeight - 30; //GPos.style.height
8318     //}

```

```

8319 //if( document.getElementById("GLog") != null ){
8320 // GLog.style.width = window.innerWidth
8321 //GLog.innerHTML = ""
8322 //}
8323 //if( document.getElementById("GLog") != null ){
8324 //GLog.innerHTML = "Resize: w=" + window.innerWidth +
8325 //, h=" + window.innerHeight
8326 //}
8327 showCurElementPosition(ev)
8328 }
8329 function GShellResize(){
8330 GShellResizeX("RESIZE")
8331 }
8332 window.onresize = GShellResize
8333 var prevNode = null
8334 var LogMouseMoveOverElement = false;
8335 function GJSH_OnMouseMove(ev){
8336 if( LogMouseMoveOverElement == false ){
8337 return;
8338 }
8339 x = ev.clientX
8340 y = ev.clientY
8341 d = new Date()
8342 t = d.getTime() / 1000
8343 if( document.elementFromPoint ){
8344 e = document.elementFromPoint(x,y)
8345 if( e != null ){
8346 if( e == prevNode ){
8347 }else{
8348 console.log('Mo-'+t+'('+x+', '+y+') '
8349 +e.nodeType+' '+e.tagName+'#'+e.id)
8350 prevNode = e
8351 }
8352 }else{
8353 console.log(t+'('+x+', '+y+') no element')
8354 }
8355 }else{
8356 console.log(t+'('+x+', '+y+') no elementFromPoint')
8357 }
8358 }
8359 window.addEventListener('mousemove',GJSH_OnMouseMove);
8360
8361 function GJSH_OnMouseMoveScreen(ev){
8362 x = ev.screenX
8363 y = ev.screenY
8364 d = new Date()
8365 t = d.getTime() / 1000
8366 console.log(t+'('+x+', '+y+') no elementFromPoint')
8367 }
8368 //screen.addEventListener('mousemove',GJSH_OnMouseMoveScreen);
8369
8370 function ScrollToElement(oe,ne){
8371 ne.scrollIntoView()
8372 ny = ne.getBoundingClientRect().top.toFixed(0)
8373 nx = ne.getBoundingClientRect().left.toFixed(0)
8374 //GLog.innerHTML = "[+ny+,"+nx+]"
8375 //window.scrollTo(0,0)
8376
8377 GTop.style.backgroundColor = "rgba(0,0,0,0.0)"
8378 GshGrid.style.left = "250px";
8379 GshGrid.style.zIndex = 0
8380 if( false ){
8381 oy = oe.getBoundingClientRect().top.toFixed(0)
8382 ox = oe.getBoundingClientRect().left.toFixed(0)
8383 y = e.getBoundingClientRect().top.toFixed(0)
8384 x = e.getBoundingClientRect().left.toFixed(0)
8385 window.scrollTo(x,y)
8386 ny = e.getBoundingClientRect().top.toFixed(0)
8387 nx = e.getBoundingClientRect().left.toFixed(0)
8388 //GLog.innerHTML = "[+oy+,"+ox+]"->["+y+,"+x+]"->["+ny+,"+nx+]"
8389 }
8390 }
8391 function showGShellPlane(){
8392 if( GShellPlane.style.zIndex == 0 ){
8393 GShellPlane.style.zIndex = 1000;
8394 GShellPlane.style.left = 30;
8395 GShellPlane.style.height = 320;
8396 GShellPlane.innerHTML = DateLong() + "<br>" +
8397 -- History --<br>" + MyHistory;
8398 }else{
8399 GShellPlane.style.zIndex = 0;
8400 GShellPlane.style.left = 0;
8401 GShellPlane.style.height = 50;
8402 GShellPlane.innerHTML = "";
8403 }
8404 }
8405 var SuppressGJShell = false
8406 function jgshCommand(kevent){
8407 if( SuppressGJShell ){
8408 return
8409 }
8410 key = kevent
8411 keycode = key.code
8412 //GStat.style.width = window.innerWidth
8413 GStat.style.backgroundColor = "rgba(0,0,0,0.4)"
8414
8415 console.log("JSGsh-Key:"+keycode+"(^-^)/")
8416 if( keycode == "Slash" ){
8417 console.log('('+x+', '+y+') '
8418 e = document.elementFromPoint(x,y)
8419 console.log('('+x+', '+y+') '+e.nodeType+' '+e.tagName+'#'+e.id)
8420 }else
8421 if( keycode == "Digit0" ){ // fold side-bar
8422 // "Zero page"
8423 showGShellPlane();
8424 }else
8425 if( keycode == "Digit1" ){ // fold side-bar
8426 primary.style.width = "94%"
8427 secondary.style.width = "0%"
8428 secondary.style.opacity = 0
8429 GStat.innerHTML = "[Single Column View]"
8430 }else
8431 if( keycode == "Digit2" ){ // unfold side-bar
8432 primary.style.width = "58%"
8433 secondary.style.width = "36%"
8434 secondary.style.opacity = 1
8435 GStat.innerHTML = "[Double Column View]"
8436 }else
8437 if( keycode == "KeyU" ){ // fold/unfold all
8438 html_fold(GshMenuFold);
8439 location.href = "#"+CurElement.id;
8440 }else
8441 if( keycode == "KeyO" || keycode == "ArrowRight" ){ // fold the element
8442 CurElement.open = 1CurElement.open;
8443 }else
8444 if( keycode == "ArrowRight" ){ // unfold the element
8445 CurElement.open = true
8446 }else
8447 if( keycode == "ArrowLeft" ){ // unfold the element
8448 CurElement.open = false
8449 }else
8450 if( keycode == "KeyI" ){ // inspect the element
8451 e = CurElement
8452 //GLog.innerHTML =
8453 GJLog_append("Current Element: " + e + "<br>"
8454 + "name="+e.nodeName + ", "
8455 + "id="+e.id + ", "
8456 + "children="+e.childNodes.length + ", "
8457 + "parent="+e.parentNode.id + "<br>"
8458 + "text="+e.textContent)
8459 GStat.style.backgroundColor = "rgba(0,0,0,0.8)"
8460 return
8461 }else
8462 if( keycode == "KeyM" ){ // memory the position
8463 MemElement = CurElement
8464 }else
8465 if( keycode == "KeyN" || keycode == "ArrowDown" ){ // next element
8466 e = nextSib(CurElement)
8467 if( e != null ){
8468 setColor(CurElement,"SUMMARY","#fff")
8469 setColor(e,"SUMMARY","#8f8") // should be complement ?
8470 oe = CurElement
8471 CurElement = e
8472 //location.href = "#"+e.id;
8473 ScrollToElement(oe,e)
8474 }
8475 }else
8476 if( keycode == "KeyP" || keycode == "ArrowUp" ){ // previous element
8477 oe = CurElement
8478 e = prevSib(CurElement)
8479 if( e != null ){
8480 setColor(CurElement,"SUMMARY","#fff")
8481 setColor(e,"SUMMARY","#8f8") // should be complement ?
8482 CurElement = e
8483 //location.href = "#"+e.id;
8484 ScrollToElement(oe,e)
8485 }else{
8486 e = document.getElementById("GshBanner")
8487 if( e != null ){
8488 setColor(CurElement,"SUMMARY","#fff")
8489 CurElement = e
8490 ScrollToElement(oe,e)
8491 }else{
8492 e = document.getElementById("primary")
8493 if( e != null ){
8494 setColor(CurElement,"SUMMARY","#fff")
8495 CurElement = e

```



```

8496         ScrollToElement(oe,e)
8497     }
8498 }
8499 }
8500 }else
8501 if( keycode == "KeyR" ){
8502     location.reload();
8503 }else
8504 if( keycode == "KeyJ" ){
8505     GshGrid.style.top = '120px';
8506     GshGrid.innerHTML = '>><(Down)';
8507 }else
8508 if( keycode == "KeyK" ){
8509     GshGrid.style.top = '0px';
8510     GshGrid.innerHTML = '^^(Up)';
8511 }else
8512 if( keycode == "KeyH" ){
8513     GshGrid.style.left = '0px';
8514     GshGrid.innerHTML = '^^(Left)';
8515 }else
8516 if( keycode == "KeyL" ){
8517     //GLog.innerHTML +=
8518     GJLog.append(
8519         'Screen='+screen.width+'px'+<br>'+
8520         'window='+window.innerWidth+'px'+<br>'+
8521         )
8522     GshGrid.style.left = (document.documentElement.clientWidth-160).toString(10)+'px';
8523     GshGrid.innerHTML = '^^(Right)';
8524 }else
8525 if( keycode == "KeyS" ){
8526     html_stop(GshMenuStop,true)
8527 }else
8528 if( keycode == "KeyF" ){
8529     html_fork()
8530 }else
8531 if( keycode == "KeyC" ){
8532     window.close()
8533 }else
8534 if( keycode == "KeyD" ){
8535     html_digest()
8536 }else
8537 if( keycode == "KeyV" ){
8538     e = document.getElementById('gsh-digest')
8539     if( e != null ){
8540         showDigest(e)
8541     }
8542 }
8543 }
8544 showCurElementPosition("[+key.code+" --");
8545 //if( document.getElementById("GPos") != null ){
8546     //GPos.innerHTML += "[+key.code+" --"
8547     //}
8548 //GShellResizeX("[+key.code+" --");
8549 }
8550 var initGSKC = false;
8551 function GShell_initKeyCommands(){
8552     if( initGSKC ){return; } initGSKC = true;
8553 }
8554 GShellResizeX("[INIT]");
8555 DisplaySize = "-- Display: "
8556 + "screen="+screen.width+'px, ' + 'window='+window.innerWidth+'px';
8557
8558 let {text, digest} = getDigest()
8559 //GLog.innerHTML +=
8560 GJLog.append(
8561     "-- GShell: " + GshVersion.innerHTML + '\n' +
8562     "-- Digest: " + digest + '\n' +
8563     DisplaySize
8564     //+ "<br>" + "-- LastVisit:<br>" + MyHistory
8565 )
8566 GShellResizeX(null);
8567 //GShell_initKeyCommands();
8568 }
8569
8570 // <a href="https://www.w3.org/TR/WebCryptoAPI/">Web Cryptography API</a>
8571 //Convert a string into an ArrayBuffer
8572 //from https://developers.google.com/web/updates/2012/06/How-to-convert-ArrayBuffer-to-and-from-String
8573 function strToArray(buf) {
8574     const buf = new ArrayBuffer(str.length);
8575     const bufView = new Uint8Array(buf);
8576     for (let i = 0, strLen = str.length; i < strLen; i++) {
8577         bufView[i] = str.charCodeAt(i);
8578     }
8579     return buf;
8580 }
8581 function importPrivateKey(pem) {
8582     const binaryDerString = window.atob(pemContents);
8583     const binaryDer = str2ab(binaryDerString);
8584     return window.crypto.subtle.importKey(
8585         "pkcs8",
8586         binaryDer,
8587         {
8588             name: "RSA-PSS",
8589             modulusLength: 2048,
8590             publicExponent: new Uint8Array([1, 0, 1]),
8591             hash: "SHA-256",
8592         },
8593         true,
8594         ["sign"]
8595     );
8596 }
8597 //importPrivateKey(ppem)
8598
8599 //key = {}
8600 //buf = "abc"
8601 //enc = "xyzxxxxx"; //crypto.publicEncrypt(key,buf)
8602 //b64 = btoa(enc)
8603 //dec = atob(b64)
8604 //GLog.innerHTML = "enc:" + b64 + ", dec:" + dec
8605 </script>
8606 */
8607
8608 //<!-- ===== Work { ===== -->
8609 //<span id="PackmonGo_WorkCodeSpan">
8610 /*
8611 <details id="PackmonGo_Section"><summary id="PackmonGo_Summary">PackmonGo</summary>
8612 </-- ===== PackmonGo // 2020-1025 SatoxITS <!-- =====
8613 <h2>PackmonGo</h2>
8614 <div id="PackmonGo_1" class="PackmonGo">
8615 <canvas id="PackmonGo_1_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8616 </div>
8617 <div id="PackmonGo_2" class="PackmonGo">
8618 <canvas id="PackmonGo_2_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8619 </div>
8620 <div id="PackmonGo_3" class="PackmonGo">
8621 <canvas id="PackmonGo_3_Canvas" class="PackmonGo_Canvas" width="150px" height="150px" draggable="true"></canvas>
8622 </div>
8623 <div id="PackmonGo_4" class="PackmonGo">
8624 <canvas id="PackmonGo_4_Canvas" class="PackmonGo_Canvas" width="400px" height="400px" draggable="true"></canvas>
8625 </div>
8626 <style>
8627 .PackmonGo {
8628     position:fixed !important;
8629     background-color:rgba(0,0,0,0.0);
8630 }
8631 .PackmonGo_Canvas {
8632     background-color:rgba(0,0,0,0.0);
8633 }
8634 </style>
8635 <script>
8636 var stopPackmonFlag = false;
8637 var PackmonInit = false;
8638 function stopPackmonMon(){
8639     stopPackmonFlag = !stopPackmonFlag;
8640 }
8641 function spawnPackmonGo(){
8642     if( PackmonInit == false ){
8643         pgw = 100;
8644         pgh = 100;
8645         pgo = 0.5;
8646
8647         ctx = PackmonGo_1_Canvas.getContext('2d');
8648         ctx.fillStyle = 'rgba(255,255,0, 'pgow')';
8649         ctx.fillRect(0,0,pgw,pgh);
8650         base = PackmonGo_1;
8651         gsh.appendChild(base);
8652         base.style.zIndex = 1000;
8653         base.style.position = "fixed";
8654         base.style.left = 0 + 'px';
8655         base.style.top = 0 + 'px';
8656         base.xinc = 4;
8657         base.yinc = 4;
8658         base.scrx = 0;
8659         base.scry = 0;
8660         base.pgw = 100;
8661         base.pgh = 100;
8662         movePgWindow(PackmonGo_1);
8663
8664         ctx = PackmonGo_2_Canvas.getContext('2d');
8665         ctx.fillStyle = 'rgba(127,255,127, 'pgow')';
8666         ctx.fillRect(0,0,pgw,pgh);
8667         base = PackmonGo_2;
8668         gsh.appendChild(base);
8669         base.style.zIndex = 1001;
8670         base.style.position = "fixed";
8671         base.style.left = 200 + 'px';
8672         base.style.top = 0 + 'px';

```

```

8673     base.xinc = 4;
8674     base.yinc = 4;
8675     base.scrx = 0;
8676     base.scry = 0;
8677     base.pgw = 100;
8678     base.pgh = 100;
8679     movePWindow(PackmonGo_2);
8680
8681     ctx = PackmonGo_3_Canvas.getContext('2d');
8682     pgo = 0.3;
8683     ctx.fillStyle = 'rgba(64,64,255,'+pgo+')';
8684     ctx.fillRect(0,0,pgw,pgh);
8685     base = PackmonGo_3;
8686     gsh.appendChild(base);
8687     base.style.zIndex = 1002;
8688     base.style.position = "fixed";
8689     base.style.left = 200 + 'px';
8690     base.style.top = 0 + 'px';
8691     base.xinc = 4;
8692     base.yinc = 4;
8693     base.scrx = 0;
8694     base.scry = 0;
8695     base.soft = 0;
8696     base.pgw = 100;
8697     base.pgh = 100;
8698     movePScreen(PackmonGo_3);
8699
8700     ctx = PackmonGo_4_Canvas.getContext('2d');
8701     pgw = pgh = 400;
8702     ctx.beginPath();
8703     ctx.strokeStyle = 'rgba(0,0,0)';
8704     ctx.arc(200,200,200,0,Math.PI*2,true);
8705     ctx.stroke();
8706     pgo = 0.4;
8707     ctx.fillStyle = 'rgba(255,31,32,'+pgo+')';
8708     ctx.fill();
8709     base = PackmonGo_4;
8710     gsh.appendChild(base);
8711     base.style.zIndex = 1002;
8712     base.style.position = "fixed";
8713     base.style.left = 200 + 'px';
8714     base.style.top = 0 + 'px';
8715     base.xinc = 4;
8716     base.yinc = 4;
8717     base.scrx = 0;
8718     base.scry = 0;
8719     base.soft = 5000;
8720     base.pgw = pgw;
8721     base.pgh = pgh;
8722     movePScreen(PackmonGo_4);
8723 }
8724 function movePWindow(base){
8725     if( stopPackmonFlag ){
8726         return;
8727     }
8728     x = parseInt(base.style.left);
8729     y = parseInt(base.style.top);
8730     w = window.innerWidth;
8731     h = window.innerHeight;
8732     if( x < 0 || w-base.pgw < x ){
8733         base.xinc = -base.xinc;
8734     }
8735     x += base.xinc;
8736     if( y < 0 || h-base.pgh < y ){
8737         //yinc = -yinc;
8738         base.yinc = -base.yinc;
8739     }
8740     y += base.yinc;
8741     base.style.left = x + 'px';
8742     base.style.top = y + 'px';
8743     //console.log('PC x:'+x+',y:'+y);
8744     //window.setTimeout(movePG,10);
8745 }
8746 function movePScreen(base){
8747     if( stopPackmonFlag ){
8748         return;
8749     }
8750     sw = screen.width;
8751     sh = screen.height;
8752     d = new Date();
8753     s = d.getTime(); // milli-seconds
8754     sx = ((s+base.soft) % 10000) * (screen.width / 10000);
8755     sy = ((s+base.soft) % 5000) * (screen.height / 5000);
8756     x = sx - window.screenX;
8757     y = sy - window.screenY;
8758     base.style.left = x + 'px';
8759     base.style.top = y + 'px';
8760 }
8761 function movePScreenCircle(base){
8762     if( stopPackmonFlag ){
8763         return;
8764     }
8765     sw = screen.width;
8766     sh = screen.height;
8767     pgw = base.pgw;
8768     pgh = base.pgh;
8769     d = new Date();
8770     s = d.getTime(); // milli-seconds
8771     ds = s + base.soft; // delay
8772     vsw = pgw + sw + pgw;
8773     vsh = pgh + sh + pgh;
8774     sx = -pgw + vsw * ((ds % 10000)/10000);
8775     sy = -pgh + vsh * ((ds % 10000)/10000);
8776     x = sx - window.screenX;
8777     y = sy - window.screenY;
8778     base.style.left = x + 'px';
8779     base.style.top = y + 'px';
8780 }
8781 if( PackmonInit == false ){
8782     //window.setTimeout(movePG,mm300);
8783     window.setInterval(movePWindow,10,PackmonGo_1);
8784     window.setInterval(movePWindow,10,PackmonGo_2);
8785     window.setInterval(movePScreen,10,PackmonGo_3);
8786     window.setInterval(movePScreen,10,PackmonGo_4);
8787     window.setInterval(movePScreenCircle,10,PackmonGo_4);
8788     window.addEventListener('click',stopPackmon);
8789     PackmonInit = true;
8790     stopPackmonFlag = false;
8791 }
8792 }
8793 function PackmonGo_Setup(e){
8794     stopPackmon();
8795     spawnPackmonGo();
8796     if( e != null ){
8797         e.stopPropagation();
8798     }
8799 }
8800 PackmonGo_Summary.addEventListener('click',PackmonGo_Setup);
8801 if( PackmonGo_Section.open == true ){
8802     PackmonGo_Setup();
8803 }
8804 </script>
8805
8806 <input id="PackmonGo_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8807 <input id="PackmonGo_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8808 <input id="PackmonGo_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8809 <span id="PackmonGo_WorkCodeView"></span>
8810 <script id="PackmonGo_WorkScript">
8811 function PackmonGo_openWorkCodeView(){
8812     function PackmonGo_showWorkCode(){
8813         showHtmlCode(PackmonGo_WorkCodeView,PackmonGo_WorkCodeSpan);
8814     }
8815     PackmonGo_WorkCodeViewOpen.addEventListener('click',PackmonGo_showWorkCode);
8816 }
8817 PackmonGo_openWorkCodeView(); // should be invoked by an event
8818 </script>
8819 </details>
8820 <!-- Template_WorkCodeSpan -->
8821 *//</span>
8822 //<!-- ===== Work } ===== -->
8823
8824
8825
8826 //<!-- ===== Work { ===== -->
8827 //<span id="SightClass_WorkCodeSpan">
8828 /*
8829 <details id="SightClass"><summary>SightClass</summary>
8830 <!-- ----- SightClass // 2020-1023 SatoxITS { -->
8831 <h2>SightClass</h2>
8832 <details><summary>meter</summary>
8833 <div id="SightClass_l_BPosition" class="SightClass_TextData">(0000, 0000)(0000 x 0000) </div>
8834 <div id="SightClass_l_BGScroffset" class="SightClass_TextData">(0000, 0000) </div>
8835 <div id="SightClass_l_GPosition" class="SightClass_TextData">(0000, 0000)(0000 x 0000) </div>
8836 <div id="SightClass_l_BGScroffset" class="SightClass_TextData">(0000, 0000)(0000 x 0000) </div>
8837 <div id="SightClass_l_Wheel" class="SightClass_TextData">Wheel</div>
8838 </details>
8839 <div id="SightClass_l_Window" class="SightClass_Window" draggable="true"></div>
8840 <style>
8841 .SightClass_TextData {
8842     color:#000;
8843     font-size:10pt;
8844 }
8845 .SightClass_Window {
8846     xzoom:2;
8847     resize:both;
8848     width:600px;
8849     height:320px;

```

```

8850 background-color:rgba(200,200,200,0.5);
8851 background-size:200%;
8852 xbackground-size:1280px 720px;
8853 background-image:url(WD-WallPaper03.png);
8854 }
8855 xbody {
8856 scroll-behavior:smooth;
8857 }
8858 </style>
8859 <script>
8860 //
8861 // https://stackoverflow.com/questions/4319487/detecting-if-the-browser-window-is-moved-with-javascript
8862 var SG_ScrollInitX = 0;
8863 var SG_ScrollInitY = 0;
8864 var SG_InitX = 0;
8865 var SG_InitY = 0;
8866 function SG_adjust(){
8867 xy = window.screenX + ' ' + window.screenY;
8868 wh = window.innerWidth + ' x ' + window.innerHeight;
8869 xywh = 'xy(' + xy + ') wh(' + wh + ')';
8870 if( SightClass.open) SightClass.l_BPosition.innerHTML = 'Browser: ' + xywh;
8871
8872 sg = SightClass.l_Window;
8873 sgr = sg.getBoundingClientRect();
8874 xy = sgr.left.toFixed(0) + ' ' + sgr.top.toFixed(0);
8875 wh = sgr.width + ' x ' + sgr.height;
8876 xywh = 'xy(' + xy + ') wh(' + wh + ')';
8877 if( SightClass.open) SightClass.l_GPosition.innerHTML = 'SiClass: ' + xywh;
8878
8879 //SightClass.l_Window.style.backgroundColor = sgr.left+'px ' + sgr.top+'px';
8880 if( SG_InitX == 0 ){
8881 SG_ScrollInitX = window.screenX;
8882 SG_ScrollInitY = window.screenY;
8883 //SightClass.l_Window.style.backgroundColor = '200%';
8884 SightClass.l_Window.style.backgroundColor = 'url("WD-WallPaper03.png")';
8885 SG_InitX = sgr.left;
8886 SG_InitY = sgr.top;
8887 }
8888 dx = SG_InitX - sgr.left;
8889 dy = SG_InitY - sgr.top;
8890
8891 dsx = SG_ScrollInitX - window.screenX;
8892 dsy = SG_ScrollInitY - window.screenY;
8893 scroff = 'Screen: ' + 'sx'+dsx + ',sy'+dsy;
8894 if( SightClass.open) SightClass.l_BGScrollOffset.innerHTML = scroff;
8895 dx += dsx;
8896 dy += dsy;
8897
8898 bgpos = dx+'px ' + dy+'px';
8899 SightClass.l_Window.style.backgroundColor = bgpos;
8900 if( SightClass.open) SightClass.l_BGPosition.innerHTML = 'BGround: '
8901 + SightClass.l_Window.style.backgroundColor;
8902 }
8903 var wheels = 0;
8904 function SG_wheel(e){
8905 wheels = 1;
8906 SightClass.l_Wheel.innerHTML = 'Mouse Wheel: ' + wheels + ' #' + e.target.id;
8907 if( e.target.id == 'SightClass.l_Window' ){
8908 e.preventDefault();
8909 }
8910 }
8911 function SG_adjust1(){
8912 SG_adjust();
8913 //sampling = 0;
8914 sampling = 20;
8915 //sampling = 200;
8916 window.setTimeout(SG_adjust1,sampling);
8917 }
8918 function SightClass_Setup(){
8919 SG_adjust();
8920 window.addEventListener('resize',SG_adjust);
8921 window.addEventListener('mousemove',SG_adjust);
8922 window.addEventListener('scroll',SG_adjust);
8923 window.addEventListener('wheel',SG_wheel,{passive:false});
8924 //window.moveTo(0,0);
8925
8926 // if focused
8927 //window.setInterval(SG_adjust,1);
8928 window.setTimeout(SG_adjust1,1);
8929 }
8930 // https://stackoverflow.com/questions/45225798/how-do-i-subscribe-to-a-window-move-event-in-the-atom-editor
8931 function Electron_Setup(){
8932 const { BrowserWindow } = require('electron');
8933 const currentWindow = BrowserWindow.getFocusedWindow();
8934 //currentWindow.on('move',function(){ SG_adjust(); });
8935 currentWindow.addEventListener('move',SG_adjust);
8936 }
8937 </script>
8938
8939 <input id="SightClass_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
8940 <input id="SightClass_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
8941 <input id="SightClass_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
8942 <span id="SightClass_WorkCodeView"></span>
8943 <script id="SightClass_WorkScript">
8944 function SightClass_openWorkCodeView(){
8945 function SightClass_showWorkCode(){
8946 showHtmlCode(SightClass_WorkCodeView,SightClass_WorkCodeSpan);
8947 }
8948 SightClass_WorkCodeViewOpen.addEventListener('click',SightClass_showWorkCode);
8949 }
8950 SightClass_openWorkCodeView(); // should be invoked by an event
8951 </script>
8952 </details>
8953 <!-- SightClass_WorkCodeSpan -->
8954 *//</span>
8955 //<!-- ===== Work ===== -->
8956
8957
8958 /*
8959 <!-- ----- GJConsole BEGIN { ----- -->
8960 <span id="gjc" data-title="GJConsole" data-author="sato@its-more.jp">
8961 <details><summary>GJ Console</summary>
8962 <p>
8963 <span id="GJE_RootNode0"></span>
8964 <span id="GJC_Container"></span>
8965 </p>
8966 <style id="GJConsoleStyle">
8967 .GJConsole {
8968 z-index:1000;
8969 width:400px; height:200px;
8970 margin:2px;
8971 color:#fff; background-color:#66a;
8972 font-size:12px; font-family:monospace,Courier New;
8973 }
8974 </style>
8975
8976 <script id="GJConsoleScript" class="GJConsole">
8977 var PSl = " "
8978 function GJC_Keydown(keyevent){
8979 key = keyevent.code
8980 if( key == "Enter" ){
8981 GJC_Command(this)
8982 this.value += "\n" + PSl // prompt
8983 }else
8984 if( key == "Escape"){
8985 SuppressGJShell = false
8986 GshMenu.focus() // should be previous focus
8987 }
8988 }
8989 var GJC_SessionId
8990 function GJC_SetSessionId(){
8991 var xd = new Date()
8992 GJC_SessionId = xd.getTime() / 1000
8993 }
8994 GJC_SetSessionId()
8995 function GJC_Memory(mem,args,text){
8996 argv = args.split(' ')
8997 cmd = argv[0]
8998 argv.shift()
8999 args = argv.join(' ')
9000 ret = ""
9001
9002 if( cmd == 'clear' ){
9003 Permanent.setItem(mem,'')
9004 }else
9005 if( cmd == 'read' ){
9006 ret = Permanent.getItem(mem)
9007 }else
9008 if( cmd == 'save' ){
9009 val = Permanent.getItem(mem)
9010 if( val == null ){ val = "" }
9011 d = new Date()
9012 val += d.getTime()/1000+" "+GJC_SessionId+" "+document.URL+" "+args+"\n"
9013 val += text.value
9014 Permanent.setItem(mem,val)
9015 }else
9016 if( cmd == 'write' ){
9017 val = Permanent.getItem(mem)
9018 if( val == null ){ val = "" }
9019 d = new Date()
9020 val += d.getTime()/1000+" "+GJC_SessionId+" "+document.URL+" "+args+"\n"
9021 Permanent.setItem(mem,val)
9022 }else{
9023 ret = "Commands: write | read | save | clear"
9024 }
9025 }
9026 return ret

```

```

9027 }
9028 // -- 2020-09-14 added TableEditor
9029 var GJE_CurElement = null; //GJE_RootNode
9030 GJE_NodeSaved = null
9031 GJE_TableNo = 1
9032 function GJE_StyleKeyCommand(kev){
9033     keycode = Kev.code
9034     console.log('GJE-Key: '+keycode)
9035     if( keycode == "Escape" ){
9036         GJE_SetStyle(this);
9037     }
9038     kev.stopPropagation()
9039     // https://developer.mozilla.org/en-US/docs/Web/API/Event/stopPropagation
9040 }
9041 var GJE_CommandMode = false
9042 function GJE_TableKeyCommand(kev,tab){
9043     wasCmdMode = GJE_CommandMode
9044     key = kev.code
9045     if( key == "Escape" ){
9046         console.log("To command mode: "+tab.nodeName+"#"+tab.id)
9047         //tab.setAttribute('contenteditable', 'false')
9048         tab.style.caretColor = "blue"
9049         GJE_CommandMode = true
9050     }else
9051     if( key == "KeyA" ){
9052         tab.style.caretColor = "red"
9053         GJE_CommandMode = false
9054     }else
9055     if( key == "KeyI" ){
9056         tab.style.caretColor = "red"
9057         GJE_CommandMode = false
9058     }else
9059     if( key == "KeyO" ){
9060         tab.style.caretColor = "red"
9061         GJE_CommandMode = false
9062     }else
9063     if( key == "KeyJ" ){
9064         console.log("ROW-DOWN")
9065     }else
9066     if( key == "KeyK" ){
9067         console.log("ROW-UP")
9068     }else
9069     if( key == "Keyw" ){
9070         console.log("COL-FORW")
9071     }else
9072     if( key == "Keyb" ){
9073         console.log("COL-BACK")
9074     }
9075     kev.stopPropagation()
9076     if( wasCmdMode ){
9077         kev.preventDefault()
9078     }
9079 }
9080 }
9081 function GJE_DragEvent(ev,elem){
9082     x = ev.clientX
9083     y = ev.clientY
9084     console.log("Dragged: "+this.nodeName+'#'+this.id+' x='+x' y='+y)
9085 }
9086 // https://developer.mozilla.org/en-US/docs/Web/API/Event/DragEvent
9087 // https://www.w3.org/TR/ulevents/#events-mouseevents
9088 function GJE_DropEvent(ev,elem){
9089     x = ev.clientX
9090     y = ev.clientY
9091     this.style.x = x
9092     this.style.y = y
9093     this.style.position = 'absolute' // 'fixed'
9094     this.parentNode = gsh // just for test
9095     console.log("Dropped: "+this.nodeName+'#'+this.id+' x='+x' y='+y
9096     +' parent='+this.parentNode.id)
9097 }
9098 }
9099 function GJE_SetTableStyle(ev){
9100     this.innerHTML = this.value; // sync. for external representation?
9101     if(false){
9102         stid = this.parentNode.id+this.id
9103         // and remove " _span" at the end
9104         e = document.getElementById(stid)
9105         //alert("SetTableStyle #'+e.id+'\n'+this.value)
9106         if( e != null ){
9107             e.innerHTML = this.value
9108         }else{
9109             console.log('Style Not found: '+stid)
9110         }
9111         //alert('event StopPropagation: '+ev)
9112     }
9113 }
9114 function setCSSofClass(cclass,cstyle){
9115     const ss = document.styleSheets[3]; // 0, 1, 2, 3, ... ?
9116     rlen = ss.cssRules.length;
9117     let tabrule = null;
9118     rulex = -1
9119     // should skip white space at the top of cstyle
9120     sel = cstyle.charAt(0);
9121     selector = sel+cclass;
9122     console.log("-- search style rule for '+selector')
9123     for(let i = 0; i < rlen; i++){
9124         cr = ss.cssRules[i];
9125         console.log('CSS rule ['+'i'+rlen+' '+cr.selectorText);
9126         if( cr.selectorText == selector ){ // css class selector
9127             tabrule = ss.cssRules[i];
9128             console.log('CSS rule found for: ['+'i'+rlen+' '+selector);
9129             ss.deleteRule(i);
9130             //rlen = ss.cssRules.length;
9131             rulex = i
9132             // should search and replace the property here
9133         }
9134     }
9135     // https://developer.mozilla.org/en-US/docs/Web/API/CSSStyleSheet/insertRule
9136     if( tabrule == null ){
9137         console.log("CSS rule NOT found for: ['+rlen+' '+selector);
9138         ss.insertRule(cstyle,rlen);
9139         ss.insertRule(cstyle,0); // override by 0?
9140         console.log("CSS rule inserted: ['+(rlen+1)+'\n'+cstyle);
9141     }else{
9142         ss.insertRule(cstyle,rlen);
9143         ss.insertRule(cstyle,0);
9144         console.log("CSS rule replaced: ['+(rlen+1)+'\n'+cstyle);
9145     }
9146 }
9147 }
9148 function GJE_SetStyle(te){
9149     console.log('Apply the style to: '+te.id+'\n');
9150     console.log('Apply the style to: '+te.parentNode.id+'\n');
9151     console.log('Apply the style to: '+te.parentNode.class+'\n');
9152     cclass = te.parentNode.class;
9153     setCSSofClass(cclass,te.value); // should get selector part from
9154     // selector { rules }
9155 }
9156 if(false){
9157     //console.log('Apply the style:')
9158     //stid = this.parentNode.id+this.id+"
9159     //stid = this.id+"style"
9160     css = te.value
9161     stid = te.parentNode.id+"style"
9162     e = document.getElementById(stid)
9163     if( e != null ){
9164         //console.log('Apply the style: '+e.id+'\n'+te.value);
9165         console.log('Apply the style: '+e.id+'\n'+css);
9166         e.innerHTML = css; //te.value;
9167         //ncss = e.sheet;
9168         //ncss.insertRule(te.value,ncss.cssRules.length);
9169     }else{
9170         console.log('No element to Apply the style: '+stid)
9171     }
9172     tblid = te.parentNode.id+"table";
9173     e = document.getElementById(tblid);
9174     if( e != null ){
9175         //e.setAttribute('style',css);
9176         e.setProperty('style',css,'!important');
9177     }
9178 }
9179 }
9180 function makeTable(argv){
9181     //tid = ''
9182     //cwe = GJE_CurElement
9183     cwe = GJCl_Container;
9184     //cwd = GJFactory;
9185     tid = 'table_' + GJE_TableNo
9186 }
9187 nt = new Text('\n')
9188 cwe.appendChild(nt)
9189 ne = document.createElement('span'); // the container
9190 cwe.appendChild(ne)
9191 ne.id = tid + "_span"
9192 ne.setAttribute('contenteditable',true)
9193 }
9194 htspar = document.createElement('span'); // html part
9195 //htspan.id = tid + "-html"
9196 //ne.innerHTML = "\n"
9197 nt = new Text('\n')
9198 ne.appendChild(nt)
9199 ne.appendChild(htspan)
9200 }
9201 htspar.id = tid
9202 htspar.setAttribute('class',tid)
9203 }

```

```

9204
9205 ne.setAttribute('draggable', 'true')
9206 ne.addEventListener('drag', GJE_DragEvent);
9207 ne.addEventListener('dragend', GJE_DropEvent);
9208
9209 var col = 3
9210 var row = 2
9211 if( argv[0] != null ){
9212     col = argv[0]
9213     argv.shift()
9214 }
9215 if( argv[0] != null ){
9216     row = argv[0]
9217     argv.shift()
9218 }
9219
9220 //ne.setAttribute('class', tid)
9221 ht = "\n"
9222 //ht += '<'+table ' + 'id="'+tid+'"' + ' class="'+tid+'"'
9223 ht += '<'+table
9224     + ' onkeydown="GJE_TableKeyCommand(event,this)"'
9225     //+ ' ondrag="GJE_DragEvent(event,this)"\n'
9226     //+ ' ondragend="GJE_DropEvent(event,this)"\n'
9227     + ' draggable="true"\n'
9228     //+ ' contenteditable="true"'
9229     + '>\n'
9230 ht += '<'+tbody>\n';
9231 for( r = 0; r < row; r++){
9232     ht += "<"+tr>\n"
9233     for( c = 0; c < col; c++){
9234         ht += "<"+td>"
9235         ht += " ABCDEFGHIJKLMNOPQRSTUVWXYZ".charAt(c) + r
9236         ht += "<"+"/td>\n"
9237     }
9238     ht += "<"+"/tr>\n"
9239 }
9240 ht += '<'+"/tbody>\n';
9241 ht += '<'+"/table>\n';
9242 htspan.innerHTML = ht;
9243 nt = new Text('\n');
9244 ne.appendChild(nt);
9245
9246 st = '#'+tid+' *{\n' // # for instanse specific
9247     + ' '+border:1px solid #aaa;\n'
9248     + ' '+background-color:#efe;\n'
9249     + ' '+color:#22;\n'
9250     + ' '+font-size:#14pt !important;\n'
9251     + ' '+font-family:monospace,Courier New !important;\n'
9252     + ' } /* * hit ESC to apply *'+"/\n"
9253
9254 // wish script to be included
9255 //n] = document.createElement('script')
9256 //ne.appendChild(n]
9257 //ne.innerHTML = 'function SetStyle(e){}'
9258
9259 // selector seems lost in dynamic style appending
9260 if(false){
9261     ns = document.createElement('style')
9262     ne.appendChild(ns)
9263     ns.id = tid + '-style'
9264     ns.innerHTML = '\n'+st
9265     nt = new Text('\n')
9266     ne.appendChild(nt)
9267 }
9268 setCSSofClass(tid, st); // should be in JavaScript script?
9269
9270 nx = document.createElement('textarea')
9271 ne.appendChild(nx)
9272 nx.id = tid + '-style_def'
9273 nx.setAttribute('class', 'GJ_StyleEditor')
9274 nx.spellcheck = false
9275 nx.cols = 60
9276 nx.rows = 10
9277 nx.innerHTML = '\n'+st
9278 nx.addEventListener('change', GJE_SetTableStyle);
9279 nx.addEventListener('keydown', GJE_StyleKeyCommand);
9280 //nx.addEventListener('click', GJE_SetTableStyle);
9281
9282 nt = new Text('\n')
9283 cwe.appendChild(nt)
9284
9285 GJE_TableNo += 1
9286 return 'created TABLE id="'+tid+'"'
9287 }
9288
9289 function GJE_NodeEdit(argv){
9290     cwe = GJE_CurElement
9291     cmd = argv[0]
9292     argv.shift()
9293     args = argv.join(' ')
9294     ret = ""
9295
9296     if( cmd == '.u' || cmd == '.un' || cmd == 'undo' ){
9297         if( GJE_Nodesaved != null ){
9298             xn = GJE_RootNode
9299             GJE_RootNode = GJE_Nodesaved
9300             GJE_Nodesaved = xn
9301             ret = '-- did undo'
9302         }else{
9303             ret = '-- could not undo'
9304         }
9305         return ret
9306     }
9307     GJE_Nodesaved = GJE_RootNode.cloneNode()
9308     if( cmd == '.c' || cmd == '.cd' || cmd == 'cd' ){
9309         if( argv[0] == null ){
9310             ne = GJE_RootNode
9311         }else{
9312             if( argv[0] == '..' ){
9313                 ne = cwe.parentNode
9314             }else{
9315                 ne = document.getElementById(argv[0])
9316             }
9317             if( ne != null ){
9318                 GJE_CurElement = ne
9319                 ret = "-- current node: " + ne.id
9320             }else{
9321                 ret = "-- not found: " + argv[0]
9322             }
9323         }else{
9324             if( cmd == '.mkt' || cmd == '.mktable' ){
9325                 makeTable(argv)
9326             }else{
9327                 if( cmd == '.m' || cmd == '.mk' || cmd == 'mk' ){
9328                     ne = document.createElement(argv[0])
9329                     //ne.id = argv[0]
9330                     ret = "-- created " + ne + " under " + cwe.tagName + "#" + cwe.id
9331                     cwe.appendChild(ne)
9332                     if( cmd == '.m' || cmd == '.mk' ){
9333                         GJE_CurElement = ne
9334                     }
9335                 }else{
9336                     if( cmd == '.n' || cmd == '.nm' || cmd == 'nm' ){
9337                         cwe.id = argv[0]
9338                     }else{
9339                         if( cmd == '.r' || cmd == '.rm' || cmd == 'rm' ){
9340                             }else{
9341                                 if( cmd == '.h' || cmd == '.sh' || cmd == 'sh' ){
9342                                     s = argv.join(' ')
9343                                     cwe.innerHTML = s
9344                                 }else{
9345                                     if( cmd == '.a' || cmd == '.sa' || cmd == 'sa' ){
9346                                         cwe.setAttribute(argv[0], argv[1])
9347                                     }else{
9348                                         if( cmd == '.l' ){
9349                                             }else{
9350                                                 if( cmd == '.i' || cmd == '.ih' || cmd == 'ih' ){
9351                                                     ret = cwe.innerHTML
9352                                                 }else{
9353                                                     if( cmd == '.p' || cmd == '.pw' || cmd == 'pw' ){
9354                                                         ret = cwe.nodeType + " " + cwe.tagName + " " + cwe.id
9355                                                         for( we = cwe.parentNode; we != null; ){
9356                                                             ret += "\n" + " " + we.nodeType + " " + we.tagName + " " + we.id
9357                                                         }
9358                                                         we = we.parentNode
9359                                                     }else{
9360                                                         {
9361                                                             ret = "Command: mk | rm \n"
9362                                                             ret += " pw -- print current node\n"
9363                                                             ret += " mk type -- make node with name and type\n"
9364                                                             ret += " nm name -- set the id #name of current node\n"
9365                                                             ret += " rm name -- remove named node\n"
9366                                                             ret += " cd name -- change current node\n"
9367                                                         }
9368                                                     }
9369                                                     //alert(ret)
9370                                                     return ret
9371         }
9372     }
9373     function GJC_Command(text){
9374         lines = text.value.split('\n')
9375         line = lines[lines.length-1]
9376         argv = line.split(' ')
9377         text.value = '\n'
9378         if( argv[0] == '%' ){ argv.shift() }
9379         args0 = argv.join(' ')
9380         cmd = argv[0]
9381         argv.shift()
9382         args = argv.join(' ')
9383     }

```

```

9381 if( cmd == 'nolog' ){
9382     StopConsoleLog = true
9383 }else
9384 if( cmd == 'new' ){
9385     if( argv[0] == 'table' ){
9386         argv.shift()
9387         console.log( argv+'argv')
9388         text.value += makeTable(argv)
9389     }else
9390     if( argv[0] == 'console' ){
9391         text.value += GJ_NewConsole('GJ_Console')
9392     }else{
9393         text.value += '-- new { console | table }'
9394     }
9395 }else
9396 if( cmd == 'strip' ){
9397     //text.value += GJF_StripeClass()
9398 }else
9399 if( cmd == 'css' ){
9400     sel = '#table 1'
9401     if( argv[0] == '0' ){
9402         rule1 = sel+'{color:#000 !important; background-color:#fff !important;}';
9403     }else
9404     rule1 = sel+'{color:#f00 !important; background-color:#eef !important;}';
9405     document.styleSheets[3].deleteRule(0);
9406     document.styleSheets[3].insertRule(rule1,0);
9407     text.value += "CSS rule added: "+rule1
9408 }else
9409 if( cmd == 'print' ){
9410     e = null;
9411     if( e == null ){
9412         e = document.getElementById('GJFactory_0')
9413     }
9414     if( e == null ){
9415         e = document.getElementById('GJFactory_1')
9416     }
9417     if( argv[0] != null ){
9418         id = argv[0]
9419         if( id == 't' ){
9420             //e = document.getElementById('GJE_RootNode');
9421         }else{
9422             e = document.getElementById(id)
9423         }
9424         if( e != null ){
9425             text.value += e.outerHTML
9426         }else{
9427             text.value += "Not found: " + id
9428         }
9429     }else{
9430         text.value += GJE_RootNode.outerHTML
9431         //text.value += e.innerHTML
9432     }
9433 }else
9434 if( cmd == 'destroy' ){
9435     text.value += GJFactory_Destroy()
9436 }else
9437 if( cmd == 'save' ){
9438     e = document.getElementById('GJFactory')
9439     Permanent.setItem('GJFactory-1',e.innerHTML)
9440     text.value += "-- Saved GJFactory"
9441 }else
9442 if( cmd == 'load' ){
9443     gjf = Permanent.getItem('GJFactory-1')
9444     e = document.getElementById('GJFactory')
9445     e.innerHTML = gjf
9446     // must restore EventListener
9447     text.value += "-- EventListener was not restored"
9448 }else
9449 if( cmd.charAt(0) == '.' ){
9450     argv0 = args0.split('.')
9451     text.value += GJE_NodeEdit(argv0)
9452 }else
9453 if( cmd == 'cont' ){
9454     bannerIsStopping = false
9455     GshMenuStop.innerHTML = "Stop"
9456 }else
9457 if( cmd == 'date' ){
9458     text.value += DateLong()
9459 }else
9460 if( cmd == 'echo' ){
9461     text.value += args
9462 }else
9463 if( cmd == 'fork' ){
9464     html_fork()
9465 }else
9466 if( cmd == 'last' ){
9467     text.value += MyHistory
9468     //h = document.createElement("span")
9469     //h.innerHTML = MyHistory
9470     //text.value += h.innerHTML
9471     //tx = MyHistory.replace("\n","")
9472     //text.value += tx.replace("<"+br>","\n") + "xxxx<"+br>yyyy"
9473 }else
9474 if( cmd == 'ne' ){
9475     text.value += GJE_NodeEdit(argv)
9476 }else
9477 if( cmd == 'reload' ){
9478     location.reload()
9479 }else
9480 if( cmd == 'mem' ){
9481     text.value += GJC_Memory('GJC_Storage',args,text)
9482 }else
9483 if( cmd == 'stop' ){
9484     bannerIsStopping = true
9485     GshMenuStop.innerHTML = "Start"
9486 }else
9487 if( cmd == 'who' ){
9488     text.value += "SessionId="+GJC_SessionId+" "+document.URL
9489 }else
9490 if( cmd == 'wall' ){
9491     text.value += GJC_Memory('GJC_Wall','write',text)
9492 }else
9493 {
9494     text.value += "Commands: help | echo | date | last \n"
9495     + '          new | save | load | mem \n'
9496     + '          who | wall | fork | nife'
9497 }
9498 }
9499
9500 function GJC_Input(){
9501     if( this.value.endsWith("\n") ){ // remove NL added by textarea
9502         this.value = this.value.slice(0,this.value.length-1)
9503     }
9504 }
9505
9506 var GCJ_Id = null
9507 function GJC_Resize(){
9508     GJC_Id.style.zIndex = 20000
9509     //GJC_Id.style.width = window.innerWidth - 16
9510     GJC_Id.style.width = '100%';
9511     GJC_Id.style.height = 300;
9512     GJC_Id.style.backgroundColor = "rgba(0,64,16,1.0)" // blackboard color
9513     GJC_Id.style.color = "rgba(255,255,255,1.0)"
9514 }
9515 function GJC_FocusIn(){
9516     this.spellCheck = false
9517     SuppressGJShell = true
9518     this.onkeydown = GJC_Keydown
9519     GJC_Resize()
9520 }
9521 function GJC_FocusOut(){
9522     SuppressGJShell = false
9523     this.removeEventListener('keydown',GJC_Keydown);
9524 }
9525 window.addEventListener('resize',GJC_Resize);
9526
9527 function GJC_OnStorage(e){
9528     //alert('Got Message')
9529     //GJC.value += "\n((ReceivedMessage))\n"
9530 }
9531 window.addEventListener('storage',GJC_OnStorage);
9532 //window.addEventListener('storage',()=>{alert('GotMessage')})
9533
9534 function GJC_Setup(gjcId){
9535     //gjcId.style.width = gsh.getBoundingClientRect().width
9536     gjcId.style.width = '100%';
9537     gjcId.value = "GJShell Console // " + GshVersion.innerHTML + "\n"
9538     //gjcId.value += "Date: " + DateLong() + "\n"
9539     gjcId.value += P81
9540     gjcId.onFocus = GJC_FocusIn
9541     gjcId.addEventListener('input',GJC_Input);
9542     gjcId.addEventListener('focusout',GJC_FocusOut);
9543     GJC_Id = gjcId
9544 }
9545 function GJC_Clear(id){
9546 }
9547 function GJConsole_initConsole(){
9548     if( document.getElementById("GJC_0") != null ){
9549         GJC_Setup(GJC_0)
9550     }else{
9551         GJC_Container.innerHTML = '<'
9552             + 'textarea id="GJC_1" class="GJConsole"><'+'/>';
9553         GJC_Setup(GJC_1)
9554         factory = document.createElement('span');
9555         gsh.appendChild(factory)
9556         GJE_RootNode = factory;
9557         GJE_CurElement = GJE_RootNode;

```

```

9558     }
9559     }
9560     var initGJCF = false;
9561     function GJConsole_initFactory(){
9562         if (initGJCF) { return; } initGJCF = true;
9563         GShell_initKeyCommands();
9564         GJConsole_initConsole();
9565     }
9566     //GJConsole_initFactory();
9567     // TODO: focus handling
9568     </script>
9569     <style>
9570     .GJ_StyleEditor {
9571         font-size:9pt !important;
9572         font-family:Courier New, monospace !important;
9573     }
9574     </style>
9575     </details>
9576     </span>
9577     <!-- GJConsole END ) ----- -->
9578     */
9579     /*
9580     <span id="BlinderText">
9581     <style id="BlinderTextStyle">
9582     #GJLinkView {
9583         xposition:absolute; z-index:5000;
9584         position:relative;
9585         display:block;
9586         left:8px;
9587         color:#fff;
9588         width:800px; height:300px; resize:both;
9589         margin:0px; padding:4px;
9590         background-color:rgba(200,200,200,0.5) !important;
9591     }
9592     .MsgText {
9593         width:578px !important;
9594         resize:both !important;
9595         color:#000 !important;
9596     }
9597     .GjNote {
9598         font-family:Georgia !important;
9599         font-size:13pt !important;
9600         color:#22a !important;
9601     }
9602     .textField {
9603         display:inline;
9604         border:0.3px solid #444;
9605         border-radius:3px;
9606         color:#000; background-color:#fff;
9607         width:106pt; height:18pt;
9608         margin:2px;
9609         padding:2px;
9610         resize:none;
9611         vertical-align:middle;
9612         font-size:10pt; font-family:Courier New;
9613     }
9614     .textLabel {
9615         border:0px solid #000 !important;
9616         background-color:rgba(0,0,0,0);
9617     }
9618     .textURL {
9619         width:300pt !important;
9620         border:0px solid #000 !important;
9621         background-color:rgba(0,0,0,0);
9622     }
9623     .VisibleText {
9624     }
9625     .BlinderText {
9626         color:#000; background-color:#eee;
9627     }
9628     .JoinButton {
9629         font-family:Georgia !important;
9630         font-size:11pt;
9631         line-height:1.1;
9632         height:18pt;
9633         width:50pt;
9634         padding:3px !important;
9635         text-align:center !important;
9636         border-color:#aaa !important;
9637         border-radius:5px;
9638         color:#fff; background-color:#44 !important;
9639         vertical-align:middle !important;
9640     }
9641     .SendButton {
9642         vertical-align:top;
9643     }
9644     .ws0_log {
9645         font-size:10pt;
9646         color:#000 !important;
9647         line-height:1.0;
9648         background-color:rgba(255,255,255,0.7) !important;
9649         font-family:Courier New,monospace !important;
9650         width:99.3%;
9651         white-space:pre;
9652     }
9653     </style>
9654     <!-- Form autofill test
9655     Location: <input id="xxserv" type="text" value="https://192.168.10.1/boafm/formLogin" size="80">
9656     <form method="POST" id="xxform" action="https://192.168.10.1/boafm/formLogin">
9657     dest? <input id="XDS" name="dest" type="text" size="80" value="/index_contents.html">
9658     -->
9659     <details><summary>Form Auto. Filling</summary>
9660     <style>
9661     .xxinput { width:260pt !important; line-height:1.1 !important; margin:1px;
9662     display:inline !important; font-size:10pt !important; padding:1px !important;
9663     }
9664     </style>
9665     <span style="font-family:Courier New;color:black;font-size:12pt;" onactive="">
9666     <form method="POST" id="xxform" action="https://192.168.10.1/" style="white-space:pre;">
9667     Location: <input id="xxserv" class="xxinput" type="text" value="https://192.168.10.1/" size="80">
9668     Username: <input id="xxuser" class="xxinput" name="user" type="text" autocomplete="on">
9669     Password: <input id="xxpass" class="xxinput" name="pass" type="password" autocomplete="on">
9670     SessionId:<input id="xxssid" class="xxinput" name="SESSION ID" type="text" size="80">
9671     <input id="xxsubm" class="xxinput" type="submit" value="Submit"></form>
9672     </span>
9673     <script>
9674     function XXSetFormAction(){
9675         xxform.setAttribute('action',xxserv.value);
9676     }
9677     xxform.setAttribute("action",xxserv.value);
9678     xxserv.addEventListener('change',XXSetFormAction);
9679     //xxserv.value = location.href;
9680     </script>
9681     </details>
9682     */
9683     /*
9684     <details id="BlinderTextClass"><summary>BlinderText</summary>
9685     <span class="gsh-src">
9686     <span id="BlinderTextScript">
9687     // https://w3c.github.io/ivevents/#event-type-keydown
9688     //
9689     // 2020-09-21 class BlinderText - textarea element not to be readable
9690     //
9691     // BlinderText attributes
9692     // bl_plainText - null
9693     // bl_hideChecksum - [false]
9694     // bl_showLength - [false]
9695     // bl_visible - [false]
9696     // data-bl_config []
9697     // - min. length
9698     // - max. length
9699     // - acceptable charset in generate text
9700     //
9701     function BlinderChecksum(text){
9702         plain = text.bl_plainText;
9703         return strCRC32(plain,plain.length).toFixed(0);
9704     }
9705     function BlinderKeydown(ev){
9706         pass = ev.target;
9707         if ( ev.code == 'Enter' ){
9708             ev.preventDefault();
9709         }
9710         ev.stopPropagation()
9711     }
9712     function BlinderKeyUp1(ev){
9713         blind = ev.target;
9714         if ( ev.code == 'Backspace' ){
9715             blind.bl_plainText = blind.bl_plainText.slice(0,blind.bl_plainText.length-1)
9716         }
9717         else if ( and(ev.code == 'KeyV', ev.ctrlKey) ){
9718             blind.bl_visible = !blind.bl_visible;
9719         }
9720         else if ( and(ev.code == 'KeyL', ev.ctrlKey) ){
9721             blind.bl_showLength = !blind.bl_showLength;
9722         }
9723         else if ( and(ev.code == 'KeyU', ev.ctrlKey) ){
9724             blind.bl_plainText = "";
9725         }
9726         else if ( and(ev.code == 'KeyR', ev.ctrlKey) ){
9727             checksum = BlinderChecksum(blind);
9728             blind.bl_plainText = checksum; //.toString(32);
9729         }
9730         if ( ev.code == 'Enter' ){

```

```

9735     ev.stopPropagation();
9736     ev.preventDefault();
9737     return;
9738 }else
9739 if( ev.key.length != 1 ){
9740     console.log('KeyUp: '+ev.code+'/'+ev.key);
9741     return;
9742 }else{
9743     blind.bl_plainText += ev.key;
9744 }
9745
9746 leng = blind.bl_plainText.length;
9747 //console.log('KeyUp: '+ev.code+'/'+blind.bl_plainText);
9748 checksum = BlinderChecksum(blind) % 10; // show last one digit only
9749
9750 visual = '';
9751 if( !blind.bl_hideChecksum || blind.bl_showLength ){
9752     visual += '(';
9753 }
9754 if( !blind.bl_hideChecksum ){
9755     visual += '#' +checksum.toString(10);
9756 }
9757 if( blind.bl_showLength ){
9758     visual += '/' + leng;
9759 }
9760 if( !blind.bl_hideChecksum || blind.bl_showLength ){
9761     visual += ')';
9762 }
9763 if( blind.bl_visible ){
9764     visual += blind.bl_plainText;
9765 }else{
9766     visual += '*'.repeat(leng);
9767 }
9768 blind.value = visual;
9769 }
9770 function BlinderKeyUp(ev){
9771     BlinderKeyUp1(ev);
9772     ev.stopPropagation();
9773 }
9774 // https://w3c.github.io/uievents/#keyboardevent
9775 // https://w3c.github.io/uievents/#uievent
9776 // https://dom.spec.whatwg.org/#event
9777 function BlinderTextEvent(){
9778     ev = event;
9779     blind = ev.target;
9780     console.log('Event: '+ev.type+'@'+blind.nodeName+'#'+blind.id)
9781     if( ev.type == 'keyup' ){
9782         BlinderKeyUp(ev);
9783     }else
9784     if( ev.type == 'keydown' ){
9785         BlinderKeyDown(ev);
9786     }else{
9787         console.log('thru-event '+ev.type+'@'+blind.nodeName+'#'+blind.id)
9788     }
9789 }
9790 //< textarea hidden id="BlinderTextClassDef" class="textField"
9791 // onkeydown="BlinderTextEvent()" onkeyup="BlinderTextEvent()"
9792 // spellcheck="false">< /textarea>
9793 //< textarea hidden id="gj_pass1"
9794 // class="textField BlinderText"
9795 // placeholder="PassWord"
9796 // onkeydown="BlinderTextEvent()"
9797 // onkeyup="BlinderTextEvent()"
9798 // spellcheck="false"> /textarea>
9799 function SetupBlinderText(parent,txa,phold){
9800     if( txa == null ){
9801         txa = document.createElement('textarea');
9802         //txa.id = id;
9803     }
9804     txa.setAttribute('class','textField BlinderText');
9805     txa.setAttribute('placeholder',phold);
9806     txa.setAttribute('onkeydown','BlinderTextEvent()');
9807     txa.setAttribute('onkeyup','BlinderTextEvent()');
9808     txa.setAttribute('spellcheck','false');
9809     //txa.setAttribute('bl_plainText','false');
9810     txa.bl_plainText = '';
9811     //parent.appendChild(txa);
9812 }
9813 function DestroyBlinderText(txa){
9814     txa.removeAttribute('class');
9815     txa.removeAttribute('placeholder');
9816     txa.removeAttribute('onkeydown');
9817     txa.removeAttribute('onkeyup');
9818     txa.removeAttribute('spellcheck');
9819     txa.bl_plainText = '';
9820 }
9821 //
9822 // visible textarea like Username
9823 //
9824 function VisibleTextEvent(){
9825     if( event.code == 'Enter' ){
9826         if( event.target.NoEnter ){
9827             event.preventDefault();
9828         }
9829     }
9830     event.stopPropagation();
9831 }
9832 function SetupVisibleText(parent,txa,phold){
9833     if( false ){
9834         txa.setAttribute('class','textField VisibleText');
9835     }else{
9836         newclass = txa.getAttribute('class');
9837         if( and(newclass != null, newclass != '') ){
9838             newclass += ' ';
9839         }
9840         newclass += 'VisibleText';
9841         txa.setAttribute('class',newclass);
9842     }
9843     //console.log('SetupVisibleText class='+txa.class);
9844     txa.setAttribute('placeholder',phold);
9845     txa.setAttribute('onkeydown','VisibleTextEvent()');
9846     txa.setAttribute('onkeyup','VisibleTextEvent()');
9847     txa.setAttribute('spellcheck','false');
9848     cols = txa.getAttribute('cols');
9849     if( cols != null ){
9850         txa.style.width = '580px';
9851         //console.log('VisualText#'+txa.id+' cols='+cols)
9852     }else{
9853         //console.log('VisualText#'+txa.id+' NO cols')
9854     }
9855     rows = txa.getAttribute('rows');
9856     if( rows != null ){
9857         txa.style.height = '30px';
9858         txa.style.resize = 'both';
9859         txa.NoEnter = false;
9860     }else{
9861         txa.NoEnter = true;
9862     }
9863 }
9864 function DestroyVisibleText(txa){
9865     txa.removeAttribute('class');
9866     txa.removeAttribute('placeholder');
9867     txa.removeAttribute('onkeydown');
9868     txa.removeAttribute('onkeyup');
9869     txa.removeAttribute('spellcheck');
9870     cols = txa.removeAttribute('cols');
9871 }
9872 </span>
9873 <script>
9874 js = document.getElementById('BlinderTextScript');
9875 eval(js.innerHTML);
9876 //js.outerHTML = ""
9877 </script>
9878
9879 </span><!-- end of class="gsh-src" -->
9880 </details>
9881 </span>
9882 */
9883
9884 /*
9885 <script id="GJLinkScript">
9886 function gjkey_hash(text){
9887     return strCRC32(text,text.length) % 0x10000;
9888 }
9889 function gj_addlog(e,msg){
9890     now = (new Date()).getTime() / 1000;.toFixed(3);
9891     tstp = '[' +now+' ]';
9892     e.value += tstp + msg;
9893     e.scrollTop = e.scrollHeight;
9894 }
9895 function gj_addlog_cl(msg){
9896     ws0_log.value += '(console.log) ' + msg + '\n';
9897 }
9898 var GJ_Channel = null;
9899 var GJ_Log = null;
9900 var gjx; // the global variable
9901 function GJ_Join(){
9902     target = gj_join;
9903     if( target.value == 'Leave' ){
9904         GJ_Channel.close();
9905         GJ_Channel = null;
9906         target.value = 'Join';
9907     }
9908     return;
9909 }
9910 var ws0;
9911 var ws0_log;

```



```

9912
9913 sav_console_log = console.error
9914 console.error = gj_addlog_cj
9915 ws0 = new WebSocket(gj_serv.innerHTML);
9916 console.error = sav_console_log
9917
9918 GJ_Channel = ws0;
9919 ws0_log = document.getElementById('ws0_log');
9920 GJ_Log = ws0_log;
9921
9922 now = (new Date().getTime() / 1000).toFixed(3);
9923 const wsstats = ["CONNECTING", "OPEN", "CLOSING", "CLOSED"];
9924 cst = wsstats[ws0.readyState];
9925 gj_addlog(ws0_log, 'stat '+ws0.readyState+'('+cst+') : GJ Linked\n');
9926
9927 ws0.addEventListener('error', function(event){
9928   gj_addlog(ws0_log, 'stat error : transport error?\n');
9929 });
9930 ws0.addEventListener('open', function(event){
9931   GJLinkView.style.zIndex = 10000;
9932   //console.log('#GJLinkView.id'+'.zIndex='+GJLinkView.style.zIndex);
9933   date1 = new Date().getTime();
9934   date2 = (date1 / 1000).toFixed(3);
9935   seed = date1.toString(16);
9936
9937   // user name and key
9938   user = document.getElementById('gj_user').value;
9939   if (user.length == 0 ){
9940     gj_user.value = 'nemo';
9941     user = 'nemo';
9942
9943     key1 = document.getElementById('gj_ukey').bl_plainText;
9944     ukey = gjkey_hash(seed+user+key1).toString(16);
9945
9946     // session name and key
9947     chan = document.getElementById('gj_chan').value;
9948     if (chan.length == 0 ){
9949       gj_chan.value = 'main';
9950       chan = 'main';
9951     }
9952     key2 = document.getElementById('gj_ckey').bl_plainText;
9953     ckey = gjkey_hash(seed+chan+key2).toString(16);
9954
9955     msg = date2 + ' JOIN ' + user + '|' + chan + ' ' + ukey + ':' + ckey;
9956     gj_addlog(ws0_log, 'send '+msg+'\n');
9957     ws0.send(msg);
9958
9959     target.value = 'Leave';
9960     //console.log('["+date2+"] #' +target.id+' '+target.value+'\n');
9961     //gj_addlog(ws0_log, label '+target.value+'\n');
9962   });
9963   ws0.addEventListener('message', function(event){
9964     now = (new Date().getTime() / 1000).toFixed(3);
9965     msg = event.data;
9966     if ( false ){
9967       gj_addlog(ws0_log, 'recv '+msg+'\n');
9968     }
9969     argv = msg.split(' ');
9970     tstamp = argv[0];
9971     argv.shift();
9972     if ( argv[0] == 'reload' ){
9973       location.reload();
9974     }
9975     gjcmd = argv[0];
9976     otstamp = '';
9977     if (gjcmd == 'CAST' ){ // from reflector
9978       otstamp = argv[0];
9979       argv.shift(); // original time stamp
9980       ofrom = argv[0];
9981       argv.shift(); // original from
9982     }
9983     argv.shift(); // command
9984     from = argv[0];
9985     argv.shift(); // from|to
9986     cmd1 = argv[0];
9987     argv.shift(); // xxxx command
9988
9989     if ( false ){
9990       gj_addlog(ws0_log, '--'
9991         + ' tstamp='+tstamp
9992         + ',gjcmd='+gjcmd
9993         + ',from='+from
9994         + ',cmd=['+cmd1+']'
9995         + ' '+argv+'\n');
9996     }
9997     if ( cmd1 == 'auth' ){
9998       // doing authorization required
9999     }
10000     if ( cmd1 == 'echo' ){
10001       now = (new Date().getTime() / 1000).toFixed(3);
10002       msg = now+ ' +RESP '+argv.join(' ');
10003       gj_addlog(ws0_log, 'send '+msg+'\n');
10004       ws0.send(msg);
10005     }
10006     if ( cmd1 == 'eval' ){
10007       argv.shift();
10008       js = argv.join(' ');
10009       ret = eval(js); // <----- eval()
10010       gj_addlog(ws0_log, 'eval '+js+' = '+ret+'\n');
10011       now = (new Date().getTime() / 1000).toFixed(3);
10012       msg = now + ' ' + 'RESP ' + ret;
10013       ws0.send(msg);
10014       gj_addlog(ws0_log, 'send '+msg+'\n');
10015     }
10016     if ( cmd1 == 'DRAW' ){
10017       if ( false ){
10018         gj_addlog(ws0_log, 'DRAW '+argv[0]+'\n')
10019       }
10020       Pointillism_RemoteDraw(argv[0]);
10021     }
10022   });
10023   ws0.addEventListener('close', function(event){
10024     if ( GJ_Channel == null ){
10025       gj_addlog(ws0_log, 'stat OK : GJ UnLinked\n');
10026       return;
10027     }
10028     GJ_Channel.close();
10029     GJ_Channel = null;
10030     target.value = 'Join';
10031     gj_addlog(ws0_log, 'stat error : close : GJ UnLinked unexpectedly\n');
10032   });
10033 }
10034 function GJ_BcastMessageUserPass(user,chan,msgbody){
10035   now = (new Date().getTime() / 1000).toFixed(3);
10036   msg = now + ' RESP '+user+' '+chan+' '+msgbody;
10037   if ( false ){
10038     gj_addlog(GJ_Log, 'send '+msg+'\n');
10039   }
10040   GJ_Channel.send(msg);
10041 }
10042 function GJ_BcastMessage(msgbody){
10043   if ( GJ_Channel == null ){
10044     gj_addlog(ws0_log, 'stat error : send : GJ not Linked\n');
10045     return;
10046   }
10047   //target = event.target;
10048   user = document.getElementById('gj_user').value;
10049   chan = document.getElementById('gj_chan').value;
10050   GJ_BcastMessageUserPass(user,chan,msgbody);
10051 }
10052 function GJ_SendMessageUserPass(user,chan,msgbody){
10053   now = (new Date().getTime() / 1000).toFixed(3);
10054   msg = now + ' RESP '+user+' '+chan+' '+msgbody;
10055   gj_addlog(GJ_Log, 'send '+msg+'\n');
10056   GJ_Channel.send(msg);
10057 }
10058 function GJ_SendMessage(msgbody){
10059   if ( GJ_Channel == null ){
10060     gj_addlog(ws0_log, 'stat error : send : GJ not Linked\n');
10061     return;
10062   }
10063   //target = event.target;
10064   user = document.getElementById('gj_user').value;
10065   chan = document.getElementById('gj_chan').value;
10066   GJ_SendMessageUserPass(user,chan,msgbody);
10067 }
10068 function GJ_Send(){
10069   msgbody = gj_sendText.value;
10070   GJ_SendMessage(msgbody);
10071 }
10072 }
10073 </script>
10074 <!-- ----- GJLINK ----->
10075 <!--
10076 - User can subscribe to a channel
10077 - A channel will be broadcasted
10078 - A channel can be a pattern (regular expression)
10079 - User is like From:(me) and channel is like To: or Recipient:
10080 - like VIABUS
10081 - watch message with SENDME, WATCH, CATCH, HEAR, or so
10082 - routing with path expression or name pattern (with routing with DNS like system)
10083 -->
10084 */
10085
10086 <span id="GJLinkGolang">
10087 // <details id="GshWebSocket"><summary>Golang / JavaScript Link</summary>
10088 // <span class="gsh-src"><!-- { -->

```

```

10089// 2020-0920 created
10090// <a href="https://pkgs.go.dev/golang.org/x/net/websocket">WS</a>
10091// <a href="https://godev.org/pkg/golang.org/x/net/websocket">WS</a>
10092// INSTALL: go get golang.org/x/net/websocket
10093// INSTALL: sudo (apt,yum) install git (if git is not installed yet)
10094// import "golang.org/x/net/websocket"
10095const gshws_origin = "https://localhost:9999"
10096const gshws_server = "localhost:9999"
10097const gshws_port = 9999
10098const gshws_path = "glink1"
10099const gshws_url = "ws://" + gshws_server + "/" + gshws_path
10100const GSHWS_MSGSIZE = (8*1024)
10101func fmtstring(fmts string, params ...interface{}) (string) {
10102    return fmt.Sprintf(fmts, params...)
10103}
10104func GSHWS_MARK(what string) (string) {
10105    now := time.Now()
10106    us := fmtstring("%06d", now.Nanosecond() / 1000)
10107    mark := ""
10108    if (!AtConsoleLineTop) {
10109        mark = "\n"
10110        AtConsoleLineTop = true
10111    }
10112    mark += "[" + now.Format(time.Stamp) + "." + us + "]" -GJ- + what + " : "
10113    return mark
10114}
10115func gchk(what string, err error) {
10116    if (err != nil) {
10117        panic(GSHWS_MARK(what) + err.Error())
10118    }
10119}
10120func glog(what string, fmts string, params ...interface{}) {
10121    fmt.Print(GSHWS_MARK(what))
10122    fmt.Printf(fmts + "\n", params...)
10123}
10124}
10125var WSV = []*websocket.Conn{}
10126func jsend(argv []string) {
10127    if len(argv) <= 1 {
10128        fmt.Printf("--I] %v [-m] command arguments\n", argv[0])
10129        return
10130    }
10131    argv = argv[1:]
10132    if (len(WSV) == 0) {
10133        fmt.Printf("--Ej-- No link now\n")
10134        return
10135    }
10136    if (1 < len(WSV)) {
10137        fmt.Printf("--Ij-- multiple links (%v)\n", len(WSV))
10138    }
10139}
10140multicast := false // should be filtered with regexp
10141if (0 < len(argv) && argv[0] == "-m") {
10142    multicast = true
10143    argv = argv[1:]
10144}
10145args := strings.Join(argv, " ")
10146}
10147now := time.Now()
10148msec := now.UnixNano() / 1000000;
10149tstamp := fmtstring("%.3f", float64(msec)/1000.0)
10150msg := fmtstring("%v SEND gshell* %v", tstamp, args)
10151}
10152if (multicast) {
10153    for i, ws := range WSV {
10154        wn, werr := ws.Write([]byte(msg))
10155        if (werr != nil) {
10156            fmt.Printf("[%v] wn=%v, werr=%v\n", i, wn, werr)
10157        }
10158        glog("SQ", fmtstring("(%v) %v", wn, msg))
10159    }
10160} else {
10161    i := 0
10162    ws := WSV[i]
10163    wn, werr := ws.Write([]byte(msg))
10164    if (werr != nil) {
10165        fmt.Printf("[%v] wn=%v, werr=%v\n", i, wn, werr)
10166    }
10167    glog("SQ", fmtstring("(%v) %v", wn, msg))
10168}
10169}
10170func ws_broadcast(msg string) (wn int, werr error) {
10171    for i, ws := range WSV {
10172        wn, werr := ws.Write([]byte(msg))
10173        if (werr != nil) {
10174            fmt.Printf("[%v] wn=%v, werr=%v\n", i, wn, werr)
10175        }
10176        glog("SQ", fmtstring("(%v) %v", wn, msg))
10177    }
10178    return wn, werr;
10179}
10180func serv1(ws *websocket.Conn) {
10181    WSV = append(WSV, ws)
10182    //fmt.Print("\n")
10183    glog("CO", "accepted connections[%v]", len(WSV))
10184    //remoteAddr := ws.RemoteAddr
10185    //fmt.Printf("-- accepted %v\n", remoteAddr)
10186    //fmt.Printf("-- accepted %v\n", ws.Config())
10187    //fmt.Printf("-- accepted %v\n", ws.Config().Header)
10188    //fmt.Printf("-- accepted %v // %v\n", ws, serv1)
10189}
10190var reqb = make([]byte, GSHWS_MSGSIZE)
10191for {
10192    rn, rerr := ws.Read(reqb)
10193    if (rerr != nil || rn < 0) {
10194        glog("SQ", fmtstring("(%v) %v", rn, rerr))
10195        break
10196    }
10197    req := string(reqb[0:rn])
10198    glog("SQ", fmtstring("(%v) %v", rn, req))
10199}
10200margv := strings.Split(req, " ");
10201margv = margv[1:]
10202if (0 < len(margv)) {
10203    if (margv[0] == "RESP") {
10204        // should forward to the destination
10205        continue;
10206    }
10207}
10208now := time.Now()
10209msec := now.UnixNano() / 1000000;
10210tstamp := fmtstring("%.3f", float64(msec)/1000.0)
10211res := fmtstring("%v +\"CAST+\" %v", tstamp, req)
10212}
10213wn := 0;
10214werr := error(nil);
10215if (0 < len(margv) && margv[0] == "BCAST") {
10216    wn, werr = ws_broadcast(res);
10217} else {
10218    wn, werr = ws.Write([]byte(res))
10219}
10220gchk("SE", werr)
10221glog("SR", fmtstring("(%v) %v", wn, string(res)))
10222}
10223glog("SF", "WS response finish")
10224}
10225wsv = []*websocket.Conn{}
10226wsx := 0
10227for i, v := range WSV {
10228    if (v != ws) {
10229        wsx = i
10230        wsv = append(wsv, v)
10231    }
10232}
10233Wsv = wsv
10234//glog("CO", "closed %v", ws)
10235glog("CO", "closed connection [%v/%v]", wsx+1, len(WSV)+1)
10236ws.Close()
10237}
10238}
10239// url := [scheme://host[:port]][/path]
10240func decomp_URL(url string) {
10241}
10242}
10243func full_wURL() {
10244}
10245func gj_server(argv []string) {
10246    gjserv := gshws_url
10247    gjport := gshws_server
10248    gjpath := gshws_path
10249    gjscheme := "ws"
10250}
10251//cmd := argv[0]
10252argv = argv[1:]
10253if (1 <= len(argv)) {
10254    serv := argv[0]
10255    if (0 < strings.Index(serv, "://")) {
10256        schemev := strings.Split(serv, "://")
10257        gjscheme = schemev[0]
10258        serv = schemev[1]
10259    }
10260    if (0 < strings.Index(serv, "/")) {
10261        pathv := strings.Split(serv, "/")
10262        serv = pathv[0]
10263        gjpath = pathv[1]
10264    }
10265    servv := strings.Split(serv, ":")
10266    host := "localhost"
10267    port := 9999
10268}

```

```

10266     if( servv[0] != "" ){
10267         host = servv[0]
10268     }
10269     if( len(servv) == 2 ){
10270         fmt.Fprintf(servv[1], "%d", &port)
10271     }
10272     //glog("LC", "hostport=%v (%v : %v)", servv, host, port)
10273     gjport = fmt.Sprintf("%v:%v", host, port)
10274     gjserv = gjscheme + "://" + gjport + "/" + gjpath
10275 }
10276 glog("LS", fmtstring("listening at %v", gjserv))
10277 http.Handle("/"+gjpath, websocket.Handler(serv1))
10278 err := error(nil)
10279 if( gjscheme == "wss" ){
10280     // https://golang.org/pkg/net/http/#ListenAndServeTLS
10281     //err = http.ListenAndServeTLS(gjport, nil)
10282 }else{
10283     err = http.ListenAndServe(gjport, nil)
10284 }
10285 gchk("LE", err)
10286 }
10287 }
10288
10289 func gj_client(argv []string) {
10290     glog("CS", fmtstring("connecting to %v", gshws_url))
10291     ws, err := websocket.Dial(gshws_url, "", gshws_origin)
10292     gchk("C", err)
10293
10294     var resb = make([]byte, GSHWS_MSGSIZE)
10295     for qi := 0; qi < 3; qi++ {
10296         req := fmtstring("Hello, GShell! (%v)", qi)
10297         wn, werr := ws.Write([]byte(req))
10298         glog("QE", fmtstring("(%v) %v", wn, req))
10299         gchk("QE", werr)
10300         rn, rerr := ws.Read(resb)
10301         glog("RE", rerr)
10302         glog("RM", fmtstring("(%v) %v", rn, string(resb)))
10303     }
10304     glog("CF", "WS request finish")
10305 }
10306
10307 //</span><!-- end of class="gsh-src" -->
10308 //</details></span>
10309 /*
10310 <details id="GJLink_Section"><summary>GJ Link</summary>
10311 <span id="GJLinkView" class="GJLinkView">
10312 <p>
10313 <note class="GJNote">Execute command "gsh gj server" on the localhost and push the Join button:</note>
10314 </p>
10315 <span id="GJLink_1">
10316 <div id="GJLink_ServerSet"></div>
10317 <div>
10318 <input id="gj_join" type="button" class="joinButton" onclick="GJ_Join()" value="Join">
10319 <span id="GJLink_Account"></span>
10320 </div>
10321 <div>
10322 <input id="gj_sendButton" type="button" class="joinButton SendButton" onclick="GJ_Send()" value="Send" data-bodyid="gj_sendText">
10323 </div>
10324 <span id="GJLink_SendArea"></span>
10325 </div>
10326 <div id="ws0_log_container"></div>
10327 </span>
10328 </span>
10329 </span>
10330 </script>
10331 function setupGJLinkArea(){
10332     GJLink_ServerSet.innerHTML = '<'+<span id="gj_serv_label"
10333     + " class="textField textLabel">Server: </span>
10334     + '<'+<span id="gj_serv" class="textField textURL" contenteditable><'+</span>';
10335
10336     GJLink_Account.innerHTML = '<'+<textarea id="gj_user" class="textField"><'+</textarea>
10337     + '<'+<textarea id="gj_ukey" class="textField"><'+</textarea>
10338     + '<'+<textarea id="gj_chan" class="textField"><'+</textarea>
10339     + '<'+<textarea id="gj_ckekey" class="textField"><'+</textarea>';
10340
10341     GJLink_SendArea.innerHTML =
10342     '<'+<textarea id="gj_sendText" class="textField MsgText" cols=60 rows=2><'+</textarea>';
10343
10344     ws0_log_container.innerHTML = '<'+<textarea id="ws0_log" class="ws0_log"
10345     + " cols=100 rows=10 spellcheck="false"><'+</textarea>';
10346 }
10347
10348 function clearGJLinkArea(){
10349     GJLink_ServerSet.innerHTML = "";
10350     GJLink_Account.innerHTML = "";
10351     GJLink_SendArea.innerHTML = "";
10352     ws0_log_container.innerHTML = "";
10353 }
10354 </script>
10355
10356 </script>
10357 function SetupGJLink(){
10358     setupGJLinkArea();
10359     SetupVisibleText(GJLink_l_gj_serv, 'GJLink5v');
10360     SetupVisibleText(GJLink_l_gj_user, 'UserName');
10361     SetupBlinderText(GJLink_l_gj_ukey, 'UserKey');
10362     SetupVisibleText(GJLink_l_gj_chan, 'ChannelName');
10363     SetupBlinderText(GJLink_l_gj_ckekey, 'ChannelKey');
10364     SetupVisibleText(GJLink_l_gj_sendText, 'Message');
10365     gj_serv.innerHTML = 'ws://localhost:9999/gjlink1'
10366 }
10367
10368 function GJLink init(){
10369     SetupGJLink();
10370 }
10371
10372 function iselem(eid){
10373     return document.getElementById(eid);
10374 }
10375
10376 function DestroyGJLink(){
10377     clearGJLinkArea();
10378     if( iselem('gj_user') ){
10379         return;
10380     }
10381     if( gj_serv_label.parentNode != gj_user ){
10382         return;
10383     }
10384     if( iselem('gj_serv_label') ) gj_user.parentNode.removeChild(gj_serv_label);
10385     if( iselem('gj_serv') ) gj_user.parentNode.removeChild(gj_serv);
10386     if( iselem('gj_user') ) gj_user.parentNode.removeChild(gj_user);
10387     if( iselem('gj_ukey') ) gj_ukey.parentNode.removeChild(gj_ukey);
10388     if( iselem('gj_chan') ) gj_chan.parentNode.removeChild(gj_chan);
10389     if( iselem('gj_ckekey') ) gj_ckekey.parentNode.removeChild(gj_ckekey);
10390     if( iselem('gj_sendText') ) gj_sendText.parentNode.removeChild(gj_sendText);
10391     if( iselem('ws0_log') ) ws0_log.parentNode.removeChild(ws0_log);
10392 }
10393
10394 DestroyGJLink = DestroyGJLink1;
10395 </script>
10396
10397 /*
10398 <style>
10399 .GJDigest {
10400     display:none;
10401 }
10402 </style>
10403 <script id="HtmlCodeview-script">
10404 function showNodeAsHtmlSourceX(otxa,code,prefix,postfix,sign){
10405     txa = document.createElement('textarea');
10406     txa.id = otxa.id;
10407     txa.setAttribute('class', 'HtmlCodeviewText');
10408     otxa.parentNode.replaceChild(txa,otxa);
10409     txa.setAttribute('spellcheck', 'false');
10410     //txa.value = code.innerHTML;
10411     //txa.innerHTML = code.innerHTML;
10412     txa.innerHTML = prefix + code.outerHTML + postfix;
10413     if( sign ){
10414         text = txa.value;
10415         tlen = txa.value.length;
10416         digest = strCRC32(text,tlen) + ' ' + tlen
10417         + ' ' + code.id + ' (' + DateShort() + ')';
10418         //alert('digest: '+digest);
10419         console.log('digest: '+digest);
10420         txa.innerHTML += '</'+<span class="GJDigest">+digest+<'+</span>\n';
10421     }
10422     txa.style.display = "block";
10423     txa.style.width = "100%";
10424     txa.style.height = "300px";
10425 }
10426
10427 function showHtmlCodeX(otxa,code,prefix,postfix,sign){
10428     if( event.target.value == 'ShowCode' ){
10429         showNodeAsHtmlSourceX(otxa,code,prefix,postfix,sign);
10430         event.target.value = 'HideCode';
10431     }else{
10432         otxa.style.display = "none";
10433         event.target.value = 'ShowCode';
10434     }
10435 }
10436
10437 function showNodeAsHtmlSource(otxa,code){
10438     showNodeAsHtmlSourceX(otxa,code,'','');
10439 }
10440
10441 function showHtmlCode(otxa,code){
10442     if( event.target.value == 'ShowCode' ){
10443         showNodeAsHtmlSource(otxa,code);
10444         event.target.value = 'HideCode';
10445     }else{
10446         otxa.style.display = "none";
10447     }
10448 }

```

```

10443     event.target.value = 'ShowCode';
10444 }
10445 }
10446</script>
10447<style id="HtmlCodeview-style">
10448 .HtmlCodeviewText {
10449     font-size:10pt;
10450     font-family:Courier New;
10451     white-space:pre;
10452 }
10453 .HtmlCodeViewButton {
10454     padding:2pt 1important;
10455     line-height:1.1 1important;
10456     border:2px inset #bbb 1important;
10457     font-size:11pt 1important;
10458     font-weight:normal 1important;
10459     font-family:Georgia 1important;
10460     border-radius:3px 1important;
10461     color:#ddd; background-color:#228 1important;
10462 }
10463</style>
10464*/
10465
10466<details><summary>Live HTML Snapshot</summary>
10467<span id="LiveHTML">
10468<!-- ----- HTML Snapshot: Edit, save and load // 2020-0924 SatoxITS { -->
10469<div class="GshMenu">
10470<span class="GshMenu" onclick="html_edit();">Edit</span>
10471<span class="GshMenu" onclick="html_save();">Save</span>
10472<span class="GshMenu" onclick="html_load();">Load</span>
10473<span class="GshMenu" onclick="html_ver0();">Ver</span>
10474</div>
10475<div>
10476<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="showLiveHTMLCode()">
10477<span id="LiveHTML_Codeview"></span>
10478</div>
10479<script id="LiveHTMLScript">
10480 function showLiveHTMLCode(){
10481     showHtmlCode(LiveHTML_Codeview,LiveHTML);
10482 }
10483
10484 var editable = false;
10485 var savSuppressGJShell = false;
10486 function ToggleEditMode(){
10487     editable = !editable;
10488     if(editable){
10489         savSuppressGJShell = SuppressGJShell;
10490         SuppressGJShell = true;
10491         gsh.setAttribute('contenteditable','true');
10492         GshMenuEdit.innerHTML = 'Lock';
10493         GshMenuEdit.style.color = 'rgba(255,0,0,1)';
10494         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10495     }else{
10496         SuppressGJShell = savSuppressGJShell;
10497         gsh.setAttribute('contenteditable','false');
10498         GshMenuEdit.innerHTML = 'Edit';
10499         GshMenuEdit.style.color = 'rgba(16,160,16,1)';
10500         GshMenuEdit.style.backgroundColor = 'rgba(255,255,255,1)';
10501     }
10502 }
10503 function html_edit(){
10504     ToggleEditMode();
10505 }
10506
10507 // Live HTML (DOM) Snapshot onto browser's localStorage
10508 // 2020-0923 SatoxITS
10509 var htRoot = gsh // -- Element-ID, should be selectable
10510 const snappedHTML = 'SnappedHTML'; // Item-ID of the HTML data in localStogate
10511 // -- should be a [map] of URL
10512 // -- should be with CSSOM as inline script
10513 const htVersionTag = 'VersionTag'; // VesionTag Element-ID in the HTML (in DOM)
10514 function showVersion(note,w,v,u,t){
10515     w.alert(note+' : '+ v + '\n'
10516         + '-- URL: ' + u + '\n'
10517         + Time() + DateLong(t*1000) + ' ');
10518 }
10519
10520 function html_save(){
10521     u = document.URL;
10522     t = new Date().getTime() / 1000;
10523     v = '<'+span id="+htVersionTag+" data-url="+u+" data-time="+t+">';
10524     w = '<'+span id="+u+'>';
10525     h = v + htRoot.outerHTML;
10526     localStorage.setItem(snappedHTML,h);
10527     showVersion("Saved",window,v,u,t);
10528 }
10529
10530 function html_load(){
10531     h = localStorage.getItem(snappedHTML);
10532     if( h == null ){
10533         alert('No snapshot taken yet');
10534         return;
10535     }
10536     w = window.open('','');
10537     d = w.document;
10538     d.write(h);
10539     w.focus();
10540     html_ver1("Loaded",w,d);
10541 }
10542
10543 function html_ver1(note,w,d){
10544     if( (v = d.getElementById(htVersionTag)) != null ){
10545         h = v.outerHTML;
10546         u = v.getAttribute('data-url');
10547         t = v.getAttribute('data-time');
10548     }else{
10549         h = 'No version info. in the page';
10550         u = '';
10551         t = 0;
10552     }
10553     showVersion(note,w,v,u,t);
10554 }
10555
10556 function html_ver0(){
10557     html_ver1("Version",window,document);
10558 }
10559</script>
10560</div>
10561<!-- LiveHTML } -->
10562</span>
10563</details>
10564*/
10565
10566<details><summary>Event sharing</summary>
10567<span id="EventSharingCodespan">
10568<!-- ----- Event sharing // 2020-0925 SatoxITS { -->
10569<div id="iftestTemplate" class="iftest" hidden="">
10570<style>_iftestbody{ color:#f22;font-family:Georgia;font-size:10pt;}</style>
10571<span id="frameMsg" class="iftestbody" onclick="frameClick()"><script>
10572     function docadd(txt){
10573         document.body.append(txt);
10574         window.scrollTo(0,100000);
10575     }
10576     function frameClick(){ 'x'+event.x + 'y'+event.y+'';
10577         //docadd('Got Click on #' +event.target.id+ '+xy+ '\n');
10578         docadd('Got Click on #' +Fid.value+ '+xy+ '\n');
10579         window.scrollTo(0,100000);
10580         window.parent.postMessage('OnClick: '+xy, '*');
10581     }
10582     function frameMouseMove(){
10583         if( false ){
10584             document.body.append('Mousemove on #' +event.target.id+ ' '
10585                 + 'x'+event.x + 'y'+event.y + '\n');
10586             peerWin = window.frames.iframe1;
10587             document.body.append('Send to peer #' +peerWin+ ' ' + '\n');
10588             window.scrollTo(0,100000);
10589             peerWin.postMessage('Hi!', '*');
10590         }
10591     }
10592     function frameKeydown(){
10593         msg = 'Got Keydown: #' +Fid.value+ ', (' +event.code+ ')';
10594         docadd(msg+'\n');
10595         window.parent.postMessage(msg, '*');
10596     }
10597     function frameOnMessage(){
10598         docadd('Message + event.data + '\n');
10599         window.scrollTo(0,100000);
10600     }
10601     if( document.getElementById('Fid') ){
10602         framebody.id = Fid.value;
10603         h = '<'+style*>';
10604         h += 'font-size:10pt;white-space:pre-wrap;';
10605         h += 'font-family:Courier New;';
10606         h += '<'+/style*>';
10607         h += 'I am '+Fid.value+'\n';
10608         document.write(h);
10609         window.addEventListener('click',frameClick);
10610         window.addEventListener('keydown',frameKeydown);
10611         window.addEventListener('message',frameOnMessage);
10612         window.addEventListener('mousemove',frameMouseMove);
10613         window.parent.postMessage('Hi parent, I am '+Fid.value, '*');
10614     }
10615 }
10616</script></span></div>
10617
10618<style>.iframeTest{margin:3px;resize:both;width:370px;height:120px;}</style>
10619<h2>Inter-window communicaiton</h2>

```

```

1062<note>
1063frame0 >>> frame1 and frame2<br>
1064frame1 >>> frame0 and frame2<br>
1065frame2 >>> frame0 and frame1<br>
1066</note>
1067<div id="iframe-test">
1068<pre id="iframeHost" style="border:1px;font-family:Courier New;font-size:10pt;"></pre>
1069<iframe id="iframe0" title="iframe0" class="iframeTest" style=""></iframe>
1070<iframe id="iframe1" title="iframe1" class="iframeTest"></iframe>
1071<iframe id="iframe2" title="iframe2" class="iframeTest"></iframe>
1072</div>
1073<script id="ifo-test-script">
1074function InterFrameComm_init(){
1075  setupFrames0();
1076  setupFrames1();
1077}
1078function setFrameSrcdoc(dst,src){
1079  if( true ){
1080    dst.contentWindow.document.write(src);
1081    // this makes browser wait close, and crash if accumulated !?
1082    // so it should be closed after write
1083    dst.contentWindow.document.close();
1084  }else{
1085    // to be erased before source dump
1086    // but should be set for live snapshot
1087    dst.srcdoc = src;
1088  }
1089}
1090function setupFrames0(){
1091  tbody = iframe0.contentWindow.document.body;
1092  iframe0.style.width = "75px";
1093  //iframeHost.innerHTML = "Message exchange at iframes' host:\n";
1094  window.addEventListener('message',messageFromChild);
1095}
1096if0 = '';
1097if0 += "<pre style='font-family:Courier New;'>";
1098if0 += "<input id='Pid' value='iframe0'>";
1099if0 += iftestTemplate.innerHTML;
1100setFrameSrcdoc(iframe0,if0);
1101}
1102function clickOnChild(){
1103  console.log('clickOn #' +this.id);
1104}
1105function moveOnChild(){
1106  console.log('moveOn #' +this.id);
1107}
1108iframe0.contentWindow.document.body.style.setProperty('white-space','pre');
1109iframe0.contentWindow.document.body.style.setProperty('font-size','9pt');
1110}
1111function setupFrames1(){
1112  if1 = <input id="Pid" value="iframe1">;
1113  if1 += iftestTemplate.innerHTML;
1114  setFrameSrcdoc(iframe1,if1);
1115  //iframe1.name = 'iframe1'; // this seems break contentWindow
1116}
1117if2 = <input id="Pid" value="iframe2">;
1118if2 += iftestTemplate.innerHTML;
1119setFrameSrcdoc(iframe2,if2);
1120}
1121iframe1.addEventListener('message',messageFromChild);
1122//iframe1.addEventListener('mouseover',moveOnChild);
1123iframe2.addEventListener('message',messageFromChild);
1124//iframe2.addEventListener('mouseover',moveOnChild);
1125iframe1.contentWindow.postMessage(['parent0 Hi iframe1 -- from parent.','*']);
1126//iframe1.contentWindow.postMessage(['parent0 Hi iframe2 -- from parent.','*']);
1127//iframe2.contentWindow.postMessage(['parent0 Hi iframe2 -- from parent.','*']);
1128//iframe2.contentWindow.postMessage(['Your peer is '+iframe1.contentWindow,'*']);
1129}
1130function messageFromChild(){
1131  from = null;
1132  forw = null;
1133  if( event.source == iframe0.contentWindow ){
1134    from = 'iframe0';
1135    forw = 'iframe1';
1136  }else
1137  if( event.source == iframe1.contentWindow ){
1138    from = 'iframe1';
1139    forw = 'iframe2';
1140  }else
1141  if( event.source == iframe2.contentWindow ){
1142    from = 'iframe2';
1143    forw = 'iframe1';
1144  }else
1145  {
1146    iframeHost.innerHTML += 'Message [unknown] '
1147    + ' orig=' + event.origin
1148    + ' data=' + event.data
1149    //+ ' from=' + event.source
1150    ;
1151  }
1152  msglog1 = from + event.data + ' --- '
1153    + from + event.source
1154    + ' orig=' + event.origin
1155    + ' name=' + event.source.name
1156    //+ ' ports=' + event.ports
1157    //+ ' euid=' + event.lastEventId
1158    + '\n'
1159    ;
1160  if( true ){
1161    if( forw == 'iframe1' || forw == 'iframe2' ){
1162      iframe1.contentWindow.postMessage(from+event.data);
1163    }
1164    if( forw == 'iframe2' || forw == 'iframe1' ){
1165      iframe2.contentWindow.postMessage(from+event.data);
1166    }
1167  }
1168  txtadd0(msglog1);
1169}
1170function txtadd0(txt){
1171  iframe0.contentWindow.document.body.append(txt);
1172  iframe0.contentWindow.scrollTo(0,100000);
1173}
1174}
1175function es_ShowSelf(){
1176  iframe1.setAttribute('src',document.URL);
1177  iframe2.setAttribute('src',document.URL);
1178}
1179</script>
1180<input class="HtmlCodeViewButton" type="button" value="ShowSelf" onclick="es_ShowSelf()">
1181<input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="es_showHtmlCode()">
1182<span id="EventSharingCodeview"></span>
1183<script id="EventSharingScript">
1184function es_showHtmlCode(){
1185  showHtmlCode(EventSharingCodeview,EventSharingCodeSpan);
1186}
1187function DestroyEventSharingCodeview = function(){
1188  //EventSharingCodeview.parentNode.removeChild(EventSharingCodeview);
1189  EventSharingCodeview.innerHTML = "";
1190  iframe0.style = "";
1191  //iframe0.srcdoc = "erased";
1192  //iframe1.srcdoc = "erased";
1193  //iframe2.srcdoc = "erased";
1194}
1195</script>
1196<!-- EventSharing -->
1197</span>
1198</details>
1199</div>
1200<!-- ***** GShell Inside Notification { -->
1201<script id="script-gshell-inside">
1202var notices = 0;
1203function noticeGShellInside(){
1204  ver = '';
1205  if( ver = document.getElementById('GshVersion') ){
1206    ver = ver.innerHTML;
1207  }
1208  console.log('GShell Inside (^~^)' +ver);
1209  notices += 1;
1210  if( 2 < notices ){
1211    document.removeEventListener('mousemove',noticeGShellInside);
1212  }
1213}
1214document.addEventListener('mousemove',noticeGShellInside);
1215noticeGShellInside();
1216}
1217const FooterName = 'GshFooter'
1218function DestroyFooter(){
1219  if( (footer = document.getElementById(FooterName)) != null ){
1220    //footer.parentNode.removeChild(footer);
1221    empty = document.createElement('div');
1222    empty.id = 'GshFooter0';
1223    footer.parentNode.replaceChild(empty,footer);
1224  }
1225}
1226function showFooter(){
1227  footer = document.createElement('div');
1228  footer.id = FooterName;
1229  footer.style.backgroundImage = "url('"+ITSmoreQR+"')";
1230  //GshFooter0.parentNode.appendChild(footer);
1231  if( document.getElementById('GshFooter0') != null ){
1232    GshFooter0.parentNode.replaceChild(footer,GshFooter0);
1233  }
1234}
1235}

```

```

1079 </script>
1079<!-- } -->
1079<!--
1080<!--
10801 border:20px inset #888;
10802-->
10803
10804<span id="VirtualDesktopCodeSpan">
10805</span>
10806</span>
10807<div id="VirtualDesktopDetails"><summary>Virtual Desktop</summary>
10808<div id="Web Virtual Desktop // 2020-0927 SatoxITS { -->
10809<div style="border: 2px solid #ccc; padding: 5px; width: 100%; height: 100%; position: relative; background-color: #f0f0f0; border-radius: 10px; overflow: hidden;">
10810VirtualSpace {
10811 z-index: 1;
10812 width: 1280px !important; height: 720px !important;
10813 border: 1px solid #ccc;
10814 background-color: rgba(32, 32, 160, 0.8);
10815 background-image: url("WD-WallPaper03.png");
10816 background-size: 100% 100%;
10817 color: #22a; font-family: Georgia; font-size: 10pt;
10818 overflow: scroll;
10819 }
10820VirtualGrid {
10821 z-index: 0;
10822 position: absolute;
10823 width: 800px; height: 500px;
10824 border: 1px solid #fff;
10825 color: rgba(192, 255, 192, 0.8);
10826 font-family: Georgia, Courier New;
10827 text-align: right;
10828 vertical-align: middle;
10829 font-size: 200px;
10830 text-shadow: 4px 4px #ccc;
10831 }
10832.WD_GridScroll {
10833 z-index: 100000;
10834 background-color: rgba(200, 200, 200, 0.1);
10835 }
10836.VirtualDesktop {
10837 z-index: 0;
10838 position: relative;
10839 resize: both !important;
10840 overflow: scroll;
10841 display: block;
10842 min-width: 20px !important; min-height: 60px !important;
10843 width: 800px;
10844 height: 500px;
10845 border: 10px inset #228;
10846 border-width: 30px; border-radius: 20px;
10847 background-image: url("WD-WallPaper03.png");
10848 background-size: 100% 100%;
10849 color: #22a; font-family: Georgia; font-size: 10pt;
10850 }
10851.comment {
10852 // overflow: scroll seems to bound children's view in the element span
10853 // specifying overflow seems fix the position of the element
10854 }
10855.VirtualBrowserSpan {
10856 z-index: 10;
10857 border: 0.5px dashed #fff !important;
10858 border-color: rgba(255, 255, 255, 0.5) !important;
10859 position: relative;
10860 left: 100px;
10861 top: 100px;
10862 display: block;
10863 resize: both !important;
10864 width: 540px;
10865 height: 320px;
10866 min-width: 40px !important; min-height: 20px !important;
10867 max-width: 1210px !important; max-height: 2880px !important;
10868 background-color: rgba(255, 200, 255, 0.1);
10869 overflow: scroll;
10870 }
10871.xVirtualBrowserLocationBar: focus {
10872 color: #f00;
10873 background-color: rgba(255, 128, 128, 0.2);
10874 }
10875.xVirtualBrowserLocationBar: active {
10876 color: #f00;
10877 background-color: rgba(128, 255, 128, 0.2);
10878 }
10879.a.VirtualBrowserLocation {
10880 color: #ccc !important;
10881 text-decoration: none !important;
10882 }
10883.a.VirtualBrowserLocation: hover {
10884 color: #fff !important;
10885 text-decoration: underline;
10886 }
10887.VirtualBrowserLocationBar {
10888 position: absolute;
10889 z-index: 100000;
10890 display: block;
10891 width: 400px;
10892 height: 20px;
10893 padding-left: 2px;
10894 line-height: 1.1;
10895 vertical-align: middle;
10896 font-size: 14px;
10897 color: #fff;
10898 background-color: rgba(128, 128, 128, 0.2);
10899 font-family: Georgia;
10900 }
10901.VirtualBrowserCommandBar {
10902 position: absolute;
10903 z-index: 200000;
10904 display: inline-block;
10905 width: 60px;
10906 height: 20px;
10907 line-height: 1.1;
10908 vertical-align: middle;
10909 font-size: 14px;
10910 color: #ff4;
10911 background-color: rgba(128, 128, 128, 0.1);
10912 font-family: Georgia;
10913 text-align: left;
10914 left: 404px;
10915 }
10916.VirtualBrowserFrame {
10917 position: absolute;
10918 z-index: 10;
10919 width: 400px; height: 240px;
10920 min-width: 60px; min-height: 30px;
10921 max-width: 5120px; max-height: 2880px;
10922 border-radius: 6px;
10923 background-color: rgba(255, 255, 255, 0.9);
10924 border-top: 20px solid;
10925 border-right: 4px solid;
10926 border-bottom: 10px solid;
10927 }
10928.WinFavicon {
10929 width: 16px;
10930 height: 16px;
10931 margin-right: 3px;
10932 vertical-align: middle;
10933 background-color: rgba(255, 255, 255, 1.0);
10934 }
10935.VirtualDesktopMenuBar {
10936 position: absolute;
10937 z-index: #fff;
10938 font-size: 7pt;
10939 text-align: right;
10940 padding-right: 4px;
10941 background-color: rgba(128, 128, 128, 0.7);
10942 }
10943.VirtualDesktopCalendar {
10944 color: #fff;
10945 font-size: 22pt;
10946 text-align: right;
10947 padding-right: 4px;
10948 background-color: rgba(255, 255, 255, 0.2);
10949 }
10950.xxxinput { width: 260pt !important; line-height: 1.1 !important; margin: 1px;
10951 display: inline !important; font-size: 10pt !important; padding: 1px !important;
10952 }
10953.WD_Config {
10954 display: inline !important;
10955 padding: 2px !important;
10956 font-size: 10pt !important;
10957 width: 60px !important;
10958 height: 12pt !important;
10959 line-height: 1.0pt !important;
10960 height: 15pt !important;
10961 }
10962.WD_Button {
10963 display: inline !important;
10964 padding: 2px !important;
10965 color: #fff !important;
10966 background-color: #228 !important;
10967 font-size: 10pt !important;

```

```

10974 width:60pt !important;
10975 height:12pt !important;
10976 line-height:1.0pt !important;
10977 height:16pt !important;
10978 border:2px inset #44a !important;
10979 }
10980 .WD_Href {
10981   display:inline !important;
10982   padding:2px !important;
10983   font-size:9pt !important;
10984   width:120pt !important;
10985   height:12pt !important;
10986   line-height:1.0pt !important;
10987   height:15pt !important;
10988 }
10989 LiveHtmlCodeviewText {
10990   font-size:10pt;
10991   font-family:Courier New;
10992   xwhite-space:pre;
10993 }
10994 }
10995
10996 .WD_Panel {
10997   x-index:100 !important;
10998   color:#000 !important;
10999   margin-left:25px !important;
11000   width:800px !important;
11001   padding:4px !important;
11002   border:1px solid #888 !important;
11003   border-radius:6px !important;
11004   background-color:rgba(220,220,220,0.9) !important;
11005   font-size:9pt;
11006   font-family:Courier New;
11007 }
11008 .WD_Help {
11009   font-size:10pt !important;
11010   font-family:Courier New;
11011   line-height:1.2 !important;
11012   color:#000 !important;
11013   width:100% !important;
11014   background-color:rgba(240,240,255,0.8) !important;
11015 }
11016 }
11017 .WB_Zoom {
11018 }
11019 }
11020 </style>
11021 <h2>CosmoScreen 0.0.8</h2>
11022 <menu>
11023 <span id="WD_Help_1" class="WD_Help"><b>ScopeControl command keys</b><br>
11024 g ... grid on/off<br>
11025 i ... zoom in<br>
11026 o ... zoom out<br>
11027 s ... save current scope<br>
11028 r ... restore saved scope<br>
11029 </span>
11030 </menu>
11031 <div class="WD_Panel" draggable="true">
11032 <p><!-- should be on the frame of the WD -->
11033 Monitor <input id="WD_Resize_1" class="WD_Button" type="button" value="Resize">
11034 <input id="WD_Height_1" class="WD_Config" type="text">
11035 wall-paper: <a href="#WD-WallPaper03.png" class="WD_Href" contenteditable="true">WD-WallPaper03.png</a>
11036 </p>
11037 <p>
11038 Desktop <input id="WS_Resize_1" class="WD_Button" type="button" value="Resize">
11039 <input id="WS_1_Width" class="WD_Config" type="text">
11040 x <input id="WS_1_Height" class="WD_Config" type="text">
11041 </p>
11042 <p>
11043 Display <input id="WD_Zoom_1" class="WD_Button" type="button" value="Zoom">
11044 <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
11045 x <input id="WD_Zoom_1_MAG" class="WD_Config" type="text" value="1.41421356">
11046 <input id="WD_Zoom_1_OUT" class="WD_Button" type="button" value="ZoomOut">
11047 <input id="WD_Zoom_1_IN" class="WD_Button" type="button" value="ZoomIn">
11048 </p>
11049 <p>
11050 Content <input id="WD_Scroll_1" class="WD_Button" type="button" value="Scroll">
11051 X <input id="WD_Left_1" class="WD_Config" type="text" value="0">
11052 Y <input id="WD_Top_1" class="WD_Config" type="text" value="0">
11053 shift+wheel for horizontal scroll
11054 </p>
11055 <p>
11056 Scopeex <input id="WD_Zoom_1_Restore" class="WD_Button" type="button" value="Restore">
11057 <input id="WD_Zoom_1_RestoreScope" class="WD_Config" type="text" value="Scope0">
11058 | <input id="WD_Zoom_1_Save" class="WD_Button" type="button" value="Save">
11059 > <input id="WD_Zoom_1_SaveScope" class="WD_Config" type="text" value="Scope0">
11060 </p>
11061 <p>
11062 Scopey <input id="WD_Zoom_1_Forw" class="WD_Button" type="button" value="Forw">
11063 <input id="WD_Zoom_1_ForwScope" class="WD_Config" type="text" value="Scope0">
11064 | <input id="WD_Zoom_1_Back" class="WD_Button" type="button" value="Back">
11065 > <input id="WD_Zoom_1_BackScope" class="WD_Config" type="text" value="Scope0">
11066 </p>
11067 <p>
11068 Scopez <input id="WD_Zoom_1_Push" class="WD_Button" type="button" value="Push">
11069 <input id="WD_Zoom_1_PushScope" class="WD_Config" type="text" value="Scope0">
11070 | <input id="WD_Zoom_1_Pop" class="WD_Button" type="button" value="Pop">
11071 > <input id="WD_Zoom_1_PopScope" class="WD_Config" type="text" value="Scope0">
11072 </p>
11073 <p>
11074 Display <input id="WD_Grid_1" class="WD_Button" type="button" value="GridOn">
11075 </p>
11076 <p>
11077 Overflow <input id="WD_Overflow_1" class="WD_Button" type="button" value="scroll">
11078 "scroll" imprisons windows inside the display
11079 </p>
11080 </div>
11081
11082 <div id="VirtualDesktop_1" class="VirtualDesktop" draggable="true" style="" contenteditable="true">
11083 <div id="VirtualDesktop_1_MenuBar" class="VirtualDesktopMenuBar" spellcheck="false">
11084 <span id="CosmoScreen 0.0.8</span></div>
11085 <div id="VirtualDesktop_1_Calendar" class="VirtualDesktopCalendar" >00:00</div>
11086 <div align="right"><div id="VirtualSpace 1.</div>
11087 <div id="VirtualDesktop_1_Content" class="VirtualSpace">
11088
11089 <div id="VirtualBrowser_1" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11090 <div id="VirtualBrowser_1_Location" class="VirtualBrowserLocationBar"></div>
11091 <span id="VirtualBrowser_1_Command" class="VirtualBrowserCommandBar">Reload</span>
11092 <iframe id="VirtualBrowser_1_Frame" class="VirtualBrowserFrame" style=""></iframe>
11093 </div>
11094
11095 <div id="VirtualBrowser_2" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11096 <div id="VirtualBrowser_2_Location" class="VirtualBrowserLocationBar"></div>
11097 <span id="VirtualBrowser_2_Command" class="VirtualBrowserCommandBar">Reload</span>
11098 <iframe id="VirtualBrowser_2_Frame" class="VirtualBrowserFrame" style=""></iframe>
11099 </div>
11100
11101 <div id="VirtualBrowser_3" class="VirtualBrowserSpan" spellcheck="false" draggable="true">
11102 <div id="VirtualBrowser_3_Location" class="VirtualBrowserLocationBar"></div>
11103 <span id="VirtualBrowser_3_Command" class="VirtualBrowserCommandBar">Reload</span>
11104 <iframe id="VirtualBrowser_3_Frame" class="VirtualBrowserFrame" style=""></iframe>
11105 </div>
11106
11107 <div id="VirtualDesktop_1_GridPlane" class="VirtualSpace"></div>
11108 </div>
11109 </div>
11110
11111 <input class="HtmlCodeViewButton" type="button" value="ShowCode" onclick="vd_showHtmlCode()">
11112 <span id="VirtualDesktopCodeview"></span>
11113 <script id="VirtualDesktopCodeScript">
11114 function vd_showHtmlCode(){
11115   codespan = document.getElementById("VirtualDesktopCodeSpan");
11116   showHtmlCode(VirtualDesktopCodeview,codespan);
11117   VirtualDesktopCodeview.setAttribute('class','LiveHtmlCodeviewText');
11118 }
11119
11120 DestroyEventSharingCodeview = function(){
11121   VirtualDesktopCodeview.innerHTML = "";
11122 }
11123
11124 function wdlog(log){
11125   if( GJ_Channel != null ){
11126     GJ_SendMessage("WD "+log);
11127   }
11128   console.log(log);
11129 }
11130
11131 var topMostWin = 10000;
11132 function onEnterWin(e){
11133   t = e.target;
11134   oindex = t.style.zIndex;
11135   //if( oindex == '' ) oindex = 0;
11136   //t.saved_zIndex = oindex;
11137   //t.style.zIndex = 10000;
11138   topMostWin += 1;
11139   t.style.zIndex = topMostWin;
11140   nindex = t.style.zIndex;
11141   wlog("Enter "+t.id+" #"+t.id+" ('+oindex+'->'+nindex+'");
11142   e.stopPropagation();
11143   e.preventDefault();
11144 }
11145
11146 function onClickWin(e) // can detect click on the thick border? t = e.target;
11147   oindex = t.style.zIndex;
11148   topMostWin += 1;
11149   t.style.zIndex = topMostWin;
11150   nindex = t.style.zIndex;
11151   wlog("Click "+t.id+" #"+t.id+" ('+oindex+'->'+nindex+'");
11152   //e.stopPropagation();

```

```

11151 //e.preventDefault();
11152 }
11153 function onLeaveWin(e){
11154     t = e.target;
11155     //oindex = t.style.zIndex;
11156     //nindex = t.saved_zIndex;
11157     //t.style.zIndex = nindex;
11158     //wlog('Leave '+e.target+' #' + e.target.id + '+' + oindex + '-' + nindex');
11159     e.stopPropagation();
11160     e.preventDefault();
11161 }
11162
11163 var WinDragstartX; // event
11164 var WinDragstartY;
11165 var WinDragstartTX; // target
11166 var WinDragstartTY;
11167
11168 function onWinDragstart(e){
11169     WinDragstartX = e.x;
11170     WinDragstartY = e.y;
11171
11172     t = e.target;
11173
11174     //WinDragstartTX = t.getBoundingClientRect().left.toFixed(0)
11175     //WinDragstartTY = t.getBoundingClientRect().top.toFixed(0)
11176     if( t.style.left == '' ){
11177         WinDragstartTX = x0 = 0;
11178         t.style.left = '0px';
11179     }else{
11180         //WinDragstartTX = x0 = Number(t.style.left);
11181         WinDragstartTX = x0 = parseInt(t.style.left);
11182     }
11183     if( t.style.top == '' ){
11184         WinDragstartTY = y0 = 0;
11185         t.style.top = '0px';
11186     }else{
11187         //WinDragstartTY = y0 = Number(t.style.top);
11188         WinDragstartTY = y0 = parseInt(t.style.top);
11189     }
11190     if( true ){ // to be undo
11191         t.wasAtX = WinDragstartTX;
11192         t.wasAtY = WinDragstartTY;
11193     }
11194     wlog('DragSTA #' + t.id
11195         + ' event(' + e.x + ', ' + e.y + ') '
11196         + ' position' + t.style.position
11197         + ' style left,top' + t.style.left + ', ' + t.style.top + ') '
11198     );
11199     e.stopPropagation();
11200     //e.preventDefault();
11201     return true;
11202 }
11203 function onWinDragEvent(wh,e,set,dolog){
11204     t = e.target;
11205     dx = e.x - WinDragstartX;
11206     dy = e.y - WinDragstartY;
11207     nx = WinDragstartTX + dx;
11208     ny = WinDragstartTY + dy;
11209     log = 'Drag '+wh+' #' + t.id
11210         + ' event(' + WinDragstartTX + ', ' + WinDragstartTY + ') '
11211         + ' event(' + e.x + ', ' + e.y + ') '
11212         + ' diff(' + dx + ', ' + dy + ') '
11213         + ' ( ' + nx + ', ' + ny + ') '
11214         + ' ( ' + t.style.left + ', ' + t.style.top + ') '
11215         + ' wasAt(' + t.wasAtX + ', ' + t.wasAtY + ') '
11216     ;
11217     if( e.x != 0 || e.y != 0 ){
11218         if( set == true ){
11219             //t.style.x = nx + 'px'; // not effective
11220             //t.style.y = ny + 'px'; // not effective
11221             t.style.left = nx + 'px';
11222             t.style.top = ny + 'px';
11223             log += ' Set';
11224         }else{
11225             log += ' NotSet';
11226             if( !dolog ){
11227                 log = '';
11228             }
11229         }
11230     }else{
11231         log += ' What?'; // the type is event start?
11232         if( !dolog ){
11233             log = '';
11234         }
11235     }
11236     if( and(dolog, log != '' ) ){
11237         wlog(log);
11238     }
11239     if( true ){
11240         // should be propagated to parent in Firefox ?
11241         e.stopPropagation();
11242     }
11243     e.preventDefault();
11244     return false;
11245 }
11246
11247 function onWinDrag(e){
11248     return onWinDragEvent('Ing',e,true,false);
11249 }
11250 function onWinDragend(e){
11251     return onWinDragEvent('End',e,false,true);
11252 }
11253 function onWinDragexit(e){
11254     return onWinDragEvent('Exit',e,false,true);
11255 }
11256 function onWinDragover(e){
11257     return onWinDragEvent('Over',e,false,true);
11258 }
11259 function onWinDragenter(e){
11260     return onWinDragEvent('Enter',e,false,true);
11261 }
11262 function onWinDragleave(e){
11263     return onWinDragEvent('Leave',e,false,true);
11264 }
11265 function onWinDragdrop(e){
11266     return onWinDragEvent('Drop',e,false,true);
11267 }
11268 function onFaviconChange(e){
11269     wlog('--Favicon #' + e.target.id + ' href=' + e.details.href);
11270 }
11271 var savedSuppressGJShell = false;
11272 function stopGShell(e){
11273     //wlog('enter Gsh STOP');
11274     savedSuppressGJShell = SuppressGJShell;
11275     SuppressGJShell = true;
11276     e.stopPropagation();
11277     e.preventDefault();
11278 }
11279 function contGShell(e){
11280     //wlog('leave Gsh STOP');
11281     SuppressGJShell = savedSuppressGJShell;
11282     e.stopPropagation();
11283     e.preventDefault();
11284 }
11285 function WD_onKeydown(e){
11286     keycode = e.code;
11287     console.log('Keydown #' + e.target.id + ' ' + keycode);
11288     if( keycode == 'KeyG' ){
11289         WD_setGrid(WD_Grid_1);
11290     }else
11291     if( keycode == 'KeyI' ){
11292         WD_doZoomIN();
11293     }else
11294     if( keycode == 'KeyO' ){
11295         WD_doZoomOUT();
11296     }else
11297     if( keycode == 'KeyR' ){
11298         WD_RestoreScope(null);
11299     }else
11300     if( keycode == 'KeyS' ){
11301         WD_SaveScope(null);
11302     }
11303     e.stopPropagation();
11304     e.preventDefault();
11305 }
11306 function WD_onKeyUp(e){
11307     e.stopPropagation();
11308     e.preventDefault();
11309 }
11310 function WD_EventSetup1(){
11311     WirtualDesktop_1.addEventListener('keydown', e => { WD_onKeydown(e); });
11312     WirtualDesktop_1_Content.addEventListener('keydown', e => { WD_onKeydown(e); });
11313     WD_Help_1.addEventListener('keydown', e => { WD_onKeydown(e); });
11314     WD_Help_1.addEventListener('keyup', e => { WD_onKeyUp(e); });
11315 }
11316 function settleWin(s,l,cmd,f,u,w,h,x,y,c,scale){
11317     function WirtualBrowserCommand(e,s,l,cmd,f){
11318         command = cmd.innerHTML;
11319         if( command == "Reload" ){
11320             href_id = e.target.href_id;
11321             d = document.getElementById(href_id);
11322             wlog('href tag #' + href_id + '\n elem #' + href_id + '\n href=' + d);
11323             url = d.innerHTML;
11324             wlog('---- Load href tag #' + href_id + '\n elem #' + href_id + '\n href=' + d
11325                 + '\n url=' + url);
11326             wlog('---- Load target #' + f.id + ' with url=' + url;
11327                 f.src = url;

```



```

11328     }else{
11329         alert('unknown command'+command+' '+e.target.id+' '+l.id+' '+f.id);
11330     }
11331 }
11332 function onKeyDown(e){
11333     if (e.code == 'Enter' ){
11334         e.stopPropagation();
11335         e.preventDefault();
11336     }
11337 }
11338 function onKeyUp(e){
11339     if (e.code == 'Enter' ){
11340         e.stopPropagation();
11341         e.preventDefault();
11342         // should reload immediately ?
11343     }
11344 }
11345
11346 if( false ){
11347     wdllog('start settle VirtualBrowser url='+u +'\n'
11348         + 'id=' + s.id + '\n'
11349         + 'width=' + s.style.width + '\n'
11350         + 'height=' + s.style.height
11351     );
11352 }
11353 // very important for WordPress ??
11354 s.style.width = f.style.width = 501; // for WordPress ...??
11355 s.style.height = f.style.height = 271; // for WordPress ...??
11356 if( false ){
11357     wdllog('midway settle VirtualBrowser url='+u +'\n'
11358         + 'id=' + s.id + '\n'
11359         + 'width=' + s.style.width + '\n'
11360         + 'height=' + s.style.height
11361     );
11362 }
11363 s.width = 502; // for WordPress ...??
11364 s.height = 272; // for WordPress ...??
11365 if( false ){
11366     wdllog('midway-2 settle VirtualBrowser url='+u +'\n'
11367         + 'id=' + s.id + '\n'
11368         + 'span-width=' + s.width + '\n'
11369         + 'span-height=' + s.height
11370     );
11371 }
11372
11373 s.style.width = w + 'px';
11374 s.style.height = h + 'px';
11375 f.style.width = w + 'px';
11376 f.style.height = h + 'px';
11377 //f.style.setProperty('--webkit-transform','scale('+scale+')');
11378 f.style.setProperty('transform','scale('+scale+')');
11379
11380 //wdllog('---l-- u='+u+' width s='+s.style.width+',f='+f.style.width);
11381 //wdllog('---x-- u='+u+' width s='+s.style.width+',f='+f.style.width);
11382 s.setAttribute('draggable','true');
11383 f.setAttribute('draggable','false'); // why necessary?
11384 l.setAttribute('draggable','false'); // why necessary?
11385 cmd.setAttribute('draggable','false'); // why necessary?
11386 s.addEventListener('dragstart', e => { onWinDragstart(e); });
11387 s.addEventListener('drag', e => { onWinDrag(e); });
11388 s.addEventListener('exit', e => { onWinDragexit(e); });
11389 s.addEventListener('dragend', e => { onWinDragend(e); });
11390 s.addEventListener('dragexit', e => { onWinDragexit(e); });
11391 s.addEventListener('dragenter', e => { onWinDragenter(e); });
11392 s.addEventListener('dragover', e => { onWinDragover(e); });
11393 s.addEventListener('dragleave', e => { onWinDragleave(e); });
11394 s.addEventListener('drop', e => { onWinDrop(e); });
11395
11396 s.addEventListener('mouseenter',e => { onEnterWin(e); });
11397 s.addEventListener('mouseleave',e => { onLeaveWin(e); });
11398
11399 if( false ){
11400     s.style.position = "absolute";
11401     s.style.x = x+'px';
11402     s.style.left = x+'px';
11403     s.style.y = y+'px';
11404     s.style.top = y+'px';
11405 }else{
11406     s.style.setProperty('position','absolute','important');
11407     s.style.setProperty('x','x','important');
11408     s.style.setProperty('left','x','important');
11409     s.style.setProperty('y','y','important');
11410     s.style.setProperty('top','y','important');
11411 }
11412
11413 favicon = './favicon.ico';
11414 uv1 = u.split('/');
11415 if( 2 <= uv1.length ){
11416     uv2 = uv1[1].split('/');
11417     if( 2 <= uv2.length ){
11418         if( uv1[0] == 'file' ){
11419             //favicon = 'file:/// ' + uv2.slice(0,uv2.length-1).join('/');
11420             // + '/favicon.ico';
11421             favicon = './favicon.ico';
11422         }else{
11423             favicon = uv1[0] + '://' + uv2[0] + '/favicon.ico';
11424         }
11425     }
11426 }
11427 //wdllog('----- favicon-url='+favicon);
11428 href_id = l.id + ' href';
11429 l.innerHTML = ''+u+'</a>');
11431 //l.addEventListener('click', e => { onClickWin(e); });
11432 l.addEventListener('mouseenter',e => { stopGShell(e); });
11433 l.addEventListener('mouseleave',e => { contGShell(e); });
11434 l.addEventListener('keydown', e => { onKeyDown(e); });
11435 l.addEventListener('keyup', e => { onKeyUp(e); });
11436
11437 cmd.href_id = href_id;
11438 wdllog('(0)cmd#'+cmd.id);
11439 wdllog('(1)href_id#'+href_id);
11440 wdllog('(2)href_id#'+cmd.href_id);
11441 cmd.addEventListener('click', e => { VirtualBrowserCommand(e,s,l,cmd,f); });
11442
11443 f.style.borderColor = c;
11444 f.src = u;
11445 //f.addEventListener('mouseenter',e => { onEnterWin(e); });
11446 //f.addEventListener('mouseleave',e => { onLeaveWin(e); });
11447
11448 //s.addEventListener('click', e => { onClickWin(e); });
11449 //f.addEventListener('click', e => { wdllog('click wbl'); });
11450 f.addEventListener('mozbrowsericonchange',onFaviconChange);
11451
11452 wdllog('done settle VirtualBrowser url='+u +'\n'
11453     + 'id=' + s.id + ' '
11454     + 'width=' + s.style.width + ' '
11455     + 'height=' + s.style.height + ' '
11456     + 'cmd=' + cmd.id
11457 );
11458 }
11459
11460 function WD_EventSetup2(){
11461     dt = VirtualDesktop1;
11462     dt.style.width = "800px";
11463     dt.style.height = "500px";
11464     dt.addEventListener('dragstart',e => { onWinDragstart(e); });
11465     dt.addEventListener('drag', e => { onWinDrag(e); });
11466     dt.addEventListener('exit', e => { onWinDragexit(e); });
11467 }
11468
11469 function GRonClick(){
11470     WD_SaveScope(null); // should be push
11471     t = event.target;
11472     x = t.getAttribute('data-leftx');
11473     y = t.getAttribute('data-topy');
11474     zoom = WD_Zoom1_XY.value;
11475     x *= zoom;
11476     y *= zoom;
11477     WD_doScrollXY(event,x,y);
11478     //alert('scroll #' + t.id + ' x=' + x + ', y=' + y);
11479 }
11480
11481 function WD_setGrid(e){
11482     t = e.target;
11483     WD_setGrid1(t);
11484 }
11485
11486 function WD_setGrid1(t){
11487     //ds = VirtualDesktop1_Content; // should be VirtualSpace_1
11488     ds = VirtualDesktop1_GridPlane;
11489     if( t.value == 'GridOn' ){
11490         for( col = 0; col < 16; col++ ){
11491             for( row = 0; row < 16; row++ ){
11492                 g1 = document.createElement('span');
11493                 g1.setAttribute('class','VirtualGrid');
11494                 leftx = col * 800;
11495                 topy = row * 500;
11496                 gid = col + ' ' + row
11497                 label = '<+span '
11498                     + 'id="'+gid+' '+class="WD_GridScroll" '
11499                     + 'contenteditable="false" onclick="GRonClick()" '
11500                     + 'data-leftx=' + leftx + ' ' + 'data-topy=' + topy + ' '
11501                     + '>';
11502                 + gid + '<+span>';
11503                 console.log('grid '+label);
11504                 g1.innerHTML = label;
11505                 g1.position = 'relative';
11506                 g1.leftx = leftx;

```

```

11505     gl.topy = topy;
11506     gl.style.left = gl.leftx + 'px';
11507     gl.style.top = gl.topy + 'px';
11508     if( col % 2 == row % 2 ){
11509         gl.style.backgroundColor = 'rgba(255,255,255,0.3)';
11510     }
11511     ds.appendChild(gl);
11512 }
11513 }
11514 t.value = 'GridOff';
11515 }else{
11516     ds.innerHTML = '';
11517     t.value = 'GridOn';
11518 }
11519 }
11520 }
11521 var sav_scrollLeft;
11522 var sav_scrollTop;
11523 var sav_nscale;
11524 function WD_SaveScope(e){
11525     sav_scrollLeft = WD_Left_1.value;
11526     sav_scrollTop = WD_Top_1.value;
11527     sav_nscale = WD_Zoom_1_XY.value;
11528     //console.log('Saved zoom='+sav_oscale+', '+sav_nscale);
11529 }
11530 function WD_RestoreScope(e){
11531     WD_Zoom_1_XY.value = sav_nscale;
11532     WD_doZoom();
11533 }
11534 WD_Left_1.value = sav_scrollLeft;
11535 WD_Top_1.value = sav_scrollTop;
11536 WD_doScroll(null);
11537 }
11538 }
11539 function ignoreEvent(e){
11540     e.stopPropagation();
11541     //e.preventDefault();
11542 }
11543 }
11544 function zoomMag(){
11545     return WD_Zoom_1_MAG.value;
11546 }
11547 }
11548 function WD_EventSetup3(){
11549     WD_Grid_1.addEventListener('click', e => { WD_setGrid(e); });
11550     WD_Zoom_1_Save.addEventListener('click', e => { WD_SaveScope(e); });
11551     WD_Zoom_1_Restore.addEventListener('click', e => { WD_RestoreScope(e); });
11552     WD_Width_1.value = dt.style.width;
11553     WD_Height_1.value = dt.style.height;
11554     WD_Width_1.addEventListener('keydown', ignoreEvent);
11555     WD_Height_1.addEventListener('keydown', ignoreEvent);
11556     WD_Width_1.addEventListener('keyup', ignoreEvent);
11557     WD_Height_1.addEventListener('keyup', ignoreEvent);
11558     WD_Zoom_1_MAG.addEventListener('keydown', ignoreEvent);
11559     WD_Zoom_1_MAG.addEventListener('keyup', ignoreEvent);
11560 }
11561 }
11562 function escale1(e,oscale,nscale){
11563     e.style.setProperty('transform', 'scale('+nscale+')');
11564     nscale = oscale / nscale;
11565     w = parseInt(e.style.width);
11566     h = parseInt(e.style.height);
11567     w = w * nscale; //(oscale/nscale);
11568     h = h * nscale; //(oscale/nscale);
11569     e.style.width = w + 'px';
11570     e.style.height = h + 'px';
11571 }
11572 }
11573 function scaleWD(ds,oscale,nscale){
11574     if( true ){
11575         escale1(WirtualBrowser_1,oscale,nscale);
11576         escale1(WirtualBrowser_1_Location,oscale,nscale);
11577         escale1(WirtualBrowser_1_Command,oscale,nscale);
11578         escale1(WirtualBrowser_1_Frame,oscale,nscale);
11579     }
11580     escale1(WirtualBrowser_2,oscale,nscale);
11581     escale1(WirtualBrowser_2_Location,oscale,nscale);
11582     escale1(WirtualBrowser_2_Command,oscale,nscale);
11583     escale1(WirtualBrowser_2_Frame,oscale,nscale);
11584     escale1(WirtualBrowser_3,oscale,nscale);
11585     escale1(WirtualBrowser_3_Location,oscale,nscale);
11586     escale1(WirtualBrowser_3_Command,oscale,nscale);
11587     escale1(WirtualBrowser_3_Frame,oscale,nscale);
11588 }
11589 }
11589 function WD_doZoom(){
11590     ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11591     oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
11592     nscale = WD_Zoom_1_XY.value;
11593     ds.style.zoom = nscale;
11594     WD_Zoom_1_XY.value = ds.style.zoom;
11595     WD_Zoom_1_XY.ovalue = ds.style.zoom;
11596     scaleWD(ds,oscale,nscale);
11597 }
11598 }
11599 function WD_EventSetup4(){
11600     WD_Zoom_1.addEventListener('click',WD_doZoom);
11601     WD_Zoom_1_XY.addEventListener('keydown',ignoreEvent);
11602     WD_Zoom_1_XY.addEventListener('keyup',ignoreEvent);
11603 }
11604 }
11605 function WD_doZoomOUT(){
11606     ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11607     oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
11608     nscale = 0;
11609     if( oscale == 0 || oscale == '' ){
11610         nscale = 1;
11611     }
11612     nscale = oscale / zoomMag();
11613     ds.style.zoom = nscale;
11614     WD_Zoom_1_XY.value = ds.style.zoom;
11615     WD_Zoom_1_XY.ovalue = ds.style.zoom;
11616     scaleWD(ds,oscale,nscale);
11617 }
11618 }
11619 function WD_doZoomIN(){
11620     ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11621     oscale = WD_Zoom_1_XY.ovalue; // hidden value for current zoom
11622     nscale = 0;
11623     if( oscale == 0 || oscale == '' ){
11624         nscale = 1;
11625     }
11626     nscale = oscale * zoomMag();
11627     if( nscale < 1 ){
11628         alert('maybe too large, zoom='+nscale);
11629         return;
11630     }
11631     ds.style.zoom = nscale;
11632     WD_Zoom_1_XY.value = ds.style.zoom;
11633     WD_Zoom_1_XY.ovalue = ds.style.zoom;
11634     scaleWD(ds,oscale,nscale);
11635 }
11636 }
11637 function WD_EventSetup5(){
11638     WD_Zoom_1_OUT.addEventListener('click',WD_doZoomOUT);
11639     WD_Zoom_1_IN.addEventListener('click',WD_doZoomIN);
11640 }
11641 }
11642 function WD_doResize(e){
11643     dt = WirtualDesktop_1;
11644     dt.style.width = WD_Width_1.value;
11645     dt.style.height = WD_Height_1.value;
11646     WD_Width_1.value = dt.style.width;
11647     WD_Height_1.value = dt.style.height;
11648 }
11649 }
11650 function WD_doRSResize(e){
11651     //alert('Resize Space');
11652     ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11653     ds.style.width = WS_1_Width.value;
11654     ds.style.height = WS_1_Height.value;
11655     WS_1_Width.value = ds.style.width;
11656     WS_1_Height.value = ds.style.height;
11657 }
11658 }
11659 function WD_EventSetup6(){
11660     ds = WirtualDesktop_1_Content; // should be WirtualSpace_1
11661     ds.style.width = '5120px';
11662     ds.style.height = '2880px';
11663     WS_1_Width.value = ds.style.width;
11664     WS_1_Height.value = ds.style.height;
11665     WS_1_Width.addEventListener('keydown', ignoreEvent);
11666     WS_1_Height.addEventListener('keydown', ignoreEvent);
11667     WS_1_Width.addEventListener('keyup', ignoreEvent);
11668     WS_1_Height.addEventListener('keyup', ignoreEvent);
11669     WS_1_Width.addEventListener('click', e => { WD_doRSResize(e); });
11670     WS_1_Height.addEventListener('click', e => { WD_doRSResize(e); });
11671 }
11672 }
11673 function WD_doScrollXY(e,left,top){
11674     dt = WirtualDesktop_1;
11675     dt.scrollLeft = left;
11676     dt.scrollTop = top;
11677     WD_Left_1.value = dt.scrollLeft;
11678     WD_Top_1.value = dt.scrollTop;
11679     console.log('--Scroll #' + dt.id + ' (' + left + ', ' + top + ')');
11680 }
11681 }
11682 function WD_doScroll(e){
11683     //dt = WirtualDesktop_1_Content;
11684     dt = WirtualDesktop_1;
11685     left = parseInt(WD_Left_1.value);
11686     stop = parseInt(WD_Top_1.value);
11687     dt.scrollLeft = left;
11688     dt.scrollTop = stop;
11689     WD_Left_1.value = dt.scrollLeft;

```

```

11682 WD_Top_1.value = dt.scrollTop;
11683 console.log('--Scroll #' + dt.id + ' (' + sleft + ', ' + stop + ')');
11684
11685 function showScrollPosition(){
11686   if( false ){
11687     console.log(
11688       'w=' + VirtualDesktop_1.style.width + 'px' +
11689       'h=' + VirtualDesktop_1.style.height + 'px' +
11690       'wss=' + VirtualDesktop_1.scrollLeft + 'px' +
11691       'wtop=' + VirtualDesktop_1_Content.style.top + 'px' +
11692       'wds=' + VirtualDesktop_1_Content.style.left + 'px' +
11693       'wds=' + VirtualDesktop_1_Content.scrollLeft + 'px' +
11694     );
11695     WD_Left_1.value = VirtualDesktop_1.scrollLeft;
11696     WD_Top_1.value = VirtualDesktop_1.scrollTop;
11697   }
11698 }
11699 function WD_EventSetup7(){
11700   WD_Scroll_1.addEventListener('click', e => { WD_DoScroll(e); });
11701   WD_Left_1.addEventListener('keydown', ignoreEvent);
11702   WD_Left_1.addEventListener('keyup', ignoreEvent);
11703   WD_Top_1.addEventListener('keydown', ignoreEvent);
11704   WD_Top_1.addEventListener('keyup', ignoreEvent);
11705 }
11706 function WD_EventSetup8(){
11707   VirtualDesktop_1.addEventListener('scroll', showScrollPosition);
11708   VirtualDesktop_1_Content.addEventListener('scroll', showScrollPosition);
11709 }
11710 if( false ){
11711   w = 1000 + 'px';
11712   dt.style.width = w;
11713   dt.style.height = '300px';
11714   dt.style.resize = 'both';
11715   dt.style.borderWidth = 50 + 'px';
11716   dt.style.borderRadius = 25 + 'px';
11717   console.log('--2----- #' + dt.id + ' style=' + dt.style);
11718   console.log('----- #' + dt.id + ' width=' + dt.style.width);
11719   console.log('----- #' + dt.id + ' left=' + dt.style.left);
11720   console.log('----- #' + dt.id + ' border=' + dt.style.border);
11721 }
11722 function onDTRize(e){
11723   dt = e.target;
11724   h = parseInt(dt.style.height);
11725   dt.style.borderWidth = (h * 0.075) + 'px';
11726   console.log('----- borderWidgh=' + dt.style.borderWidth);
11727 }
11728
11729 VirtualDesktopDetails.addEventListener('toggle', VirtualDesktop_init);
11730 function VirtualDesktop_init(){
11731   if( !VirtualDesktopDetails.open ){
11732     return;
11733   }
11734   //GJ_Join();
11735   VirtualDesktop_1.addEventListener('resize', e => { onDTRize(e); });
11736   //console.log('----- #' + dt.id
11737   // + ' borderWidgh=' + dt.style.getProperty('border-width'));
11738
11739   settleWin(
11740     VirtualBrowser_1,
11741     VirtualBrowser_1_Location,
11742     VirtualBrowser_1_Command,
11743     VirtualBrowser_1_Frame,
11744     document.URL,
11745     500, 280, 50, 20, '#262', 1.0);
11746   settleWin(
11747     VirtualBrowser_2,
11748     VirtualBrowser_2_Location,
11749     VirtualBrowser_2_Command,
11750     VirtualBrowser_2_Frame,
11751     'https://its-more.jp/ja_jp/',
11752     500, 280, 150, 100, '#448', 1.0);
11753   settleWin(
11754     VirtualBrowser_3,
11755     VirtualBrowser_3_Location,
11756     VirtualBrowser_3_Command,
11757     VirtualBrowser_3_Frame,
11758     // ..gshell/gsh.go.html',
11759     // http://gshell.org/gshell/gsh.go.html',
11760     'https://golang.org',
11761     500, 280, 250, 180, '#444', 1.0);
11762     //1000, 720, 0, 0, '#444', 0.125);
11763     //1200, 720, -100, -50, '#444', 0.4);
11764   function WD_ClockUpdate(e){
11765     VirtualDesktop_1_Clock.innerHTML = DateShort();
11766     VirtualDesktop_1_Calender.innerHTML = DateHourWin();
11767   }
11768   window.setInterval(WD_ClockUpdate, 500);
11769
11770   WD_EventSetup1();
11771   WD_EventSetup2();
11772   WD_EventSetup3();
11773   WD_EventSetup4();
11774   WD_EventSetup5();
11775   WD_EventSetup6();
11776   WD_EventSetup7();
11777   WD_EventSetup8();
11778 }
11779 //VirtualDesktop_init();
11780
11781 Destroy_VirtualDesktop = function(){
11782   VirtualDesktop_1.style = '';
11783
11784   VirtualBrowser_1.removeAttribute('style');
11785   VirtualBrowser_1_Location.innerHTML = '';
11786   VirtualBrowser_1_Frame.removeAttribute('src');
11787   VirtualBrowser_1_Frame.removeAttribute('style');
11788   VirtualBrowser_1_Frame.style = '';
11789
11790   VirtualBrowser_2.removeAttribute('style');
11791   VirtualBrowser_2_Location.innerHTML = '';
11792   VirtualBrowser_2_Frame.removeAttribute('src');
11793   VirtualBrowser_2_Frame.style = '';
11794
11795   VirtualBrowser_3.removeAttribute('style');
11796   VirtualBrowser_3_Location.innerHTML = '';
11797   VirtualBrowser_3_Frame.removeAttribute('src');
11798   VirtualBrowser_3_Frame.style = '';
11799
11800   GJFactory_1.style = '';
11801   iframe0.style = '';
11802   VirtualDesktop_1.style = '';
11803 }
11804
11805 </script>
11806 <!-- VirtualDesktop -->
11807 </details>
11808 * //</span>
11809
11810 <!-- ===== Work { ===== -->
11811 <span id="SBSidebar_WorkCodeSpan">
11812 *
11813 <details><summary>SBSidebar</summary>
11814 <!-- ----- SBSidebar // 2020-0928 SatoxITS { -->
11815 <h2>SBSidebar</h2>
11816 <input id="SBSidebar_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
11817 <input id="SBSidebar_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
11818 <input id="SBSidebar_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
11819 <span id="SBSidebar_WorkCodeView"></span>
11820 <script id="SBSidebar_WorkCodeView">
11821 function SBSidebar_openWorkCodeView(){
11822   function SBSidebar_showWorkCode(){
11823     showHtmlCode(SBSidebar_WorkCodeView, SBSidebar_WorkCodeSpan);
11824   }
11825   SBSidebar_WorkCodeViewOpen.addEventListener('click', SBSidebar_showWorkCode);
11826 }
11827 SBSidebar_openWorkCodeView(); // should be invoked by an event
11828
11829 console.log('-- SbSlider // 2020-1006-01 SatoxITS --');
11830 function SetSideBar(){
11831   sidebar = document.getElementById('secondary');
11832   console.log('primary=' + primary + ' secondary=' + sidebar + ' main=' + main + ');
11833   wrap = sidebar.parentNode;
11834   console.log('SbSlider parent is ' + wrap + ', #' + wrap.id + ' . ' + wrap.class);
11835   //wrap = wrap.parentNode;
11836   //console.log('SbSlider parent is ' + wrap + ', #' + wrap.id + ' . ' + wrap.class);
11837   //wrap = wrap.parentNode;
11838   //console.log('SbSlider parent is ' + wrap + ', #' + wrap.id + ' . ' + wrap.class);
11839   //nsb = sidebar.cloneNode();
11840   nsb = sidebar;
11841   nsb.style.width = '100%';
11842   slider = document.createElement('div');
11843   slider.id = 'SbSlider';
11844   slider.appendChild(nsb);
11845   slider.setAttribute('class', 'SbSlider');
11846   nsb.style.position = 'relative';
11847   slider.style.position = 'fixed';
11848   slider.style.display = 'block'; // inline;
11849   slider.style.zIndex = 100000;
11850   // nsb.style.zIndex = 200000;
11851   nsb.style.position = 'absolute';
11852   nsb.style.minWidth = '80px';
11853   nsb.style.left = '0px';
11854   nsb.style.top = '0px';
11855
11856   w = window.innerWidth;
11857   console.log('SliderWidth ' + w + ': ', (w/3) + 'px');
11858   if( w < 640 ){

```

```

11859     slider.style.setProperty('width',(w/3) + 'px','important');
11860   }
11861   main.style.marginLeft = (parseInt(slider.style.width)/2 - 20) + 'px';
11862 }
11863 slider.style.resize = "both";
11864 slider.draggable = "true";
11865 wrap.appendChild(slider);
11866 console.log('-- added Sbslider');
11867 //nsb.addEventListener('scroll',SbScrolled);
11868
11869 buttons = document.createElement('div');
11870 buttons.id = 'NaviButtons';
11871 buttons.setAttribute('class','NaviButtons');
11872 buttons.align = "center";
11873 buttons.innerHTML += '<'+p><a href="#TopOfPost" draggable="true">TOP</a><'+p>';
11874 buttons.innerHTML += '<'+p><a href="#EndOfPost" draggable="true">END</a><'+p>';
11875 page.appendChild(buttons);
11876 buttons.style.position = 'fixed';
11877 buttons.style.zindex = 30000;
11878 buttons.style.width = '180%';
11879 buttons.style.top = '320px';
11880 buttons.style.left = parseInt(w) * 1.0 + 'px';
11881 console.log('-- Sbslider installed (^~/ SatoxITS');
11882 }
11883
11884 //window.addEventListener('load',SetSidebar); // after load
11885 DestroyNaviButtons = function(){
11886   nb = document.getElementById('NaviButtons');
11887   if( nb != null ){
11888     nb.parentNode.removeChild(NaviButtons);
11889   }
11890 }
11891 </script>
11892
11893 // 2020-1006 its-more.jp-blog-60000-style.css
11894 <!--
11895 <style>
11896 #NaviButtons {
11897   position:fixed;
11898   display:block;
11899   width:100%;
11900   xtop:100px;
11901   xleft:100px;
11902   z-index:30000;
11903   font-size:10pt;
11904   color:#2ff !important;
11905   text-align:center;
11906   background-color:rgba(230,230,230,0.01);
11907 }
11908 #NaviButtons a {
11909   color:#2a2 !important;
11910   font-size:20px;
11911   text-align:center;
11912   xtext-shadow:2px 2px #8ff;
11913   resize:both;
11914   padding:6px;
11915   margin:10px;
11916   border:1px solid #288 !important;
11917   border-radius:3px;
11918   background-color:rgba(160,160,160,0.05);
11919 }
11920 #Sbslider {
11921   overflow:auto;
11922   resize:both !important;
11923   xoverflow-y:hidden !important;
11924   height:100px !important;
11925   display:inline !important;
11926   position:fixed !important;
11927   left:0px;
11928   top:0px;
11929   xwidth:180px;
11930   width:24%;
11931   min-width:80px;
11932   height:100% !important;
11933   background-color:rgba(100,100,200,0.1);
11934 }
11935 #secondary {
11936   position:fixed;
11937   left:0px;
11938   top:0px;
11939   xxxz-index:60000;
11940   scroll-behavior: overflow !important;
11941   xxxxxxxxwidth:18% !important;
11942   padding-left:4pt;
11943   color:#fff;
11944   font-size:0.5em;
11945   background-color:rgba(64,160,64,0.6) !important;
11946   white-space:nowrap;
11947 }
11948 #secondary a {
11949   color:#fff !important;
11950   text-decoration:disable !important;
11951 }
11952 #primary {
11953   position:relative;
11954   width:75% !important;
11955   left:25% !important;
11956 }
11957 #main {
11958   position:relative;
11959   width:75% !important;
11960   left:25% !important;
11961 }
11962 #site-navigation {
11963   position:relative;
11964   left:120px;
11965 }
11966 #adswc countertext {
11967   color:#4169e1;
11968   font-size:16pt !important;
11969   xfont-size:108 !important;
11970   font-weight:bold;
11971 }
11972 #nowTime {
11973   color:#0ffa0;
11974   font-size:16pt !important;
11975   xfont-size:108 !important;
11976   font-weight:bold;
11977   text-shadow:1px 1px #fff;
11978 }
11979 .navigation-top {
11980   color:#2a2 !important;
11981   border:0px;
11982   background-color:rgba(220,220,220,0.1);
11983 }
11984 .visible-scrollbar, .invisible-scrollbar, .mostly-customized-scrollbar {
11985   display: block;
11986   xwidth: 1em;
11987   xoverflow: auto;
11988   xxheight: 1em;
11989 }
11990 .invisible-scrollbar::-webkit-scrollbar {
11991   xdisplay: none;
11992   width:1px !important;
11993   height:1px !important;
11994 }
11995 .mostly-customized-scrollbar::-webkit-scrollbar {
11996   width: 2px;
11997   height: 2px;
11998   xxbackground-color: #aaa; xxx:or add it to the track;
11999 }
12000 .mostly-customized-scrollbar::-webkit-scrollbar-thumb {
12001   background: #000;
12002 }
12003 </style>
12004 } -->
12005
12006 </details>
12007 <!-- SBSidebar_WorkCodeSpan } -->
12008 </span> //</span>
12009 <!-- ===== Work } ===== -->
12010 <!-- ===== Work { ===== -->
12011 </span id="Affiliate_WorkCodeSpan">
12012 </
12013 <!-- ===== Work { ===== -->
12014 </span id="Affiliate_Test"><summary>Affiliate/</summary>
12015 <!-- ----- Affiliate // 2020-1010 SatoxITS { -->
12016 <div id="AffiliateDock">
12017 <div id="AffView" class="AffView" draggable="true" style="">
12018 <div id="AffSet" class="AffPlate">
12019 <iframe id="aff_0" class="AffItem"></iframe>
12020 <iframe id="aff_1" class="AffItem"></iframe>
12021 <iframe id="aff_2" class="AffItem"></iframe>
12022 <iframe id="aff_3" class="AffItem"></iframe>
12023 <iframe id="aff_4" class="AffItem"></iframe>
12024 <iframe id="aff_5" class="AffItem"></iframe>
12025 </div>
12026 </div></div>
12027 <h2>Supportive Affiliate</h2>
12028 <style>
12029 .AffView {
12030   z-index:0;
12031   overflow-x:scroll;
12032   overflow-y:scroll;
12033   position:fixed;
12034 }
12035

```

```

12036 max-width:2560px;
12037 max-height:1008;
12038 width:470px;
12039 left:75%;
12040 height:958;
12041 resize:both;
12042 xleft:-10%;
12043 margin-top:40px;
12044 xleft:0;
12045 xalign:right;
12046 display:block;
12047 border:4px inset rgba(255,255,255,0.1);
12048 background-color:rgba(255,255,255.0.1);
12049 }
12050.AffView:hover {
12051 z-index:1;
12052 width:300px;
12053 overflow:scroll;
12054 border:4px inset #fcc;
12055 background-color:rgba(80,80,255.0.2);
12056 background-color:#ffe;
12057 }
12058.AffPlate:hover {
12059 border-left:4px dashed #888;
12060 }
12061.AffPlate{
12062 overflow-x:visible;
12063 border-left:4px dashed rgba(255,255,255,0.1);
12064 max-width:2560px;
12065 max-height:2880px;
12066 margin-top:10px;
12067 margin-bottom:10px;
12068 margin-left:4px;
12069 width:300px;
12070 xheight:1440px;
12071 }
12072.AffItem {
12073 overflow-x:visible;
12074 overflow-y:scroll;
12075 max-width:2560px;
12076 max-height:1440px;
12077 z-index:0;
12078 display:block;
12079 xposition:fixed;
12080 xposition:absolute;
12081 position:relative;
12082 //left:300px;
12083 xresize:both;
12084 padding:0px;
12085 width:600px;
12086 height:400px;
12087 max-height:800px !important;
12088 margin-top:0%;
12089 margin-left:0%;
12090 margin-right:0% !important;
12091 border:10px inset #ccc;
12092 transform:scale(0.5);
12093 background-color:rgba(255,255,255,0.2);
12094 xalign:right;
12095 }
12096.AffItem:hover {
12097 border:16px inset #bbf;
12098 }
12099 </style>
12100<input id="Affiliate_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12101<input id="Affiliate_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12102<input id="Affiliate_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12103<span id="Affiliate_WorkCodeView"></span>
12104<script id="Affiliate_WorkScript">
12105function Affiliate_openWorkCodeView(){
12106 function Affiliate_showWorkCode(){
12107 showHtmlCode(Affiliate_WorkCodeView,Affiliate_WorkCodeSpan);
12108 }
12109 Affiliate_WorkCodeViewOpen.addEventListener('click',Affiliate_showWorkCode);
12110 }
12111Affiliate_openWorkCodeView(); // should be invoked by an event
12112
12113</iframe id="aff_8" xxsrc="https://stackoverflow.com/tags" class="AffItem"></iframe>
12114</iframe id="aff_9" xxsrc="https://developer.mozilla.org/en-US/docs/Web" class="AffItem"></iframe>
12115var Aff_isSetup = false;
12116Affiliate_test.addEventListener('click',Aff_Setup);
12117function Aff_Setup(){
12118 if(Aff_isSetup){return;}Aff_isSetup = true;
12119 parent = document.documentElement;
12120 parent.appendChild(AffView);
12121 AffView.style.top = '0px';
12122 AffView.style.left = (window.innerWidth - 280) + 'px';
12123
12124 var off = 100;
12125 zoom = 0.5;
12126 ozoom = 0.3;
12127 leftx = window.innerWidth - 300;
12128 left = leftx + 'px';
12129 left = -130 + 'px';
12130 console.log('aff-init window.innerWidth='+window.innerWidth);
12131 w = 1000;
12132 h = 560;
12133
12134 aff_0.src = './gshell/gsh.go.html';
12135 aff_1.src = 'https://golang.org';
12136 aff_2.src = 'https://drafts.csswg.org/';
12137 aff_3.src = 'https://html.spec.whatwg.org/dev/';
12138 aff_4.src = 'https://wikipedia.org';
12139 aff_5.src = 'https://www.bing.com/translator';
12140
12141 //parent.appendChild(aff_0);
12142 aff_0.style.width = zoom*w+'px';
12143 aff_0.style.height = zoom*h+'px';
12144 aff_0.style.left = left;
12145 //aff_0.style.top = off+'px'; off += ozoom*h;
12146 aff_0.draggable = 'true';
12147
12148 //parent.appendChild(aff_1);
12149 aff_1.style.width = zoom*w+'px';
12150 aff_1.style.height = zoom*h+'px';
12151 aff_1.style.left = left;
12152 //aff_1.style.top = off+'px'; off += ozoom*h;
12153 aff_1.style.top = '-150px';
12154 aff_1.draggable = 'true';
12155
12156 //parent.appendChild(aff_2);
12157 aff_2.style.width = zoom*w+'px';
12158 aff_2.style.height = zoom*h+'px';
12159 aff_2.style.left = left;
12160 //aff_2.style.top = off+'px'; off += ozoom*h;
12161 aff_2.style.top = '-300px';
12162 aff_2.draggable = 'true';
12163
12164 //parent.appendChild(aff_3);
12165 aff_3.style.transform = 'scale(0.25)';
12166 aff_3.style.width = 2*zoom*w+'px';
12167 aff_3.style.height = 2*zoom*h+'px';
12168 aff_3.style.border = '32px inset #ccc';
12169 //aff_3.style.left = -390 + 'px'; //left+2;
12170 //aff_3.style.left = (leftx - 265) + 'px';
12171 aff_3.style.left = -395 + 'px';
12172 //aff_3.style.top = (-155+off)+'px'; off += ozoom*h;
12173 aff_3.style.top = '-600px';
12174 aff_3.draggable = 'true';
12175
12176 //parent.appendChild(aff_4);
12177 aff_4.style.width = zoom*w+'px';
12178 aff_4.style.height = zoom*h+'px';
12179 aff_4.style.left = left;
12180 //aff_4.style.top = off+'px'; off += ozoom*h;
12181 aff_4.style.top = '-900px';
12182 aff_4.draggable = 'true';
12183
12184 //parent.appendChild(aff_5);
12185 aff_5.style.transform = 'scale(0.300)';
12186 aff_5.style.width = zoom*(w*1.67)+'px';
12187 aff_5.style.height = zoom*(h*1.67)+'px';
12188 aff_5.style.border = '25px inset #ccc';
12189 aff_5.style.left = -308+'px';
12190 //aff_5.style.left = (-175+leftx)+'px';
12191 //aff_5.style.top = (-95+off)+'px'; off += ozoom*h;
12192 //aff_5.style.left = '0px';
12193 //aff_5.style.top = '0px';
12194 aff_5.style.top = '-1150px';
12195 //aff_5.style.align = 'right';
12196 aff_5.draggable = 'true';
12197
12198 window.addEventListener('resize',affresize);
12199 }
12200function affresize(){
12201 AffView.style.left = (window.innerWidth - 280) + 'px';
12202 leftx = window.innerWidth - 400;
12203 left = leftx + 'px';
12204 console.log('aff-resize window.innerWidth='+window.innerWidth);
12205 }
12206//window.addEventListener('resize',affresize);
12207//document.addEventListener('resize',affresize);
12208//gsh.addEventListener('resize',affresize);
12209
12210function ResetAffView(){
12211 AffViewDock.appendChild(AffView);
12212 AffView.removeAttribute('style');

```

```

12213 aff_0.removeAttribute('src');
12214 aff_0.removeAttribute('style'); aff_0.removeAttribute('draggable');
12215 aff_1.removeAttribute('src');
12216 aff_1.removeAttribute('style'); aff_1.removeAttribute('draggable');
12217 aff_2.removeAttribute('src');
12218 aff_2.removeAttribute('style'); aff_2.removeAttribute('draggable');
12219 aff_3.removeAttribute('src');
12220 aff_3.removeAttribute('style'); aff_3.removeAttribute('draggable');
12221 aff_4.removeAttribute('src');
12222 aff_4.removeAttribute('style'); aff_4.removeAttribute('draggable');
12223 aff_5.removeAttribute('src');
12224 aff_5.removeAttribute('style'); aff_5.removeAttribute('draggable');
12225 }
12226 </script>
12227 </details>
12228 <!-- Affiliate_WorkCodeSpan -->
12229 <!--/span>
12230 <!-- Work } ===== -->
12231
12232
12233
12234 <!-- Work { ===== -->
12235 </span id="TextCanvas_WorkCodeSpan">
12236 />
12237 <details id="TextCanvas Section"><summary id="TextCanvas_Summary">TextCanvas</summary>
12238 <!-- TextCanvas // 2020-1013 SatoxITS { -->
12239
12240 <details id="FontSelect Section"><summary id="FontSelect_Summary">Font Selection</summary>
12241 <h2>Font Selection</h2>
12242 <div>
12243 <div id="FontList"></div>
12244 </div>
12245 <style>
12246 #FontList {
12247   overflow:visible;
12248   background-color:rgba(240,245,255,1.0) !important;
12249 }
12250 #FontList td {
12251   font-size:16px;
12252   padding:0px;
12253   padding-left:2px;
12254   padding-right:2px;
12255   margin:0px;
12256   line-height:1.2;
12257   border:0px;
12258 }
12259 #FontList td:hover {
12260   background-color:#228;
12261 }
12262 #FontList tr:hover {
12263   color:#fff;
12264   background-color:#000;
12265   xborder:1px solid #000;
12266 }
12267 .xcourier { colr:#000; font-size:16px; font-family:courier; }
12268 .xcursive { colr:#000; font-size:16px; font-family:cursive; }
12269 .xfantasy { colr:#000; font-size:16px; font-family:fantasy; }
12270 .xgeorgia { colr:#000; font-size:16px; font-family:georgia; }
12271 .xmonospace { colr:#000; font-size:16px; font-family:monospace; }
12272 </style>
12273 </script>
12274 function fontstr(name,text){
12275   //tr = '<+tr style="font-family:'+name+';">\n';
12276   tr = '<+tr style="font-family:'+name+';">\n';
12277   tr += '<+td style="font-family:Arial;font-size:12pt;">'+name+'<+td>';
12278   tr += '<+td data-fsyt="n">+text+'<+td>';
12279   tr += '<+td data-fsyt="b">+b+'<+td>'+text+'<+td>';
12280   tr += '<+td data-fsyt="i">+i+'<+td>'+text+'<+td>';
12281   tr += '<+td data-fsyt="bi">+b+'<+td>'+i+'<+td>'+text+'<+td>';
12282   tr += '<+td>';
12283   return tr;
12284 }
12285 function lsfont(){
12286   text = 'GShell-Go012#x1f923;';
12287
12288   fl = '';
12289   fl += <table>\n';
12290   fl += fontstr('Arial',text);
12291   fl += fontstr('Courier',text);
12292   fl += fontstr('Courier New',text);
12293   fl += fontstr('Georgia',text);
12294   fl += fontstr('Helvetica',text);
12295   fl += fontstr('Verdana',text);
12296   fl += fontstr('Times',text);
12297
12298   fl += fontstr('Osaka',text);
12299   fl += fontstr('Meiryo',text);
12300   fl += fontstr('YuMincho',text);
12301
12302   //fl += fontstr('Roman',text);
12303   //document.fonts.load("30px cursive");
12304   fl += fontstr('Serif',text);
12305   fl += fontstr('Sans-serif',text);
12306   fl += fontstr('System-UI',text);
12307   fl += fontstr('Monospace',text);
12308   fl += fontstr('Cursive',text);
12309   fl += fontstr('Fantasy',text);
12310   fl += '</table>\n';
12311
12312   if( false ){
12313     fss = document.fonts.entries(); // FontFaceSet
12314     console.log('FSS='+fss);
12315     while( true ){
12316       font = fss.next();
12317       if( font.done ){
12318         break;
12319       }
12320       fl += font.value[0] + '<br>';
12321     }
12322     FontList.innerHTML = fl;
12323   }
12324 }
12325 function selectFont(e){
12326   t = e.target;
12327   let fsyt = '';
12328   for( i = 0; i < 4; i++){
12329     //console.log('FontSelect '+t.nodeName+' #' +t.id+' '+t.style);
12330     if( t.nodeName == 'TD' ){
12331       //console.log('FontSelect '+t.outerHTML);
12332       if( t.hasAttribute('data-fsyt') ){
12333         fsyt = t.getAttribute('data-fsyt');
12334         //console.log('FontSelect = '+ fsyt);
12335       }
12336     }
12337     if( t.nodeName != 'TD' ){
12338       if( t.style != '' ){
12339         if( t.style.fontFamily != '' ){
12340           break;
12341         }
12342       }
12343     }
12344     t = t.parentNode;
12345   }
12346   if( t.style != '' ){
12347     font = t.style.fontFamily;
12348     //console.log('FontSelect: '+font);
12349     //console.log('FontSelect == '+ fsyt);
12350     if( font != '' ){
12351       sel = document.getElementById("TextCanvas_1_Font");
12352       if( sel != null ){
12353         if( fsyt != '' ){
12354           TextCanvas_1_Bold.checked = 0 <= fsyt.indexOf('b');
12355           TextCanvas_1_Italic.checked = 0 <= fsyt.indexOf('i');
12356         }
12357         sel.value = font;
12358         RedrawTextCanvas();
12359       }else{
12360         alert('Event: ' + e.target.nodeName + ' #' + font);
12361       }
12362     }
12363   }
12364 }
12365 FontList.addEventListener('click',selectFont);
12366 document.fonts.onloadingdone = function(fsse){
12367   //alert('font-loaded '+fsse.fontfaces.length);
12368 }
12369 function FontList_Setup(){
12370   if( FontSelect_Summary.open ){
12371     lsfont();
12372   }
12373 }
12374 FontSelect_Summary.addEventListener('click',lsfont);
12375 </script>
12376 </details>
12377
12378 <h2>Drawing Text on Canvas</h2>
12379 <!-- 2020-1012 -- Drawing Text on Canvas // SatoxITS { -->
12380 <div id="TextCanvas_1_Panel" class="CanvasLabel">
12381 <input id="TextCanvas_1_Clear" class="PanelButton" type="button" value="Clear">
12382 <input id="TextCanvas_1_Draw" class="PanelButton" type="button" value="Draw">
12383 <input id="TextCanvas_1_Font" class="CanvasPanel" type="text" size="14" value="Georgia">
12384 <input id="TextCanvas_1_Bold" class="CanvasBox" type="checkbox" checked="">Bold
12385 <input id="TextCanvas_1_Italic" class="CanvasBox" type="checkbox" checked="">Italic
12386 <br>
12387 <input id="TextCanvas_1_Size" class="CanvasPanel" type="text" size="3" value="64">Pixels
12388 <input id="TextCanvas_1_Color" class="CanvasPanel" type="text" size="6" value="#22a">

```

```

12390<!-- to be PBlue series ? -->
12391<p>
12392<input id="TextCanvas_1_Text" class="TextCanvasText" type="text" size="50" value="GShell">
12393</p>
12394</div>
12395<p>
12396<canvas id="TextCanvas_1" class="TextCanvas" width="400px" height="100px" draggable="true"></canvas>
12397</p>
12398<div class="CanvasLabel">
12399<input id="TextCanvas_1_ToImage" class="PanelButton" type="button" value="ToImage">
12400<input id="TextCanvas_1_ToPNG" class="PanelRadio" type="radio" name="ImageType" value="ToPNG" checked="">PNG
12401<input id="TextCanvas_1_ToJPG" class="PanelRadio" type="radio" name="ImageType" value="ToJPEG">JPEG
12402<input id="TextCanvas_1_DataURL" class="CanvasBox" type="checkbox" checked="">DataURL
12403<div class="TextCanvas_1_BgImage"><img id="TextCanvas_1_Image" class="CanvasImage" src=""></div>
12404<div id="TextCanvas_1_DataURLView" class="CanvasInImage"><br>(background image)</div>
12405</div>
12406</div>
12407<div id="TextCanvas_1_DataURLView" class="DataURLView"><span id="TextCanvas_1_DataURLText"></span></div>
12408</div>
12409<style>
12410.CommandUsageText {
12411  font-family:Courier New;
12412}
12413.TextCanvas {
12414  border:1px solid #000;
12415  resize:both;
12416  display:inline !important;
12417}
12418.DataURLView {
12419  width:100%;
12420  font-size:10pt;
12421  font-family:Courier New, monospace;
12422  color:#000;
12423  xbackground-color:#eee;
12424  margin-bottom:10px;
12425  xdisplay:block;
12426  xxoverflow:scroll;
12427}
12428.CanvasImage {
12429  border:1px dashed #000;
12430}
12431.TextCanvasText {
12432  font-size:12pt;
12433  width:100%;
12434}
12435.CanvasLabel {
12436  font-size:10pt;
12437  color:#000;
12438}
12439.CanvasInImage {
12440  width:100%;
12441  height:160px;
12442  margin-bottom:10px;
12443  color:rgba(32,160,32,0.5);
12444  text-shadow:3px 3px #eee;
12445  background-color:#eee;
12446  xborder:1px solid #000;
12447  font-size:8pt;
12448  vertical-align:middle;
12449}
12450.PanelRadio {
12451  font-size:12pt !important;
12452  color:#000 !important;
12453  vertical-align:middle;
12454}
12455.CanvasBox {
12456  vertical-align:middle;
12457  margin-left:4px !important;
12458  margin-right:8px !important;
12459}
12460.CanvasPanel {
12461  vertical-align:middle !important;
12462  height:14pt !important;
12463  width:inherit !important;
12464  padding:2px !important;
12465  margin:4px !important;
12466  margin-left:4px !important;
12467  margin-right:4px !important;
12468  font-size:10pt !important;
12469  font-family:Georgia !important;
12470  color:#000;
12471  display:inline !important;
12472}
12473.TextCanvasPanel {
12474  vertical-align:middle;
12475  font-size:10pt !important;
12476  font-family:Georgia !important;
12477  color:#000;
12478  display:inline !important;
12479}
12480.PanelButton {
12481  font-size:10pt !important;
12482  font-family:Georgia !important;
12483  vertical-align:middle;
12484  width:45pt !important;
12485  height:14pt !important;
12486  line-height:1.2 !important;
12487  padding:2px !important;
12488  margin:1px !important;
12489  display:inline !important;
12490  padding:1px !important;
12491  color:#fff !important;
12492  background-color:#228 !important;
12493}
12494</style>
12495<script>
12496function DrawTextCanvas(){
12497  ctx = TextCanvas_1.getContext('2d');
12498  var textfont = '';
12499  if( TextCanvas_1_Italic.checked ) textfont += ' italic';
12500  if( TextCanvas_1_Bold.checked ) textfont += ' bold';
12501  textfont += " "+TextCanvas_1_Size.value+'px';
12502  textfont += " "+TextCanvas_1_Font.value;
12503  //ctx.font = 'italic bold 64px Georgia';
12504  //console.log('font:'+textfont);
12505  ctx.fillStyle = TextCanvas_1_Color.value; //'#22a';
12506  ctx.font = textfont;
12507  ctx.fillText(TextCanvas_1_Text.value,10,80);
12508}
12509TextCanvas_1_Draw.addEventListener('click',DrawTextCanvas);
12510function ClearTextCanvas(){
12511  cv = TextCanvas_1;
12512  ctx = cv.getContext('2d');
12513  ctx.clearRect(0,0,cv.width,cv.height);
12514}
12515TextCanvas_1_Clear.addEventListener('click',ClearTextCanvas);
12516function RedrawTextCanvas(){
12517  ClearTextCanvas();
12518  DrawTextCanvas();
12519}
12520function ab2str(buf) {
12521  return String.fromCharCode.apply(null, new Uint16Array(buf));
12522}
12523// 2020-1024, canvas to image
12524function CanvasToImage(){
12525  canvas = TextCanvas_1;
12526  // https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement/toDataURL
12527  if( TextCanvas_1_ToPNG.checked ){
12528    url = canvas.toDataURL("image/png");
12529  }else{
12530    url = canvas.toDataURL("image/jpeg",1.0);
12531  }
12532  //alert('CanvasToImage: length='+url.length+'\n'+url);
12533  TextCanvas_1_Image.src = url;
12534  TextCanvas_1_BgImage.style.backgroundColor = 'url('+url+')';
12535  if( TextCanvas_1_DataURL.checked ){
12536    //TextCanvas_1_DataURLView.innerHTML = url;
12537    txa = TextCanvas_1_DataURLView;
12538    utxa = document.createElement('textArea');
12539    utxa.id = 'TextCanvas_1_DataURLText';
12540    utxa.style.width = '100%';
12541    utxa.style.height = '50pt';
12542    utxa.value = url;
12543    txa.parentNode.replaceChild(utxa,txa);
12544  }
12545  return TextCanvas_1_Image;
12546}
12547TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12548var image = new Image();
12549image.src = url;
12550url = str2ab(url);
12551blob = new Blob(url,{type:'text/plain'});
12552link = document.createElement('a');
12553link.href = URL.createObjectURL(blob);
12554link.download = 'character.txt';
12555link.click();
12556return image;
12557}
12558TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12559TextCanvas_1_ToImage.addEventListener('click',CanvasToImage);
12560if( TextCanvas_Summary.open ){
12561  DrawTextCanvas();
12562}
12563TextCanvas_Summary.addEventListener('click',DrawTextCanvas);
12564</script>
12565<!-- } -->
12566

```

```

1256 <script>
1256 //TextCanvas_1_Panel.addEventListener('mouseover',OffGJShell);
1256 //TextCanvas_1_Panel.addEventListener('mouseout',OnGJShell);
1257 </script>
1257 <!-- Clicking the textarea is necessary to see upto the end of text. why? -->
1257 <input id="TextCanvas_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1257 <input id="TextCanvas_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1257 <input id="TextCanvas_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1257 <span id="TextCanvas_WorkCodeView"></span>
1257 <script id="TextCanvas_WorkCodeScript">
1257 function TextCanvas openWorkCodeView(){
1257     function TextCanvas_showWorkCode(){
1257         showHtmlCode(TextCanvas_WorkCodeView,TextCanvas_WorkCodeSpan);
1257     }
1258     TextCanvas_WorkCodeViewOpen.addEventListener('click',TextCanvas_showWorkCode);
1258 }
1258 TextCanvas openWorkCodeView(); // should be invoked by an event
1258 </script>
1258 </details>
1258 <!-- TextCanvas_WorkCodeSpan -->
1258 *! //</span>
1258 //<!-- ===== Work } ===== -->
1258
1259 //<!-- ===== Work { ===== -->
1259 //<span id="Shading_WorkCodeSpan">
1259 //</span>
1259 <details><summary>Shading Canvas</summary>
1259 <!-- ----- Shading Canvas // 2020-1011 SatoxITS { -->
1259 <h2>Shading Canvas</h2>
1259 <note class="CommandUsageText">
1259 <b>Command</b>
1259 Placement Mode<br>
1260 a ... apply (into absolute position)<br>
1260 j ... bring down (ArrowDown)<br>
1260 k ... bring up (ArrowUp)<br>
1260 h ... bring left (ArrowLeft)<br>
1260 l ... bring right (ArrowRight)<br>
1260 0 ... z-index = 0<br>
1260 + ... z-index += 1<br>
1260 - ... z-index -= 1<br>
1260 r ... return to here (relative position)<br>
1260 c ... clear the log text<br>
1261 Note: the HTML text must be contenteditable to catch Key Event.<br>
1261 </note>
1261
1261
1261 <div id="Shading_1" class="ShadingPlate" draggable="true" contenteditable="true">
1261 <div id="Shading_1_Log" class="ShadingLog" draggable="true"></div>
1261 <div id="Shading_1_Canvas" class="ShadingCanvas" width="300px" height="300px" draggable="true" style="">My Canvas.</div>
1261
1261 <style>
1262 .ShadingPlate {
1262     z-index:0;
1262     position:static;
1262     overflow:scroll;
1262     display:block;
1262     width:100%;
1262     height:400px;
1262     font-size:9pt;
1262     font-family:Courier New;
1262     border:1px dashed #000;
1262     color:#444;
1262 }
1263 .ShadingLog {
1263     z-index:1;
1263     position:relative;
1263     display:block;
1263     top:0px;
1263     left:0px;
1263     overflow:scroll;
1263     width:100%;
1263     font-size:9pt;
1263     font-family:Courier New;
1263     color:#666;
1263     overflow:scroll;
1263     background:rgba(200,255,200,0.4);
1263     height:400px;
1263 }
1264 .ShadingHtml {
1264     z-index:2;
1264     position:relative;
1264     display:block;
1264     top:0px;
1264     left:0px;
1264     overflow:scroll;
1264     width:100%;
1264     font-size:12pt;
1264     font-family:Courier New;
1264     color:#666;
1264     overflow:scroll;
1264     background:rgba(200,255,200,0.4);
1264     height:400px;
1264 }
1265 .ShadingCanvas {
1265     z-index:3;
1265     position:relative;
1265     xdisplay:block;
1265     top:0px;
1265     left:100px;
1265     resize:both;
1265     border:1px solid #000;
1265 }
1266 </style>
1266 <input id="Shading_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
1266 <input id="Shading_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
1266 <input id="Shading_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
1266 <span id="Shading_WorkCodeView"></span>
1266 <script id="Shading_WorkCodeScript">
1266 function Shading openWorkCodeView(){
1266     function Shading_showWorkCode(){
1266         showHtmlCode(Shading_WorkCodeView,Shading_WorkCodeSpan);
1266     }
1267     Shading_WorkCodeViewOpen.addEventListener('click',Shading_showWorkCode);
1267 }
1267 const BR = '<br>';
1267 Shading openWorkCodeView(); // should be invoked by an event
1267 function sh_onClick(e){
1267     Shading_1_Log.innerHTML += 'Click '+e.target.nodeName+'#'+e.target.id
1267     + ' offset('+e.offsetX+', '+e.offsetY+')'
1267     + ' client('+e.clientX+', '+e.clientY+')'
1267     + ' page('+e.pageX+', '+e.pageY+')'
1267     + ' screen('+e.screenX+', '+e.screenY+')'
1267     +BR;
1267     e.stopPropagation();
1267     e.preventDefault();
1267 }
1267 function sh_onKeyUp(e){
1267     if( Shading_1.style.zIndex == '' ){
1267         Shading_1.style.zIndex = 0;
1267     }
1267     zi = parseInt(Shading_1.style.zIndex);
1267     if( e.key.length == 1 ){
1267         Shading_1_Log.innerHTML += e.key;
1267     }
1267     if( e.key == '0' ){ zi = 0; }else
1267     if( e.key == '+' ){ zi += 1; }else
1267     if( e.key == '-' ){ zi -= 1; }else
1267     if( e.key == 'c' ){
1267         Shading_1_Log.innerHTML = '';
1267     }else
1267     if( e.key == 'r' ){
1267         Shading_1.style.position = "relative";
1267         Shading_1.style.top = '0px';
1267         Shading_1.style.left = '0px';
1267         zi = 0;
1267     }else
1267     if( e.key == 'j' || e.code == 'ArrowDown' ){
1267         topx = parseInt(Shading_1.style.top) + 50;
1267         Shading_1.style.top = topx + 'px';
1267     }else
1267     if( e.key == 'k' || e.code == 'ArrowUp' ){
1267         topx = parseInt(Shading_1.style.top) - 50;
1267         Shading_1.style.top = topx + 'px';
1267     }else
1267     if( e.key == 'l' || e.code == 'ArrowRight' ){
1267         lefty = parseInt(Shading_1.style.left) + 50;
1267         Shading_1.style.left = lefty + 'px';
1267     }else
1267     if( e.key == 'h' || e.code == 'ArrowLeft' ){
1267         lefty = parseInt(Shading_1.style.left) - 50;
1267         Shading_1.style.left = lefty + 'px';
1267     }else
1267     if( e.key == 'a' ){
1267         Shading_1.style.position = "absolute";
1267         Shading_1.style.top = '0px';
1267         Shading_1.style.left = '0px';
1267     }else{
1267     }
1267     Shading_1.style.zIndex = zi;
1267     Shading_1_Log.innerHTML += 'KeyUp. '+e.target.nodeName+'#'+e.target.id
1267     + '0('+e.keys+'/'+e.code+')'
1267     + 'z-index:'+zi+'/' +Shading_1.style.zIndex
1267     + 'top:'+Shading_1.style.top

```



```

12744     +BR;
12745     e.stopPropagation();
12746     e.preventDefault();
12747   }
12748   function sh_onkeydown(e){
12749     Shading_1_Log.innerHTML += 'keydown'+e.target.nodeName+'#'+e.target.id
12750     + 'Down('+e.keys+''+e.code+')'+BR;
12751     e.stopPropagation();
12752     e.preventDefault();
12753   }
12754 function Shading_Setup(){
12755   Shading_1_Log.innerHTML += '<'+h4>Click here<'+h4>';
12756   Shading_1.addEventListener('keydown',sh_onkeydown);
12757   Shading_1.addEventListener('keyup',sh_onkeyup);
12758   Shading_1.addEventListener('click',sh_onClick);
12759   Shading_1.addEventListener('click',sh_onClick);
12760 }
12761 Shading_1_Log.style.top = "-400px";
12762 Shading_1_Log.style.left = "200px";
12763 }
12764 Shading_1.appendChild(Shading_1_Canvas);
12765 Shading_1_Canvas.style.width = "300px";
12766 Shading_1_Canvas.style.height = "300px";
12767 Shading_1_Canvas.style.position = "relative";
12768 Shading_1_Canvas.style.top = "-750px";
12769 Shading_1_Canvas.style.left = "100px";
12770 }
12771 const ctx = Shading_1_Canvas.getContext('2d');
12772 ctx.fillStyle = 'rgba(160,0,0,0.9)';
12773 ctx.fillRect(50,50,40,40);
12774 ctx.fillStyle = 'rgba(0,160,0,0.9)';
12775 ctx.fillRect(60,60,40,40);
12776 ctx.fillStyle = 'rgba(0,0,160,0.9)';
12777 ctx.fillRect(70,70,40,40);
12778 }
12779 function Reset_ShadingCanvas(){
12780   Shading_1_Log.removeAttribute('style');
12781   Shading_1_Log.innerHTML = "";
12782   Shading_1_Canvas.style = "";
12783   //Shading_1_Canvas.removeAttribute('style');
12784 }
12785 }
12786 </script>
12787 </details>
12788 <!-- Shading_WorkCodeSpan -->
12789 *//</span>
12790 //<!-- ===== Work } ===== -->
12791 }
12792 }
12793 //<!-- ===== Work { ===== -->
12794 <span id="Charmap_WorkCodeSpan">
12795 />
12796 <details id="Charmap_Work"><summary>Character Map Mandala</summary>
12797 <!-- ----- UnicodeCharmap // 2020-1015 SatoxITS { -->
12798 <h2>Unicode Character Map</h2>
12799 <note>code 0x0000 - 0xFFFF / 16px / 3200 x 3200 px / zoom:0.25</note>
12800 <div id="Charmap_1_Frame" style="border: 1px solid black; padding: 5px;">
12801 <div id="Charmap_1_Text" class="Charmap">
12802 </div>
12803 </div>
12804 <br>
12805 }
12806 <style>
12807 #Charmap_1_Frame {
12808   overflow:scroll;
12809   height:3200px; width:3200px;
12810   transform:scale(0.5);
12811   zoom:0.25;
12812   resize:both;
12813   background-color:#fff;
12814 }
12815 .Charmap {
12816   zoom:0.25;
12817   font-size:16px;
12818   line-height:1.0;
12819   xfont-family:Georgia;
12820   color:#000;
12821 }
12822 </style>
12823 <script>
12824 function charmapgen(){
12825   text = "";
12826   for( cc = 0; cc < 0x10000; cc++){
12827     text += String.fromCharCode(cc);
12828   }
12829   Charmap_1_Text.innerHTML = text;
12830 }
12831 Charmap_Work.addEventListener('click',charmapgen);
12832 //charmapgen();
12833 </script>
12834 <input id="Charmap_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
12835 <input id="Charmap_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
12836 <input id="Charmap_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
12837 <span id="Charmap_WorkCodeView"></span>
12838 <script id="Charmap_WorkCodeScript">
12839 function Charmap_openWorkCodeView(){
12840   function Charmap_showWorkCode(){
12841     showHtmlCode(Charmap_WorkCodeView,Charmap_WorkCodeSpan);
12842   }
12843   Charmap_WorkCodeViewOpen.addEventListener('click',Charmap_showWorkCode);
12844 }
12845 Charmap_openWorkCodeView(); // should be invoked by an event
12846 </script>
12847 </details>
12848 <!-- Charmap_WorkCodeSpan -->
12849 *//</span>
12850 //<!-- ===== Work } ===== -->
12851 }
12852 }
12853 //<!-- ===== Work { ===== -->
12854 <span id="Pointillism_WorkCodeSpan">
12855 />
12856 <details><summary>Collaborated Pointillism</summary>
12857 <!-- ----- CollaboratedPointillism // 2020-1016 SatoxITS { -->
12858 <h2><a name="Pointillism" class="Pointillism"><a href="#Pointillism">Collaborated Pointillism</a></h2>
12859 }
12860 <input id="Pointillism_1_Share" type="checkbox" class="HtmlCodeViewButton" value="Share"> Share
12861 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ResetCanvas()" value="Reset">
12862 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Clear">
12863 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ReplayCanvas()" value="Replay">
12864 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_RepeatCanvas()" value="Repeat">
12865 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Save">
12866 <input type="button" class="HtmlCodeViewButton" onclick="Pointillism_1_ClearCanvas()" value="Load">
12867 <div id="Pointillism_1" class="Pointillism">
12868 <span id="Pointillism_1_Unit_1" class="Pointillism_Unit">
12869 <span id="Pointillism_1_XY_1" class="Pointillism_XY">XY1</span>
12870 <span id="Pointillism_1_XY_1_Remote" class="Pointillism_XY_Remote">XY1-remote</span>
12871 <canvas id="Pointillism_1_Canvas_1" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12872 </span>
12873 <span id="Pointillism_1_Unit_2" class="Pointillism_Unit">
12874 <span id="Pointillism_1_XY_2" class="Pointillism_XY">XY2</span>
12875 <span id="Pointillism_1_XY_2_Remote" class="Pointillism_XY_Remote">XY2-remote</span>
12876 <canvas id="Pointillism_1_Canvas_2" class="Pointillism_Canvas" width="300px" height="300px"></canvas>
12877 </span>
12878 </div>
12879 <br>
12880 }
12881 <style>
12882 .Pointillism {
12883   display:block;
12884   resize:both;
12885   width:600px;
12886   height:380px;
12887   min-width:240px;
12888   min-height:270px;
12889   background-color:#eee;
12890   overflow:scroll;
12891   font-size:16px;
12892   font-family:Georgia;
12893   color:#000;
12894   vertical-align:middle;
12895 }
12896 .Pointillism_Unit {
12897   position:relative;
12898   top:10px;
12899   display:block;
12900   overflow:scroll;
12901   width:300px;
12902   height:350px;
12903   margin:5px;
12904   padding:10px;
12905   background-color:rgba(255,255,127,0.7);
12906 }
12907 .Pointillism_XY {
12908   display:block;
12909   vertical-align:middle;
12910   width:290px;
12911   xheight:20px;
12912   font-size:12px;
12913   line-height:1.2;
12914   padding:5px;
12915   margin:0px;
12916   color:#fff;
12917 }

```

```

12921 background-color:#44c;
12922 }
12923 .Pointillism_XY_Remote {
12924 display:block;
12925 vertical-align:middle;
12926 width:200px;
12927 xheight:20px;
12928 font-size:12px;
12929 line-height:1.2;
12930 padding:5px;
12931 color:#fff;
12932 background-color:#4a4;
12933 }
12934 .Pointillism_Canvas {
12935 display:block;
12936 position:relative;
12937 xpadding:20px;
12938 xleft:20px;
12939 xtop:20px;
12940 background-color:#333;
12941 }
12942 </style>
12943 <script>
12944 var points = [];
12945 var replay = [];
12946 var replayx = 0;
12947 function pClearCanvas(can){
12948   ctx = can.getContext('2d');
12949   ctx.clearRect(0,0,can.width,can.height);
12950 }
12951 function Pointillism_1_ClearCanvas(){
12952   pClearCanvas(Pointillism_1_Canvas_1);
12953   pClearCanvas(Pointillism_1_Canvas_2);
12954 }
12955 function PointsReset(){
12956   points = [];
12957   replay = [];
12958   inRepeat = false;
12959   inReplay = false;
12960   Pointillism_1_ClearCanvas();
12961 }
12962 function Pointillism_1_ResetCanvas(){
12963   PointsReset();
12964   if( Pointillism_1_Share.checked ){
12965     //alert('---broadcast reset\n');
12966     GJ_BoastMessage('DRAW RESET');
12967   }
12968 }
12969 function Pointillism_1_ResetCanvasReceive(){
12970   //alert('---received reset\n');
12971   PointsReset();
12972 }
12973 function drawPoint(can,x,y,r,g,b){
12974   const ctx = can.getContext('2d');
12975   ctx.fillStyle = 'rgba('+r+', ' +g+', ' +b+', 0.7)';
12976   ctx.fillRect(x,y,8,8);
12977 }
12978 function waitMs(serno,ms){
12979   console.log('--- wait #' +serno+ ' '+ms+'ms');
12980   until = new Date();
12981   now = until.getTime();
12982   untilMs = now + ms;
12983   for( wi = 0; ; wi++ ){
12984     now = new Date();
12985     nowMs = now.getTime();
12986     remMs = untilMs - nowMs;
12987     //console.log('wait '+wi+': '+remMs+' '+ms);
12988     if( remMs < 0 ){
12989       break;
12990     }
12991   }
12992 }
12993 var inReplay = false;
12994 function replay(){
12995   rx = replayx;
12996   if( replay.length <= rx ){
12997     return;
12998   }
12999   replayx += 1;
13000   pl = replay[rx];
13001   if( pl[1] == 1 ){
13002     can = Pointillism_1_Canvas_1;
13003   }else{
13004     can = Pointillism_1_Canvas_2;
13005   }
13006   drawPoint(can,pl[2],pl[3],pl[4],pl[5],pl[6]);
13007   if( inReplay == false ){
13008     console.log('wait '+replayx+' Stopped');
13009     return;
13010   }
13011   if( rx < replay.length-1 ){
13012     prevMs = replay[rx][0].getTime();
13013     nextMs = replay[rx+1][0].getTime();
13014     delayMs = nextMs - prevMs;
13015     //console.log('wait '+replayx+' '+delayMs+'ms');
13016     window.setTimeout(replay,delayMs);
13017   }else{
13018     console.log('wait '+replayx+' Finished');
13019     if( inRepeat ){
13020       window.setTimeout(repeat1,1000);
13021     }
13022   }
13023 }
13024 function Pointillism_1_ReplayCanvas(can){
13025   Pointillism_1_ClearCanvas();
13026   replay = points;
13027   replayx = 0;
13028   inReplay = true;
13029   replay1();
13030 }
13031 var inRepeat = false;
13032 function repeat1(){
13033   Pointillism_1_ClearCanvas();
13034   replay = points;
13035   replayx = 0;
13036   replay1();
13037   if( inRepeat ){
13038     //window.setTimeout(repeat1,1000);
13039   }
13040 }
13041 function Pointillism_1_RepeatCanvas(can){
13042   if( inRepeat ){
13043     inRepeat = false;
13044     inReplay = false;
13045   }else{
13046     inRepeat = true;
13047     inReplay = true;
13048     repeat1();
13049   }
13050 }
13051 }
13052 function CopyLocal(){ return Pointillism_1_Share.checked == false; }
13053 function Pointillism_Setup(){
13054   var moveCount1 = 0;
13055   var moveCount2 = 0;
13056 }
13057 var gJlinked = false;
13058 function GJdraw(msg){
13059   if( gJlinked == false ){
13060     //GJLink_Section.open = true;
13061     GJ_Join();
13062     gJlinked = true;
13063   }
13064   GJ_BoastMessage('DRAW '+msg);
13065 }
13066 function showXY1(e){
13067   moveCount1 += 1;
13068   x = e.offsetX;
13069   y = e.offsetY;
13070   Pointillism_1_XY_1.innerHTML = 'XY1: ' + 'x='+x + ', y='+y + ' /'+moveCount1+'/' +points.length;
13071   Pointillism_1_XY_2_Remote.innerHTML = 'XY1: ' + 'x='+x + ', y='+y + ' /'+moveCount1;
13072   if( e.buttons || CopyLocal() ){
13073     points.push([new Date(),1,x,y,64,64,255]);
13074     drawPoint(Pointillism_1_Canvas_1,x,y,128,128,255);
13075     if( CopyLocal() ){
13076       drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
13077     }
13078     GJdraw('1','x+', 'y');
13079   }
13080 }
13081 function showXY2(e){
13082   moveCount2 += 1;
13083   x = e.offsetX;
13084   y = e.offsetY;
13085   Pointillism_1_XY_2.innerHTML = 'XY2: ' + 'x='+x + ', y='+y + ' /'+moveCount2+'/' +points.length;
13086   Pointillism_1_XY_1_Remote.innerHTML = 'XY2: ' + 'x='+x + ', y='+y + ' /'+moveCount1;
13087   if( e.buttons || CopyLocal() ){
13088     points.push([new Date(),2,x,y,64,255,64]);
13089     drawPoint(Pointillism_1_Canvas_2,x,y,128,255,128);
13090     if( CopyLocal() ){
13091       drawPoint(Pointillism_1_Canvas_1,x,y,160,255,160);
13092     }
13093     //GJdraw('2','x+', 'y');
13094   }
13095 }
13096 Pointillism_1_Canvas_1.addEventListener('mousemove',showXY1);
13097 Pointillism_1_Canvas_2.addEventListener('mousemove',showXY2);

```

```

13098 Pointillism_1_Unit_2.style.left = '340px';
13099 Pointillism_1_Unit_2.style.top = '-375px';
13100
13101 function Pointillism_RemoteDraw(arg){
13102 //alert('Draw at '+arg);
13103 //drawPoint(Pointillism_1_Canvas_2,x,y,160,160,255);
13104 if( arg == "RSSM" ){
13105 Pointillism_1_ResetCanvasReceive();
13106 }else{
13107   argv = arg.split(',');
13108   x = argv[1];
13109   y = argv[2];
13110   Pointillism_1_XY_2_Remote.innerHTML = 'XYR: ' + 'x'+x +', y='+y + '/' +points.length;
13111   drawPoint(Pointillism_1_Canvas_2,x,y,255,0,0);
13112 }
13113 }
13114 </script>
13115
13116
13117 <input id="Pointillism_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13118 <input id="Pointillism_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13119 <input id="Pointillism_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13120 <span id="Pointillism_WorkCodeView"></span>
13121 <script id="Pointillism_WorkScript">
13122 function Pointillism_openWorkCodeView(){
13123   function Pointillism_showWorkCode(){
13124     showHtmlCode(Pointillism_WorkCodeView,Pointillism_WorkCodeSpan);
13125   }
13126   Pointillism_WorkCodeViewOpen.addEventListener('click',Pointillism_showWorkCode);
13127 }
13128 Pointillism_openWorkCodeView(); // should be invoked by an event
13129 </script>
13130 </details>
13131 <!-- Pointillism_WorkCodeSpan } -->
13132 *//</span>
13133 //<!-- ===== Work } ===== -->
13134
13135
13136
13137 //<!-- ===== Work { ===== -->
13138 //<span id="StatCounter_WorkCodeSpan">
13139 /*
13140 <details><summary>StatCounter</summary>
13141 <!-- ----- StatCounter// 2020-1018 SatoxITS { -->
13142 <h2>StatCounter</h2>
13143
13144 <div class="statcounter"><a title="hit counter" href="https://statcounter.com/" target="_blank"></a>
13145 (counter as image tag)</div>
13146 <style>
13147 .statcounter {
13148   vertical-align:middle;
13149 }
13150 #sc_SatoxITS {
13151   color:#000;
13152   font-size:12pt;
13153   height:30px;
13154   width:100%;
13155   background-color:#ddd;
13156 }
13157 </style>
13158
13159 <div>
13160 <script>
13161   var sc_project=12411639;
13162   var sc_invisible=0;
13163   var sc_security="1aeb2a3a";
13164   var sc_https="1";
13165   var scJsHost = "https://";
13166 </script>
13167 <!-- /script src="https://statcounter.com/counter/counter.js" -->
13168 </script --> (counter by inline script)
13169 </div>
13170
13171 <input id="StatCounter_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
13172 <input id="StatCounter_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
13173 <input id="StatCounter_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
13174 <span id="StatCounter_WorkCodeView"></span>
13175 <script id="StatCounter_WorkScript">
13176 function StatCounter_openWorkCodeView(){
13177   function StatCounter_showWorkCode(){
13178     showHtmlCode(StatCounter_WorkCodeView,StatCounter_WorkCodeSpan);
13179   }
13180   StatCounter_WorkCodeViewOpen.addEventListener('click',StatCounter_showWorkCode);
13181 }
13182 StatCounter_openWorkCodeView(); // should be invoked by an event
13183 </script>
13184 </details>
13185 <!-- StatCounter_WorkCodeSpan } -->
13186 *//</span>
13187 //<!-- ===== Work } ===== -->
13188
13189
13190
13191 //<!-- ===== Work { ===== -->
13192 //<span id="CascadedCanvasBook_WorkCodeSpan">
13193 /*
13194 <details id="CascadedCanvasBook_Section"><summary id="CascadedCanvasBook_Summary">CascadedCanvasBook</summary>
13195 <!-- ----- CascadedCanvasBook // 2020-1031 SatoxITS { -->
13196 <h2>Cascaded Canvas Book</h2>
13197
13198 <div id="CBPanel" class="CBPanel">
13199 <input id="CS_new" type="button" value="NewCanvas">
13200 </div>
13201 <div>
13202 <h3>Undo / Redo / Replay</h3>
13203 <div id="CanvasTool_UndoRedo">
13204 <span id="DrawReplay" class="CanvasPool" draggable="true" contenteditable="">
13205 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13206 <input class="CV_Button" type="button" value="Redraw">
13207 From <input data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
13208 To <input id="DrawingSernoView" data-name="D" class="ColorParam" type="text" value="0" onwheel="OnWheelInt()">
13209 </span>
13210 </div>
13211 <script>
13212 function childByName(node,name){
13213   for( let i = 0; i < node.children.length; i++){
13214     ch = node.children[i];
13215     name1 = ch.getAttribute('data-name');
13216     if( name1 == name ){
13217       return ch;
13218     }
13219   }
13220   return null;
13221 }
13222 function OnWheelInt(){
13223   event.preventDefault();
13224   t = event.target;
13225   n = t.nodeName;
13226   i = t.id;
13227   p = t.parentNode;
13228   y = event.deltaY;
13229   //console.log('OnWheelInt '+ys' '+n+'#'+i' '+t.value);
13230   if( y < 0 ){ // scroll forward (up)
13231     inc = -y;
13232   }else{
13233     inc = y;
13234   }
13235   inc /= 6;
13236   val = parseFloat(t.value) + inc;
13237   t.value = val.toFixed(0);
13238   return val;
13239 }
13240 var DrawingSerno = 0;
13241 function saveDrawing(){
13242   DrawingSerno += 1;
13243   DrawingSernoView.value = DrawingSerno;
13244 }
13245 function to2x(x){
13246   if( x <= 0xF ){
13247     return '0'+x.toString(16);
13248   }else{
13249     return x.toString(16);
13250   }
13251 }
13252 </script>
13253 <div id="InstaColorPicker" draggable="true">
13254 <h3>Color Picker</h3>
13255 <div id="CanvasColor">
13256
13257 Select value by
13258 <input id="ICPMotion" type="checkbox" checked="">Mouse Motion
13259 <input id="ICPautoAddMotion" type="checkbox" checked="">Auto. add to history
13260 <input id="ICPautoAddWheel" type="checkbox" checked="">Mouse Wheel
13261 <input id="ICPautoAddWheel" type="checkbox" checked="">Auto. add to history
13262
13263 <div data-name="Fore" id="CanvasTool_Color_Fore" class="CanvasPool" onchange="showColorISample()">
13264 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13265 <input class="CV_Button" type="button" value="Fore">
13266 <input data-name="R" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">

```

```

13275 <input data-name="G" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13276 <input data-name="B" class="CanvasParam" type="text" value="32" onwheel="OnWheelHex()">
13277 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13278 <input data-name="C" class="ColorParam" type="text" value="#000000ff">
13279 <span data-name="Sample">Sample</span>
13280 </div>
13281
13282 <div data-name="Fill" id="CanvasTool_Color_Fill" class="CanvasTool" onchange="showColorSample()">
13283 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13284 <input class="CV_Button" type="button" value="Fill">
13285 <input data-name="R" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13286 <input data-name="G" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13287 <input data-name="B" class="CanvasParam" type="text" value="0" onwheel="OnWheelHex()">
13288 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13289 <input data-name="C" class="ColorParam" type="text" value="#000000ff">
13290 <span data-name="Sample">Sample</span>
13291 </div>
13292
13293 <div data-name="Back" id="CanvasTool_Color_Back" class="CanvasTool" onchange="showColorSample()">
13294 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13295 <input class="CV_Button" type="button" value="Back">
13296 <input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13297 <input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13298 <input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13299 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13300 <input data-name="C" class="ColorParam" type="text" value="ffffff">
13301 <span data-name="Sample">Sample</span>
13302 </div>
13303
13304 <div data-name="Border" id="CanvasTool_Color_Border" class="CanvasTool" onchange="showColorSample()">
13305 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13306 <input class="CV_Button" type="button" value="Border">
13307 <input data-name="R" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13308 <input data-name="G" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13309 <input data-name="B" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13310 <input data-name="A" class="CanvasParam" type="text" value="255" onwheel="OnWheelHex()">
13311 <input data-name="C" class="ColorParam" type="text" value="ffffff">
13312 <span data-name="Sample">Sample</span>
13313 </div>
13314
13315 <div id="ColorComposition" class="ColorComposition">
13316 Sample
13317 </div>
13318 <br>
13319 <input class="LargeButton CV_Button" type="button" value="Clear Color History" onclick="ClearColorHistory()">
13320 <input id="LenColorHistory" class="CanvasParam" type="text" value="0">
13321 / Max <input id="MaxColorHistory" class="CanvasParam" type="text" value="200">
13322 <div id="Color0" class="Color1" onclick="SelectThisColor()"></div>
13323 <div id="Color1" class="ColorHistory"></div>
13324 </div>
13325 </div>
13326
13327 <style>
13328 .LargeButton {
13329 font-size:14px !important;
13330 width:160px !important;
13331 color:#f00 !important;
13332 }
13333 .ColorComposition {
13334 color:#000000ff !important;
13335 font-size:16pt !important;
13336 padding:5px !important;
13337 border:2px solid #000000ff !important;
13338 }
13339 .Color1 {
13340 width:100% !important;
13341 height:10px !important;
13342 font-family:Courier New !important;
13343 font-size:10px !important;
13344 background-color:#000 !important;
13345 }
13346 .Color1:hover {
13347 border:1px solid #000;
13348 }
13349 .ColorHistory {
13350 resize:vertical;
13351 //overflow:scroll;
13352 width:100% !important;
13353 height:200px !important;
13354 }
13355 </style>
13356
13357 <script>
13358 var ColorID = 0;
13359 var LastColor = Color1;
13360 var HistLength = 0;
13361 function ClearColorHistory(){
13362 ColorHistory.innerHTML = '';
13363 cl = Color0.cloneNode();
13364 cl.id = 'Color1';
13365 LastColor = document.getElementById('Color1');
13366 ColorHistory.appendChild(cl);
13367 HistLength = 0;
13368 LenColorHistory.value = '0';
13369 }
13370 function OnWheelHex(){
13371 event.preventDefault();
13372 t = event.target;
13373 n = t.nodeName;
13374 i = t.id;
13375 p = t.parentNode;
13376 y = event.deltaY;
13377 inc = -y; // scroll forward (up)
13378 inc /= 6;
13379
13380 val = parseFloat(t.value) + inc;
13381 val = val.toFixed(0);
13382 if( val < 0 ) val = 0;
13383 if( 255 < val ) val = 255;
13384 if( t.value != val || event.ctrlKey ){
13385 t.value = val;
13386 if (ICPaintAddWheel.checked){
13387 showColorSample();
13388 }
13389 }
13390 return val;
13391 }
13392 function SelectThisColor(){
13393 ok = confirm('Pick this color ? '+event.target.innerHTML);
13394 if( ok ){
13395 // add to Picked
13396 }
13397 }
13398 }
13399 }
13400 var LastMotion = 0;
13401 function motionColor(){
13402 if( !ICMotion.checked ){
13403 return;
13404 }
13405 d = new Date();
13406 if( d.getTime() - LastMotion < 100 ){
13407 return;
13408 }
13409 LastMotion = d.getTime();
13410
13411 t = CanvasTool_Color_Fore;
13412 R = childByName(t,'R');
13413 G = childByName(t,'G');
13414 B = childByName(t,'B');
13415 A = childByName(t,'A');
13416 // console.log('mouse motion '+event.x+', '+event.y+' target#'+t);
13417
13418 updated = false;
13419
13420 val = 255 * (event.x/window.innerWidth);
13421 val = val.toFixed(0);
13422 if( B.value != val || event.ctrlKey ){
13423 B.value = val;
13424 updated = true;
13425 }
13426 val = 255 * (event.y/window.innerHeight);
13427 val = val.toFixed(0);
13428 if( G.value != val || event.ctrlKey ){
13429 G.value = val;
13430 updated = true;
13431 }
13432 if( updated ){
13433 showColorSample1(t,ICPaintAddMotion.checked);
13434 }
13435 }
13436 function scrollColor(){
13437 if( !ICMotion.checked ){
13438 return;
13439 }
13440 d = new Date();
13441 if( d.getTime() - LastMotion < 100 ){
13442 return;
13443 }
13444 LastMotion = d.getTime();
13445
13446 t = CanvasTool_Color_Fore;
13447 R = childByName(t,'R');
13448 G = childByName(t,'G');
13449 B = childByName(t,'B');
13450 A = childByName(t,'A');
13451

```

```

13452 updated = false;
13453
13454 y = gsh.getBoundingClientRect().top.toFixed(0)
13455 if( y < 0 ) y += -1;
13456 size = 10000;
13457 val = 255 * (y/size);
13458 val = val.toFixed(0);
13459 if( 255 < val ) val = 255;
13460 if( R.value != val || event.ctrlKey ){
13461 R.value = val;
13462 updated = true;
13463 }
13464 if( updated ){
13465 showColorSample(t,ICPautoAddMotion.checked);
13466 }
13467 }
13468 }
13469
13470 function showColorSample(){
13471 t = event.target.parentNode;
13472 showColorSample(t,ICPautoAddWheel);
13473 }
13474
13475 function showColorSample(t,add){
13476 name = t.getAttribute('data-name');
13477
13478 R = childByName(t,'R').value;
13479 G = childByName(t,'G').value;
13480 B = childByName(t,'B').value;
13481 A = childByName(t,'A').value;
13482
13483 R = parseInt(R);
13484 G = parseInt(G);
13485 B = parseInt(B);
13486 A = parseInt(A);
13487
13488 R = to02x(R); //R.toString(16);
13489 G = to02x(G); //G.toString(16);
13490 B = to02x(B); //B.toString(16);
13491 A = to02x(A); //A.toString(16);
13492
13493 color = '#'+R+G+B+A;
13494 //console.log(name+' color='+color);
13495
13496 C = childByName(t,'C');
13497 C.value = color;
13498 S = childByName(t,'Sample');
13499 S.style.color = color;
13500
13501 ColorID += 1;
13502 cl = ColorI;
13503 clid = cl.id;
13504 cl.id = "color_"+ColorID;
13505 cl.innerHTML = cl.id +
13506 '<div style="background-color:black">'+color+'</div><div style="font color:white">'+color+'</div>';
13507 if( name == 'Fore' ){
13508 ColorComposition.style.setProperty('color',color,'important');
13509 cl.style.setProperty('color',color,'important');
13510 cl.style.setProperty('background-color',color,'important');
13511 }
13512 if( name == 'Fill' ){
13513 ColorComposition.style.setProperty('border-color',color,'important');
13514 cl.style.setProperty('color',color,'important');
13515 cl.style.setProperty('background-color',color,'important');
13516 }
13517 if( name == 'Back' ){
13518 ColorComposition.style.setProperty('background-color',color,'important');
13519 cl.style.setProperty('background-color',color,'important');
13520 }
13521 if( name == 'Border' ){
13522 ColorComposition.style.setProperty('border-color',color,'important');
13523 cl.style.setProperty('background-color',color,'important');
13524 cl.style.setProperty('border-color',color,'important');
13525 }
13526 ccl = cl.cloneNode(true);
13527 cl.id = clid;
13528
13529 if( add ){
13530 max = parseInt(MaxColorHistory.value);
13531 if( HistLength < max ){
13532 HistLength++;
13533 LenColorHistory.value = HistLength;
13534 ColorHistory.insertBefore(ccl,LastColor);
13535 LastColor = ccl;
13536 }
13537 if( max <= HistLength ){
13538 LenColorHistory.style.setProperty('background-color','#f00','important');
13539 }else{
13540 LenColorHistory.style.setProperty('background-color','#fff','important');
13541 }
13542 }
13543
13544 function InstaColor_Setup1(){
13545 window.addEventListener('mousemove',motionColor);
13546 window.addEventListener('scroll',scrollColor);
13547 //window.addEventListener('click',motionColor); // click is generated for animation
13548
13549 fi = document.getElementById('FeaturesView');
13550 if( fi != null ){
13551 fi.appendChild(InstaColorPicker);
13552 //cs = document.getElementById('InstaColorSpan');
13553 //cs.appendChild(InstaColorPicker);
13554 //ci.hidden = false;
13555 //ci.open = true;
13556 }
13557 }
13558
13559 function InstaColor_Setup(){
13560 if( CascadedCanvasBook.Section.open ){
13561 InstaColor_Setup1();
13562 }
13563 CascadedCanvasBook_Summary.addEventListener('click',InstaColor_Setup1);
13564 }
13565 }
13566 </script>
13567
13568 <h3>Colors</h3>
13569
13570 <div id="CanvasColors">
13571 <div id="CanvasTool_Color_0" class="CanvasTool" onchange="showColorSample()">
13572 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13573 <input class="CV_Button" type="button" value="Trans">
13574 <span data-name="1" class="ColorParam" type="text"><input type="text" value="0.0"></span>
13575 <input data-name="BW" class="CanvasParam" type="text" value="1">
13576 <input data-name="FC" class="ColorParam" type="text" value="0.9">
13577 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13578 <span data-name="FCS" class="ColorSample">xxxx</span>
13579 <span data-name="BCS" class="ColorSample">xxxx</span>
13580 <input data-name="BC" class="ColorParam" type="text" value="#000000">
13581 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13582 </div>
13583 <div id="CanvasTool_Color_1" class="CanvasTool" onchange="showColorSample()">
13584 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13585 <input class="CV_Button" type="button" value="Mono">
13586 <span data-name="1" class="ColorParam" type="text"><input type="text" value="1"></span>
13587 <input data-name="BW" class="CanvasParam" type="text" value="1">
13588 <input data-name="FC" class="ColorParam" type="text" value="0.9">
13589 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13590 <span data-name="FCS" class="ColorSample">xxxx</span>
13591 <span data-name="BCS" class="ColorSample">xxxx</span>
13592 <input data-name="BC" class="ColorParam" type="text" value="#000000">
13593 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13594 </div>
13595 <div id="CanvasTool_Color_2" class="CanvasTool" onchange="showColorSample()">
13596 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13597 <input class="CV_Button" type="button" value="Red">
13598 <span data-name="1" class="ColorParam" type="text"><input type="text" value="1"></span>
13599 <input data-name="BW" class="CanvasParam" type="text" value="1">
13600 <input data-name="FC" class="ColorParam" type="text" value="#c82020">
13601 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13602 <span data-name="FCS" class="ColorSample">xxxx</span>
13603 <span data-name="BCS" class="ColorSample">xxxx</span>
13604 <input data-name="BC" class="ColorParam" type="text" value="0.9">
13605 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13606 </div>
13607 <div id="CanvasTool_Color_3" class="CanvasTool" onchange="showColorSample()">
13608 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13609 <input class="CV_Button" type="button" value="Green">
13610 <span data-name="1" class="ColorParam" type="text"><input type="text" value="1"></span>
13611 <input data-name="BW" class="CanvasParam" type="text" value="1">
13612 <input data-name="FC" class="ColorParam" type="text" value="0.9">
13613 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13614 <span data-name="FCS" class="ColorSample">xxxx</span>
13615 <span data-name="BCS" class="ColorSample">xxxx</span>
13616 <input data-name="BC" class="ColorParam" type="text" value="#20c820">
13617 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13618 </div>
13619 <div id="CanvasTool_Color_4" class="CanvasTool" onchange="showColorSample()">
13620 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13621 <input class="CV_Button" type="button" value="Blue">
13622 <span data-name="1" class="ColorParam" type="text"><input type="text" value="1"></span>
13623 <input data-name="BW" class="CanvasParam" type="text" value="1">
13624 <input data-name="FC" class="ColorParam" type="text" value="#680f0f">
13625 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13626 <span data-name="FCS" class="ColorSample">xxxx</span>
13627 <span data-name="BCS" class="ColorSample">xxxx</span>
13628 <input data-name="BC" class="ColorParam" type="text" value="#c0c0c0">
13629 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13630 </div>
13631 <div id="CanvasTool_Color_5" class="CanvasTool" onchange="showColorSample()">
13632 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">

```

```

13629 <input class="CV_Button" type="button" value="Yellow">
13630 #span data-name="I" class="ColorParam" type="text"><0-5</span>
13631 #W<input data-name="B" class="CanvasParam" type="text" value="1">
13632 #C<input data-name="FC" class="ColorParam" type="text" value="#fae600">
13633 <input data-name="FO" class="CanvasParam" type="text" value="0.9">
13634 #span data-name="BSC" class="ColorSample">xxx</span>
13635 #span data-name="BCS" class="ColorSample">xxx</span>
13636 #C<input data-name="BC" class="ColorParam" type="text" value="#ffffff">
13637 <input data-name="BO" class="CanvasParam" type="text" value="0.9">
13638 </div>
13639 </div>
13640
13641 <script>
13642 // https://developer.mozilla.org/en-US/docs/Web/CSS/color_value
13643 function genCSSColorStyle(color,opa){
13644 opa = parseFloat(opa);
13645 opa = opa;
13646 opa = opa.toFixed(0);
13647 if( 0xFF < opa ) opa = 0xFF;
13648 opa = parseInt(opa);
13649 opa = opa.toString(16);
13650 if( opa < 0x10 ) opa = '0' + opa;
13651 color += opa;
13652 return color;
13653 }
13654 function CV_GenColorStyle(cole,forFill){
13655 if( forFill ){
13656 col = childByName(cole, 'FC').value;
13657 opa = childByName(cole, 'FO').value;
13658 }else{
13659 col = childByName(cole, 'BC').value;
13660 opa = childByName(cole, 'BO').value;
13661 }
13662 color = genCSSColorStyle(col,opa);
13663 return color;
13664 }
13665 function xxxshowColorSample(){
13666 t = event.target;
13667 p = t.parentNode;
13668 alert('showColorSample '+event.target.nodeName+'/#'+t.id);
13669 }
13670 function showColorSample(){
13671 var csv = CanvasColors.children;
13672 //console.log('colors'+csv.length);
13673 for( i = 0; i < csv.length; i++ ){
13674 fc = childByName(csv[i], 'FC').value;
13675 bc = childByName(csv[i], 'BC').value;
13676 //console.log('colors'+csv[i].id+' fc='+fc+' bc='+bc);
13677 fcs = childByName(csv[i], 'FCS');
13678 fcs.style.color = fc;
13679 fcs.style.borderColor = fc;
13680 fcs.style.backgroundColor = bc;
13681
13682 bcs = childByName(csv[i], 'BCS');
13683 bcs.style.color = bc;
13684 bcs.style.borderColor = fc;
13685 bcs.style.backgroundColor = fc;
13686 }
13687 CV_redrawParts();
13688 }
13689 </script>
13690
13691 <style>
13692 .ColorSample {
13693 color:#fff;
13694 background-color:#fff;
13695 border:1px solid #000;
13696 margin:0px;
13697 padding:0px;
13698 width:12pt !important;
13699 height:12pt !important;
13700 }
13701 .ColorParam {
13702 color:#000 !important;
13703 font-family:Courier New !important;
13704 font-size:9pt !important;
13705 padding:2px !important;
13706 line-height:1.1 !important;
13707 height:14pt !important;
13708 width:55pt !important;
13709 text-align:left !important;
13710 display:inline !important;
13711 vertical-align:middle !important;
13712 }
13713 .CanvasParam {
13714 color:#000 !important;
13715 font-family:Courier New, Monospace !important;
13716 font-size:9pt !important;
13717 padding:2px !important;
13718 line-height:1.1 !important;
13719 height:14pt !important;
13720 width:30pt !important;
13721 text-align:right !important;
13722 display:inline !important;
13723 vertical-align:middle !important;
13724 }
13725 .CV_Button {
13726 padding:2pt !important;
13727 line-height:1.1 !important;
13728 border:2px inset #bbb !important;
13729 font-size:9pt !important;
13730 font-weight:normal !important;
13731 font-family:Georgia !important;
13732 border-radius:3px !important;
13733 color:#ddd; background-color:#66a !important;
13734 width:50pt;
13735 }
13736 .xxHtmlCodeviewText {
13737 font-size:9pt;
13738 font-family:Courier New;
13739 white-space:pre;
13740 }
13741 .xxCV_Button {
13742 font-family:Arial, Monospace, Courier New;
13743 color:#000;
13744 font-size:9pt;
13745 line-height:1.2;
13746 width:50pt;
13747 }
13748 </style>
13749
13750 <h3>Parts</h3>
13751 <div id="AppendToCanvas" class="CanvasTool" draggable="true">
13752 Resize<input class="CanvasParam" type="text" value="100" omhheel="OnWheelIntRedraw()">%
13753 Zoom<input class="CanvasParam" type="text" value="100" omhheel="OnWheelZoom()">%
13754 <input id="RedrawImmediate" type="checkbox" value="Redraw" checked="">Redraw Immediate
13755 </div>
13756 #span id="CanvasTools" class="CanvasTools" draggable="true" contenteditable=""
13757 <div id="CanvasTool_Clear" class="CanvasTool">
13758 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13759 <input data-name="rdr" class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
13760 #span data-name="ovid" class="CanvasParam" type="text">CI-0</span>
13761 #input type="checkbox" value="Fill" checked=""
13762 #input data-name="X" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">
13763 #input data-name="Y" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">
13764 #input data-name="Z" class="CanvasParam" type="text" value="1" omhheel="OnWheelIntRedraw()">
13765 #input data-name="W" class="CanvasParam" type="text" value="1000" omhheel="OnWheelIntRedraw()">
13766 #input data-name="H" class="CanvasParam" type="text" value="1000" omhheel="OnWheelIntRedraw()">
13767 #input data-name="R" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">
13768 #C<input data-name="C" class="CanvasParam" type="text" value="0">
13769 </div>
13770 <div id="CanvasTool_Rect" class="CanvasTool">
13771 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13772 <input data-name="rdr" class="CV_Button" type="button" value="Rect" onclick="CV_drawRect()">
13773 #span data-name="ovid" class="CanvasParam" type="text">RE-0</span>
13774 #input data-name="X" class="CanvasParam" type="text" value="40" omhheel="OnWheelIntRedraw()">
13775 #input data-name="Y" class="CanvasParam" type="text" value="60" omhheel="OnWheelIntRedraw()">
13776 #input data-name="Z" class="CanvasParam" type="text" value="2" omhheel="OnWheelIntRedraw()">
13777 #input data-name="W" class="CanvasParam" type="text" value="120" omhheel="OnWheelIntRedraw()">
13778 #input data-name="H" class="CanvasParam" type="text" value="80" omhheel="OnWheelIntRedraw()">
13779 #input data-name="R" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">
13780 #C<input data-name="C" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">
13781 #C<input data-name="C" class="CanvasParam" type="text" value="1">
13782 </div>
13783 <div id="CanvasTool_Circle" class="CanvasTool">
13784 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13785 <input data-name="rdr" class="CV_Button" type="button" value="Circle" onclick="CV_drawCircle()">
13786 #span data-name="ovid" class="CanvasParam" type="text">CI-0</span>
13787 #input data-name="F" type="checkbox" value="Fill" checked=""
13788 #input data-name="X" class="CanvasParam" type="text" value="100" omhheel="OnWheelIntRedraw()">
13789 #input data-name="Y" class="CanvasParam" type="text" value="100" omhheel="OnWheelIntRedraw()">
13790 #input data-name="Z" class="CanvasParam" type="text" value="3" omhheel="OnWheelIntRedraw()">
13791 #input data-name="R" class="CanvasParam" type="text" value="24" omhheel="OnWheelIntRedraw()">
13792 #input data-name="W" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">
13793 #input data-name="H" class="CanvasParam" type="text" value="360" omhheel="OnWheelIntRedraw()">
13794 #C<input data-name="C" class="CanvasParam" type="text" value="2">
13795 </div>
13796 <div id="CanvasTool_Packman" class="CanvasTool">
13797 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()">
13798 <input data-name="rdr" class="CV_Button" type="button" value="Packman" onclick="CV_drawPackman()">
13799 #span data-name="ovid" class="CanvasParam" type="text">PA-0</span>
13800 #input data-name="F" type="checkbox" value="Fill" checked=""
13801 #input data-name="X" class="CanvasParam" type="text" value="240" omhheel="OnWheelIntRedraw()">
13802 #input data-name="Y" class="CanvasParam" type="text" value="130" omhheel="OnWheelIntRedraw()">
13803 #input data-name="Z" class="CanvasParam" type="text" value="6" omhheel="OnWheelIntRedraw()">
13804 #input data-name="R" class="CanvasParam" type="text" value="45" omhheel="OnWheelIntRedraw()">
13805 #input data-name="S" class="CanvasParam" type="text" value="0" omhheel="OnWheelIntRedraw()">

```

```

13806 <input data-name="E" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()"/>
13807 #CO-<input data-name="C" class="CanvasParam" type="text" value="5">
13808 </div>
13809 <div id="CanvasTool_Ellipse" class="CanvasTool">
13810 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13811 <input data-name="rdr" class="CV_Button" type="button" value="Ellipse" onclick="CV_drawEllipse()"/>
13812 #<span data-name="ovid" class="CanvasParam" type="text">BA-0</span>
13813 <input data-name="F" type="checkbox" value="Fill" checked="">
13814 <input data-name="X" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()"/>
13815 <input data-name="Y" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()"/>
13816 <input data-name="Z" class="CanvasParam" type="text" value="4" onwheel="OnWheelIntRedraw()"/>
13817 <input data-name="RX" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()"/>
13818 <input data-name="RY" class="CanvasParam" type="text" value="20" onwheel="OnWheelIntRedraw()"/>
13819 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13820 <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13821 <input data-name="E" class="CanvasParam" type="text" value="360" onwheel="OnWheelIntRedraw()"/>
13822 #CO-<input data-name="C" class="CanvasParam" type="text" value="4">
13823 </div>
13824
13825 <div id="CanvasTool_Balloon" class="CanvasTool">
13826 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13827 <input data-name="rdr" class="CV_Button" type="button" value="Balloon" onclick="CV_drawBalloon()"/>
13828 #<span data-name="ovid" class="CanvasParam" type="text">BA-0</span>
13829 <input data-name="F" type="checkbox" value="Fill" checked="">
13830 <input data-name="X" class="CanvasParam" type="text" value="280" onwheel="OnWheelIntRedraw()"/>
13831 <input data-name="Y" class="CanvasParam" type="text" value="40" onwheel="OnWheelIntRedraw()"/>
13832 <input data-name="Z" class="CanvasParam" type="text" value="5" onwheel="OnWheelIntRedraw()"/>
13833 <input data-name="RX" class="CanvasParam" type="text" value="50" onwheel="OnWheelIntRedraw()"/>
13834 <input data-name="RY" class="CanvasParam" type="text" value="30" onwheel="OnWheelIntRedraw()"/>
13835 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13836 <input data-name="S" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13837 <input data-name="E" class="CanvasParam" type="text" value="100" onwheel="OnWheelIntRedraw()"/>
13838 #CO-<input data-name="C" class="CanvasParam" type="text" value="4">
13839 </div>
13840
13841 <div id="CanvasTool_Piechart" class="CanvasTool">
13842 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13843 <input data-name="rdr" class="CV_Button" type="button" value="Piechart" onclick="CV_drawPiechart()"/>
13844 #<span data-name="ovid" class="CanvasParam" type="text">BA-0</span>
13845 <input data-name="F" type="checkbox" value="Fill" checked="">
13846 <input data-name="X" class="CanvasParam" type="text" value="250" onwheel="OnWheelIntRedraw()"/>
13847 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()"/>
13848 <input data-name="Z" class="CanvasParam" type="text" value="7" onwheel="OnWheelIntRedraw()"/>
13849 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13850 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13851 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13852 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()"/>
13853 <input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()"/>
13854 #CO-<input data-name="C" class="CanvasParam" type="text" value="3">
13855 </div>
13856
13857 <div id="CanvasTool_XArc1" class="CanvasTool">
13858 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13859 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()"/>
13860 #<span data-name="ovid" class="CanvasParam" type="text">XA-0</span>
13861 <input data-name="F" type="checkbox" value="Fill" checked="">
13862 <input data-name="X" class="CanvasParam" type="text" value="460" onwheel="OnWheelIntRedraw()"/>
13863 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()"/>
13864 <input data-name="Z" class="CanvasParam" type="text" value="6" onwheel="OnWheelIntRedraw()"/>
13865 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13866 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13867 <input data-name="R0" class="CanvasParam" type="text" value="150" onwheel="OnWheelIntRedraw()"/>
13868 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()"/>
13869 <input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()"/>
13870 #CO-<input data-name="C" class="CanvasParam" type="text" value="4">
13871 </div>
13872
13873 <div id="CanvasTool_XArc2" class="CanvasTool">
13874 <input class="CV_Button" type="button" value="Clone" onclick="CV_cloneTool()"/>
13875 <input data-name="rdr" class="CV_Button" type="button" value="XArc" onclick="CV_drawXArc()"/>
13876 #<span data-name="ovid" class="CanvasParam" type="text">XA-0</span>
13877 <input data-name="F" type="checkbox" value="Fill" checked="">
13878 <input data-name="X" class="CanvasParam" type="text" value="580" onwheel="OnWheelIntRedraw()"/>
13879 <input data-name="Y" class="CanvasParam" type="text" value="130" onwheel="OnWheelIntRedraw()"/>
13880 <input data-name="Z" class="CanvasParam" type="text" value="8" onwheel="OnWheelIntRedraw()"/>
13881 <input data-name="RX" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13882 <input data-name="RY" class="CanvasParam" type="text" value="45" onwheel="OnWheelIntRedraw()"/>
13883 <input data-name="R0" class="CanvasParam" type="text" value="0" onwheel="OnWheelIntRedraw()"/>
13884 <input data-name="S" class="CanvasParam" type="text" value="210" onwheel="OnWheelIntRedraw()"/>
13885 <input data-name="E" class="CanvasParam" type="text" value="180" onwheel="OnWheelIntRedraw()"/>
13886 #CO-<input data-name="C" class="CanvasParam" type="text" value="2">
13887 </div>
13888
13889 <canvas id="CV_partsCanvas" data-name="canvas" class="CS_Canvas" width="740" height="200"></canvas>
13890 </span>
13891 </script>
13892 function CV_redrawParts(){
13893 // search Z-Index and sort
13894 var parts = CanvasTools.children;
13895 //console.log("parts="+parts.length);
13896 np = [];
13897 for( i = 0; i < parts.length; i++ ){
13898 p = parts[i];
13899 z = childByName(p,'Z');
13900 if( z != null ){
13901 //console.log("P'+p.id+' z="+z+'+z.value);
13902 np.push([z.value,p]);
13903 }
13904 }
13905 np.sort(function(np1,np2){ return np1[0] - np2[0]; });
13906 CV_clearRect();
13907 for( i = 0; i < np.length; i++ ){
13908 p = np[i][1];
13909 redraw = childByName(p,'rdr');
13910 //console.log("Redraw Z="+np[i][0]+' #' +np[i][1].id' redraw="+redraw.onclick);
13911 if( redraw != null ){
13912 redraw.click();
13913 }
13914 }
13915 }
13916
13917 function DrawingCanvas_Setup(){
13918 showColorSample();
13919 CV_redrawParts();
13920 }
13921
13922 function OnWheelZoom(){
13923 val = OnWheelInt();
13924 canvas = CV_partsCanvas;
13925 canvas.style.zoom = val + '%';
13926 CV_redrawParts();
13927 }
13928
13929 function OnWheelResize(){
13930 val = OnWheelInt();
13931 canvas = CV_partsCanvas;
13932 if( !canvas.hasAttribute('data-width') ){
13933 w = canvas.width;
13934 h = canvas.height;
13935 canvas.setAttribute('data-width',w);
13936 canvas.setAttribute('data-height',h);
13937 sw = canvas.getAttribute('data-width');
13938 sh = canvas.getAttribute('data-height');
13939 console.log("Zoom save original w="+w+',h="+h' sw="+sw+', sh="+sh);
13940 }
13941 w = canvas.getAttribute('data-width');
13942 h = canvas.getAttribute('data-height');
13943 console.log("Zoom got original size w="+w+',h="+h);
13944 nw = w * (val/100.0);
13945 nh = h * (val/100.0);
13946 //console.log("Zoom mw="+nw+', nh="+nh);
13947
13948 CV_partsCanvas.width = nw;
13949 CV_partsCanvas.height = nh;
13950 CV_redrawParts();
13951 }
13952
13953 function OnWheelIntRedraw(){
13954 OnWheelInt();
13955
13956 t = event.target;
13957 n = t.nodeName;
13958 i = t.id;
13959 p = t.parentNode;
13960 y = event.delta.toFixed(0);
13961 //console.log("OnWheelIntRedraw '+y+' '+n+'#'+i+' '+t.value+' #' +p.id);
13962
13963 if( true ){
13964 CV_redrawParts();
13965 }else{
13966 if( RedrawImmediate.checked ){
13967 CV_clearRect();
13968 //If( p.id == 'CanvasTool_Circle' ){ CV_drawCircle(CanvasTool_Circle); }
13969 //If( p.id == 'CanvasTool_Rect' ){ CV_drawRect(CanvasTool_Rect); }
13970 CV_drawCircle(CanvasTool_Circle);
13971 CV_drawRect(CanvasTool_Rect);
13972 }
13973 }
13974
13975 function CV_setCtxStyle(ctx,p){
13976 C = childByName(p,'C').value;
13977 c = document.getElementById('CanvasTool_Color_'+C);
13978 ctx.fillStyle = CV_GenColorStyle(c,true);
13979 ctx.strokeStyle = CV_GenColorStyle(c,false);
13980 return ctx;
13981 }
13982
13983 function CV_clearRect(){
13984 cv = document.getElementById('CV_partsCanvas');
13985 ctx = cv.getContext('2d');
13986 ctx.clearRect(0,0,cv.width,cv.height);

```

```

13983}
13984function CV_drawRect1(rect){
13985    canvas = document.getElementById('CV_partsCanvas');
13986    ctx = canvas.getContext('2d');
13987    ctx = CV_setCtxStyle(ctx,p);
13988
13989    p = rect;
13990    F = childByName(p,'F').checked;
13991    X = childByName(p,'X').value;
13992    Y = childByName(p,'Y').value;
13993    Z = childByName(p,'Z').value;
13994    p.style.zIndex = Z;
13995    W = childByName(p,'W').value;
13996    H = childByName(p,'H').value;
13997
13998    if( F ){
13999        ctx.fillRect(X,Y,W,H);
14000    }else{
14001        ctx.strokeRect(X,Y,W,H);
14002    }
14003    saveDrawing();
14004}
14005function CV_drawRect(rect){
14006    CV_drawRect1(CanvasTool_Rect);
14007}
14008function CV_drawCircle1(circle){
14009    canvas = document.getElementById('CV_partsCanvas');
14010    ctx = canvas.getContext('2d');
14011    ctx = CV_setCtxStyle(ctx,p);
14012
14013    p = circle;
14014    F = childByName(p,'F').checked;
14015    X = childByName(p,'X').value;
14016    Y = childByName(p,'Y').value;
14017    Z = childByName(p,'Z').value;
14018    p.style.zIndex = Z;
14019    R = childByName(p,'R').value;
14020    S = childByName(p,'S').value;
14021    E = childByName(p,'E').value;
14022
14023    //console.log('Circle'+X+', '+Y+' F='+F);
14024    ctx.beginPath();
14025    SA = (S / 180) * Math.PI;
14026    EA = (E / 180) * Math.PI;
14027    ctx.arc(X,Y,R,SA,EA);
14028    if( F ){
14029        ctx.fill();
14030    }else{
14031        ctx.stroke();
14032    }
14033    saveDrawing();
14034}
14035function CV_drawCircle(){
14036    CV_drawCircle1(CanvasTool_Circle);
14037}
14038function CV_drawEllipse1(circle){
14039    canvas = document.getElementById('CV_partsCanvas');
14040    ctx = canvas.getContext('2d');
14041    ctx = CV_setCtxStyle(ctx,p);
14042
14043    p = circle;
14044    X = childByName(p,'X').value;
14045    Y = childByName(p,'Y').value;
14046    Z = childByName(p,'Z').value;
14047    RX = childByName(p,'RX').value;
14048    RY = childByName(p,'RY').value;
14049    p.style.zIndex = Z;
14050    RO = childByName(p,'RO').value;
14051    S = childByName(p,'S').value;
14052    E = childByName(p,'E').value;
14053
14054    ctx.beginPath();
14055    SA = (S / 180) * Math.PI;
14056    EA = (E / 180) * Math.PI;
14057    ROA = (RO / 180) * Math.PI;
14058    ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14059
14060    F = childByName(p,'F').checked;
14061
14062    if( F ){
14063        ctx.fill();
14064    }
14065    // if( S ){
14066    {
14067        ctx.stroke();
14068    }
14069    saveDrawing();
14070}
14071function CV_drawEllipse(){
14072    CV_drawEllipse1(CanvasTool_Ellipse);
14073}
14074function CV_drawBaloon1(baloon){
14075    canvas = document.getElementById('CV_partsCanvas');
14076    ctx = canvas.getContext('2d');
14077    ctx = CV_setCtxStyle(ctx,p);
14078
14079    p = baloon;
14080    F = childByName(p,'F').checked;
14081    X = childByName(p,'X').value;
14082    Y = childByName(p,'Y').value;
14083    Z = childByName(p,'Z').value;
14084    RX = childByName(p,'RX').value;
14085    RY = childByName(p,'RY').value;
14086    p.style.zIndex = Z;
14087    RO = childByName(p,'RO').value;
14088    S = childByName(p,'S').value;
14089    E = childByName(p,'E').value;
14090
14091    //console.log('Ellipse'+X+', '+Y+' F='+F);
14092    ctx.beginPath();
14093
14094    SA = (S / 180) * Math.PI;
14095    EA = (E / 180) * Math.PI;
14096    ROA = (RO / 180) * Math.PI;
14097    ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14098
14099    PX = parseInt(X);
14100    PY = parseInt(Y);
14101    //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14102    PX -= 25;
14103    PY += 40;
14104    //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14105    ctx.lineTo(PX,PY);
14106    ctx.closePath();
14107
14108    if( F ){
14109        ctx.fill();
14110    }else{
14111        ctx.stroke();
14112    }
14113    saveDrawing();
14114}
14115function CV_drawBaloon(){
14116    CV_drawBaloon1(CanvasTool_Baloon);
14117}
14118function CV_drawPackman1(circle){
14119    canvas = document.getElementById('CV_partsCanvas');
14120    ctx = canvas.getContext('2d');
14121    ctx = CV_setCtxStyle(ctx,p);
14122
14123    p = circle;
14124    F = childByName(p,'F').checked;
14125    X = childByName(p,'X').value;
14126    Y = childByName(p,'Y').value;
14127    Z = childByName(p,'Z').value;
14128    p.style.zIndex = Z;
14129    R = childByName(p,'R').value;
14130    S = childByName(p,'S').value;
14131    E = childByName(p,'E').value;
14132
14133    //console.log('Packman'+X+', '+Y+' F='+F);
14134    ctx.beginPath();
14135    SA = (S / 180) * Math.PI;
14136    //SA += 0.15 * Math.PI;
14137    EA = SA + Math.PI; //(E / 180) * Math.PI;
14138    ctx.arc(X,Y,R,SA,EA);
14139    if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14140
14141    ctx.beginPath();
14142    E = 180 - E;
14143    SA += (E/180) * Math.PI;
14144    EA += (E/180) * Math.PI;
14145    ctx.arc(X,Y,R,SA,EA);
14146    if( F ){ ctx.fill(); }else{ ctx.stroke(); }
14147
14148    saveDrawing();
14149}
14150function CV_drawPackman(){
14151    CV_drawPackman1(CanvasTool_Packman);
14152}
14153function CV_drawPiechart1(baloon){
14154    canvas = document.getElementById('CV_partsCanvas');
14155    ctx = canvas.getContext('2d');
14156    ctx = CV_setCtxStyle(ctx,p);
14157
14158    p = baloon;
14159    F = childByName(p,'F').checked;

```



```

14160 X = childByName(p, 'X').value;
14161 Y = childByName(p, 'Y').value;
14162 Z = childByName(p, 'Z').value;
14163 RX = childByName(p, 'RX').value;
14164 RY = childByName(p, 'RY').value;
14165 p.style.zIndex = Z;
14166 RO = childByName(p, 'RO').value;
14167 S = childByName(p, 'S').value;
14168 E = childByName(p, 'E').value;
14169
14170 //console.log('Ellipse'+X+', '+Y+' F='+F);
14171 ctx.beginPath();
14172
14173 SA = (S / 180) * Math.PI;
14174 EA = (E / 180) * Math.PI;
14175 ROA = (RO / 180) * Math.PI;
14176 ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14177
14178 PX = parseInt(X);
14179 PY = parseInt(Y);
14180 //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14181 //PX -= 25;
14182 //PY += 40;
14183 //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14184 ctx.lineTo(PX,PY);
14185 ctx.closePath();
14186
14187 if( F ){
14188 ctx.fill();
14189 }else{
14190 ctx.stroke();
14191 }
14192 saveDrawing();
14193 }
14194 function CV_drawPiechart(){
14195 CV_drawPiechart(CanvasTool_piechart);
14196 }
14197
14198 function CV_drawXArc(baloon){
14199 canvas = document.getElementById('CV_partsCanvas');
14200 ctx = canvas.getContext('2d');
14201 ctx = CV_setCtxStyle(ctx,p);
14202
14203 p = baloon;
14204 F = childByName(p, 'F').checked;
14205 X = childByName(p, 'X').value;
14206 Y = childByName(p, 'Y').value;
14207 Z = childByName(p, 'Z').value;
14208 RX = childByName(p, 'RX').value;
14209 RY = childByName(p, 'RY').value;
14210 p.style.zIndex = Z;
14211 RO = childByName(p, 'RO').value;
14212 S = childByName(p, 'S').value;
14213 E = childByName(p, 'E').value;
14214
14215 //console.log('Ellipse'+X+', '+Y+' F='+F);
14216 ctx.beginPath();
14217
14218 if( true ){
14219 d = new Date();
14220 ms = d.getTime();
14221 id = (ms % 300) / 10;
14222 //S = parseFloat(E) + id/2;
14223 E = parseFloat(E) + id;
14224 xd = (ms % 10000) / 5;
14225 if( 1000 < xd ){
14226 xd = 2000 - xd;
14227 }
14228 xd *= 0.8;
14229 X = parseFloat(X) + xd - 600;
14230 //console.log('Ellipse S='+S+', E='+E+' id='+id);
14231
14232 SA = (S / 180) * Math.PI;
14233 EA = (E / 180) * Math.PI;
14234 ROA = (RO / 180) * Math.PI;
14235 ctx.ellipse(X,Y,RX,RY,ROA,SA,EA);
14236 }
14237
14238 PX = parseInt(X);
14239 PY = parseInt(Y);
14240
14241 //console.log('Ellipse A '+PX+', '+PY+' F='+F);
14242 //console.log('Ellipse B '+PX+', '+PY+' F='+F);
14243
14244 ctx.lineTo(PX,PY);
14245 ctx.closePath();
14246
14247 if( F ){
14248 ctx.fill();
14249 }else{
14250 ctx.stroke();
14251 }
14252 saveDrawing();
14253 }
14254 function CV_drawXArc(){
14255 t = event.target;
14256 CV_drawXArc(t.parentNode);
14257 }
14258 var AnimateIvtl = window.setInterval(CV_redrawParts,30);
14259
14260 </script>
14261
14262 <h3>Animation</h3>
14263 <span id="Animation" class="CanvasTool" draggable="true" contenteditable="">
14264 <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14265 <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14266 <input class="CanvasParam" type="text" value="ANIMA-0"/>
14267 <input data-name="r" class="ColorParam" type="text" value="Rotate">
14268 <input data-name="T" class="CanvasParam" type="text" value="30" onwheel="OnWheelInt()">ms
14269 <input data-name="r" class="CanvasParam" type="text" value="10" onwheel="OnWheelInt()">s
14270 </span>
14271
14272 <h3>Canvas</h3>
14273 <span id="AppendToCanvas" class="CanvasTool" draggable="true">
14274 <input class="CV_Button" type="button" value="Append"> the above part
14275 </span>
14276 <span id="CanvasWrapTemplate" class="CanvasWrap" draggable="true">
14277 <input class="CV_Button" type="button" value="Clone" onclick="cloneParent()">
14278 <input class="CV_Button" type="button" value="Clear" onclick="CV_clearRect()">
14279 <input data-name="cvid" class="CanvasParam" type="text">0</span>
14280 <input data-name="width" class="CanvasParam" type="text" onchange="CS_setSize()" value="700">
14281 <input data-name="height" class="CanvasParam" type="text" onchange="CS_setSize()" value="400">
14282 <input data-name="zoom" class="CanvasParam" type="text" onchange="CS_setSize()" value="100" onwheel="OnWheelCanvasZoom()">*8
14283 <input class="CV_Button" type="button" value="Remove" onclick="removeParent()"><br>
14284 <div class="DrawingCanvas" data-name="canvas" class="CS_Canvas" width="700" height="400"></div>
14285 </span>
14286 </script>
14287 function OnWheelCanvasZoom(){
14288 val = OnWheelInt();
14289 DrawingCanvas.style.zoom = val + '%';
14290 //CanvasWrapTemplate.style.width = (700*(val/100.0)+20) + 'px';
14291 CanvasWrapTemplate.style.height = (400*(val/100.0)+30) + 'px';
14292 }
14293 </script>
14294
14295 <h3>CanvasBook</h3>
14296 <div id="CanvasBook"></div>
14297 <br>
14298 <style>
14299 .CanvasBook {
14300 overflow:scroll;
14301 }
14302 .CBPanel {
14303 }
14304 .CanvasTool {
14305 font-size:9pt;
14306 font-family:Courier New;
14307 color:#000 !important;
14308 xborder:1px solid #aaa;
14309 background-color:rgba(127,127,127,0.5);
14310 width:740px;
14311 white-space:nowrap;
14312 xheight:30;
14313 margin:4px;
14314 padding-left:4px;
14315 padding-right:4px;
14316 overflow:auto;
14317 display:inline-block;
14318 resize:both;
14319 vertical-align:top;
14320 zoom:1.0;
14321 }
14322 .CanvasWrap {
14323 font-size:9pt;
14324 font-family:Courier New;
14325 color:#000 !important;
14326 border:1px solid #aaa;
14327 background-color:rgba(200,200,200,0.2);
14328 width:740px;
14329 height:430px;
14330 margin:4px;
14331 padding-left:4px;
14332 padding-right:4px;
14333 overflow:auto;
14334 display:inline-block;
14335 resize:both;
14336 vertical-align:top;

```

```

14337 zoom:1.0;
14338 }
14339 .CS_Panel {
14340 padding:3px;
14341 vertical-align:middle;
14342 }
14343 .CS_Canvas {
14344 border:1px dashed #fcc;
14345 resize:both;
14346 zoom:1.0;
14347 }
14348 </style>
14349 <script>
14350 var CanvasID = 0;
14351 function removeParent(){
14352 e = event.target;
14353 p = e.parentNode;
14354 if( p == CanvasWrapTemplate ){
14355 return;
14356 }
14357 pp = p.parentNode;
14358 alert('removeParent #' + pp.id + '/' + p.id + '/' + e.id);
14359 p.parentNode.removeChild(p);
14360 }
14361 function cloneParent(){
14362 b = event.target;
14363 w = b.parentNode;
14364 CanvasID = 1;
14365 //pp = w.parentNode;
14366 cw = w.cloneNode(true);
14367 cw.id = 'Canvas_' + CanvasID;
14368 childByName(cw, 'cvid').innerHTML = CanvasID;
14369 childByName(cw, 'cvid').value = CanvasID;
14370 CanvasBook.appendChild(cw);
14371 }
14372 function CS_setSize(){
14373 e = event.target;
14374 p = e.parentNode;
14375 console.log('resize ' + e.nodeName + ' ' + p.nodeName);
14376 c = childByName(p, 'canvas');
14377 w = childByName(p, 'width').value;
14378 h = childByName(p, 'height').value;
14379 console.log('resize ' + c.nodeName + ' ' + w + 'x' + h);
14380 c.width = w;
14381 c.height = h;
14382 p.style.width = w + 'px';
14383 p.style.height = h + 'px';
14384 console.log("c=" + c + " w=" + w + " h=" + h + " c.width=" + w + " c.height=" + h);
14385 }
14386 function CS_newFunc(){
14387 cvt = CanvasWrapTemplate;
14388 cw = CanvasWrapTemplate.cloneNode(true); //needs an argument, otherwise 'function not found'
14389 CanvasID = 1;
14390 cw.id = 'Canvas_' + CanvasID;
14391 //childByName(cw, 'cvid').contentEditable = false;
14392 childByName(cw, 'cvid').innerHTML = CanvasID;
14393 childByName(cw, 'cvid').value = CanvasID;
14394 childByName(cw, 'width').value = w = childByName(cvt, 'width').value;
14395 childByName(cw, 'height').value = h = childByName(cvt, 'height').value;
14396 cw.style.width = w + 'px';
14397 //ncv = document.createElement('canvas');
14398 ncv = childByName(cw, 'canvas');
14399 //ncv.setAttribute('class', 'CS_Canvas');
14400 //ncv.setAttribute('data-name', 'canvas');
14401 ncv.width = w;
14402 ncv.height = h;
14403 //cvt.replaceChild(ncv, childByName(cw, 'canvas'));
14404 CanvasBook.appendChild(cw);
14405 }
14406 //CS_new.addEventListener('click', CS_newFunc);
14407 </script>
14408
14409 <input id="CanvasBook_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14410 <input id="CanvasBook_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14411 <input id="CanvasBook_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14412 <span id="CanvasBook_WorkCodeView"></span>
14413 <script id="CanvasBook_WorkScript">
14414 function CanvasBook openWorkCodeView(){
14415 function CanvasBook_showWorkCode(){
14416 showHtmlCode(CanvasBook_WorkCodeView, CascadedCanvasBook_WorkCodeSpan);
14417 }
14418 CanvasBook_WorkCodeViewOpen.addEventListener('click', CanvasBook_showWorkCode);
14419 }
14420 CanvasBook_openWorkCodeView(); // should be invoked by an event
14421 </script>
14422 </details>
14423 <!-- CanvasBook_WorkCodeSpan -->
14424 <!-- /CanvasBook_WorkCodeSpan -->
14425 <!-- Work -->
14426 <!-- /Work -->
14427 <!-- Work { -->
14428 <!-- /Work -->
14429 <!-- /Work { -->
14430 <span id="SVG_WorkCodeSpan" open=""><a href="#SVG">SVG</a></span></a name="SVG">SVG</a>
14431 </span>
14432 <details id="SGV_Section"><summary id="SGV_Summary">SVG Getting Started</summary>
14433 <!-- ----- SVG // 2020-1108 SatoxITS { -->
14434 <h2>Getting Started SVG</h2>
14435 <div>
14436 <svg id="xSVG_01" class="SVG100">
14437 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14438 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14439 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14440 <circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14441 </svg>
14442 <svg id="xSVG_02" class="SVG100" viewBox="0 0 100 100">
14443 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14444 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14445 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14446 <circle cx="50" cy="60" r="20" stroke="black" stroke-width="2" fill="#00000000"></circle>
14447 </svg>
14448 <svg id="xSVG_03" class="SVG100" viewBox="0 0 100 100">
14449 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14450 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14451 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14452 <path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14453 </svg>
14454 <svg id="xSVG_04" class="SVG100" viewBox="0 0 100 100">
14455 <circle cx="50" cy="50" r="40" stroke="white" stroke-width="0" fill="yellow"></circle>
14456 <circle cx="35" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14457 <circle cx="65" cy="35" r="5" stroke="white" stroke-width="0" fill="#000000ff"></circle>
14458 <path fill="orange" d="M 25 50 A 10 10 0 0 0 75 50 Z"></path>
14459 </svg>
14460 <svg id="xSVG_05" class="SVG100" viewBox="0 0 100 100">
14461 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="1" fill="yellow"></circle>
14462 <circle cx="35" cy="35" r="5" stroke="black" stroke-width="1" fill="#00000000"></circle>
14463 <circle cx="65" cy="35" r="5" stroke="black" stroke-width="1" fill="#00000000"></circle>
14464 <path stroke="black" d="M 20 50 A 10 10 0 0 0 80 50 fill="yellow"></path>
14465 </svg>
14466 </div>
14467 </script>
14468 var svg_deg = 0;
14469 function rotateSVG(){
14470 //ms = new Date().getTime();
14471 //deg = (ms - msFixed(0))/10 % 360;
14472 svg_deg += 2;
14473 xSVG_04.style.transform = 'rotate(' + svg_deg + 'deg)';
14474 xSVG_05.style.transform = 'rotate(' + svg_deg + 'deg)';
14475 }
14476 window.setInterval(rotateSVG, 30);
14477 </script>
14478
14479 <input id="SVG_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14480 <input id="SVG_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14481 <input id="SVG_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14482 <span id="SVG_WorkCodeView"></span>
14483 <script id="SVG_WorkScript">
14484 function SVG openWorkCodeView(){
14485 function SVG_showWorkCode(){
14486 showHtmlCode(SVG_WorkCodeView, SVG_WorkCodeSpan);
14487 }
14488 SVG_WorkCodeViewOpen.addEventListener('click', SVG_showWorkCode);
14489 }
14490 SVG_openWorkCodeView(); // should be invoked by an event
14491 </script>
14492 </details>
14493 <!-- SVG_WorkCodeSpan -->
14494 <!-- /CanvasBook_WorkCodeSpan -->
14495 <!-- Work -->
14496 <!-- /Work -->
14497 <!-- Work { -->
14498 <!-- /Work -->
14499 <span id="Xyeyes_WorkCodeSpan">

```

```

14514 /*<details open=""><summary>XYeyes</summary>
14515 <a href="#XYeyes">XYeyes</a></a><a name="XYeyes">XYeyes</a>
14516 </--<!-- XYeyes / 2020-11-3 Sat 07:25 { -->
14517 <h2>XYeyes</h2>
14518
14519 <style>
14520 .SVG100b { width:100px; height:100px; }
14521 </style>
14522 <div id="XYeyes1">
14523 <svg class="SVG100b" viewBox="0 0 100 100">
14524 <circle cx="50" cy="50" r="40" fill="#ff0000"/></circle>
14525 <circle cx="38" cy="42" r="10" stroke="black" stroke-width="1" fill="#000000"/></circle>
14526 <circle cx="62" cy="42" r="10" stroke="black" stroke-width="1" fill="#000000"/></circle>
14527 <path stroke="black" d="M 20 50 A 10 0 0 0 80 50 fill:#ffff00"/></path>
14528 </svg>
14529 <svg class="SVG100b" viewBox="0 0 100 100">
14530 <circle cx="50" cy="50" r="40" fill="#ff0000"/></circle>
14531 <circle cx="50" cy="50" r="20" fill="#ffffff"/></circle>
14532 <circle cx="50" cy="50" r="10" fill="#0000ff"/></circle>
14533 </svg>
14534 <svg class="SVG100b" viewBox="0 0 100 100">
14535 <circle cx="50" cy="50" r="40" fill="#400000"/></circle>
14536 <circle cx="66" cy="50" r="20" fill="#ffffff"/></circle>
14537 <circle cx="76" cy="50" r="10" fill="#0000ff"/></circle>
14538 </svg>
14539 <svg class="SVG100b" viewBox="0 0 100 100">
14540 <circle cx="50" cy="50" r="40" fill="#000000"/></circle>
14541 <circle cx="40" cy="40" r="20" fill="#ffffff"/></circle>
14542 <circle cx="72" cy="34" r="10" fill="#0000ff"/></circle>
14543 </svg>
14544 </div>
14545 <div id="XYeyes_0_0" class="SVG100b" viewBox="0 0 100 100">
14546 <circle cx="50" cy="50" r="40" fill="#ff0000"/></circle>
14547 <circle cx="50" cy="34" r="20" fill="#ffffff"/></circle>
14548 <circle cx="50" cy="24" r="10" fill="#0000ff"/></circle>
14549 </svg>
14550 <div id="XYeyes_0_1" class="SVG100b" viewBox="0 0 100 100">
14551 <circle cx="50" cy="50" r="40" fill="#ff0000"/></circle>
14552 <circle cx="50" cy="34" r="20" fill="#ffffff"/></circle>
14553 <circle cx="50" cy="24" r="10" fill="#0000ff"/></circle>
14554 </svg>
14555 <div class="XYZvalues">
14556 <input id="XYeyes_0_Move" class="XYZcheckbox" type="checkbox" checked="">Move
14557 <input id="XYeyes_0_Cont" class="XYZcheckbox" type="checkbox" checked="">Cont
14558 <input id="XYeyes_0_Degree" class="XYZvalue" type="text" value="0"></input>
14559 <input id="XYeyes_0_dist0" class="XYZvalue" type="text" value="0"></input>
14560 <input id="XYeyes_0_dist1" class="XYZvalue" type="text" value="0"></input>
14561 </div>
14562 </div>
14563 </div>
14564 <script>
14565 function XYeyes_Setup0(){
14566   function lookat(eye,pos,x,y){
14567     x0 = eye.getBoundingBoxRect().left.toFixed(0);
14568     y0 = eye.getBoundingBoxRect().top.toFixed(0);
14569     pos.value = x0 + ', ' + y0;
14570   }
14571   var rad = 0;
14572   function rotate(){
14573     if ( !XYeyes_0_Move.checked ){
14574       return;
14575     }
14576     if ( XYeyes_0_Cont.checked ){
14577       rad += 2;
14578     }else{
14579       rad += 12;
14580     }
14581     XYeyes_0_Degree.value = rad % 360;
14582     XYeyes_0_0.style.transform = 'rotate('+rad+'deg)';
14583     XYeyes_0_1.style.transform = 'rotate('+rad+'deg)';
14584     lookat(XYeyes_0_0,XYeyes_0_dist0,x,y);
14585     lookat(XYeyes_0_1,XYeyes_0_dist1,x,y);
14586   }
14587   window.setInterval(rotate,100);
14588 }
14589 XYeyes_Setup0();
14590 </script>
14591 </div>
14592 <div id="XYeyes_1" class="XYeyes">
14593 <h3>XY eyes</h3>
14594 <svg id="XYeyes_1_x0" class="SVG100e" viewBox="0 0 100 100">
14595 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="6" fill="#ffffff00"/></circle>
14596 <circle id="XYeyes_1_x0p" cx="50" cy="30" r="20" fill="#000000ff"/></circle>
14597 </svg>
14598 <svg id="XYeyes_1_x1" class="SVG100f" viewBox="0 0 100 100">
14599 <circle cx="50" cy="50" r="40" stroke="black" stroke-width="6" fill="#ffffff00"/></circle>
14600 <circle id="XYeyes_1_x1p" cx="50" cy="30" r="20" fill="#000000ff"/></circle>
14601 </svg>
14602 <svg id="XYeyes_1_0" class="SVG100c" viewBox="0 0 100 100">
14603 <circle cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
14604 <circle cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
14605 <circle cx="50" cy="24" r="10" fill="#0000ffff"/></circle>
14606 </svg>
14607 <div id="XYeyes_1_1" class="SVG100c" viewBox="0 0 100 100">
14608 <circle data-name="ball" cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
14609 <circle id="XYeyes_1_1_wheye" data-name="wheye" cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
14610 <circle id="XYeyes_1_1_bleye" data-name="bleye" cx="50" cy="24" r="10" fill="#0000ffff"/></circle>
14611 </div>
14612 <div id="XYeyes_1_2" class="SVG100c" viewBox="0 0 100 100">
14613 <circle data-name="ball" cx="50" cy="50" r="40" fill="#ff0000ff"/></circle>
14614 <circle id="XYeyes_1_2_wheye" data-name="wheye" cx="50" cy="34" r="20" fill="#ffffffff"/></circle>
14615 <circle id="XYeyes_1_2_bleye" data-name="bleye" cx="50" cy="24" r="10" fill="#0000ffff"/></circle>
14616 </div>
14617 <div class="XYZvalues">
14618 <input id="XYeyes_1_Move" class="XYZcheckbox" type="checkbox" checked="">Move
14619 <input id="XYeyes_1_Cont" class="XYZcheckbox" type="checkbox" checked="">Cont
14620 <input id="XYeyes_1_MousePos" class="XYZvalue" type="text" value="0"></input>
14621 <input id="XYeyes_1_dist0" class="XYZvalue" type="text" value="0"></input>
14622 <input id="XYeyes_1_dist1" class="XYZvalue" type="text" value="0"></input>
14623 <input id="XYeyes_1_dist0" class="XYZvalue" type="text" value="0"></input>
14624 <input id="XYeyes_1_dist1" class="XYZvalue" type="text" value="0"></input>
14625 <input id="XYeyes_1_dist2" class="XYZvalue" type="text" value="0"></input>
14626 </div>
14627 </div>
14628 </div>
14629 <style>
14630 .SVG100c { width:100px; height:100px; display:inline; }
14631 .SVG100e { width:100px; height:100px; display:inline; }
14632 .SVG100f { width:100px; height:100px; position:relative; left:-20px; display:inline; }
14633 .XYeyes {
14634   text-align:left;
14635 }
14636 .XYZcheckbox {
14637   width:20px;
14638   height:20px;
14639 }
14640 .XYZvalues {
14641   z-index:2;
14642   display:block;
14643   background-color:#eee;
14644   height:100px;
14645   text-align:left;
14646 }
14647 .XYZvalue {
14648   font-size:10pt !important;
14649   width:70pt !important;
14650   line-height:1.1 !important;
14651   margin:1px !important;
14652   display:inline !important;
14653   text-align:right;
14654   padding:1px !important;
14655 }
14656 </style>
14657 <script>
14658 function XYeyes_Setup1(){
14659   function lookat(eye,pos,x,y){
14660     x0 = eye.getBoundingBoxRect().left.toFixed(0);
14661     y0 = eye.getBoundingBoxRect().top.toFixed(0);
14662     dx = x - x0;
14663     dx *= 65;
14664     dy = y - y0;
14665     dy *= 75;
14666     degx = Math.acos(dx/Math.sqrt(dx*dx+dy*dy));
14667     degx *= 180 / Math.PI;
14668     degx = degx.toFixed(0);
14669     degx = parseInt(degx);
14670     if ( dx <= 0 && dy <= 0 ){
14671       deg1 = -45;
14672       degx = -(degx + 90);
14673     }else
14674     if ( dx <= 0 && 0 < dy ){
14675       deg1 = -135;
14676       degx = degx + 90;
14677     }else
14678     if ( 0 < dx && dy <= 0 ){
14679       deg1 = 45;
14680       degx = 90 - degx;
14681     }else
14682     if ( 0 < dx && 0 < dy ){
14683       deg1 = 135;
14684       degx = degx + 90;
14685     }
14686     if ( XYeyes_1_Cont.checked ){
14687

```

```

14691     deg1 = degx;
14692     }
14693     eye.style.transform = 'rotate('+deg1+'deg)';
14694     //pos.value = dx+', '+dy+' / '+deg1+' '+degx;
14695     pos.value = dx+', '+dy+' / '+deg1;
14696 }
14697 function motion(){
14698     if ( !XYeyes_1.Move.checked ){
14699         return;
14700     }
14701     x = event.x;
14702     y = event.y;
14703     XYeyes_1.MousePos.value = x + ' ' + y;
14704     lookat(XYeyes_1_0,XYeyes_1_dist0,x,y);
14705     lookat(XYeyes_1_1,XYeyes_1_dist1,x,y);
14706     lookat(XYeyes_1_2,XYeyes_1_dist2,x,y);
14707     lookat(XYeyes_1_0,XYeyes_1_dist0,x,y);
14708     lookat(XYeyes_1_1,XYeyes_1_dist1,x,y);
14709     }
14710     window.addEventListener('mousemove',motion);
14711     fi = document.getElementById('FeaturesView');
14712     if( fi != null ){
14713         fi.appendChild(XYeyes_1);
14714     }
14715 }
14716 XYeyes_Setup1();
14717 </script>
14718
14719
14720 <input id="XYeyes_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
14721 <input id="XYeyes_WorkOpensnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
14722 <input id="XYeyes_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
14723 <span id="XYeyes_WorkCodeView"></span>
14724 <script id="XYeyes_WorkScript">
14725     function XYeyes_openWorkCodeView(){
14726         function XYeyes_showWorkCode(){
14727             showHtmlCode(XYeyes_WorkCodeView,XYeyes_WorkCodeSpan);
14728         }
14729         XYeyes_WorkCodeViewOpen.addEventListener('click',XYeyes_showWorkCode);
14730     }
14731     XYeyes_openWorkCodeView(); // should be invoked by an event
14732 </script>
14733 </details>
14734 <!-- XYeyes_WorkCodeSpan -->
14735 </span>
14736 </!-- ===== Work } ===== -->
14737
14738
14739
14740
14741
14742
14743 </!-- ===== Work { ===== -->
14744 </span id="X3DOM_WorkCodeSpan"><a href="#X3DOM">X3DOM</a></a name="X3DOM">X3DOM</a>
14745 *
14746 <details id="X3DOM Section" open=""><summary id="X3DOM Summary">X3DOM Getting Started</summary>
14747 <!-- X3DOM // 2020-1111 SatoxITS -->
14748 <h2>X3D-ROFL</h2>
14749
14750 <!-- x3dom JavaScript and CSS BEGIN -->
14751 <h3><a href="/gshell/x3dom-1.8.2-dev/">x3dom-1.8.2-dev</a></h3>
14752
14753
14754 <div>
14755 Viewpoint
14756 position=<input id="X3dRofl_ViewPosition_Value" class="X3DOM_Param" type="text" value="0 0 0">
14757 orientation=<input id="X3dRofl_ViewOrientation_Value" class="X3DOM_Param" type="text" value="0 0 0">
14758 </div>
14759 <div>
14760 Box
14761 position=<input id="X3dRofl_BoxPosition" class="X3DOM_Param" type="text" value="0 0 0 0">
14762 rotation=<input id="X3dRofl_BoxRotation_Value" class="X3DOM_Param" type="text" value="0 0 0 0">
14763 </div>
14764
14765 <span id="x3domInclude">
14766 <script type="text/javascript" src="/gshell/x3dom-1.8.2-dev/x3dom.js"></script>
14767 <link rel="stylesheet" type="text/css" href="/gshell/x3dom-1.8.2-dev/x3dom.css">
14768 </span>
14769 <!-- x3dom JavaScript and CSS END -->
14770
14771 <div class="x3dRofl">
14772 <x3d id="Box1View" class="x3dRoflScene" width="600px" height="300px" draggable="false">
14773 <scene id="Box1Scene">
14774 <viewpoint id="X3dRofl_ViewPoint" position="0.8 0.7 4"></viewpoint>
14775 <transform id="x3d" translation="1.0 0.5 0.5" rotation="1 1 1 1">
14776 <shape id="xBox1Box">
14777 <appearance>
14778 <!--
14779 <script>putTextureTag("Box1Texture");</script>
14780 <imagetexture id="Box1Texture" url="GShellInside00.png"></imagetexture>
14781 -->
14782 <imagetexture id="Box1Texture" url="Wd-WallPaper03.png"></imagetexture>
14783 <material id="Box1Material" diffusecolor="0 1 0"></material>
14784 </appearance>
14785 <box id="xBox1Box" size="1 1 2"></box>
14786 </shape>
14787 </transform>
14788 </scene>
14789 </x3d>
14790 </div>
14791 <style>
14792 .x3dRofl {
14793     background-color:#e0f0e0;
14794     position:relative;
14795     display:block;
14796     overflow:visible !important;
14797     resize:both !important;
14798 }
14799 .x3dRoflScene {
14800     z-index:100;
14801     background-color:#e0f0f8;
14802     position:relative;
14803     display:block;
14804     overflow:visible !important;
14805     resize:both !important;
14806     zoom:2.0;
14807     xtransform:scale(2.0);
14808 }
14809 .X3DOM_Param {
14810     color:#000 !important;
14811     font-family:Courier New, Monospace !important;
14812     font-size:9pt !important;
14813     padding:2px !important;
14814     line-height:1.1 !important;
14815     height:14pt !important;
14816     width:120pt !important;
14817     text-align:left !important;
14818     display:inline !important;
14819     vertical-align:middle !important;
14820 }
14821 </style>
14822
14823 <script>
14824 </transform translation="0 0 0">
14825 </transform id="Box1Box" translation="0 0 0">
14826 </transform>
14827 function showVP(){
14828     console.log('color0:'+Box1Material.getAttribute('diffusecolor'));
14829     m = document.getElementById('Box1Material');
14830     //console.log('color1:'+Box1Material.getFieldValue('diffusecolor'));
14831     console.log('color1:'+m.getFieldValue('diffusecolor'));
14832     //console.log('rotation:'+Box1Box.getFieldValue('rotation'));
14833     console.log('Box1TransRotation'+Box1Trans.getFieldValue('rotation'));
14834
14835     e = document.getElementById('Box1View');
14836     console.log('Box1View'+e.runtime.properties());
14837     e = document.getElementById('X3dRofl_ViewPoint');
14838     console.log('Box1Box'+e.outerHTML);
14839
14840     //e.runtime.showAll();
14841     //e.runtime.resetView();
14842
14843     //console.log('Transform'+x3dom.runtime.getCurrentTransform(Box1View));
14844     //console.log('Transform'+Box1View.runtime.getCurrentTransform());
14845     //console.log('Transform'+Box1Scene.runtime.getCurrentTransform());
14846     //console.log('WPosition'+X3dRofl_ViewPoint.position);
14847     console.log('WPosition'+X3dRofl_ViewPoint.getFieldValue('position'));
14848     console.log('WOrientation'+X3dRofl_ViewPoint.getFieldValue('orientation'));
14849     //console.log('WPositionProp'+Box1View.runtime.properties());
14850     console.log('WPositionProp'+X3dRofl_ViewPoint.runtime.properties());
14851     //sid = document.getElementById('Box1Scene');
14852     //console.log('Box1Scene'+sid.runtime.properties());
14853     //X3dRofl_ViewPoint.value = X3dRofl_ViewPoint.position;
14854
14855     X3dRofl_ViewPosition_Value.value = X3dRofl_ViewPoint.getFieldValue('position');
14856     X3dRofl_ViewOrientation_Value.value = X3dRofl_ViewPoint.getFieldValue('orientation');
14857     X3dRofl_BoxPosition_Value.value = Box1Trans.getFieldValue('translation');
14858     X3dRofl_BoxRotation_Value.value = Box1Trans.getFieldValue('rotation');
14859
14860     box1 = document.getElementById('Box1');
14861     //X3dRofl_ViewOrientation_Value.value = box1.runtime.viewpoint();
14862 }
14863 window.setInterval(showVP,500);
14864 function newVP(){
14865     console.log('ViewPoint changed'+event);
14866 }
14867 vp = document.getElementById('X3dRofl_ViewPoint');

```



```
15045//<!-- ===== Work { ===== -->
15046//<span id="OriginalSource_WorkCodeSpan">
15047/*
15048<details open=""><summary>Original Source</summary>
15049</>----- OriginalSource // 2020-1009 SatoxITS { -->
15050<input id="OriginalSource_WorkCodeViewOpen" class="HtmlCodeViewButton" type="button" value="ShowCode">
15051<input id="OriginalSource_WorkOpenSnapshot" class="HtmlCodeViewButton" type="button" value="Snapshot">
15052<input id="OriginalSource_WorkCodeSignature" class="HtmlCodeViewButton" type="button" value="Signature">
15053<span id="OriginalSource_TextElement"></span>
15054<span id="OriginalSource_WorkCodeView"></span>
15055<script id="OriginalSource_WorkScript">
15056function OriginalSource_openWorkCodeView(){
15057    function OriginalSource_showWorkCode(){
15058        //showHtmlCode(OriginalSource_WorkCodeView,OriginalSource_WorkCodeSpan);
15059        //OriginalSource_TextElement = OriginalSourceNode;
15060        //console.log('src3=\n'+OriginalSourceNode.outerHTML);
15061        showHtmlCodeX(OriginalSource_WorkCodeView,OriginalSourceNode,
15062            '\n\n',
15063            '\n',true);
15064    }
15065    OriginalSource_WorkCodeViewOpen.addEventListener('click',OriginalSource_showWorkCode);
15066}
15067//OriginalSourceNode = document.documentElement.cloneNode();
15068//OriginalSourceNode = gsh.cloneNode(true); //=====
15069//console.log('src0=\n'+document.documentElement.outerHTML);
15070//console.log('src1=\n'+gsh.outerHTML);
15071//console.log('src2=\n'+OriginalSourceNode.innerHTML);
15072OriginalSource_openWorkCodeView(); // should be invoked by an event
15073//showNodeAsHtmlSource(OriginalSource_WorkCodeView,OriginalSourceNode);
15074function SaveOriginalNode(){
15075    if( false ){
15076        m0 = performance.memory;
15077        mu0 = m0.usedJSHeapSize;
15078        console.log('-- heap bef clone: '
15079            +m0.usedJSHeapSize+'/'+m0.totalHeapSize+'/'+m0.jsHeapSizeLimit);
15080    }
15081    OriginalSourceNode = gsh.cloneNode(true);
15082    if( false ){
15083        m1 = performance.memory;
15084        mu1 = m1.usedJSHeapSize;
15085        mu = mu1 - mu0;
15086        //alert('-- clone: used heap '+mu0+' -> '+mu1+' = '+mu+' bytes');
15087        console.log('-- heap aft clone: '
15088            +m1.usedJSHeapSize+'/'+m1.totalHeapSize+'/'+m1.jsHeapSizeLimit);
15089        //OriginalSourceNode = document.documentElement.cloneNode(true);
15090    }
15091}
15092}
15093}
15094function Gsh_setupPage(){
15095    GshSetImages();
15096    //Indexer_afterLoaded();
15097    //GShell_initKeyCommands();
15098    //GJConsole_initConsole();
15099    GJConsole_initFactory();
15100    GJLink_init();
15101    InterFrameComm_init();
15102    Gshell_initTopbar();
15103    //VirtualDesktop_init();
15104    Banner_init();
15105    // Aff_Setup();
15106    Shading_Setup();
15107    window.setInterval(ShowResourceUsage,1000);
15108    //document.addEventListener('keydown',jgsbCommand); // should be applied later?
15109    FontLillism_Setup();
15110    FontList_Setup();
15111    showFooter();
15112    GshInsideIconSetup();
15113    SightClass_Setup();
15114    //spawnPackmonGo();
15115    //PackmonGo_Setup(null);
15116    DrawingCanvas_Setup();
15117    InstaColor_Setup();
15118}
15119}
15120function OnLoad(){
15121    SaveOriginalNode();
15122    Gsh_setupPage();
15123}
15124document.addEventListener('load',Gsh_setupPage);
15125</script>
15126</details>
15127</>----- OriginalSource_WorkCodeSpan } -->
15128*//</span>
15129//<!-- ===== Work } ===== -->
15130}
15131}
15132</div>
15133//<script><script>OnLoad();</script></span>
15134
```