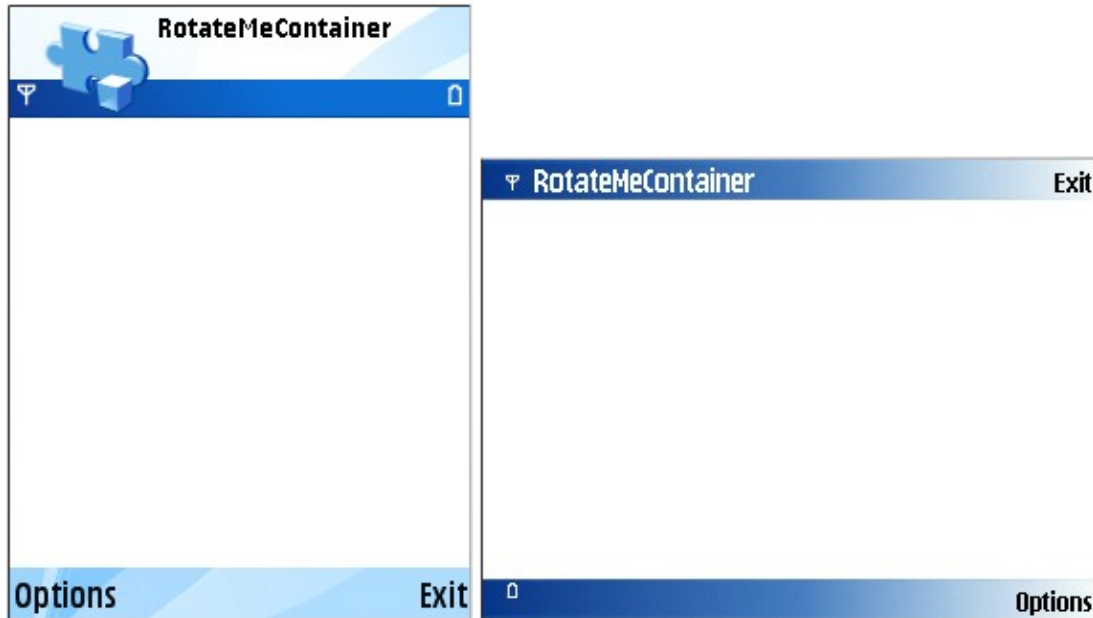




The following example shows how to change the screen orientation of an S60 UI application, from portrait to landscape or vice-versa.



The method that is used to do the rotation is `CAknAppUiBase::SetOrientationL()`. There is also another method to get the current screen orientation, `CAknAppUiBase::Orientation()`.

The following example shows a method that will change the orientation to portrait if the current one is landscape and vice-versa.

```
void CMyClass::RotateMe()
{
    // Changing from portrait to landscape or vice versa.
    iIsPortrait = !iIsPortrait;

    // Change the screen orientation.
    if (iIsPortrait)
    {
        AppUi() ->SetOrientationL(CAknAppUi::EAppUiOrientationPortrait);
    }
    else
    {
        AppUi() ->SetOrientationL(CAknAppUi::EAppUiOrientationLandscape);
    }
}
```

If the application wants just be in a single UI orientation then it should use the `CAknAppUi` flags to force certain orientation: `EAppOrientationPortrait` or `EAppOrientationLandscape`.

```
CMyAppUi::ConstructL ()
{
    BaseConstructL (EAknEnableSkin|EAppOrientationLandscape);
}
```

Change_screen_orientation_of_UI_application

```
//...  
}
```

Note: The flags are names and defined differently to those used in SetOrientationL. SetOrientationL e.g. uses EAppUiOrientationPortrait whereas CAknAppUI flag for the same orientation is EAppOrientationPortrait. This is error prone and has created lot of confusion as developers do not realized the slight naming difference.

Internal links

- [Scalable UI](#)
- [Layout-awareness challenges in custom UIs](#)
- [How to find out the correct location for softkey labels](#)
- [How to get the Current Orientation](#)

External links

- [S60 Platform: Scalable UI Guideline](#)
- [S60 Platform: Custom UI and Screen Rotation Example](#)