Archived:Creating a condition-based schedule with RScheduler

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*}}.

Overview

This code snippet shows how to create a condition-based schedule using the class `RScheduler`. One condition-based schedule can contain one or more `TTaskSchedulerCondition` schedule entries. An entry contains criteria based on the Publish and Subscribe variable and when this variable has a certain value, the Task Scheduler server launches scheduled tasks. The user can also give a default runtime for the tasks to run regardless of the condition variable values.

In this example, a condition-based schedule with one task is created. After that, the Symbian Publish and Subscribe mechanism is used to trigger and execute the scheduled task.

This snippet can be self-signed.

MMP file

The following libraries are required:

```
LIBRARY schsvr.lib
```

Preconditions

ExampleTaskHandler.exe must be created before this code snippet can be executed. See Creating and registering a task handler with RScheduler

Resource files

.rss

```
RESOURCE MENU_PANE r_schedulerexample_menu
    {
    items =
        {
         //...
        MENU_ITEM {command = ECreateConditionExample;  txt = "CreateConditionExample";},
         //...
        MENU_ITEM {command = EAknSoftkeyExit;          txt = "Exit";}
        };
    }
```

.hrh

```
enum TSchedulerExampleIds
    {
    //...
    ECreateConditionExample,
     //...
    };
```

Header file

```
#ifndef __SCHEDULEREXAMPLEAPPUI_H__
#define __SCHEDULEREXAMPLEAPPUI_H__

#include <csch_cli.h> // RScheduler
#include <schinfo.h>  // TSchedulerItemRef, TTaskInfo...

class CSchedulerExampleAppUi : public CAknAppUi
    {
    //...
public:
    void HandleCommandL(TInt aCommand);
    //...
private:
    void CreateConditionVariableL();

    void SetConditionVariableL(TInt aKeyValue);
```


(C) Copyright Nokia 2013. All rights reserved.

```
    void DeleteConditionVariableL();

    TInt CreateConditionScheduleWithTaskL(TSchedulerItemRef& aSchedulerItem,
                                          RScheduler& aScheduler,
                                          const TUid& aConditionCategory,
                                          TUint aConditionKey,
                                          const TTsTime& aDefaultRunTime,
                                          TInt& aNewTaskId);

    //...
private:

    RScheduler iScheduler;

    TSchedulerItemRef iConditionScheduleHandle;
    };

#endif // __SCHEDULEREXAMPLEAPPUI_H__
```

Source file

```
#include <e32property.h>

const TUid KExampleUid = TUid::Uid(0xE0060BBB); //UID3 from .mmp file
const TInt KExampleKey = 1;

void CSchedulerExampleAppUi::ConstructL()
    {
    //...
    User::LeaveIfError(iScheduler.Connect());

    _LIT(KExampleTaskHandlerExe, "ExampleTaskHandler.exe");
    TFileName exampleHandler(KExampleTaskHandlerExe);

    User::LeaveIfError(iScheduler.Register(exampleHandler, CActive::EPriorityStandard));
    }

CSchedulerExampleAppUi::~CSchedulerExampleAppUi()
    {

    //...
    iScheduler.Close();

    DeleteConditionVariableL();
    }

void CSchedulerExampleAppUi::HandleCommandL(TInt aCommand)
  {
  TBuf<100> Text1; //first line of dialog text
  TBuf<100> Text2; //second line of dialog text

  switch(aCommand)
    {
    case EEikCmdExit:
    case EAknSoftkeyExit:
      Exit();
      break;
//...
    case ECreateConditionExample:
      {
      TInt err(KErrNone);

      //create condition variable for condition-based schedule
      CreateConditionVariableL();

      TSchedulerItemRef conditionScheduleHandle;

      TTsTime defaultTime;
      TTime time;
      time.HomeTime();
      time = time.operator+(TTimeIntervalMonths(1));
      defaultTime.SetLocalTime( time );

      TInt newConditionTaskId=-1;
      err = CreateConditionScheduleWithTaskL(iConditionScheduleHandle,
 iScheduler,  KExampleUid, KExampleKey, defaultTime, newConditionTaskId );

      if(err == KErrNone)
        {
        Text1.Append(_L("ConditionTask:"));
        Text1.AppendNum(newConditionTaskId);
        Text1.Append(_L(" Scheduled"));

        CEikonEnv::Static()->InfoWinL(Text1, _L(""));

        //Set condition variable and trigger task scheduler to execute the example task
        SetConditionVariableL(1);
        }
      else
        {
        //Creating condition schedule failed, do something...
         }
       }
     break;

   default:
     //Panic(ESchedulerExampleUi);
     break;
     }
  }


void CSchedulerExampleAppUi::CreateConditionVariableL()
  {
  TInt ret = RProperty::Define(KExampleUid, KExampleKey, RProperty::EInt);

  if (ret != KErrAlreadyExists)
    {
    User::LeaveIfError(ret);
    }
  }
```


(C) Copyright Nokia 2013. All rights reserved.

```
void CSchedulerExampleAppUi::SetConditionVariableL(TInt aKeyValue)
  {
  TInt ret = RProperty::Set(KExampleUid, KExampleKey, aKeyValue);

  User::LeaveIfError(ret);
  }

void CSchedulerExampleAppUi::DeleteConditionVariableL()
  {
  TInt ret = RProperty::Delete(KExampleUid, KExampleKey);

  if (ret != KErrNotFound)
    {
    User::LeaveIfError(ret);
    }
  }

//create a condition-based schedule with one example task
TInt CSchedulerExampleAppUi::CreateConditionScheduleWithTaskL(
                  TSchedulerItemRef& aSchedulerItem,
                  RScheduler& aScheduler,
                  const TUid& aConditionCategory,
                  TUint aConditionKey,
                  const TTsTime& aDefaultRunTime,
                  TInt& aNewTaskId)
  {
  TInt ret(KErrNone);

  aSchedulerItem.iName = _L("Condition Schedule Example");

  CArrayFixFlat<TTaskSchedulerCondition>* conditions =
new (ELeave) CArrayFixFlat<TTaskSchedulerCondition>(1);

  CleanupStack::PushL(conditions);
  TTaskSchedulerCondition exampleCondition;
  exampleCondition.iCategory = aConditionCategory;
  exampleCondition.iKey    = aConditionKey;
  exampleCondition.iState  = 1;
  exampleCondition.iType  = TTaskSchedulerCondition::EEquals;
  conditions->AppendL(exampleCondition);

  ret = aScheduler.CreatePersistentSchedule(aSchedulerItem,
                                 *conditions, aDefaultRunTime);

  if (ret == KErrNone)
    {
    TTaskInfo taskInfo;
    taskInfo.iTaskId = 0;
    taskInfo.iName = _L("ExampleTask3");
    taskInfo.iPriority = 2;
    taskInfo.iRepeat = 0;
    HBufC* data = _L("ExampleData3").AllocLC();

    // schedule the example task
    ret = aScheduler.ScheduleTask(taskInfo, *data, aSchedulerItem.iHandle);

    aNewTaskId = taskInfo.iTaskId;

    CleanupStack::PopAndDestroy(); // data
    }
  else
    {
    //Creating persistent schedule failed, do something...
    }

  CleanupStack::PopAndDestroy(); // conditions

  return ret;
  }
```

Postconditions

The condition-based schedule with one example task has been created and the task execution is triggered using the Publish and Subscribe mechanism.

See also

- Creating and registering a task handler with RScheduler
- Archived:Creating persistent and transient schedules with RScheduler
- Archived:Getting schedule and task info using RScheduler
- Archived:Getting schedule and task count using RScheduler
- Archived:Editing a schedule using RScheduler
- Archived:Deleting schedules and tasks using RScheduler

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.