



ID	CS000987	Creation date	May 28, 2008
Platform	S60 3rd Edition, FP1	Tested on devices	Nokia N93
Category	Symbian C++	Subcategory	Files/Data

Keywords (APIs, classes, methods, functions): RScheduler, TSchedulerItemRef, TScheduleEntryInfo2, TTaskInfo, TTsTime, RScheduler::Connect(), RScheduler::Register(), RScheduler::Close(), RScheduler::CreatePersistentSchedule(), RScheduler::ScheduleTask(), TTsTime::SetLocalTime()

Overview

This code snippet shows how to create persistent and transient time-based schedules using the class `RScheduler`. The persistent schedules have their task data saved to the disk and the Task Scheduler server reads this data back into the memory whenever the device reboots. The transient schedules are deleted automatically when tasks are finished and their task data is not saved the same way. One time-based schedule can contain one or more schedule entries. The type of interval used by a schedule entry can be `EHourly`, `EDaily`, `EMonthly`, or `EYearly`.

In this example persistent and transient time-based schedules with one daily task are created. The example tasks are executed one minute after creation.

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 [CS000986 - Creating and registering a task handler with RScheduler](#)

Resource files

.RSS

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

.hrh

```
enum TSchedulerExampleIds
{
    ECreateTimeBasedExample = 1,
    //...
};
```

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:
    TInt CreatePersistentDailyScheduleWithTaskL(TSchedulerItemRef& aSchedulerItem,
        RScheduler& aScheduler,
        const TTsTime& aStartTime, TInt& aNewTaskId);

    TInt CreateTransientDailyScheduleWithTaskL(TSchedulerItemRef& aSchedulerItem,
        RScheduler& aScheduler,
        const TTsTime& aStartTime, TInt& aNewTaskId);

    //...
private:

    RScheduler iScheduler;

    TSchedulerItemRef iTransientScheduleHandle;
    TSchedulerItemRef iPersistentScheduleHandle;
};
```

```
#endif // __SCHEDULEREXAMPLEAPPUI_H__
```

Source file

```
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();
}

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 ECreateTimeBasedExample:
    {
        TTsTime startTime;
        TTime time;
        time.HomeTime();
        time = time.operator+(TTimeIntervalMinutes(1));
        startTime.SetLocalTime( time );

        // create a persistent daily schedule with one example task
        TInt newPersistentTaskId=-1;
        TSchedulerItemRef persistentScheduleHandle;
        User::LeaveIfError(CreatePersistentDailyScheduleWithTaskL(iPersistentScheduleHandle,
iScheduler, startTime, newPersistentTaskId));

        // create a transient daily schedule with one example task
        TInt newTransientTaskId=-1;
        TSchedulerItemRef transientScheduleHandle;
        User::LeaveIfError(CreateTransientDailyScheduleWithTaskL(iTransientScheduleHandle,
iScheduler, startTime, newTransientTaskId));

        Text1.Append(_L("PersistentTask:"));
        Text1.AppendNum(newPersistentTaskId);
        Text1.Append(_L(" Scheduled"));
    }
}
```

CS000987_-_Creating_persistent_and_transient_schedules_with_RScheduler

```
Text2.Append(_L("TransientTask:"));
Text2.AppendNum(newTransientTaskId);
Text2.Append(_L(" Scheduled"));

CEikonEnv::Static()->InfoWinL(Text1, Text2);
}
break;

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

//create a persistent daily schedule with one example task
TInt CSchedulerExampleAppUi::CreatePersistentDailyScheduleWithTaskL(
    TSchedulerItemRef& aSchedulerItem,
    RScheduler& aScheduler,
    const TTsTime& aStartTime,
    TInt& aNewTaskId)
{
    TInt ret(KErrNone);

    CArrayFixFlat<TScheduleEntryInfo2>* entries =
new (ELeave) CArrayFixFlat<TScheduleEntryInfo2>(1);
    CleanupStack::PushL(entries);

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

    //IMPORT_C TScheduleEntryInfo2(const TTsTime &aStartTime,
//TIntervalType aIntervalType, TInt aInterval,
//TTimeIntervalMinutes aValidityPeriod);
    TScheduleEntryInfo2 exampleEntry(aStartTime,EDaily, 1, 10);

    entries->AppendL(exampleEntry);
    ret = aScheduler.CreatePersistentSchedule(aSchedulerItem, *entries);

    if(ret == KErrNone)
    {
        //create the example task
        TTaskInfo taskInfo;
        taskInfo.iName = _L("ExampleTask1");
        taskInfo.iTaskId = 0;
        taskInfo.iPriority = 2;
        taskInfo.iRepeat = 1; // -1 repeat until explicitly deleted
        HBufC* data = _L("ExampleData1").AllocLC();

        ret = aScheduler.ScheduleTask(taskInfo, *data, aSchedulerItem.iHandle);

        aNewTaskId = taskInfo.iTaskId;

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

    CleanupStack::PopAndDestroy(); // entries
}
```

CS000987_-_Creating_persistent_and_transient_schedules_with_RScheduler

```
return ret;
}

//create a transient daily schedule with one example task
TInt CSchedulerExampleAppUi::CreateTransientDailyScheduleWithTaskL(
    TSchedulerItemRef& aSchedulerItem,
    RScheduler& aScheduler,
    const TTsTime& aStartTime,
    TInt& aNewTaskId)
{
    TInt ret(KErrNone);

    CArrayFixFlat<TScheduleEntryInfo2>* entries =
    new (ELeave) CArrayFixFlat<TScheduleEntryInfo2>(1);
    CleanupStack::PushL(entries);

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

    //IMPORT_C TScheduleEntryInfo2(const TTsTime &aStartTime,
    //TIntervalType aIntervalType, TInt aInterval,
    //TTimeIntervalMinutes aValidityPeriod);

    TScheduleEntryInfo2 exampleEntry(aStartTime,EDaily, 1, 10);

    entries->AppendL(exampleEntry);

    //create the example task
    TTaskInfo taskInfo;
    taskInfo.iName = _L("ExampleTask2");
    taskInfo.iTaskId = 0;
    taskInfo.iPriority = 2;
    taskInfo.iRepeat = 1; //-1 repeat until explicitly deleted
    HBufC* data = _L("ExampleData2").AllocLC();

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

    CleanupStack::PopAndDestroy(2); // data, entries

    aNewTaskId = taskInfo.iTaskId;

    return ret;
}
```

Postconditions

Two time-based schedules (persistent and transient) with a daily task are created and executed after one minute of the creation.

See also

- [CS000986 - Creating and registering a task handler with RScheduler](#)
- [CS000988 - Creating a condition-based schedule with RScheduler](#)

CS000987_-_Creating_persistent_and_transient_schedules_with_RScheduler

- [CS000989 - Getting schedule and task info using RScheduler](#)
- [CS000990 - Getting schedule and task count using RScheduler](#)
- [CS000991 - Editing a schedule using RScheduler](#)
- [CS000992 - Deleting schedules and tasks using RScheduler](#)