```
/*=========================================================================

IWeb Sample --

=========================================================================*/

// 1. include file

#include "AEEWeb.h"                // Web Interface definitions


// 2. reserve space for web object(s), responses and callback functions in
//    applet structure

typedef struct _MyApp
{
      AEEApplet     a;                  // first element must be AEEApplet
      IWeb*         pIWeb;              // pointer to web object
      IWebResp*     pIWebResponse       // pointer to response
      WebRespInfo*  pWebRespInfo;       // pointer to info about web response
      ISource*      pISource;           // pointer to response stream object
      AEECallback   WebCBStruct;        // structure for Iweb callback function
} MyApp;


// 3. In initialization code, create iweb object

if (ISHELL_CreateInstance(pMe->a.m_pIShell, AEECLSID_WEB,
                          (void **)&pMe->pIWeb) != SUCCESS){
            pMe->pIWeb = NULL;
            return FALSE;
            }
            else {

            // 3a. Set up the callback function to receive response from server

            CALLBACK_Init(&pMe->WebCBStruct, IWebCB, pMe);  // out, in, in
            Return TRUE;
            }


// 4.  To issue an HTTP request to a URL, use IWEB_GetResponse macro
//     Note use of inter parentheses and two pIWeb pointers!
//     URL must be fully specified; could be within a C-string variable

IWEB_GetResponse(pMe->pIWeb,
                  (pMe->pIWeb, &pMe->IWebResponse,
                   &pMe->WebCBStruct,
                   “http://www.someplace.com/somepage.html”,
                   WEBOPT_END));

return; // response processed in callback
```

```
// 5.  Response from the webserver is received by the callback function

void IWebCB (MyApp* pMe) {

      // get info about the response
      pMe->pWebRespInfo = IWEBRESP_GetInfo(pMe->pIWebResponse);

      // 5.a  check error code
      if (!WEB_ERROR_SUCCEEDED(pMe->pWebRespInfo->nCode)) {
            DisplayError(…);
            return;
            }

      // 5.b  get pointer to Source object
      pMe->pISource = pMe->pWebRespInfo->pisMessage);
      if (pMe->pISource == NULL) {
            ProcessError(…);
            return;
            }

      // 5.c  register Isource Read callback
      CALLBACK_Init(&pMe->WebCBStruct, ReadFromWebCB, pMe);  // out, in, in

      // 5.d  post a read; data is processed by ISource callback
      ISOURCE_Readable(pMe->pISource, &pMe->WebCBStruct);
      return;
      }


// 6.  Data from webserver is read from ISource stream
//     Data arrives in chunks and is processed chunk at a time.

void ReadFromWebCB(MyApp* pMe) {

      char buf[1024];         // allocate buffer
      int byteCount;

      // read data from stream; get number of bytes read
      byteCount = ISOURCE_Read(pMe->pISource, (char*)buf, sizeof(buf));

      switch (byteCount) {
            case ISOURCE_WAIT:      // Buffer empty, but more data expected
                                    // post another read
                  ISOURCE_Readable(pMe->pISource,&pMe->WebCBStruct);
                  return;

            case ISOURCE_ERROR:     // Error occurred
                  ProcessError(…);
                  return;

            case ISOURCE_END:       // Buffer empty; all data received
                  ProcessData(…);
                  return;

            default:                // data read; copy from chunk buffer
                  MEMCPY(finalDestination, buf, byteCount);
                  finalDestination =+ byteCount;
```

```
                // post another read
                ISOURCE_Readable(pMe->pISource,&pMe->WebCBStruct);
                return;
        }
    }


// 7. NOTE: No Web related events in event handler


// 8. Free object(s) when exiting program; typically in CleanUp()

    // cancel callback
    CALLBACK_Cancel(&pMe->WebCBStruc);

    if (pMe->pIWebResp)
    {
        IWEBRESP_Release(pMe->pIWebResponse);
        pMe->pIWebResp = NULL;
    }

    if (pMe->pIWeb)
    {
        IWEB_Release(pMe->pIWeb);
        pMe->pIWeb = NULL;
    }

// 9. Don't forget to free any dynamically allocated buffers, for example:

    FREE(finalDestination);
```