Archived:Installing and uninstalling MIDlets programmatically

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Note: This API is not part of the public SDK. It can be found in the SDK API Plug-in.

Description

The following code demonstrates how to install and uninstall a MIDlet programmatically using the Software Installer APIs.

Solution

Installing a MIDlet

A MIDlet can be installed using the SW installer APIs as below:

Capability requirements: *TrustedUI*

```
SwiUI::RSWInstSilentLauncher iLauncher;
SwiUI::TInstallOptions iOptions;
SwiUI::TInstallOptionsPckg iOptionsPckg;

// Connect to software installer server
User::LeaveIfError(iLauncher.Connect());

iOptions.iUpgrade = SwiUI::EPolicyNotAllowed;
iOptions.iOCSP = SwiUI::EPolicyAllowed;
iOptions.iDrive = 'C';
iOptionsPckg = iOptions;

iLauncher.SilentInstall( iStatus, KPathAndFileNameOfJarFile, iOptionsPckg );
SetActive();
```

The above code is meant to be run by an active object (derived from `CActive`) and uses the asynchronous version of `SilentInstall()` method.

It is also possible to install a MIDlet using the CDocumentHandler API. For more information on this API, refer to SDK Help.

Uninstalling a MIDlet

Note: This method is deprecated and no longer works from S60 3rd Edition, Feature Pack 2 onwards.

When a MIDlet suite (possibly containing several MIDlets) is installed, the system creates a Suite UID for it and stores it in the Java registry. However, the only way to uninstall a MIDlet is to uninstall the parent MIDlet Suite. The MIDlet Suite UID must be passed to `SwiUI::RSWInstSilentLauncher`. However, this UID cannot be retrieved with methods from `RApaLsSession`.

Every time a MIDlet is installed, a new UID is generated for the application. Installer also creates some files for each installed MIDlet:

```
C:\private\102033E6\MIDlets\[UID]\    (directory named with MIDlet Suite UID)
```

MIDlet .jar and .jad files are copied to this directory. It is possible to extract the MIDlet context (Suite) UID by searching the `C:\System\install\registry\` directory for "<UID>.reg" files. If the MIDlet name appears inside the .reg file, it can be assumed that the registry file belongs to the correct MIDlet Suite. This UID can then be passed to `SwiUI::RSWInstSilentLauncher` in order to uninstall the Suite.

```
void CJarUninstallAppUi::UninstallMIDletL()
 {
 RFs fileSession;
 CDir* dirList;
 TInt i;
 TFileName fullPath;
 TUint contextUid;
 TFileName fileName;

 _LIT(KDirName1, "c:\\system\\install\\registry\\");
 _LIT(KFileSpec1,"c:\\system\\install\\registry\\*.*");

 User::LeaveIfError( fileSession.Connect() );
 CleanupClosePushL( fileSession );

 // Get the file list, sorted by name
 User::LeaveIfError( fileSession.GetDir( KFileSpec1,
                                         KEntryAttMaskSupported,
                                         ESortByName, dirList ) );
```


(C) Copyright Nokia 2013. All rights reserved.

```
CleanupStack::PushL(dirList);

for ( i=0; i < dirList->Count(); i++ )
  {
  fileName = (*dirList)[i].iName;
  if( fileName.Compare(_L("temp")) == KErrNone )
    {
    continue;
    }

  fullPath = KDirName1;
  fullPath.Append(fileName);
  RFile lFile;
  TInt err = lFile.Open( fileSession, fullPath, EFileShareAny | EFileRead );

  if(err == KErrNone)
    {
    TInt lSize;
    lFile.Size(lSize);
    RBuf8 lBuf;
    lBuf.Create(lSize);
    lBuf.CleanupClosePushL();
    lFile.Read(lBuf);//read contents into buffer
    RBuf buff;
    buff.Create(lSize);
    buff.CleanupClosePushL();
    buff.Copy(lBuf);//Copy into 16 bit buffer
    lFile.Close();
    if (err == KErrNone)
      {
      // Check for MIDlet name in the file
      err = buff.Find(_L("somemidlet.jar"));
      if (err != KErrNotFound)
        {
        TLex lex( fileName.Mid(0,8) );
        lex.Val( contextUid, EHex );
        SwiUI::RSWInstSilentLauncher inst;
        inst.Connect();
        SwiUI::TUninstallOptions options;
        SwiUI::TUninstallOptionsPckg optionsPckg;
        options.iKillApp = SwiUI::EPolicyAllowed;
        options.iBreakDependency = SwiUI::EPolicyAllowed;
        optionsPckg = options;
        TInt resp = inst.SilentUninstall( TUid::Uid(contextUid),
                                          optionsPckg,
                                          SwiUI::KJavaMIMEType );
        inst.Close();
        }
      }
    CleanupStack::PopAndDestroy(2); // buff, lBuf
    }
  }
CleanupStack::PopAndDestroy(2); // dirList, fileSession
}
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.