Archived:Personalisation and Skins Settings Symbian API

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*}}.

The Personalisation and Skin Setting API can be used for retrieving the following information:

- Location of the active skin (theme) and its UID
- Available color schemes
- Active color scheme and its UID
- Bitmap index/path of background image in idle state and Favorites
- UID of the background image in idle state
- Screen saver path name
- System default screen saver
- Type of wallpaper, status of full-screen wallpaper

Header files

```
#include <aknskinsinternalcrkeys.h>
#include <centralrepository.h> // for CRepository
```

Link against

```
LIBRARY centralrepository.lib
```

```
CRepository* iRepository = CRepository::NewL( KCRUidPersonalisation );
```

Commonly used variables

```
TInt iIndex;
TInt iId;
TInt iErrCode;
TBuf<100> iBufInfo;
TBuf<5>   iBufErrCode;
TUid iUid;
TLex iIntConv;
```

Code examples

Retrieving location of the active skin (integer value):

Possible values: 0: Phone (drives C:\ and Z:\) 2: MMC (E:\)

```
iErrCode = iRepository->Get(KPslnActiveSkinLocation, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
{
switch(iId)
  {
  case 0:
    {
    iBufInfo.Copy(_L("C or Z drive"));
    }
    break;
  case 2:
    {
    iBufInfo.Copy(_L("E drive"));
    }
    break;
  default:
    {
    iBufInfo.Copy(_L("Err"));
    }
    break;
  }
  CEikonEnv::InfoWinL(_L("Loc of active skin"), iBufInfo);
}
else
{
CEikonEnv::InfoWinL(_L("Err in getting active skinloc"),iBufErrCode);
}
```


(C) Copyright Nokia 2013. All rights reserved.

```
iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving active skin UID:

```
iErrCode = iRepository->Get(KPslnActiveSkinUid, iBufInfo);
iBufErrCode.AppendNum(iErrCode);
if( iErrCode == KErrNone )
{
iIntConv.Assign(iBufInfo);
iIntConv.Val(iUid.iUid);
iBufInfo.Copy(iUid.Name());
CEikonEnv::InfoWinL(_L("Uid of active skin"), iBufInfo);
}
else
{
CEikonEnv::InfoWinL(_L("Err in getting active skin uid"),iBufErrCode);
}
iBufErrCode.Zero();
iBufInfo.Zero();
iUid = TUid::Null();
```

Retrieving active color palette UID:

```
iErrCode = iRepository->Get(KPslnColorPaletteUid, iBufInfo);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  iIntConv.Assign(iBufInfo);i
  iIntConv.Val(iUid.iUid);
  iBufInfo.Copy(iUid.Name());
  CEikonEnv::InfoWinL(_L("Uid of Colour Palette is:"), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Colour Palette Uid:"), iBufErrCode);
  }

iBufErrCode.Zero();
iBufInfo.Zero();
iUid = TUid::Null();
```

Retrieving the color scheme:

Possible values: 0: Nokia "Classic" blue 1: Nokia green 2: Nokia purple 3: red 4: pink 5: orange

```
iErrCode = iRepository->Get(KPslnColorPalette, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  switch( iId )
    {
    case 0:
      {
      iBufInfo.Copy(_L("blue"));
      }
      break;
    case 1:
      {
      iBufInfo.Copy(_L("green"));
      }
      break;
    case 2:
      {
      iBufInfo.Copy(_L("purple"));
      }
      break;
    case 3:
      {
      iBufInfo.Copy(_L("red"));
      }
      break;
    case 4:
      {
      iBufInfo.Copy(_L("pink"));
      }
      break;
    case 5:
      {
      iBufInfo.Copy(_L("orange"));
      }
      break;
    default:
      {
      iBufInfo.Copy(_L("err"));
      }
      break;
    }
    CEikonEnv::InfoWinL(_L("Color scheme"), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Colour "),iBufErrCode);
  }

iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving bitmap index of active dimmed background image in idle state:

```
iErrCode = iRepository->Get(KPslnDimmedIdleBackgroundImageIndex, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  iBufInfo.AppendNum(iId);
  CEikonEnv::InfoWinL(_L("Bitmap index of the active dimmed background image in the idle state:"), iBufInfo);
  }
```


(C) Copyright Nokia 2013. All rights reserved.

```
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Bitmap Image Index:"), iBufErrCode);
  }

iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving bitmap index of the active background image in Favorites:

```
iErrCode = iRepository->Get(KPslnFavouritesBackgroundImageIndex, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  iBufInfo.AppendNum(iId);
  CEikonEnv::InfoWinL(_L("Favourites Background Image Index:"),iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Bitmap Image Index:"),iBufErrCode);
  }
iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving the bitmap index of the active background image in the idle state:

```
iErrCode = iRepository->Get(KPslnIdleBackgroundImageIndex, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  iBufInfo.AppendNum(iId);
  CEikonEnv::InfoWinL(_L("Idle Background Image Index:"), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Bitmap Image Index:"),iBufErrCode);
  }

iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving path name of the active background image file in the idle state:

```
iErrCode = iRepository->Get(KPslnIdleBackgroundImagePath, iBufInfo);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  CEikonEnv::InfoWinL(_L("Path of BgImg in Idle state:"), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Path"), iBufErrCode);
  }
iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving path name of the active background image file in Favorites:

```
iErrCode = iRepository->Get(KPslnFavouritesBackgroundImagePath, iBufInfo);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  CEikonEnv::InfoWinL(_L("Path of BgImg in Favourites:"), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Path"), iBufErrCode);
  }
iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving the active icon set UID:

```
iErrCode = iRepository->Get(KPslnIconSetUid, iBufInfo);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  iIntConv.Assign(iBufInfo);
  iIntConv.Val(iUid.iUid);
  iBufInfo.Copy(iUid.Name());
  CEikonEnv::InfoWinL(_L("Uid Active icon set:"), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting active icon uid"),iBufErrCode);
  }
iBufErrCode.Zero();
iBufInfo.Zero();
iUid = TUid::Null();
```

Retrieving active screen saver UID:

```
iErrCode = iRepository->Get(KPslnScreenSaverUid, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  iUid = TUid::Uid(iId);
  CEikonEnv::InfoWinL(_L("Active screen savers Uid:"),iUid.Name());
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Active screen savers Uid."),
  iBufErrCode);
```


(C) Copyright Nokia 2013. All rights reserved.

```
    }
  iBufErrCode.Zero();
  iBufInfo.Zero();
  iUid = TUid::Null();
```

Retrieving system default screen saver (read-only):

```
  iErrCode = iRepository->Get(KPslnSystemDefaultScreenSaver, iBufInfo);
  iBufErrCode.AppendNum(iErrCode);
  if ( iErrCode == KErrNone )
    {
    iIntConv.Assign(iBufInfo);
    iIntConv.Val(iUid.iUid);
    iBufInfo.Copy(iUid.Name());
    CEikonEnv::InfoWinL(_L("Default screen saver:"),iBufInfo);
    }
  else
    {
    CEikonEnv::InfoWinL(_L("Err in getting Default screen saver"),iBufErrCode);
    }
  iBufErrCode.Zero();
  iBufInfo.Zero();
  iUid = TUid::Null();
```

Retrieving active background image UID in idle state:

```
  iErrCode = iRepository->Get(KPslnIdleBackgroundImageUid, iBufInfo);
  iBufErrCode.AppendNum(iErrCode);
  if ( iErrCode == KErrNone )
    {
    iIntConv.Assign(iBufInfo);
    iIntConv.Val(iUid.iUid);
    iBufInfo.Copy(iUid.Name());
    CEikonEnv::InfoWinL(_L("Idle Background Image Uid"),iBufInfo);
    }
  else
    {
    CEikonEnv::InfoWinL(_L("Err in getting IdleBackgroundImage Uid."),
  iBufErrCode);
    }
  iBufErrCode.Zero();
  iBufInfo.Zero();
  iUid = TUid::Null();
```

Retrieving active background image UID in Favorites:

```
  iErrCode = iRepository->Get(KPslnFavouritesBackgroundImageUid, iBufInfo);
  iBufErrCode.AppendNum(iErrCode);
  if ( iErrCode == KErrNone )
    {
    iIntConv.Assign(iBufInfo);
    iIntConv.Val(iUid.iUid);
    iBufInfo.Copy(iUid.Name());
    CEikonEnv::InfoWinL(_L("Favourites Background Image Uid:"),iBufInfo);
    }
  else
    {
    CEikonEnv::InfoWinL(_L("Err in getting Favourites Background Image Uid"),
  iBufErrCode);
    }
    iBufErrCode.Zero();
    iBufInfo.Zero();
    iUid = TUid::Null();
```

Retrieving list of supported color schemes:

```
  iErrCode = iRepository->Get(KPslnAvailableColorPalettes, iBufInfo);
  iBufErrCode.AppendNum(iErrCode);
  if ( iErrCode == KErrNone )
    {
    CEikonEnv::InfoWinL(_L("AvailableColorPalettes numbered [0 so on]:"),iBufInfo);
    }
  else
    {
    CEikonEnv::InfoWinL(_L("Err in getting AvailableColorPalettes"),   iBufErrCode);
    }
  iBufErrCode.Zero();
  iBufInfo.Zero();
```

Retrieving highlight animation enabled/disabled status:

```
  TInt value;
  iErrCode = iRepository->Get(KPslnHighlightAnimationEnabled, value);
  iBufErrCode.AppendNum(iErrCode);
  if( iErrCode == KErrNone )
    {
    // To display message
    _LIT(message0,"Disabled");
    _LIT(message1,"Enabled");
    CAknInformationNote* informationNote = new (ELeave)CAknInformationNote;
    if( value == 0 )
      {
      informationNote->ExecuteLD(message0); //disabled
      }
    else if( value == 1 )
      {
      informationNote->ExecuteLD(message1); //enabled
      }
    }
  else
    {
    CEikonEnv::Static()->AlertWin(_L("Err in getting HighlightAnimStatus"), iErrCode);
    }
    iBufErrCode.Zero();
    iBufInfo.Zero();
```


(C) Copyright Nokia 2013. All rights reserved.

Retrieving status of full-screen wallpaper:

```
// supported from S60 3rd Edition, Feature Pack 2 onwards
```

```
TInt value;
iErrCode = iRepository->Get(KPslnFullScreenWallpaperEnabled, value);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  _LIT(message0,"Disabled");
  _LIT(message1,"Enabled");
  CAknInformationNote* informationNote = new (ELeave) CAknInformationNote;
  if( value == 0 )
    {
    informationNote->ExecuteLD(message0); //disabled
    }
  else if( value == 1 )
    {
    informationNote->ExecuteLD(message1); //enabled
    }
  }
else
  {
  CEikonEnv::Static()->AlertWin(_L("Err in getting FullScreenWallpaperStatus"), iBufErrCode);
  }
iBufErrCode.Zero();
iBufInfo.Zero();
```

Retrieving the wallpaper type:

```
// Supported from S60 3rd Edition, Feature Pack 2 onwards
```

```
iErrCode = iRepository->Get(KPslnWallpaperType, iId);
iBufErrCode.AppendNum(iErrCode);
if ( iErrCode == KErrNone )
  {
  switch( iId )
    {
    case 0:
      {
      iBufInfo.Copy(_L("None"));
      }
      break;
    case 1:
      {
      iBufInfo.Copy(_L("Normal"));
      }
      break;
    case 2:
      {
      iBufInfo.Copy(_L("SlideSet"));
      }
      break;
    default:
      {
      iBufInfo.Copy(_L("err"));
      }
      break;
    }
  CEikonEnv::InfoWinL(_L("Wallpaper Type : "), iBufInfo);
  }
else
  {
  CEikonEnv::InfoWinL(_L("Err in getting Wallpaper type:"), iBufErrCode);
  }
iBufErrCode.Zero();
iBufInfo.Zero();
```

Example project

The following application has been tested with Nokia N78: File:PrsnAndSkinSettingsFor3rdFP2.zip

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.