

Project Contents

Overview

Project File

Project Manager

Project Manager Tabs

Building Applications

Project Conventions

Initial Scripts

Overview

A project file lets you work with applications that are built from several scripts. You can specify which scripts are to be included in the application, which are required for development purposes only, and how the application is to be built.

While the scripts are maintained individually during development, they can be compiled into a single output script for distribution/runtime/installation purposes. Alternatively, you can specify a build script that is run to create the final application, which may consist of several scripts.

The Project Manager lets you work with project files. Run the Project Manager from menu Edit|Project Manager... or by pressing Ctrl-B.

Project files distinguish between scripts you write specifically for the project, and library scripts distributed with the system that are needed in the project. The project system automatically includes any files required by the scripts you specify.

Typically all the files in a project, apart from library files, will be stored in a single directory, for example, a subdirectory of user\projects. Within the directory will be a project file, plus the project source scripts.

The project facilities require that you adhere to various naming and coding conventions.

Project File

A project file is a script file with extension .ijp that defines the project. It is created and maintained by the Project Manager.

A project file defines the following nouns:

PRIMARYFILES	Project-specific files, required for the application
PRIMARYLIBS	Standard libraries needed for the application
DEVFILES	Project-specific files, required for development only
DEVLIBS	Standard libraries required for development only
OTHERFILES	Other files accessible in Project Manager
TARGETFILE	Target file for the application
TARGETLOCALE	Target locale
TARGETHEADER	Header comments written to target file
TARGETEXTRA	Extra code appended to target file
BUILD_OPTS	Build options (numeric list)
BUILDFILE	Special build file
NOTES	Free-form notes
WINDOWS	Project open window positions
WINDOWSTATE	Project open window state

The distinction between PRIMARY or DEVELOPMENT is that when you create an application from the project, only the primary files are used. The development files are assumed to be required for development purposes only.

FILES refer to files specific to the project, and LIBS to library files that are supplied with the J system. Library files are referenced by their short names. Note that library files stdlib.ijs, colib.ijs and winlib.ijs are assumed to be always available, and therefore do not appear in the selections for library files.

OTHERFILES is an additional list of files that you can access when using Project Manager. For example, these include the project file itself, the target file (if specified), and any other files that you want to access easily, but not load with the project.

The project system uses the jproject locale for storing the project definitions. However, the project scripts are loaded into the TARGETLOCALE or base locale if not given.

TARGETFILE is the file created when building the final application. If TARGETLOCALE is given, the locale is set after any library files are loaded. TARGETEXTRA is an optional line of code to be appended to the targetfile, typically to run the application when it is loaded. TARGETHEADER comments are lines prefixed by NB. that are added to the top of the file. Use this for the file description.

BUILD_OPTS configure the targetfile. BUILDFILE is an optional script file run when building the application.

A project file is a plain text file and can be edited directly, as long as you preserve the same names. You should not add new definitions to the file, since the Project Manager will overwrite them when you next use it. Instead, add any new definitions to other script files in the project.

Project Manager

The Project Manager lets you work with project files. Run the Project Manager from menu Edit|Project Manager... or by pressing Ctrl-B.

From the Project Manager dialog, open a project file by either:

- selecting from the Project list
- running menu File|New...
- running menu File|Recent...

The Project list shows all files with extension .ijp found in the Look In folder and its subfolders. If the project filename is the same as its folder name, for example, mygraph\mygraph.ijp, then only the filename is shown.

You should set Look In to point to the folders (directories) where you are creating J projects. For example, the default Look In folder is user\projects. You might then create projects in directories such as user\projects\work, user\projects\test etc.

You can define and modify the Look In folders by pressing the >> button.

The menu File|Recent... shows the last 10 projects that have been loaded.

When a project is open, use Load to load the project on top of existing definitions. Select menu Project|Clean Load to reset the system, clearing out existing definitions, and then load the project. Select Project|Clean Load Primaries to reset the system, and then load the project primary files only; this is useful for testing the final application. The verb loadp can also be used for this purpose.

Use menu Project|Build Options... to specify how the application is to be built. Use menu Project|Build Application to build the application.

Notes

The Notes editbox in menu File|Notes is for free-form comments. These comments are stored in the project file only, and do not appear in the final application.

Project Manager Tabs

Source

The Source tab lists the files in the project. These are the scripts in the project other than library scripts.

Files in the same directory as the project file are shown without any directory path. Other files have their directory paths.

Use the menu Options|Mark as Dev Only to mark a script as being for development only; its name is then shown followed by (d).

Required Files

If a source script depends on other scripts, for example, if it contains a line of the form

```
load 'source\util\myutil'
```

then the required file will also be shown, with a name followed by (r) or (dr).

This feature is primarily intended for library files. Typically, project source files need not require other files, since it would be simpler to add them explicitly to the source file list.

Library

The Library tab shows the library files in the project, as well as the library files that are available. Files are shown with their short name only.

As with Source files, use the menu Options|Mark as Dev Only to mark files that are for development only.

Required files have names followed by (r) or (dr). Unlike Source files, some library files are expected to require other files.

Other

The Other tab shows other files in the project, including the project file itself, the target file and build file if specified, and any other files that may be added. The project, target and build files are marked with (p), (t) and (b) respectively.

Building Applications

You build the application by clicking the Build button in the Project Manager and Build Options dialogs.

The Build Options dialog lets you set up two ways to build the application. You can use either or both.

- If the Target File is given, then it is built using the options specified.
- If the Special Build File script is given, then it is run afterwards. Use this for complex builds. Note that such scripts should not be included in the Source files, since otherwise they would be run whenever the project is loaded. They are, however, accessible from the Other tab.

Target File

The extension defaults to ijs for an ordinary script file, but you can specify ijl or ijr for a locked or runtime file.

If Load in locale is given as for example, "myloc", the script will include a line:

```
coclass 'myloc'
```

after any standard libraries are loaded, and before the project files are loaded.

If Append to file is given, it is added as the last statement in the application. Use this to run the application when it is loaded.

Include Files

You can choose whether to include or require files when building the targetfile; "include" means the scripts form part of the target file, "require" means there is a line of the form:

```
require 'filename'
```

written in the target file.

For applications that will be run when the J development system is loaded, you would typically want to require any project libraries, and either include or require project source.

For standalone files, you should require libraries only if you know that the application will be run on your own machine, and so the library files will be available. Otherwise, you should select to include all library files; in this case, any library script referenced by the project will be included in the targetfile.

Project Conventions

The project facilities require that you adhere to the following conventions:

- File extensions must be:

ordinary scripts	ijs
locked scripts	ijl
runtime scripts	ijr
project files	ijp

- Dependencies must be given in script files with lines of the form:

```
load 'dates regex strings'
```

where the verb is either `load`, `require` or `corequire`, and the files are given in a text string. Specifically, the first word on the line must be one of the allowed verbs, and this must be followed by a character string. Blank separators are ignored.

Note that where dependencies are given in the right form, the project system will automatically include the required files in the project.

There are no requirements as to project directories. However, it makes for easier searching if you group the project subdirectories in a small number of directories. For example, the recommended project directory is `user\projects`, and you might then create subdirectories such as `user\projects\mywork`, `user\projects\utils` etc. Also, you may wish to name project files with the same name as the directory, since this simplifies the display in the Look In list box.

Since directories are searched for project files when you use the Project Manager, you should avoid using directories where there are a large number of subdirectories not being used for projects. For example, you would typically not include projects under the J system subdirectory.

Initial Scripts

You can specify scripts that are read in by the standard profile when you first load J. This lets you specify files and libraries that will always be available in the development environment. The script names are stored in the system configuration file.

These scripts are automatically included in all projects, by default as development files and libraries.

Specifically:

- When you load J, the initial scripts are loaded.
- When you open any other project, the Project Manager ensures that it at least includes these scripts; if necessary adding them to the lists of development files and libraries.

Define these scripts from menu File|Initial Scripts...