Constructing a Symbian custom control from a resource

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 article is believed to be still valid for the original topic scope.

Overview

This code snippet shows how to create `CCoeControl` from a resource. This example extends the code snippet How to create a custom control in Symbian C++.

`CMyControl::ConstructFromResourceL()` initiates the control in the same way as the `CMyControl::ConstructL()` method, but the parameters are obtained from TResourceReader. The resource reader is a utility class that reads bytes from a resource file stream; therefore, reading must be in the same order and the data sizes must match those defined in the resource structure declaration.

MMP file

The following capabilities and libraries are required:

```
LIBRARY bafl.lib //TResourceReader
```

Header

Add a new function that is called when creating the class from the resource and add the `BARSREAD.H` include. The default constructor `CMyControl()` should be moved to public.

```
#include <BARSREAD.H>

public:
    CMyControl();
    void ConstructFromResourceL(TResourceReader& aReader);
```

Source

This code reads text for `CEikLabel` from the resource.

```
void CMyControl::ConstructFromResourceL(TResourceReader& aReader)
    {
    // No parent owner, so create an own window
    CreateWindowL();

    // Initialize component array
    InitComponentArrayL();

    // Create contained controls
    iStatusText = new (ELeave) CEikLabel;
    iStatusText->SetContainerWindowL(*this);

    // Read label from resource
    TPtrC label = aReader.ReadTPtrC16();
    iStatusText->SetTextL(label);

    // Store component to component array
    Components().AppendLC(iStatusText);
    CleanupStack::Pop(iStatusText);

    // Set component rect to CMultiViewsAppUi::ClientRect()
    CMultiViewsAppUi* appui = (static_cast<CMultiViewsAppUi*>(iEikonEnv->AppUi()));
    appui->ClientRect();
    if (appui)
        {
        SetRect(appui->ClientRect());
        }

    ActivateL();
    }
```

CMyControl component resource customcontrol.rh file

Resource config parameters of the CMyControl component

(C) Copyright Nokia 2013. All rights reserved.

```
STRUCT CUSTOMCONTROL
{
LTEXT txt;
}
```

Resource .rss file

Make the following changes to the Multiviews example application resource file `multiviews.rss`.

```
// New include that is our own component resource config
#include "customcontrol.rh"

// Defining resource for our component.
RESOURCE CUSTOMCONTROL r_custom_control
    {
    txt = STRING_r_custom_control;
    }
```

`STRING_r_custom_control` is defined in `multiviews.rls`.

Creating component from resource

Create the resource reader, create the component via the default constuctor, and call `ConstructFromResourceL()`

```
// Creating control from resource
TResourceReader reader;
iEikonEnv->CreateResourceReaderLC(reader, R_CUSTOM_CONTROL);
iContainer2 = new (ELeave) CMyControl;
iContainer2->ConstructFromResourceL(reader);
iContainer2->SetRect(ClientRect());
CleanupStack::PopAndDestroy(); // reader
```

Postconditions

`CMyControl` is created by defined resource parameters.

See also

Custom Control Series:

- How to create a custom control in Symbian C++ Defining a custom control
- Creating a Symbian container control Creating a container control
- Changing the focus of a Symbian custom control Handling key events and changing active custom control focus
- Archived:Using scrollbars in Symbian container control Adding scroll bar to custom control
- Using a Symbian custom control in a dialog Adding a custom control into CAknDialog
- File:CustomControl.zip Example code patch

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.