Archived:Carbide.vs Criando e instalando projetos no hardware

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*}}.

Following the withdrawal of Carbide.vs, Nokia Developer no longer supports the Microsoft Visual Studio IDE for Symbian C++ development.

Introdução

Este artigo provê um guia para iniciantes de como criar projetos com o Carbide.vs (2.0.1) e depois instalar a aplicação no dispositivo. O SDK usado neste artigo é o S60 3rd Edition MR SDK [1].

Criando um projeto

Passo 1

Para se criar um projeto utilizando o Carbide.vs, é preciso selecionar o menu **File > New > Project** e então escolher "Symbian OS Project". É importante que o caminho utilizado para o projeto não contenha espaços, ou o processo de compilação irá falhar (a propósito, o wizard não aceita que se use espaços no nome).

Passo 2

A seguir, a janela "Create new Symbian OS Project" será exibida. Nessa janela é possível escolher que tipo de projeto será gerado, entre os modelos pré-definidos.

O item "Project Type" indica se a plataforma será o Symbian 9 ou alguma versão anterior ("Classic"). Para aparelhos baseados em S60 3rd Ed., é necessário escolher Symbian 9. A seção "Project Template" contém os modelos de projetos pré-definidos. Para uma aplicação normal, a opção correspondente é "S60 3rd Ed. Hello World Application".

Na parte de baixo da janela, é possível escolher para quais SDKs o projeto poderá ser compilado. Para cada SDK, pode-se escolher também quais configurações serão utilizadas. A configuração "WINSCW" indica que a aplicação será construída para ser executada no emulador. Já a configuração "GCCE" é a responsável por construir a aplicação para ser executada no aparelho, usando o compilador gcc. Caso existam outros SDKs instalados na máquina, é possível habilitá-los ou não para serem escolhidos através do botão "Enable/Disable SDKs". Por favor, escolha ambas as configurações.

Passo 3

Na janela seguinte, é possível configurar características do projeto. Na seção "Basic Settings", é possível indicar o nome base da aplicação (que será usado pelo assistente para nomear as classes) e também o UID (Universal Identification Number) da aplicação. No Symbian OS, toda aplicação deve possuir um UID (32 bits) e este deve ser único no sistema. Para se obter um UID válido, é necessário requisitar ao consórcio Symbian através do endereço uid@symbiandevnet.com. Entretanto, para o processo de desenvolvimento é possível utilizar qualquer identificador entre 0x01000000 e 0x0fffffff. No caso do Carbide.vs, o assistente irá gerar um identificador automaticamente.

Na seção "Project Directories", é possível especificar quais serão os diretórios usados no projeto. Cada diretório possui uma funcionalidade. Os diretórios gerados pelo Carbide.vs, por padrão, são:

- data;
- gfx;
- group;
- inc;
- sis;
- src;
- tsrc.

O diretório data guarda guarda arquivos de recursos (.rss) que são usados para definir elementos da interface gráfica e strings de texto.

(C) Copyright Nokia 2014. All rights reserved.

O diretório gfx armazena arquivos de imagens (ícones) da aplicação.

O diretório group guarda o arquivo bld.inf e um arquivo .mmp. O arquivo bld.inf é obrigatório e contém instruções para o processo de construção da aplicação. O arquivo .mmp corresponde a um arquivo com intruções de compilação (makefile) do projeto.

O diretório inc armazena os arquivos .h (cabeçalhos) utilizados no projeto. Os arquivos fontes (.c, .cpp) são armazenados no diretório src. No diretório tsrc, são armazenados fontes para classes de teste.

O diretório sis contém um arquivo .pkg que é usado na geração do arquivo de instalação .sis, para um determinado aparelho. Os arquivos .sis são armazenados nesse diretório.

A seção "Other settings" permite que se altere textos que aparecem nos comentários de código e também a mensagem que a aplicação exibe quando se seleciona o primeiro item do seu menu.

Resultado

O assistente irá criar quatro classes (para um nome base igual a HelloSymbian): CHelloSymbianApplication, CHelloSymbianDocument, CHelloSymbianAppView e CHelloSymbianAppUi. Essas classes são derivadas das classes básicas do framework Avkon, que é o framework de interface gráfica da plataforma S60. Esse framework segue o padrão MVC – Model, View, Controller [2]. Por isso, uma aplicação simples possui os seguintes componentes:

- Application View (CHelloSymbianAppView): Representa a janela principal da aplicação, que contém todos os outros controles;
- Application UI (CHelloSymbianAppUi): Corresponde ao Controller do MVC, sendo responsável por criar a View e tratar eventos disparados pelos controles de interface;
- Application Document (CHelloSymbianDocument): Responsável por armazenar dados de aplicação e por criar o Application UI;
- Application (CHelloSymbianApplication): Representa a aplicação principal, que cria o Documento. Esse componente também é responsável por determinar o UID da aplicação.

Importando projetos

Muitos exemplos de projetos em Symbian OS são preparados para serem compilados e executados diretamente pela linha de comando, o que pode tornar o processo complicado. Dessa forma, é comum que as IDEs ofereçam um assistente para se importar projetos.

No Carbide.vs, esse assistente poderá ser invocado através do menu **File > Import Symbian Project**.

No campo "Project File" é necessário especificar o arquivo bld.inf ou .mmp do projeto que deve ser importado. O campo "Project Type" indica qual é o tipo de projeto a ser importado, e o assistente tentará deduzir o tipo baseado no arquivo informado.

A próxima janela pode possuir uma seção ("General settings"), caso um arquivo .mmp tenha sido especificado, ou duas seções ("Solution Settings" e "Project Settings") caso tenha sido usado o arquivo bld.inf.

Para arquivos bld.inf, os SDKs e suas configurações a serem usadas podem ser escolhidas na seção "Solution Settings". Modificar o campo "Solution Root" não possui efeito já que o assistente salva os arquivos de projeto no diretório raiz do projeto original. Na seção "Project Settings" é possível escolher o nome do arquivo de projeto a ser salvo. Essa seção contém uma entrada para cada projeto existente no arquivo bld.inf.

A seção "General Settings", combina elementos das seções anteriores.

Especificando bibliotecas

Algumas funcionalidades da API do Symbian são implementadas em DLLs. Dessa forma, é preciso especificar os arquivos de ligação (.lib) para o compilador.

No Carbide.vs, isso pode ser feito acessand**o-se o menu Project > <Nome do Projeto**> properties. A seguir, acessar a pasta **Linker > Input**. Na entrada **Additional Dependencies**, colocar o nome da biblioteca requerida.

Esse procedimento funciona quando se está gerando um executável para o compilador. A configuração que gera o executável para o aparelho utiliza as mesmas bibliotecas especificadas na outra configuração.

Distribuição da aplicação

Para instalar a aplicação no telefone celular, é preciso primeiro gerar o pacote .sis correspondente. Para isso, é necessário trocar a configuração de compilação para que o executável seja gerado para o aparelho. No Carbide.vs, isso pode ser feito ao se

(C) Copyright Nokia 2014. All rights reserved.

acessar o menu **Project > SDK Configuration Manager** e escolher a opção **GCCE**. A seguir, é necessário acessar o menu **Build > Configuration Manager** e escolher a configuração apropriada para o aparelho (**GCCE**, no caso).

A partir da versão 9.0 do Symbian, é necessário assinar digitalmente o arquivo .sis para que seja possível instalá-lo no aparelho. Dependendo do tipo de funcionalidade requerida pela aplicação, pode ser precido adquirir um certificado diretamente com o consórcio Symbian. Entretanto, se a aplicação não fizer uso de funcionalidades restritas, pode-se simplesmente "auto-assinar" ("self-sign") a aplicação. Para isso, é preciso criar um certificado digital e uma chave privada.

O SDK do Symbian possui ferramentas para realizar essa tarefa: makekeys e signsis. A ferramenta makekeys pode ser utlizada para se criar uma chave privada e um certificado digital. Exemplo de uso:

```
makekeys -cert [-password <password>] [-len <key-length>] -dname <distinguised-name-
string> <private-key-file> <public-key-cert>

makekeys -req [-password <password>] -dname <distinguised-name-string> <private-key-
file> <public-key-cert> <cert-request-file>

makekeys -view <certificate-file>
```

A seguir, seque um exemplo de criação de uma chave privada chamada mykey.key e um certificado mycert.cer.

```
makekeys -cert -password yourpassword -len 2048 -dname "CN=Test User OU=Development
OR=Company CO=FI EM=test@company.com" mykey.key mycert.cer
```

Para assinar o certificado, o SDK possui o programa signsis (que normalmente está no diretório Epoc32\tools do SDK.

No Carbide.vs, pode-se usar o seguinte procedimento: No menu **Tools > Manage Certificates**, o certificado criado pelo makekeys pode ser incluído na lista de certificados, o que deve ser feito primeiro. Para se assinar o .sis, é necessário clicar com o botão direito sobre o arquivo .sis no "Solution explorer" do projeto, e escolher a opção "**SignSis**". Quando essa opção for escolhida, uma janela será aberta e pedirá que se escolha um certificado, a partir da lista de certificados cadastrados. Escolhendo-se um certificado, o pacote .sis original será assinado e um novo arquivo .sis será gerado e armazenado no mesmo diretório do original.

De posse do arquivo .sis assinado, é necessário instalá-lo no dispositivo, o que pode ser realizado através do Nokia PC Suite ou copiando diretamente para o dispositivo.

Na primeira opção, se o Nokia PC Suite estiver instalado, é possível clicar no arquivo diretamente na IDE para que o Nokia PC Suite seja invocado. Nesse caso, o Nokia PC Suite irá perguntar se o aplicativo deve ser instalado. Caso assim se deseje, o telefone será contactado e o Nokia PC Suite irá orientar para que se complete a operação no aparelho.

Na segunda opção, o telefone deverá estar funcionamento em modo de transferência de dados. Nesse caso, é preciso copiar o arquivo .sis para o telefone, através do Windows Explorer ou do Phone Browser do Nokia PC Suite, por exemplo. Uma vez copiado, é necessário procurar o arquivo no telefone, e pedir para instalá-lo.

Pode acontecer de o telefone se recusar a instalar a aplicação e informar que o certificado está expirado. Quando isso acontece, a data e hora atual do aparelho estão fora da validade do certificado. Nesse caso é preciso alterar a data e hora do telefone, ou usar outro certificado cuja validade esteja de acordo com a configuração do aparelho.

Isso é particularmente importante quando se importa para uma IDE um projeto que foi gerado em outra ferramenta, e que já foi executado alguma vez pelo emulador.

Como assinar automaticamente o pacote

No Carbide.vs, é possível configurar o ambiente para gerar o arquivo .sis já assinado. Para isso é preciso acessar o menu **Project > Carbide.vs properties**. Uma janela será aberta, e na opção "Security properties" é possível escolher o certificado padrão a ser usado marcar a opção para assinar automaticamente o arquivo.

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.