The Software AG IDL Compiler

The Software AG IDL Compiler generates interface objects, skeletons and wrappers. It uses a Software AG IDL file and a template file that controls the generated output.

This chapter covers the following topics:

- Introduction
- Starting the IDL Compiler
- IDL Compiler Usage Examples
- Writing your own Wrappers and Stubs

Introduction

The IDL Compiler is used to generate stubs, skeletons and wrappers from two specific input files:

- the IDL file (extension .idl), which describes the interface between client and server.
- the template file (extension .tpl), which controls the generated output files and their contents. In principle the template describes the target programming language.

The IDL Compiler first reads through the IDL file and builds tables and structures that form an internal representation of the interface. If there is a related server mapping file, it is also read implicitly (see *Server Mapping Files for Natural*). The IDL Compiler then loops through this internal representation and uses the template file to generate its output, the source code for the target programming language.

The following wrappers use the IDL Compiler as their generation tool:

- *EntireX DCOM Wrapper*
- *EntireX C Wrapper*
- *EntireX .NET Wrapper*
- *EntireX COBOL Wrapper*
- *EntireX PL/I Wrapper*

The IDL Compiler and the template files for the target programming languages above are fully integrated in the *EntireX Workbench*. For automation purposes, the IDL Compiler can also be started at the command prompt.

Starting the IDL Compiler

To start the IDL Compiler

- At a command prompt, enter

```
Java -classpath "%ProgramFiles%\software
ag\entirex\classes\exxidlcompiler.jar;%ProgramFiles%\software
ag\entirex\classes\saglic.jar"
"-Dsagcommon=%CommonProgramFiles%\Software AG"
com/softwareag/entirex/idlcompiler/TplParser -t template file
[-Doption=value] [-Fbasename] [-Ppreprocessor variable]
[-ooutput-directory] [-Ttrace-level] [-deprecated] -idl
idlfile.idl
```

Parameter	**Description**
`-help`	Displays help using the command-line options.
`-D` *`option=value`*	Passes the option to the templates. See also *Using Options*.
`-F` *`basename`*	Indicates that the output file base name follows. It is used as given - thus it should be provided without a path and extension. Only relevant if supported by the template used. Default is the base name of the *idl-file* without path and extension. See *Specifying the Name of the Output File*.
`-t` *`template`*	Name of the initial template file.
`-P` *`preprocessor variable`*	Specifies a preprocessor variable that can be controlled in template files with `#ifdef`, `#elif`, `#else`, and be closed by `#endif`. Only relevant if supported by the template used. See *Using Template `#if` Preprocessing Statements*.
`-o` *`output-directory`*	Specifies the directory for the output file. See *Specifying the Name of the Output File*.
`-T` *`trace-level`*	Trace information is entered as comment lines into the generated output. **Trace level** — **Description** 0 — No trace output will be created. This is the default trace level. 1 — Generates only template file name and the line number in the output file. This trace level helps to identify the specific line in the template file. 2 — This is the full tracing mode for the template file. Every action of the IDL Compiler will be displayed. 3 — Additional traces for preparation of the output statement will be created.
`-deprecated`	Deprecated mode allows compilation of templates for statements already deprecated. Each deprecated statement will create a warning.
`-idl` *`idl-file`*	Name of the IDL file. Multiple IDL files can be provided.

IDL Compiler Usage Examples

Calling the IDL Compiler under UNIX

The following applies to the UNIX Bourne, Korn and C shell.

```
$JAVA_HOME/java -Dsagcommon=/opt/softwareag/EntireX\common\conf
-Dentirex.home=$EXXDIR -classpath
$EXXDIR/classes/exxidlcompiler.jar:$EXXDIR/classes/saglic.jar
com/softwareag/entirex/idlcompiler/TplParser -t
$EXXDIR/template/client.tpl -idl aaclient.idl
```

An *erxidl.bsh* shell script file is provided with a preconfigured invocation of the IDL Compiler. In addition, the "-deprecated" mode is set as default in this shell script file.

```
erxidl.bsh -t ...\EntireX\template\client.tpl -idl aaclient.idl
```

Calling the IDL Compiler under Windows

```
java -classpath "%ProgramFiles%\software
ag\entirex\classes\exxidlcompiler.jar;%ProgramFiles%\software
ag\entirex\classes\saglic.jar"
"-Dsagcommon=%CommonProgramFiles%\Software AG"
com/softwareag/entirex/idlcompiler/TplParser -t
...\EntireX\template\client.tpl -idl aaclient.idl
```

An *erxidl.bat* batch file for the Windows command shell is provided with a preconfigured invocation of the IDL Compiler. In addition, the "-deprecated" mode is set as default in this batch file.

```
erxidl -t ...\EntireX\template\client.tpl -idl aaclient.idl
```

Writing your own Wrappers and Stubs

Additional programming languages can be adopted with user-written templates (see *Writing Template Files for Software AG IDL Compiler*). The syntax for IDL template files in a formal notation is presented in the section *Grammar for IDL Template Files*.

Integration with the IDL Editor can be accomplished using the plug-in technique provided (see *Using EntireX Custom Wrappers*).