Camera key handling in Nokia Belle

Nokia Belle's Camera application has changed its handling of the dedicated camera key. This article describes how 3rd party applications can adapt to this change and remain able to handle the camera key event.

Summary

If you are developing a camera application, or any other utility app which intents to use the camera key event for custom user initiated actions, one of the essential tasks is to capture the camera key even and prevent its default handling, which would be starting the built-in Camera application and bringing it to foreground.

The task itself is easy to achieve, all that is required is a call to Symbian's RWindowGroup::CaptureKey() API in order to capture the key pressed event of the camera key, as in line #73 of the code snippet below.

```
RWsSession &wsSession = CCoeEnv::Static()->WsSession();
```

```
TUint wGroupHandle = wsSession.GetWindowGroupHandle(wGroupId);
```

```
if (wGroupHandle) {
```

```
    RWindowGroup wGroup(wsSession);
```

```
    wGroup.Construct(wGroupHandle);
```

```
    TBool captureOK = ETrue;
```

```
    captureOK = captureOK && (wGroup.CaptureKey(EKeyCamera, 0, 0, 100) > 0);
```

```
    if (!captureOK)
```

```
        qWarning("Unable to register for camera capture key events, SwEvent capability may be missing");
```

```
}
```

In Nokia Belle, the Camera app is started on the camera key down event so capturing the key pressed event is not enough, your application would have to capture the up and down events as well. This is easily solved with an additional call to RWindowGroup::CaptureKeyUpAndDowns(), as illustrated at line #74 in the code below.

```
RWsSession &wsSession = CCoeEnv::Static()->WsSession();
```

```
TUint wGroupHandle = wsSession.GetWindowGroupHandle(wGroupId);
```

```
if (wGroupHandle) {
```

```
    RWindowGroup wGroup(wsSession);
```

```
    wGroup.Construct(wGroupHandle);
```

```
    TBool captureOK = ETrue;
```

```
    captureOK = captureOK && (wGroup.CaptureKey(EKeyCamera, 0, 0, 100) > 0);
```

```
    captureOK = captureOK && (wGroup.CaptureKeyUpAndDowns(EStdKeyDevice7, 0, 0, 100) > 0);
```

```
    if (!captureOK)
```

```
        qWarning("Unable to register for camera capture key events, SwEvent capability may be missing");
```


(C) Copyright Nokia 2013. All rights reserved.

```
    }
```

Of course the application will have to release the key events to their normal handling as soon as the application no longer requires them. The corresponding calls to RWindowGroup::CancelCaptureKey() and RWindowGroup::CancelCaptureKeyUpAndDowns() will for example have to be made each time your camera application is sent by the user to the background, just as you would also release the camera resource at the same time, in order to allow other applications to make use of them.

Note: The above code snippets are slightly modified extracts from QtMobility's camera example application (camerakeyevent_symbian.cpp), available in the Qt SDK 1.2 under \QtSDK\Examples\QtMobility\camera. The code makes use of Symbian C++ APIs so similar code is to be used on Symbian C++ applications.

Please also note that the CaptureKey* APIs require that your application has the SwEvent capability, for which self-signing is not supported. Before using this API in your application make sure that you have applied for a valid developer certificate and enable its use for signing in your IDE's settings.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.