

**"Danger, Will Robinson!"**

Please note that the recognizer framework was not designed for the purpose of providing an autostart mechanism for 3rd party applications. It provides no fail-safe mechanism and will render the phone useless if an defective recognizer is installed. This is a **hack** and should be used only if the autostart feature is critical to your application and no other solutions is available for implementing it. In fact, Symbian provides the Start-On-Boot Registration API plug-in that enables the autostart feature for both the S60 2nd Edition and UIQ platforms.

Autostart applications in S60 2nd Edition phones. This is taken from mika raento's web-site,for information check this [link](#).

## Contents

- [1 In .mmp file](#)
- [2 Header File](#)
- [3 In .cpp File](#)
- [4 In .Pkg file](#)
- [5 Tips](#)

### In .mmp file

Create resource file (.mdl) to start application on bootup.

```
target CL_AUTOSTART.MDL
targettype MDL
targetpath \system\recogs\
uid 0x10003A19 0x09A770B5
SOURCEPATH ..\src
SOURCE cl_autostart.cpp
LANG SC
USERINCLUDE ..\inc
SYSTEMINCLUDE . \epoc32\include \epoc32\include\libc
library EUSER.LIB APMIME.LIB efsrv.lib apparc.lib apgrfx.lib
LIBRARY efile.lib
```

### Header File

```
#if !defined(__CL_AUTOSTART_H__)
#define __CL_AUTOSTART_H__

#if !defined(__APMREC_H__)
```

## Autostart\_applications\_on\_S60\_2nd\_Edition\_phones

```
#include <apmrec.h>
#endif

#include <apgcli.h>
#include <f32file.h>
#include <apacmdl.h>
#include <e32std.h>
#include <apmstd.h>

class CclAutostart : public CApaDataRecognizerType
{
public: // from CApaDataRecognizerType
    CclAutostart()
        : TUint PreferredBufSize
        : TDataType SupportedDataTypeL(TInt /*aIndex*/ const);
    static void StartThread();
    static TInt StartAppThreadFunction(TAny* aParam);
    static void StartAppThreadFunctionL();

private: // from CApaDataRecognizerType
    void DoRecognizeL(const TDesC& aName, const TDesC8& aBuffer);
};

#endif
```

## In .cpp File

We need to write a recognizer to autostart third-party applications on the Series 60 version 1. A recognizer is basically a dll (containing a class) meant to handle certain kinds of documents or files. Recognizers are loaded at start up so we can put code in one that starts the application(s) we want.

```
#include <apmrec.h>
#include <apmstd.h>
#include "cl_autostart.h"

const TUid KUidemAclAutostart={0x09A770B5};

CclAutostart::CclAutostart()
:CApaDataRecognizerType(KUidemAclAutostart, CApaDataRecognizerType::ENormal)
{
    iCountDataTypes
}

TUint CclAutostart::PreferredBufSize()
{
    // no buffer recognition yet
    return 0;
}

TDataType CclAutostart::SupportedDataTypeL(TInt /*aIndex*/) const
{
    return TDataType();
}

void CclAutostart::DoRecognizeL(const TDesC& /*aName*/, const TDesC8&
/*aBuffer*/)
{
    // this function is never called
}
```

## Autostart\_applications\_on\_S60\_2nd\_Edition\_phones

```

}

void CclAutostart::StartThread()
{
    TErrNone;

//create a new thread for starting our application
    RThreadAppThread;
    startAppThread();

    ::LeaveIfError( res = startAppThread->Create(
("Autostart_starter"),
        ::StartAppThreadFunction,
        KDefaultStackSize,
        KMinHeapSize,
        KMinHeapSize,
NULL,
        ) );EOwnerThread

    startAppThreadPriority(EPriorityNormal/*EPriorityLess*/);

    startAppThread();

    startAppThread();
}

TInt CclAutostart::StartAppThreadFunction(TAny* /*aParam*/)
{
//wait 5 seconds...
    RTimer/timer asynchronous timer and ...
    TRequestStatus timerStatus its associated request status
    CreateLocal(); // Always created for this thread.
// get current time (microseconds since OAD nominal Gregorian)
    time
time.HomeTime();
// add ten seconds to the time
    TTimeIntervalSeconds timeInterval(1000000);
time += timeIntervalSeconds;
// issue and wait
    At(timerStatus,time);
    ::WaitForRequest(timerStatus);

// create a TRAP cleanup
    CTrapCleanup = CTrapCleanup::New();
    TInt err
if( cleanup == NULL )
{
    = KErrNoMemory;
}
else
{
    ( err, StartAppThreadFunctionL() );
}
delete cleanup;

if (err!=KErrNone)

```

## Autostart\_applications\_on\_S60\_2nd\_Edition\_phones

```

        ::Panic(_L("Autostart"), err);

return err;
}

void CclAutostart::StartAppThreadFunctionL()
{
#ifdef __WINS__
//This is the uid of the starter application, which you want to autostart.
const TUid starter_uid= { 0x05CCC0B0 };

        RApaLsSession ls

        ::LeaveIfError(ls.Connect());

        CleanupClosePushL

        (file_n,IT"); // dummy

        TThreadId dummy

        ::LeaveIfError( ls.StartDocument(file_n, starter_uid, dummy) );

        CleanupStackAndDestroy();
#else
//Replace this starter.app with the app which you want to autostart.
        TFileName fnAppPath(_L("system\\apps\\starter.app"));

        RProcess server

        CleanupClosePushL;

        ::LeaveIfError(server.Create(fnAppPath, _L("")));

        Resume();

        CleanupStackAndDestroy();
#endif
}

```

some devices above fun wont work use below code for 2nd edition devices eg:6600

```

void CclAutostart::StartAppThreadFunctionL()
{
// absolute file path to our application
TFileName fnAppPath = _L("\\system\\apps\\Myapp\\Myapp.app");

RFs fsSession; //file server session
User::LeaveIfError(fsSession.Connect());
CleanupClosePushL(fsSession);
TFindFile findFile( fsSession );

User::LeaveIfError( findFile.FindByDir(fnAppPath, KNullDesC) );

```

## Autostart\_applications\_on\_S60\_2nd\_Edition\_phones

```
CApaCommandLine* cmdLine = CApaCommandLine::NewLC();
cmdLine->SetLibraryNameL( findFile.File() );
cmdLine->SetCommandL( EApaCommandOpen );

RApaLsSession ls;
User::LeaveIfError(ls.Connect());
CleanupClosePushL(ls);

User::LeaveIfError( ls.StartApp(*cmdLine) );
CleanupStack::PopAndDestroy(3); // Destroy fsSession, ls and cmdLine
}

EXPORT_C CApaDataRecognizerType* CreateRecognizer()
{
    CApaDataRecognizerType= new CclAutostart();

    //start thread for our application
    CclAutostartThread();
return thing;
}

// DLL entry point
GLDEF_C TInt E32Dll(TDllReason /*aReason*/)
{
return KErrNone;
}
```

### In .Pkg file

Attach .mdl file to package.

```
"cl_autostart.mdl"          -"!:\system\recogs\cl_autostart.mdl"
```

### Tips

- It is always recommended that the time interval to autostart the app should be equal to or more than 10 seconds. This is because in slower phones for e.g. Nokia 6600 or Nokia 3230 the boot process after loading a number of applications becomes very slow. If the autostart application is to load during this time the boot becomes much slower.
- The above code starts the application on phone boot up and the application remains in the foreground. The application staying in the foreground may not be the actual behavior as wanted. You may want that the application should autostart and stay in the background. This can be achieved for e.g. by using some flag file, but the launch cannot be hidden. The application will come in the foreground and then go to the background.