Brew Job Interview Questions And Answers

Interview Questions Answers

https://interviewquestionsanswers.org/

About Interview Questions Answers

Interview Questions Answers . ORG is an interview preparation guide of thousands of Job Interview Questions And Answers, Job Interviews are always stressful even for job seekers who have gone on countless interviews. The best way to reduce the stress is to be prepared for your job interview. Take the time to review the standard interview questions you will most likely be asked. These interview questions and answers on Brew will help you strengthen your technical skills, prepare for the interviews and quickly revise the concepts.

If you find any **question or answer** is incorrect or incomplete then you can **submit your question or answer** directly with out any registration or login at our website. You just need to visit Brew Interview Questions And Answers to add your answer click on the *Submit Your Answer* links on the website; with each question to post your answer, if you want to ask any question then you will have a link *Submit Your Question*; that's will add your question in Brew category. To ensure quality, each submission is checked by our team, before it becomes live. This Brew Interview preparation PDF was generated at **Wednesday 29th November, 2023**

You can follow us on FaceBook for latest Jobs, Updates and other interviews material.
www.facebook.com/InterviewQuestionsAnswers.Org

Follow us on Twitter for latest Jobs and interview preparation guides.
https://twitter.com/InterviewQA

If you need any further assistance or have queries regarding this document or its material or any of other inquiry, please do not hesitate to contact us.

Best Of Luck.

Interview Questions Answers.ORG Team
https://InterviewQuestionsAnswers.ORG/
Support@InterviewQuestionsAnswers.ORG

Brew Interview Questions And Answers Guide.

Question - 1:

Why does ISOCKET_Release() return one when we are expecting a return value of zero?

Ans:

When an application calls ISOCKET_Release(), the internal state of the ISocket object changes to "closing," and BREW begins waiting for the asynchronous "closed" event. Since the closed event is received in a callback, the reference count of the ISocket object is incremented to prevent it from being released before its internal state changes to closed.

View All Answers

Question - 2:

Why does ISHELL_CreateInstance return ECLASSNOTSUPPORT when I try and create an instance of the net manager (AEECLSID_Net)?

Ans:

This is because of the permissions on the MIF file for your applet. Please open your MIF file in the MIF editor and check the checkbox for Network privileges, then save

View All Answers

Question - 3:

Can we have a listening TCP socket?

Ans:

No. You use UDP, specifically, ISOCKET_Bind and ISOCKET_RecvFrom

View All Answers

Question - 4:

Does BREW support blocking sockets?

Ans:

No. BREW uses asynchronous sockets. You can use ISOCKET_Readable or ISOCKET_Writeable to be notified when it is safe to read/write.

View All Answers

Question - 5:

How to handle the case when we lose cellular coverage?

Ans:

When the phone goes out of a coverage area, the data call will drop. When this happens, BREW cleans up the underlying OEM sockets that are in use. The ISocket(s) used by an app continue to exist, and the app will get appropriate error responses when it tries to use them.
For synchronous (TCP/IP) data communication, there are two ways to handle the potentially bad sockets:
Check the return value from all Socket operations
Writing or reading from the socket will cause AEE_NET_ERROR to be returned. In your code, you should check the return value from the Socket connect, read, write and take appropriate action.

```
 // Connect callback
static void SampleApp_ConnectCB(void *cxt, int err)  {
SampleApp *pMe = (SampleAppApp*)cxt;
if (err) {
// connect failed
SampleApp_CleanUp(pMe);
//Clean up net manager and
// sockets
ShowMainMenu(pMe);
return;
}
}
// Writing
```

```
iRet = ISOCKET_Write(pMe->m_piSock, (byte*)Request,
(uint16)STRLEN(Request));
if (iRet == AEE_NET_ERROR) {
// Write error
SampleApp_CleanUp(pMe);
// Clean up net manager and sockets
ShowMainMenu(pMe);
return;
}
// Reading
iRet = ISOCKET_Read(pMe->m_piSock, (byte*)buf,
sizeof(buf));
if (iRet == AEE_NET_ERROR) {
// Read error
SampleApp_CleanUp(pMe);
// Clean up net manager and sockets
ShowMainMenu(pMe);
return;
}
```

Register for change in network/socket state

Another way to handle the phone going out of coverage while a data call is in progress is to register for Network and Socket events using INETMGR_OnEvent() and take appropriate action when notified of change in network/socket status. In the example below, a flag is set to indicate that the socket is bad. The socket state is checked before it is used.

For more information on Network and Socket events, please refer to NetMgrEvent and NetState enums in aeenet.h include file.

For example:

```
 // Register to receive Network and Socket events
INETMGR_OnEvent(piNet, (PFNNETMGREVENT)CheckNetStatus,
(void*)pMe, TRUE);
//Callback invoked when Network/Socket event is received
static void CheckNetStatus(void* cxt, NetMgrEvent evt,
uint32, dwData) {
SampleApp *pMe = (SampleApp*)cxt;
if(evt == NE_PPP) {
if(dwData == NET_PPP_CLOSING || dwData ==
NET_PPP_CLOSED) {
// flag set to false to indicate socket is bad
pMe->m_SocketState = FALSE;
// clean up network and socket
ReleaseNetAndSocket(pMe);
}
}
if(evt == NE_SO_CLOSING || evt == NE_SO_CLOSED) {
pMe->m_SocketState = FALSE;
ReleaseNetAndSocket(pMe);
}
if(evt == NE_SO_CONNECTED) {
pMe->m_SocketState = TRUE;
}
return;
}
// Check socket state before using it
static void RoadWarriorApp_ReadCB(void *cxt)  {
int rc;
SampleApp *pMe = (SampleApp *)cxt;
if(pMe->m_SocketState == FALSE) {  //socket is bad
ReleaseNetAndSocket(pMe);
ShowMainMenu(pMe);
return;
}
rc = ISOCKET_Read(piSock, (byte*)buf, sizeof(buf));
}
```

When registering for network and socket events using INETMGR_InEvent(), you must de- register before terminating the application by using INETMGR_OnEvent() with bRegister = FALSE.

Note: After the phone goes out of coverage, the sockets are unusable. You must release your ISocket(s) and reinstantiate them when next required. The INetMgr does not need to be released and re-instantiated before using it for the next network operation, however there is no harm in doing so.

To handle potentially bad sockets for asynchronous(UDP) data communication, register for change in network/socket state (described in number 2 above).

View All Answers

Question - 6:

When we terminate a data call by closing an open socket, why does the phone still indicate that a data call is in progress?

Ans:

After the last connected socket is closed, BREW waits for a certain linger time before terminating the PPP connection. Hence the lingering call-in-progress indication. The default linger time is 30 seconds. To change the linger time, use INETMGR_SetLinger().

View All Answers

Question - 7:

Why does connect callback not timeout when service is lost (while attempting to connect) or the server does not respond?

Ans:

This is due to a bug in the BREW version 1.0.1 SDK - connect callback is not invoked under the following circumstances:
 Service is lost while connect is being attempted
 Server does not respond
As a workaround, you should implement a timer in association with the callback. If the connect callback does not occur in 30 seconds, you should timeout the connection.
For example:

```
 // initialize pMe->connectCBInvoked = FALSE;
// Set timer for 30 seconds before invoking ISOCKET_Connect()
ISHELL_SetTimer(pMe->a.m_pIShell, 30000, ConnectTimeout_CB, (void *)pMe);
ISOCKET_Connect(pMe->m_pISocket, nodeINAddr, AEE_htons(USAGE_TEST_PORT),
SampleApp_ConnectCB, pMe);
// Set flag indicating connect CB was called
void SampleApp_ConnectCB(void *cxt, int err)  {
SampleApp *pMe = (SampleApp*)cxt;
// Set flag indicating connect CB was called
pMe->connectCbInvoked = TRUE;
if (err) {
DisplayOutput((IApplet*)pMe, 3, "Connect failed!");
return;
}
DisplayOutput((IApplet*)pMe, 3, "Connected!");
}
// When timer expires, check if connect CB was invoked
static void ConnectTimeout_CB(void* cxt) {
SampleApp *pMe = (SampleApp *)cxt;
if(pMe->connectCbInvoked == FALSE) {
// Callback was not invoked within 30seconds - cancel connect callback
ISOCKET_Cancel(pMe->m_pISocket, 0, 0);
DisplayOutput((IApplet*) pMe, 3, "Connection timed out");
}
else {
// Callback invoked.  Set flag to default value FALSE
pMe->connectCbInvoked = FALSE;
}
}
```

Note: Multiple TCP sockets are not supported on the Kyocera 3035. It allows one TCP socket and one UDP socket at a given time.

View All Answers

Question - 8:

What precautions should we take when I invoke INETMGR_GetHostByName() to perform DNS lookup?

Ans:

You must ensure that your app cancels its pending DNS callback, if any, using CALLBACK_Cancel() in all exit routes. This includes when the app is suspended and when the app is stopped (both via the End key and the Clear key).
For example, if you used the following code to lookup a DNS address:

```
 CALLBACK_Init(&pMe->cbkLookup, GetHostByNameCB, pMe);
INETMGR_GetHostByName(pINetMgr,&pMe->dnsresult, hostName,
&pMe->cbkLookup);
```

You should use the following code to cancel the callback in all exit routes:

```
// Check if the callback can be cancelled. If NULL it has
// already happened and cannot be cancelled.
if(pMe->cbkLookup.pfnCancel != NULL) {
CALLBACK_Cancel(&pMe->cbkLookup);
}
```

The callback cancel code should be incorporated in the Suspend event handler code (EVT_APP_SUSPEND), the app's FreeAppData function (which is called when the app is stopped, i.e. receives EVT_APP_STOP event), and in the Clear key event handler code (EVT_KEY and keycode AVK_CLR).
Note: Please note that your app will not pass TRUE BREW Testing if it does not properly cancel its pending DNS callback. Some symptoms that might indicate that your app is not canceling its p ending DNS callback are:,/p>
 Phone crashes upon starting up app
 Phone displays "Please Re-insert Battery" error upon starting up app
Both might imply that the previously running app did not cancel its pending DNS callback.

View All Answers

Question - 9:

Are there any restrictions on what value we can set my applications network linger time to?

Ans:

The INETMGR_SetLinger() function is provided to give developers flexibility in designing their application. However, it is not recommended that you alter the OEM's chosen default linger time (usually 30 seconds), unless your application has a compelling reason to do so. Altering the default linger time in either direction may cause your application to fail TRUE BREW Testing, or carrier rejection of your application.

View All Answers

Question - 10:

Is there any way to tell if a socket is already connected?

Ans:

There are two ways to determine if a socket is connected. The easiest is to check the return value of ISOCKET_Connect(). When a socket is already connected, ISOCKET_Connect() will return EISCONN.

It is also possible to monitor the state of a socket connection by registering for the socket status change events NE_SO_CLOSING, and NE_SO_CLOSED with INETMGR_OnEvent(). Using this method, your application can avoid trying to connect an already connected socket.
For more information about network and socket events, including NE_SO_CLOSING and NE_SO_CLOSED, please refer to the following include file: AEENet.h.
For Example:

```
 // Register to receive Network and Socket events
INETMGR_OnEvent(piNet, (PFNNETMGREVENT)CheckNetStatus,
(void*)pMe, TRUE);
//Callback invoked when Network/Socket event is received
static void CheckNetStatus(void* cxt, NetMgrEvent evt,
uint32 dwData)
 {
SampleApp *pMe = (SampleApp*)cxt;
if(evt == NE_SO_CLOSING || evt == NE_SO_CLOSED) {
// flag set to false to indicate socket is disconnected
pMe->m_SocketConnected = FALSE;
ReleaseNetAndSocket(pMe);
}
if(evt == NE_SO_CONNECTED) {
// flag set to true to indicate socket is connected
pMe->m_SocketConnected = TRUE;
}
return;
}
```

View All Answers

Question - 11:

Can a BREW-enabled device be used as a server?

Ans:

In addition to the obvious memory and performance limitations, it is not possible to listen on a socket connection when a BREW application is running on a phone. These factors make implementing a server on BREW difficult at best.

View All Answers

Question - 12:

Is it possible to transfer data between two phones?

Ans:

Peer to peer connections between two phones have been found to be unreliable, failing when the phones are on the same subnet. It is best therefore to use a proxy server, transferring data between the phones using the server as a go between.

View All Answers

Question - 13:

How to check the status of my sockets and PPP connection?

Ans:

We can check on the status of the PPP connection (active, inactive, opening or closing) using INETMGR_NetStatus().

```
 NetState netState = NET_INVALID_STATE;
AEENetStats netStats;
netState = INETMGR_NetStatus (pINetMgr, &netStats);
```

We can use INETMGR_OnEvent() to monitor both the status of the PPP connection and socket connections. Refer to NetMgrEvent enum in AEENet.h for a list of all network and socket events.

```
 // Register to receive Network and Socket events
INETMGR_OnEvent(piNet, (PFNNETMGREVENT)CheckNetStatus,
(void*)pMe, TRUE);
//Callback invoked when Network/Socket event is received
static void CheckNetStatus(void* cxt,
NetMgrEvent evt, uint32 dwData) {
SampleApp *pMe = (SampleApp*)cxt;
if(evt == NE_PPP) {
// network event.  dwData  = NetState enum value
// refer NetState data structure in BREW API Reference doc
}
if(evt == NE_SO_CLOSING || evt == NE_SO_CLOSED) {
// socket event - closing or closed.
// DwData is pointer to socket
}
â€¦ â€¦
return;
}
```

View All Answers

Question - 14:

How to allow the input of symbols in my text control?

Ans:

Certain limited symbol entry is available by pressing a certain key (usually 1) multiple times. This set of symbols is OEM dependent. On the Kyocera 3035, the following symbols are available by pressing the '1' key multiple times: .&@,-':;?/"(). On the Sharp Z- 800, the following symbols are available by pressing the 1 key

multiple times: .,@:!?/.
To allow a greater breadth of symbol entry in your text control, you must associate your text control to a soft key menu using ITEXTCTL_SetSoftKeyMenu(). The set of symbols available using this method are:
-.&'()_!?*#%":+<>=Â¿Â¡""/@,~{}$[]^;.
ITEXTCTL_SetSoftKeyMenu() adds an item to the Soft Key menu that allows the user to change the text entry mode (the text string for this item is the currently selected mode - for example, MultiTap). When it receives this command, the text control displays a menu allowing the user to select the new text entry mode. After the user selects the new mode, the text control is reactivated and the user may continue to enter text. When entering text, the user may press the Select Key to leave text-edit mode and activate the SoftKey Menu. While the Soft Key menu is active, the user may press the UP key to return to edit mode without making a menu selection.
The following is an example of how to allow multiple input modes in your text control:

```
 // Create text control
ISHELL_CreateInstance(pMe->a.m_pIShell, AEECLSID_TEXTCTL,
(void **)&pMe->m_pIText);
if (pMe->m_pIText)  {
// Create soft key menu
ISHELL_CreateInstance(pMe->a.m_pIShell, AEECLSID_SOFTKEYCTL,
(void **)&pMe->m_pISoftKey);
if (!pMe->m_pISoftKey)  {
ITEXTCTL_Release (pMe->m_pIText);
pMe->m_pIText = NULL;
        return FALSE;
}
}
else {
return FALSE;
}
// Set the soft key menu rectangle
IMENUCTL_SetRect(pMe->m_pISoftKey, &softKeyRect);
// Set text control title, size, properties, rectangle
ITEXTCTL_SetTitle(pMe->m_pIText, NULL, NULL, titleString);
ITEXTCTL_SetMaxSize(pMe->m_pIText, 100);
ITEXTCTL_SetProperties(pMe->m_pIText, TP_FRAME | TP_MULTILINE );
ITEXTCTL_SetRect(pMe->m_pIText, &rect);
// Initialize the softkey menu of the text control.
ITEXTCTL_SetSoftKeyMenu (pMe->m_pIText, pMe->m_pISoftKey);
// Set both the soft key and text control active.
IMENUCTL_SetActive(pMe->m_pISoftKey,TRUE);
ITEXTCTL_SetActive (pMe->m_pIText, TRUE);
In the app's HandleEvent function, you must handle the EVT_KEY and EVT_COMMAND
events as follows:
case EVT_COMMAND:
// Process EVT_COMMAND event - change of input mode
if((pMe->m_pIText != NULL) && ITEXTCTL_HandleEvent(pMe->m_pIText,
eCode, wParam, dwParam)) {
return TRUE;
}
return FALSE;
case EVT_KEY:
if ((pMe->m_pIText != NULL) && ITEXTCTL_HandleEvent(pMe->m_pIText,
EVT_KEY, wParam, dwParam))   {
//Event handled by text control
        return TRUE;
        }
if(pMe->m_pISoftKey != NULL && IMENUCTL_HandleEvent(
pMe->m_pISoftKey, EVT_KEY, wParam, dwParam)) {
//Event handled by menu control
return TRUE;
}
return FALSE;
```

View All Answers

Question - 15:

Can we modify the display buffers directly?

Ans:

No. There is no way for BREW to access these and the display data is stored as the vendor's proprietary format.

View All Answers

Question - 16:

Is it possible to get/manipulate the palette information of the device?

Ans:

No. The palettes are hard-coded by the vendor and will vary from device to device.

View All Answers

Question - 17:

Can we remove the multitap item from the softkey menu associated with my text control?

Ans:

Yes, We can remove it by following the steps outlined below:
Associate the soft key menu with your text control using ITEXTCTL_SetSoftKey().
Call IMENUCTL_DeleteAll() to delete the Multitap item.
Add items using IMENUCTL_AddItem().
View All Answers

Question - 18:

How cancwe add images to the items in an IMENUCTL?

Ans:

We can use the IMENUCTL_AddItemEx method with CtlAddItem structure.
View All Answers

Question - 19:

How to create a scroll bar if menu is larger than the screen size?

Ans:

The screen rectangle in which the menu is to be drawn (specified by IMENUCTL_SetRect) must exceed the screen height by at least the height of a single menu item. Otherwise the menu item will be clipped and no scroll bar will appear.
View All Answers

Question - 20:

How to create a dialog?

Ans:

Use the BREW Resource Editor to create a dialog. You can also construct the dialog manually by creating data structures in your application to define the contents of the dialog.
Note: If a bmp file is being used, make sure that the bmp format is supported and that the bmp file is a valid full path, all in lowercase.
Once the dialog has been created, it will have a resource ID and a resource file (.bar file). Use ISHELL_CreateDialog() to create the dialog:

```
 ISHELL_CreateDialog(pMe->a.pIShell, SAMPLEAPP_RES_FILE,
RESOURCE_ID, NULL);
// SAMPLEAPP_RES_FILE is the resource file (.bar file) and
// RESOURCE_ID is the resource ID specified in the resource
// editor
```

Process the following events (return TRUE at the very least) in the app handler function:

```
 case EVT_DIALOG_START:
return TRUE;
case EVT_DIALOG_INIT:
return TRUE;
case EVT_DIALOG_END:
return TRUE;
```

Call ISHELL_EndDialog when the dialog object is no longer needed.

```
 ISHELL_EndDialog(pMe->a.pIShell);
```

View All Answers

Question - 21:

How to draw a line in a specific color?

Ans:

IDISPLAY_DrawHLine() and IDISPLAY_DrawVLine() always draw lines in black. Therefore setting CLR_USER_LINE to the desired color and then invoking IDISPLAY_DrawHLine() or IDISPLAY_DrawVLine() will not work.
The definitions of these two IDISPLAY macros are below. To draw a line in a color other than black, use the code contained in the macro definition and change to the desired fill color.

```
 #define IDISPLAY_DrawHLine(p,x,y,len)
{AEERect rc;SETAEERECT(&rc,(x),(y),(len),1); IDISPLAY_FillRect((p),&rc,
 RGB_BLACK);}
 #define IDISPLAY_DrawVLine(p,x,y,len)
{AEERect rc;SETAEERECT(&rc,(x),(y),1,(len)); IDISPLAY_FillRect((p),&rc,
 RGB_BLACK);}
```

View All Answers

Question - 22:

Can you please explain the difference between Multi-tap and T9 text entry modes for text input?

Ans:

In Multi-tap mode, a key must be tapped several times in order to specify a letter. For example, to specify the letter 'r', tap the number '7' three times. In T9 mode, a key is tapped only once per letter. T9 Text Input determines the most commonly used word matching the input numeric sequence. If more than once word matches the sequence, the most common word is selected, with the ability to scroll to the next most commonly used word.
The default text entry mode is Multi-tap. T9 text entry mode is not supported on the emulator.
View All Answers

Question - 23:

How to control animation by time?

Ans:

One way to do this is to use the IImage interface and set the rate of animation (IImage_SetParm). An example in which the animation rate is set to 750ms is shown below:

```
 IIMAGE_SetParm(pMe->m_pIImage, IPARM_RATE, 750, 0);
```

You can also use a timer to trigger the image display function. Use ISHELL_SetTimer() to set a timer:

```
 ISHELL_SetTimer(pMe->a.m_pIShell, TIMER_RATE,
(PFNNOTIFY)(ManipulateBitmap), pMe);
```

After TIMER_RATE ms expires, ManipulateBitmap function will be triggered, within which you can manipulate your image.

View All Answers

Question - 24:

Does BREW support animation?

Ans:

BREW SDK version 1.0 includes support for animated BMP. This is done by placing the frames side-by-side and specifying the width of each frame with IIMAGE_SetParm with the IPARM-CXFRAME flag. Please see the IIMAGE example in the Examples directory
BREW SDK Version 1.1 has added support for BREW Compressed Image (BCI) animation. A BCI file contains one or more compressed small graphic image(s), each with a specified duration in milliseconds. The duration represents how long you want each image to be shown before it is replaced by the next image in the series. You can use the BCI Authoring Tool included in BREW SDK Version 1.1 to create your BCI file. Please refer to 'Using the BREW Compressed Image Authoring Tool' document included in the SDK for more information.

View All Answers

Question - 25:

How to determine the character limit for the display of application names on the phone?

Ans:

Different phones have different display characteristics, so there is no unique answer to this question. You can determine whether your application name will fit on the phone's display by comparing the width of the application name to the width of the display.
Use IDISPLAY_MeasureText() to determine the pixel width of your application name string. Use ISHELL_GetDeviceInfo() to determine the pixel width of the screen

View All Answers

Question - 26:

How to suppress the title area from showing on my IStatic?

Ans:

The title area cannot currently be suppressed in SDK version 1.0. This has been added in BREW version 1.1.

View All Answers

Question - 27:

Why do we get memory errors such as "memheap.c 0696" when using IDISPLAY_BitBlt() to draw a bitmap image?

Ans:

Ensure that you are freeing the memory allocated by CONVERTBMP. Check the last Boolean parameter of CONVERTBMP. If True, a reallocation was done and the memory must be freed using SYSFREE.
For example:

```
    pBmp = CONVERTBMP (pDataBytes, &imageInfo, &bVal);
   IDISPLAY_BitBlt (pIDisplay, xDest, yDest, cxDest,
   cyDest, pBmp, xSrc, ySrc, dwRopCode);
        IDISPLAY_Update (pIDisplay);
   if(bVal)    //free only if realloc was done
   SYSFREE (pBmp);
```

View All Answers

Question - 28:

How to create a text control with horizontal and vertical scrollbars?

Ans:

If the text is larger than the Text Control size, then a vertical or horizontal scrollbar will appear and you can navigate through the text using the directional keys.

View All Answers

Question - 29:

How to move the cursor to the end of the text input?

Ans:

The cursor can be moved to the end of the string by continuously invoking ITEXTCTL_HandleEvent() method n times on the string, where n is the length of the string.
For example:

```
    while(len) {
   ITEXTCTL_HandleEvent(pMe->m_pIText, EVT_KEY,
   AVK_RIGHT, dwParam);
   len--;
   }
```

View All Answers

Question - 30:

What is the transparent color on color and monochrome phones?

Ans:

Magenta is the transparent color for color devices. White is the transparent color on monochrome and 4-level gray-scale devices.

View All Answers

Question - 31:

What image formats are supported in BREW?

Ans:

BREW supports any BMP file with a color depth up to that which is provided on the device it is running on. BREW does not yet support GIF and JPEG images. For now, you need to convert GIF and JPEG images into BMP images. PNG format and BREW Compressed Image (BCI) format will be supported in BREW SDK version 1.1.
In SDK versions prior to 1.1, the emulator is capable of emulating only 1, 4, and 8 bit color depth BMPs. In version 1.1, the emulator can also display 2 bit color depth BMPs.

View All Answers

Question - 32:

When reading from a socket the phone reads whatever it can in one go, while the emulator reads large packets in chunks. Why?

Ans:

This is a limitation of the phone.
Program should call ISOCKET_Readable() to be informed when more data can be read from the stream. The callback routine registered by ISOCKET_Readable() will be invoked whenever data becomes available---you can usually call ISOCKET_Read() at this point. Continue calling ISOCKET_Readable() for as long as your program expects more data.

```
 rv = ISOCKET_Read(piSock, (byte *)szBuf, sizeof(szBuf));
if (rv == AEE_NET_WOULDBLOCK) {
// WOULDBLOCK => no more data available at the moment
    // Register the callback to read the data later.
    ISOCKET_Readable(piSock, CommonReadCB, (void*)pMe);
return;
}
else if (rv == AEE_NET_ERROR) {
// error reading from socket
ReleaseNetAndSocket (pMe);
}
else if (rv > 0)  {
// rv bytes of data has been read from the socket into
// szBuf
// Read remaining data
ISOCKET_Readable(piSock, CommonReadCB, (void*)pMe);
}
else  {  // rv == 0
// There is no more data to be received.
// The peer has shut down the connection.
ReleaseNetAndSocket (pMe);
}
```

View All Answers

Question - 33:

Are network callbacks invoked in the context of the main thread, and if so how is the data integrity preserved in the current context?

Ans:

All network callbacks occur within the same thread context, at some point in time after the caller returns control to the AEE event loop. If your application is busy doing something, callbacks will be queued, then invoked once your application returns control to the AEE event loop, ensuring data integrity..

View All Answers

Question - 34:

How many sockets can be opened simultaneously?

Ans:

This limit is OEM specific, and will vary depending on the device that your applet is running on---it is not set by BREW.

View All Answers

Question - 35:

When transmitting large files, do we have to break the file up into packets before sending, or does BREW do this for me?

Ans:

You should simply send what you can, and ISOCKET_Write() will tell you how much data was actually sent. If there is data remaining to be sent, simply offset your data pointer by the amount indicated by your last call to ISOCKET_Write().
For Example:

```
 void CommonWriteCB (void *pUser) {
char *psz = NULL;
    int  cbWrite;
â€¦ â€¦ â€¦
SampleApp * pMe = (SampleApp*)pUser;
// pszData is the data to be written
psz = pszData;
cbWrite = STRLEN (pszData);
// Loop till all the bytes are written.
while (cbWrite > 0) {
rv = ISOCKET_Write(pMe->m_pISocket, (byte *)psz,
(uint16)cbWrite);
if (rv == AEE_NET_WOULDBLOCK) {
// No byte were written successfully. Register
// callback to try again later.
ISOCKET_Writeable(pMe->m_pISocket,
CommonWriteCB, (void *)pMe);
return;
} else if (rv == AEE_NET_ERROR) {
// Write Error.
ReleaseNetAndSocket(pMe);
return;
}
// In case of parital write, loop and write the rest
cbWrite -= rv;
psz += rv;
}
â€¦ â€¦ â€¦
}
```

View All Answers

Question - 36:

What is the largest packet size supported by BREW?

Ans:

The maximum packet size is OEM dependant, and may vary from one manufacturer's phone to another. BREW has no control over this limitation.

View All Answers

Question - 37:

With no TCP/IP Flush command available in BREW, how can I confirm that a command is sent right away?

Ans:

It is not possible to determine when a TCP/IP command is sent; BREW does not provide this capability.

View All Answers

Question - 38:

Why do we get the value of 0.0.0.0 in dnsresult.addrs when I use GetHostByName()?

Ans:

This may be due to incorrect implementation of error handling in the GetHostByName() callback function. The error handling implementation in the GetHostByName() callback of the NetSocket example is incorrect. You can use the following sample implementation:

```
 if(pMe->dnsresult.nResult > 0 && pMe->dnsresult.nResult
<= AEEDNSMAXADDRS) {
// DNS lookup  success
// Process dnsresult.addrs
for(i = 0; i < pMe->dnsresult.nResult; i++)  {
        SPRINTF(szMsg,"Addr=: %x",pMe->dnsresult.addrs[i]);
        DisplayOutput(pMe,i+4,szMsg);
      }
} else {
// DNS Failure - error code is dnsresult.nResult
SPRINTF(szMsg, "DNS: error %d", pMe->dnsresult.nResult);
      DisplayOutput(pMe, 2, szMsg);
}
```

View All Answers

Question - 39:

Why does GetHostByName() not work on my phone?

Ans:

For INETMGR_GetHostByName() to function correctly, the phone's Domain Name Server (DNS) settings must be properly configured. This is generally taken care of by the Service Provider, in which case the phone should be returned to the place of purchase, or an authorized service center so that its DNS settings can be properly configured.

View All Answers

Question - 40:

When will BREW offer HTTP Support?

Ans:

HTTP Support is available through the AEEWeb interface in version 1.1 of the BREW SDK and above. With earlier versions of the SDK, use the ISocket interface to

connect to a server's HTTP port, and send HTTP "get" and "post" requests.
View All Answers

Question - 41:
How to obtain the phone number of the device on which my application is running?

Ans:
An application can obtain the phone number of the device on which it is running by calling ITAPI_GetStatus(). The phone number is in TAPIStatus.szMobileID.
For example:

```
 TAPIStatus tapiStat;
if (ITAPI_GetStatus(pMe->p_tapi, &tapiStat) == EBADPARM) {
DisplayOutput((IApplet*)pMe, 6, "TAPI Status fetch failed");
}
DisplayOutput((IApplet*)pMe, 4, "Mobile ID:");
DisplayOutput((IApplet*)pMe, 5, tapiStat.szMobileID);
```

View All Answers

Question - 42:
When making a voice call immediately after a data call, why do we see the Privacy Alert again instead of "Return to Application"?

Ans:
Refer to above FAQ
In addition to the more common causes, stack overrun can also lead to SWI Exceptions.
Note: It is recommended that large buffers be allocated on the heap rather than on the stack as automatic variables.
View All Answers

Question - 43:
What guidelines should follow when making a voice call immediately after a data call?

Ans:
After the last socket is released, BREW waits for the network linger time (default linger time is 30s) to expire before terminating the PPP connection. The actual tearing down of the PPP connection takes about 3s. Therefore, (linger time + 3 seconds) must elapse between releasing the last socket and invoking ITAPI_MakeVoiceCall(). You should introduce a (linger time + 3 seconds) timer between releasing the last socket and making the voice call.
For example:

```
ReleaseNetAndSocket(pMe);
//LINGER_TIME below is in seconds
ISHELL_SetTimer(pMe->a.m_pIShell, (LINGER_TIME +3) * 1000,
Timer_CB, (void *)pMe);
```

In the timer callback code, you can make the voice call using ITAPI_MakeVoiceCall().
View All Answers

Question - 44:
After making a voice call using ITAPI_MakeVoiceCall, why does my application seem to be restarted when I respond No to the "Return to Application" prompt?

Ans:
Make sure that the parameter clsReturn (application to start when the call ends) in the call to ITAPI_MakeVoiceCall is 0 (zero). If you specify clsReturn as your app's Class ID, then BREW tries to create another instance of your app, rather than resume your app.
View All Answers

Question - 45:
When making a voice call using ITAPI_MakeVoiceCall, responding No to the Privacy Alert hangs the application. What is the workaround?

Ans:
This is a bug in BREW SDK version 1.0.1, and will be fixed in an upcoming release.
The recommended workaround for BREW SDK Version 1.0.1 is for the user to press 'No' twice (i.e. press Select Key twice). When the Privacy Alert Yes/No dialog is displayed, all key events up until the first Select key press go to the Dialog. After the first Select key (for the dialog), subsequent key events go to the application.
Steps are outlined below:
Added a boolean 'madeVoiceCall' in the Applet structure
If ITAPI_MakeVoiceCall() returns Success, set madeVoiceCall to TRUE

```
 retValue = ITAPI_MakeVoiceCall(pMe->p_tapi, PHONE_NUM,
AEECLSID_SAMPLEAPP);
ITAPI_Release(pMe->p_tapi);
if(retValue == SUCCESS) {
pMe->madeTapiCall = TRUE;
}
```

Handle EVT_KEY in the app handler function. If the Select key was pressed and madeVoiceCall is TRUE, the user has selected No in response to the Privacy Alert. Set madeVoiceCall = FALSE and redraw screen.

```
 case EVT_KEY:
if (wParam == AVK_SELECT && pMe->madeTapiCall ==
TRUE) {
// Redraw screen
pMe->madeTapiCall = FALSE;
}
return TRUE;
```

If the user selected 'Yes' to the Privacy Alert, the application would have been suspended. When resumed, the madeVoiceCall flag must be cleared.

```
 case EVT_APP_RESUME:
    if(pMe->madeTapiCall == TRUE) {
    pMe->madeTapiCall = FALSE;
      }
    â€¦ â€¦ â€¦
    â€¦ â€¦ â€¦
    return TRUE;
```

View All Answers

Question - 46:

What settings are required to allow a running BREW application to be suspended when a non-BREW SMS message is received on the Sharp Z-800?

Ans:

On the Sharp Z-800 handset, you must set the following in order to receive non-BREW SMS:
Main Menu -> Setup/Tool -> BREW Setting -> OFF.

View All Answers

Question - 47:

Can incoming SMS messages be emulated with the SDK?

Ans:

SMS emulation will be available in version 1.1 of the BREW SDK.

View All Answers

Question - 48:

How to deal with a Low Battery Warning?

Ans:

BREW sends the running application an EVT_APP_SUSPEND event in the case of a Low Battery Warning. In order to handle low battery condition, you must correctly handle EVT_APP_SUSPEND and EVT_APP_RESUME events.

View All Answers

Question - 49:

Tell me what events must an applet handle?

Ans:

In addition to the obvious EVT_APP_START and EVT_APP_STOP, your applet must support EVT_APP_SUSPEND and EVT_APP_RESUME in order to pass True BREW Testing.

View All Answers

Question - 50:

What difference is there between using the phones "End" key to close an applet, and using the "Clear" key to close an applet?

Ans:

An OEM of a particular device (phone) designates key(s) for the following two activities:
A key, that when pressed, will close the current application. Most OEMs designate this to be the AVK_CLR key.
A key, that when pressed, will close all applications. Most OEMs designate this to be the AVK_END key.
When AVK_END is pressed, BREW will immediately send EVT_APP_STOP to the active applet without first sending the AVK_END key. In addition, the FreeAppData() callback routine provided to AEEApplet_New() will be called prior to unloading the applet; no other events or callbacks will occur.
When AVK_CLR is pressed, BREW will first send this event to the applet. If the applet does not handle the event (i.e. returns FALSE in HandleEvent), only then BREW will close the application. In the implementation of your AVK_CLR handler, remember to also call FreeAppData as follows:

```
case AVK_CLR:
if (pMe->OnMainMenu == TRUE) {
// App is on main menu.  Therefore pressing CLR key should cause app to exit
HelloWorld_FreeAppData(pi);  //clean up
return FALSE;  //return FALSE so that BREW will now close application
}
else {  // Not on main menu.
// Therefore pressing CLR key should cause app to undo one level of menu
// nesting.  Show previous menu in menu hierarchy
return TRUE;
}
```

Make sure that your FreeAppletData() properly cleans up all allocated memory and resources. All objects and interfaces created by CreateInstance, CreateDialog, MALLOC, etc., must have an associated call to Release or FREE.

View All Answers

Question - 51:

On what devices is BREW supported?

Ans:

Currently, support for BREW is only available for devices based on the following QUALCOMM chipsets: MSM3100, MSM3300, MSM5000, MSM5100, and MSM5105. In addition, for a particular phone or other device to support BREW, it must be "BREW enabled" with a version of the BREW runtime provided by the device's manufacturer.

View All Answers

Question - 52:

Can we link to Windows DLLs from the SDK?

Ans:

No. You should treat the SDK as a fully integrated stand-alone platform, which includes avoiding the C Standard Libraries. BREW has provided a port of the most common functions.

View All Answers

Question - 53:

Does BREW support multi-threading?

Ans:

BREW does not support multi-threading. BREW does have support for cooperative multi-tasking.

View All Answers

Question - 54:

What are the Operating System requirements of the BREW SDK?

Ans:

Due to the requirement for Unicode support, the SDK runs only on the Microsoft Windows NTâ„¢ 4.0, Windows 2000â„¢ and Windows XPâ„¢ platforms. Windows 98 is not supported by the BREW SDK.

View All Answers

Question - 55:

What to do if we installed ARM BREW Builder or ARM Development Suite into a path with spaces?

Ans:

The solution is to modify your makefile in the following fashion:
Comment out the ARMBIN, ARMINC, and ARMLIB declaration.
Add the full path to the ARMCC, LD, and HEXTOOL variables. Make sure you double quote the path.

View All Answers

Question - 56:

What guidelines should be followed when compiling an application in Thumb mode?

Ans:

The function AEEMod_Load() must be moved to a different source file and this file must be compiled in ARM mode. AEEMod_Load() must always be compiled in ARM mode.
All files that are compiled in thumb mode MUST have the INTERWORK compiler option turned on. If this option is NOT turned on in the make file, the app will very likely crash.

View All Answers

Question - 57:

What compilers can be used to compile BREW applications?

Ans:

The ARM BREW Builder can be used to compile BREW applications.

View All Answers

Question - 58:

How to debug output on emulator and phone?

Ans:

When an application is running on the BREW Emulator, DBGPRINTF outputs messages to the Visual C++ Debug Window. Currently, DBGPRINTF messages are not available while running on a phone. Support for outputting debug messages to a host PC via serial connection is under development.

View All Answers

Question - 59:

How to upload a device configurator file (.qsc) to a phone?

Ans:

The device configurator file (.qsc) is used by the BREW Emulator application, and is not required to run applications on the phone.

View All Answers

Question - 60:

How to compile an application to run on a handset?

Ans:

To build applications for the phone requires the ARM Developer Suite of tools, which are used to compile and link a project that has been developed using the PC-based BREW SDK, and related tools. Running an application built using the ARM tools requires a BREW-enabled phone, a data cable for establishing a serial connection to the phone, and file copying software such as QUALCOMM's BREW AppLoader application. See How to Build a Downloadable BREW Application for more details.

View All Answers

Question - 61:

How to transfer compiled applications to a phone?

Ans:

Use BREW AppLoader to upload applications to the phone. The following example shows files and locations for an application with the name " brewApp."
/brew/ brewApp.mif

- Generated by BREW MIF Editor.
/brew/ brewApp/ brewApp.bar

- Generated by BREW Resource Editor.
/brew/ brewApp/ brewApp.mod

- Compiled and linked with ARM BREW Builder.
/brew/ brewApp/ brewApp.sig

- Digital Signature.
All directory and file names on the phone must be in lower case. Any additional files specific to your application may also be copied to the application directory, or subdirectories of the application directory.
The Digital Signature file is generated using the BREW TestSig Generator, then renamed with the name of the application. The first part of the .sig filename must be the same as the first part of the .mod filename.
After uploading files, reset the phone.

View All Answers

Question - 62:

Can an SMS message be sent to an inactive applet?

Ans:

Yes, SMS notification events can be delivered to an application, even if it is not currently active. To do so, BREW first loads the applet, and then sends the EVT_APP_MESSAGE event to it. The application may start itself by calling ISHELL_StartApplet(), or can "silently" process the event without calling ISHELL_StartApplet().

View All Answers

Question - 63:

How to send SMS messages from a BREW application?

Ans:

Currently, BREW does not expose a means to send SMS messages from within a BREW applet. BREW applications can only receive SMS messages.

View All Answers

Question - 64:

How to test an application on the emulator to determine if it behaves properly under low memory conditions?

Ans:

This can be achieved by reducing the heap size in the device configurator file (.qsc file) to almost the size of the .mod file. Any run time allocations by the applications should fail and you can test if your application deals with those failures correctly.
Since the size of the mod file a is a lot less then the dll that runs in the emulator, the emulator approximates the size of the mod file to be 10% of dll.
To properly run this test, create a separate directory that contains only the MIF of the application under test and have emulator point to this new directory (File->Change MIF Dir).

View All Answers

Question - 65:

What events must an applet handle?

Ans:

In addition to the obvious EVT_APP_START and EVT_APP_STOP, your applet must support EVT_APP_SUSPEND and EVT_APP_RESUME in order to pass True BREW Testing.

View All Answers

Question - 66:

Can you please explain the difference between MALLOC() and IHEAP_Malloc() function in BREW-SDK?

Ans:

MALLOC and IHEAP_Malloc() are identical, just as FREE and IHEAP_Free() are identical. In earlier versions of the BREW SDK, malloc was part of the IHEAP Interface, and was later made into a helper function for ease of use. Both are still available for backward compatibility, but it is recommended that, use MALLOC and FREE rather than IHEAP_Malloc() and IHEAP_Free().
An app needs to create IHeap interface only when it wants to get current memory usage statistics (IHEAP_GetMemStats()) or check if a memory block of a certain size can be allocated (IHEAP_CheckAvail()).

View All Answers

Question - 67:

Why cant an applet be run directly from flash RAM?

Ans:

BREW loads the apps into Heap RAM because files on EFS are stored as non- contiguous blocks.

View All Answers

Question - 68:

What is the size limit for an applet?

Ans:

The size of a BREW applet is limited by the amount of free file system space that is available, and by the amount of available RAM.
BREW applets, when executed, are loaded into RAM; any RAM left over can be used for memory allocation, loading resources, creating controls, etc. How much RAM is available depends on the type of phone, and its configuration. On a QCP-3035, for instance, the available RAM is 90K, 30K of which belongs to BREW and other phone related tasks, leaving 60K available for use by an applet. The memory available on each of the phones running BREW is available here.
Another limiting factor for applet size is heap fragmentation. If the largest available memory block is smaller than the size of the applet, then the applet will not be loaded. Use IHEAP_CheckAvail() to determine whether a memory block of sufficient size can be allocated. IHEAP_GetMemStats() can be used to determine the amount of RAM already used. If you have an applet that is too large for a particular device, a work-around is to split the application into multiple applets. Memory Allocation provides more information on how to avoid heap fragmentation.

View All Answers

Question - 69:

How does the memory architecture in BREW work?

Ans:

BREW device has Flash RAM and Heap RAM. You can treat Flash RAM as a hard drive. Programs are stored there, but not run from it. You can treat Heap RAM as you would always treat a memory heap. Say you have an application that is 35k in size. This will take 35k from the Flash RAM to store on the device. When the application is run, the 35k will be loaded into Heap RAM in its entirety plus however much Heap RAM it needs to allocate. When the applet is released, this Heap RAM is freed. It will still occupy space on the Flash RAM until it is removed from the device.

View All Answers

Question - 70:

Why does the emulator display a blank screen with my applications name when my application exits?

Ans:

Problem: When an application exits while running on the emulator, instead of returning to the main menu screen, a blank screen is displayed with the application name in the upper left corner.
This behavior is caused by not freeing all allocated memory before exiting the application. Any buffer you allocate using MALLOC or IHEAP_Malloc() must be freed using FREE or IHEAP_Free(). For any instance of a BREW class that you create, you must call the corresponding release function to free that instance.

View All Answers

Question - 71:

What is an "SWI Exception"?

Ans:

An "SWI Exception" is usually related to heap memory mismanagement. The most common causes of heap memory corruption are: overwriting the bounds of your allocated arrays, freeing objects more than once, and writing to wild or NULL pointers.
In addition to the more common causes, stack overrun can also lead to SWI Exceptions.

View All Answers

Question - 72:

What is a "Pref Abort Exception"?

Ans:

A "Pref Abort Exception" usually indicates that memory important to the phone operating system has been trashed. The two most common causes of this exception are data corruption (i.e. running off the end of an array), and stack overruns.

View All Answers

Question - 73:

What is a "Re-entrant Data Abort"?

Ans:

The "Re-entrant Data Abort" exception is often caused by stack overrun. When an applet is running on the phone the stack size is small, and you are more likely to see stack overrun problems than when running on the emulator. Solutions to this problem are reducing the size or number of objects on the stack, and using objects allocated on the heap instead of automatic variables.

View All Answers

Question - 74:

How to recover the phone from an "UndefInst Exception"?

Ans:

"Undef Inst Exception" stands for "undefined instruction exception," and means that the program pointer has jumped to a code segment that contains an undefined instruction. This can be the result of memory corruption, a stack overrun, or version-related incompatibilities between applet code and the BREW image on the phone.

Memory is tighter on the phone than when running on the emulator. Therefore, memory and stack overrun problems will be more likely to show up on the phone. You should double check memory related areas of your code, especially auto variable arrays and the maximum total size of allocated memory.
To clear the phone, pull the battery out for a few seconds.

View All Answers

Question - 75:

How to test application for proper handling of Suspend/Resume events on a non- provisioned phone?

Ans:

On the Kyocera 3035, you can test your app's handling of Suspend/Resume events by enabling automatic Keyguard (Main Menu->Settings->Keyguard) before running your app. When the Keyguard kicks in, the running app will receive a Suspend Event. The app will receive Resume event when the screen is unlocked.
On the Sharp Z-800, you can test your app's handling of Suspend/Resume events by setting the alarm to go off a few minutes in the future (Main Menu->Setup/Tools->Alarm->Daily Alarm) and then running the BREW app. Please note that in order for the app to be suspended when the alarm goes off, BREW Priority Setting must be off (Main Menu->Setup/Tools->BREW Priority Setting).

View All Answers

Question - 76:

What notification events can an app register for?

Ans:

An app can register for the following System notifications:
TAPI (Class ID: 0x01001007)
NMASK_TAPI_STATUS
0x0001
TAPI Status change event
NMASK_TAPI_SMS_TEXT
0x0002
Incoming SMS
NMASK_TAPI_SMS_TS
0x0004
SMS message on specific Teleservice ID
INETMGR
NMASK_OPENED
0x0001
Network layer is available
NMASK_CLOSED
0x0002
Network layer is closed
NMASK_IDLE
0x0004
Network layer available and idle

View All Answers

Question - 77:

Explain usage or BREW?

Ans:

Brew OS is used by some mobile phone manufacturers and mobile networks, however most often the end-user does not know this since mobile phones running Brew most often lack any Brew OS branding and Brew runs in the background with the custom "skins" of the mobile phone manufacturer or operator on-top. Brew OS is used by Sprint Nextel, metroPCS, Cricket Wireless, U.S. Cellular, Verizon, Syringa Wireless, and AT&T in the US and by the 3 network in much of Europe, the UK and Australia on many mobile phones produced especially for their network.

View All Answers

Question - 78:

Explain about Binary Runtime Environment (BREW)?

Ans:

Binary Runtime Environment for Wireless (Brew MP, Brew, or BREW) is an application development platform created by Qualcomm, originally for code division multiple access (CDMA) mobile phones, featuring third-party applications such as mobile games. It is offered in some feature phones but not in smartphones. It debuted in September 2001.

View All Answers

Mobile OS Most Popular & Related Interview Guides