

<b>ID</b>	TSS000430	<b>Creation date</b>	October 5, 2006
<b>Platform</b>	S60 3rd Edition	<b>Devices</b>	
<b>Category</b>	Symbian C++	<b>Subcategory</b>	

**Keywords (APIs, classes, methods, functions):**

## Overview

Writing a screen saver for S60 3rd Edition devices

## Description

This solution shows how to write a screen saver for S60 3rd Edition.

### General information:

- The example code is attached to this solution, both as a [Carbide.c++ project](#) and a [standalone MMP project](#).
- The Carbide.c++ project may not work at the first try because there are some hard-coded paths in the project properties, but these should be easy to fix to match your own paths.
- The .sis and .sixx files are signed with a private Developer Certificate, so they won't install into your test device. You need to obtain a Developer Certificate and sign the .sis file with your own Developer Certificate.

### Applicability

This solution focuses on building screen savers for S60 3rd Edition phones; if you want to create a screen saver for S60 2nd Edition, see [Series 60 2nd Edition: Screen Saver Example](#).

### Important points:

- Setting the TARGETTYPE directive to PLUGIN in your .mmp file: This is new in S60 3rd Edition and specifies that you are building an ECom plug-in.
- 0x10009D8D is the ECom dll Recognition UID, while 0x01EF299A is a unique dll UID.
- Due to the ReadDeviceData and WriteDeviceData capabilities requirement, you will not be able to write a screen saver without having a Developer Certificate bound to an ACS Publisher ID, because only those DevCerts have the extended capability set required by the screen saver plug-in framework.

### Plug-in registration resource file:

As screen savers are now ECom plug-ins, you need to create a registration resource file called ExampleScreenSaver.rss (see source code).

### Important points in this file:

- dll\_uid and implementation\_uid are the dll UIDs. interface\_uid comes straight from the screensaverpluginintdef.hrh file and identifies this plug-in as a plug-in for the screen saver engine.
- display\_name is the friendly name which will be displayed in the Themes application when you configure the current theme to use your screen saver (see more details below).

### Source code:

You need to inherit your screen saver class from `CScreensaverPluginInterfaceDefinition` that implements some ECom construction/destruction code and in turn inherits from `MScreensaverPlugin`, which defines the actual interface you will implement in order to make your screen saver work.

The important functions you need to implement in order to have real functionality in your plug-in are:

`InitializeL(MScreensaverPluginHost *aHost)`

This will be called once, when the plug-in is initialized for the first time, and it will pass you a pointer to the plug-in host (`MScreensaverPluginHost` class) which you can use for important things such as requesting the backlight to be on or off, setting the timer refresh value, and setting the active display area (so you save power by only activating screen areas where you are actually drawing something), among others.

`Draw(CWindowGc& aGc);`

This will be called each time the refresh timer expires, passing you a `CWindowGc` graphics context that you'll use to perform the actual drawing of your animations.

`HandleScreensaverEventL(TScreensaverEvent aEvent, TAny* aData);`

Here you can handle some screen saver events that interest you. See `TScreensaverEvent` class documentation in the S60 3rd Edition SDK.

See `ExampleScreenSaver.cpp` for the implementations of these functions in the example.

The last thing you need to do is to export a function that provides the mapping between the UID of the implementation and its respective creation function. This is pretty standard ECom code and it is documented in the `ImplementationProxy.cpp` source file.

### **Building, packaging, and installing the example**

If you use `Carbide.c++` or `CodeWarrior`, just perform a full rebuild of the project. If you are building it from the command line, just `cd` to the group folder and type the command:

```
abld build gcce urel
```

This will build `ExampleScreenSaver.rsc` and `ExampleScreenSaver.dll` files, which will be copied to `\resource\plugins` and `\sys\bin\` folders in the device, respectively.

Next, create the `.sis` file that will be installed on the phone. Again, if you are using an IDE, the `.sis` file will be built (and signed, depending on your configuration) for you automatically. Otherwise, you must `cd` to the `sis` folder and build the package by typing:

```
makesis ExampleScreenSaver.pkg
```

which will build a `ExampleScreenSaver.sis`. Prior to installing, you need to sign it by using the following command:

```
signsis ExampleScreenSaver.sis ExampleScreenSaver.sisx your_cert_file.cer your_key_file.key
```

where `your_cert_file.cer` is your Developer Certificate, obtained as described at the beginning of this topic, and `your_key_file.key` is the private key file for this certificate.

Installing your screen saver on the device is easy: Just transfer it via Bluetooth, cable (using Nokia PC Suite), or by downloading it over the Internet.

[Carbide.c++ project](#)

[Standalone MMP project](#)