

```

1      ;=====
2      ;weather_main.s
3      ;This version of the program requires
4      ;loading the Flyer routines as a module.
5      ;It must be called from weather.s,
6      ;which loads the Flyer module first.
7      ;=====
8      0  program weather_main
9
10     0  include library
11     7  LIST 0
12
13     7  LIST
14
15     7  list 0
16
17     7  list
18
19     7  include flyer.e
20
21     7  IMPORT FLYER ; 8/22/14
22
23     28E  EXT FUNC BYTE transflyer AT $028E
24
25     1AD  EXT FUNC BYTE readflyer AT $01AD
26
27     102  EXT FUNC BYTE sendflyer AT $0102
28
29     9B   EXT FUNC BYTE openflyer AT $009B
30
31     21   EXT FUNC WORD getstat AT $0021
32
33     9    EXT PROC closeflyer AT $0009
34
35     8    EXT DATA BYTE cmdchan AT $0008
36
37     7    EXT DATA BYTE datachan AT $0007
38
39
40     8  con maxdays=8 ;expected amt of data
41
42
43     ;initialized variables
44     7  data word url="HTTP:www.lyonlabs.org/fw/fc"
45     9  data word stateurl="HTTP-POST:9,state="
46     B  data word cityurl="HTTP-POST:9,city="
47
48
49     ;uninitialized variables
50     0  byte foreground
51     1  byte background
52     2  byte border
53     3  byte running
54     4  byte result
55     5  byte key
56     0  byte state[3]
57     3  byte city[31]
58     22 byte stateparm[80]
59     72 byte cityparm[80]
60     C2 byte raw[8192]
61     20C2 byte days[20,maxdays]
62     2162 byte forecasts[160,8]
63     2662 byte fword[20]
64     2676 byte errmsg[40]
65     6  word day
66     8  word w
67
68
69     ;-----
70

```

```
256 4E proc showerr
257 1 arg byte err
258 4E begin
259 51 closeflyer
260 54 abort "#cerror: $#h",err
261 5E end
262
263 ;-----
264
265 6F func byte readkey
266 6F begin
267 72 m[cndx]=0 ;clear kbd buffer
268 79 repeat
269 79 nothing ;let irq do it
270 79 until m[cndx] <> 0
271 84 m[cndx]=0 ;in case of key repeat
272 8B return m[$277] ;first key in queue
273 92 end
274
275 ;-----
276
277 96 proc hitkey
278 93 begin
279 96 curset 28,0
280 9F put "hit a key...",cr
281 A9 key=readkey
282 AE end
283
284 ;-----
285
286 BF func byte parse
287 1 word i
288 3 word j
289 5 byte result
290 269E own byte ack[4]
291 C2 begin
292 C2 for i=0 to 2
293 CB ack[i]=raw[i]
294 D6 ack[3]=0
295 DD if cmpstr(ack,"=", "ACK") and (raw[3]=$0a or raw[3]=0)
296 100 movstr "No data received.",errmsg
297 10A result=false
298 110 else if cmpstr(ack,"=", "ACK")
299 ;set up forecasts
300 120 day=0
301 124 i=3
302 129 repeat
303 129 j=0
304 12D repeat
305 12D days[j,day]=raw[i]
306 13D i=i+1
307 144 j=j+1
308 14B until raw[i]=$7c ;pipe
309 156 days[j,day]=0
310 162 i=i+1 ;past pipe
```

```
311 169      j=0
312 16D      repeat
313 16D          forecasts[j,day]=raw[i]
314 17D          i=i+1
315 184          j=j+1
316 18B      until raw[i]=$7c      ;pipe
317 196      forecasts[j,day]=' ' ;for showcast
318 1A3      j=j+1
319 1AA      forecasts[j,day]=0
320 1B6      i=i+2 ;past second pipe
321 1BD      day=day+1
322 1C4      until raw[i]=$0a or raw[i]=0
323 1D7      result=true
324 1DD      else if cmpstr(ack,"=", "NAK")
325          ;set up error message
326 1ED      i=3
327 1F2      j=0
328 1F6      repeat
329 1F6          errmsg[j]=raw[i]
330 200          i=i+1
331 207          j=j+1
332 20E      until raw[i]=$0a or raw[i]=0
333 221      errmsg[j]=0
334 227      result=false
335 22D      else
336 22D          movstr "Invalid response: ",errmsg
337 237          movstr ack,errmsg[18]
338 245          result=false
339 248      return result
340 24B      end
341
342      ;-----
343
344 286      proc showcast
345      1      word i
346      3      word w
347      5      word column
348      7      word row
349 286      begin
350 289      for i=$4c8 to $7bf ;clear forecast
351 294          m[i]=$20
352 29C      curset 0,5
353 2A5      put #days[0,day],cr
354 2B7      i=0
355 2BB      w=0
356 2BF      row=6
357 2C4      column=0
358 2C8      repeat ;word wrap forecast
359 2C8          while forecasts[i,day] <> ' '
360 2D9              fword[w]=forecasts[i,day]
361 2E9              w=w+1
362 2F0              i=i+1
363 2FA      i=i+1 ;past blank
364 301      if column = 40-w ;fits exactly at eol
365 30D          fword[w]=0 ;drop blank
```

```
366 316     else
367 316         fword[w]=' ' ;keep blank
368 31D         w=w+1
369 324         fword[w]=0
370 32A         if column > 40-w ;doesn't fit
371 336             row=row+1 ;new line
372 33D             column=0
373 341             curset column,row
374 349             put fword
375 350             column=column+lenstr(fword)
376 35D             w=0
377 361         until forecasts[i,day] = 0
378 371         end
379
380             ;=====
381
382 375         begin
383             ;save screen colors
384 372         foreground=m[$0286]
385 37A         background=m[$d021]
386 382         border=m[$d020]
387 38A         m[$0286]=12 ;characters grey 2
388 392         m[$d021]=0 ;background black
389 399         m[$d020]=0 ;border black
390
391 3A0         running=true
392 3A3         while running
393 3A8             put $93,$12,"PROMAL Weather",$92,cr,cr
394 3BE             put "City:",cr,"State:"
395 3CB             curset 6,2
396 3D4             getl city,30
397 3DE             if city[0]='x'
398 3E9                 break
399 3EC             curset 7,3
400 3F6             getl state,2
401 3FF             if state[0]='x'
402 40A                 break
403
404 40D             curset 15,24
405 417             put "sending..."
406 41E             movstr stateurl,stateparm
407 428             movstr state,stateparm+lenstr(stateparm)
408 43B             movstr cityurl,cityparm
409 445             movstr city,cityparm+lenstr(cityparm)
410
411 458             result=openflyer(url)
412 460             if result
413 465                 showerr(result)
414 46B             result=sendflyer(stateparm,cmdchan)
415 477             if result
416 47C                 showerr(result)
417 482             result=sendflyer(cityparm,cmdchan)
418 48E             if result
419 493                 showerr(result)
420 499             result=transflyer
```

```
421 49E    if result
422 4A3      showerr(result)
423
424 4A9      curset 15,24
425 4B3      put "reading..."
426 4BA      result=readflyer(raw,datachan)
427 4C6      if result
428 4CB        showerr(result)
429 4D1      closeflyer
430 4D4      curset 15,24
431 4DE      put "parsing..."
432 4E5      if parse
433 4EB        day=0
434 4EF        showcast
435 4F2        curset 5,24
436 4FC        put $12,"f1:new f3:prev f5:next f8:quit",$92
437 509      repeat
438 509        key=readkey
439 50E        choose key
440 510        f3 ;prev day
441 515        if day > 0
442 51D          day=day-1
443 524          showcast
444 52A        f5 ;next day
445 52F        if day < maxdays-1
446 53A          day=day+1
447 541          showcast
448 547        f8
449 54C        running=false
450 552      else
451 553        nothing
452 553      until key=f1 or key=f8
453 564    else
454 564      curset 0,24
455 56D      for w=lenstr(errmsg) to 38
456 57D        errmsg[w]=' '
457 585      errmsg[39]=0
458 58C      put $12,errmsg,$92
459 599      hitkey
460
461          ;restore screen colors
462 59F      m[$0286]=foreground
463 5A7      m[$d020]=background
464 5AF      m[$d021]=border
465 5B7      put $93 ;clear screen
466 61E      end
```

WEATHER_MAI compiled:

466 Source lines

\$061E (1566) bytes Obj.

\$A Scalar, \$26AC (9900) tot. Vars.

Table usage:

Symbols: 25% Fwd. Refs.: 4%

Strings: 2%