Graphics memory handling

Overview

This article describes how to handle Graphics memory issue on Symbian^3 device.

Note: It has been tested on Nokia N8 and the API will work from Symbian^3 onwards.

Basic Idea

With some devices having graphics accelerator, there are also dedicated additional Graphics memory that is used with the accelerator to speed up the graphics handling. The Graphics memory is independent from the normal RAM memory used by applications, and solely used for graphics content.

Similarly as the RAM the graphics memory is also limited resource which can run out. With Symbian^3 OS there is a Graphics memory monitoring application called Graphics Out Of Memory (GOOM) monitor, which is monitoring the Graphics memory usage, and once it determines that the Graphics memory is running low, it will start sending shut-down events to the processes utilizing Graphics memory.

GOOM event handling with Qt

With Qt applications the graphics memory allocations & freeing is automated. Thus the graphics memory is allocated when an UI element is shown. Thus the amount of graphics memory used depends on the drawing size of the UI elements, i.e. for example the size of an image drawn to the screen.

With Qt, the graphics memory is freed when the application loses focus, though with large memory allocations, this can take some time. And if an application that requires large amounts of data is started when the Qt application is running, there is a possibility that GOOM starts sending shut-down events, effectively closing the Qt application.

The shut-down event sent is `EEventUser` type, and the first `EventData` is `EApaSystemEventShutdown` and the second item in `EventData` is then the `KGoomMemoryLowEvent`.

If this event is not handled, the application will exit immediately. With Qt applications since the memory is freed automatically, only thing you need to do is then to stop this event. This can be done by implementing `symbianEventFilter()` function, and returning true for this event.

```
bool MyApplication::symbianEventFilter( const QSymbianEvent* symbianEvent ) {
  const TWsEvent *event = symbianEvent->windowServerEvent();

  if( !event ) {
    return false;
  }

  switch( event->Type() ) {
    case EEventUser: {

          TApaSystemEvent* eventData = reinterpret_cast<TApaSystemEvent*>(event-
>EventData());
          if ((*eventData) == EApaSystemEventShutdown)
          {
                eventData++;
                if ((*eventData) == KGoomMemoryLowEvent)
                {

                    return true;
                }
           }
```


(C) Copyright Nokia 2014. All rights reserved.

```
        break;
    }
  }

  return false;
}
```

- Full example for the GOOM event handling: GoomTest.zip. If using the Qt SDK to compile this example, it might be necessary to add libraries for GOOM handling to the embedded Symbian SDK. Extract the contents of this file: File:GoomForQtSDK.zip to C:\QtSDK\Symbian\SDKs\Symbian3Qt472\epoc32 (adapt this path to the Qt/Symbian SDK version you are using when compiling).

Note, that if the application is using some APIs (such as Open-GL) inside the Qt application, these parts might not be automatically handled, and the application logic should take care of handling the Graphics memory for these items.

For additional guidance on how to handle GOOM from a Qt (Quick) application: QCamera, MirrorHouse

GOOM event handling with Symbian Open GL

With Symbian Open GL the graphics memory is allocated when Open GL elements are loaded, and the memory is not freed automatically. Thus each Open GL application should take care of releasing the graphics memory by themselves. Easiest way to handle the graphics memory with Open GL application would be to mimic the Qt application behavior, i.e. to free all resources when application loses focus, and then re-allocate the resource when application gains focus.

In this way the **WSERV** GOOM event could be handled exactly the same way. However, if the memory is not freed and the GOOM low memory event is stopped, then other applications might not work correctly. For example camera application might fail to fully initialize, and it will ask the user to manually restart the device.

Thus if the application does not wish to free the resources every time it goes in to background, it must at least free them once it gets the GOOM low memory event, or otherwise it should not handle the event at all, and simply let the event to close the application.

- Full example on Open-GL GOOM event handling: OpenGL Slideshow GOOM.zip

GOOM event handling with pure Symbian

Normal Symbian applications are not using the GPU by default, thus they are also not receiving any GOOM events and thus they are not required to handle the GOOM events in any ways.

Reserving memory with GOOM

When application would require Graphics memory (generally more than1-2Mb), it should use `RequestFreeMemory()` function to inform GOOM of the total amount of memory required.

This would allow the GOOM to free some of the Graphics memory and thus make it more likely that the application would succeed on reserving the memory.

Note: The memory freeing might take some time, as well as when the `RequestFreeMemory()` return `KErrNoMemory`, it might be a good idea to try couple of times, just to allow GOOM to process the memory freeing actions.

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.