MMP file

Syntax overview

A .mmp project definition file specifies the properties of a project in a platform and compiler independent way. The makmake tool converts project definition files into makefiles for particular platforms. The abld.bat tool wraps calls to makmake, and can be more convenient to use than makmake directly.

A project definition file has extension `.mmp`.

Note that:

- Each statement occupies a single line.
- Use the C++ style comment syntax for comments.
- A trailing backslash is used to indicate a line continuation. Therefore, specify directories without their trailing backslash, for example: Do `SYSTEMINCLUDE \epoc32\include`, rather than `SYSTEMINCLUDE \epoc32\include\`

Example project

TARGETTYPE

From v9.0, a GUI application is an exe. This is specified with the `targettype` statement:

```
TARGETTYPE exe
```

The target filename's extension can be either **.app** or **.exe**. This is specified using a `target` statement:

```
TARGET <appname>.exe
```

UID

Specify the UID for the application in a `uid` statement:

```
UID 0x100039CE <UID3>
```

The UID2 value `0x100039CE` should be used for all applications. The UID3 value is the unique identifier for the particular application.

Languages

Specify the languages that the application is localised for in the **.mmp** file:

```
LANG 01 02 03
```

Stack size

By default, processes have a stack size of *8K*. This can be insufficient for some applications. To increase the size, use an `epocstacksize` statement:

```
// set stack to 20K
epocstacksize 0x5000
```

UI resource

Specify the application's UI resource file using a `start resource` statement:

```
START RESOURCE  <appname>.RSS
HEADER
TARGETPATH      \Resource\Apps
END
```

The resource should be built into the `\Resource\Apps` directory. This is specified using the `TARGETPATH` part of the statement.

The header line tells the resource compiler to produce a file `\epoc32\include\<appname>.rsg`, which defines macro constants through which C++ programs can refer to resource structures.

Caption/icon resource

An application can specify localisable captions and icons to display on the shell using a `LOCALISABLE_APP_INFO` resource, either in the UI resource file or in a separate resource file.

(C) Copyright Nokia 2013. All rights reserved.

Registration file

Specify the application's registration file using another `start resource` statement:

```
START RESOURCE  <appname>_reg.rss
TARGETPATH      \private\10003a3f\apps
END
```

The registration file must be built to the private system directory `\private\10003a3f\apps` directory.

Note that for security reasons a registration file cannot be installed directly to this location on a target device. When creating a PKG file (software install package file) for an application, the registration file must be installed to the `\private\10003a3f\import\apps\` directory.

Icons

A bitmap file containing icons for an application can be specified using a `start bitmap` statement:

```
START BITMAP    <appname>.mbm
TARGETPATH      \Resource\Apps
SOURCE          <color-depth> <source-bitmap-list>
END
```

Icon files, like the UI resource file, should be built into the `\Resource\Apps` directory.

Example

An example of a real GUI application project file is given below:

```
TARGET HelloWorld.exe
TARGETTYPE exe
UID 0x100039CE 0x10004299
VENDORID 0x70000001
epocstacksize 0x5000

SOURCEPATH .
SOURCE HelloWorld_Main.cpp
SOURCE HelloWorld_Application.cpp
SOURCE HelloWorld_Document.cpp
SOURCE HelloWorld_AppUi.cpp
SOURCE HelloWorld_AppView.cpp
USERINCLUDE .
SYSTEMINCLUDE \epoc32\include

START RESOURCE  HelloWorld.RSS
HEADER
TARGETPATH      \Resource\Apps
end

START RESOURCE  HelloWorld_reg.rss
TARGETPATH      \private\10003a3f\apps
END

START BITMAP    HelloWorld.mbm
TARGETPATH      \Resource\Apps
SOURCE          c8,1 icon24.bmp icon2m.bmp icon32.bmp icon3m.bmp
                icon48.bmp icon4m.bmp
END

LIBRARY euser.lib apparc.lib cone.lib eikcore.lib
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.