GradC++ ToolBox: Administration Guide

DocumentId:GradSoft-AD-e-04.09.2000-v1.1.0

April 29, 2001

Contents

1 Introduction

1.1 Subject

GradSoft C++ Toolbox (which will be called Gen below) consists from 4 components:

1. ProgOptions - class for hight level handling of program options.
2. ServiceOptions - specialization of ProgOptions for CORBA services.
3. Logger - class for logging support.
4. Threading - set of classes to design of multithreading programs.

This is Administration Guide for version 1.1 of the package. Current version of Gen is always avaible on our website: http://www.gradsoft.com.ua/eng

1.2 Possible desksides

The package Gen can be installed as something integer "by means of single moving of the hand" (is recommended). Besides, the eache of single components and pared-down version of the package composed of three non-CORBA components (i.e. ProgOptions + Logger + Threading) can be installed apart.

2 Installation

2.1 Software needed

1. CORBA ORB (not needed, if you not use ServiceOptions):
   - Unix: one of omniORB-3.0 or higher, TAO-5.9b or higher, ORBacus 3.3.2 or higher, ORBacus 4.0.1 or higher
   - WindowsNT: ORBacus 3.x and 4.x is supported
2. C++ compiler:
   - Unix: gcc-2.95.2 or higher, SunProc C++ 4.2
   - Windows NT: Microsoft Visual C v. 6.0 or higher.
3. make:
   - Unix: gnu make is necessary
   - WindowsNT: nmake from MSVC++

2.2 Installing for UNIX

1. Make sure that necessary software are installed and are in working state.
2. Extract files from archive Gen.tar.gz in some catalog. Let's name this catalog `<project_root>`
3. cd to `<project_root>`.

4. Run configure with command `./configure [options]` (options list is accessible via command `./configure --help`; for example, you can set `--prefix=<smth>` option to set directory of installation or `--with-corba=<yes/no>` option to set volume of compilation and installation).
5. Start compilation process with command `gmake`
6. Become superuser.
7. Run install process with command `gmake install`
8. For deinstallation you can use command `gmake uninstall`

2.3 Installing for Windows NT

1. Check, that all necessary software is installed and is in working state; put pathes to `nmake`, `cl` and `xcopy` utilities to the environment variable PATH.
2. Extract files from archive Gen.tar.gz in some directory. Let's name of this catalogue `<project_root>`
3. Edit file env_inc.nt.mak in directory `<project_root>\config`: set values of next nmake variables:

name	*description*
PROJECT_ROOT	`<project_root>` (root directory of project)
INSTALL_IDL_DIR	directory, where idl modules would be installed
INSTALL_INC_DIR	directory, where include files will be installed
INSTALL_LIB_DIR	directory, where libraries will be installed
ORB_DIR	root directory of ORB
MSVC_DIR	root directory of Microsoft Visual Studio

Footnotes:

(a) Precisely, include files will be installed into `$(INSTALL_INC_DIR)\GradSoft` subdirectory

(b) To use of Gen along with other GradSoft products, we reccomend to set
INSTALL_IDL_DIR,
INSTALL_INC_DIR and
INSTALL_LIB_DIR
variables in compliance with next procedure:

- Choose directory, where GradSoft products must be installed and set INSTALL_DIR to one.
- set INSTALL_IDL_DIR, INSTALL_INC_DIR and INSTALL_LIB_DIR via INSTALL_DIR, as in default settings.

4. For package compilation go to directory `<project_root>` and type

```
make build
```

5. For package installation go to directory `<project_root>` and type

   `make install`

6. If you want to deinstall installed package, go to directory `<project_root>` and type

   `make uninstall`

7. For cleaning of package (i.e. removing of automatically generated files, such as *.obj and *.lib, from `<project_root>` and its subdirs) go to directory `<project_root>` and type

   `make clean`

8. For compilation, installation, deinstallation or cleaning of single (apart) subpackage move over to directory suitable (one of four:
   `<project_root>\ProgOptions`,
   `<project_root>\CORBA\ServiceOptions`
   `<project_root>\Threading`
   or
   `<project_root>\Logger`)
   and use "`make <option>`" with suitable option (see section "Details of installation under Windows NT" 5 for more details)

9. For uniform compilation, installation, deinstallation or cleaning of tree non-CORBA subpackages (this is the "cut" configuration mentioned) move over to `project_root` and use command "`make.cut <option>`" with suitable option (see section "Details of installation under Windows NT" 5 for more details)

3 Testing

3.1 Common information

Demo examplex, included in Gen, for ProgOptions, ServiceOptions Й Logger are related to examples, which are described in Programmers Guide.

3.2 Procedure

3.2.1 ProgOptions

1. Read ProgrammingGuide in file
   `<project_root>`/ProgOption/docs/ProgrammingGuide_rus.pdf

2. cd to directory `<project_root>`/ProgOptions/demo

3. run ProgOptionsDemo few times with next options:

```
--help
--qqq
--zzz
--zzz <arg>
--zz1 <arg1> --qqq
--zz3
<arg2>
--<arg3>
```

4. check, that program output is expected

3.2.2 ServiceOptions

1. Go to directory `<project_root>`/CORBA/ServiceOptions/demo
2. Start HelloWorldServer with option `--help`; read help message.
3. Start HelloWorldServer with option `--ior-stdout`; make sure that Interoperable Object Reference is displayed on standard output
4. Start HelloWorldServer with options: `--ior-file-HelloService hello.ref` Make sure that Interoperable Object Reference is placed into file hello.ref
5. Start HelloWorldServer with options:
   - `-OAport 1025` if you use ORBacus
   - `-ORBport 1025` if you use TAO
   - `-ORBpoa_iiop_port 1025` if you use omniORB
6. Start HelloWorldClient with options:
   - if you use omniORB3 or ORBacus 4.x :
     `-ORBInitRef HelloService=corbaloc::127.0.0.1:1025/HelloService`
   - if you use TAO or ORBacus 3.x :
     `-ORBInitRef HelloService=iiop://127.0.0.1:1025/HelloService`

   Port 1025 must be not busy by some other application, and of course, you can use any port number instead 1025
7. Make sure that string "Hello World" is printed on you terminal
8. Check you NameService daemon
9. Start program HelloWorldServer with option `--with-naming`
10. Start HelloWorldClient with option `--with-naming`
11. Make sure that string "Hello World" is printed on you terminal

3.2.3 Logger

1. Read ProgrammingGuide in file
   `<project_root>/Logger/docs/ProgrammingGuide_rus.pdf`
2. Move out to directory `<project_root>/Logger/demo`
3. Run demo executable
4. Make sure that file.log is generated and its content correspond with usage rules described in programming Guide.

3.2.4 Treading

Move out in series to

1. `project_root/Treading/demo/Philosophs`
2. `project_root/Treading/demo/Container`

and make sure that executable files is present and is in working state.

4 Using

Programs, which use ProgOptions, ServiceOptions and Logger must be compiled with the next libs:

For Unix:

package	*static libraries*	*dinaminc libraries*
ProgOptions	libProgOptions.a	libProgOptions.so
ServiceOptions	libProgOptions.a libServiceOptions.a	libProgOptions.so libServiceOptions.so
Logger	libLogger.a	libLogger.so
Threading	libThreading.a	libThreading.so

For Windows NT:

package	*static libraries*	*dinaminc libraries* *)
ProgOptions	ProgOptions.lib	
ServiceOptions	ProgOptions.lib ServiceOptions.lib	
Logger	Logger.lib	
Threading	Threading.lib	

*) to be impemented in next versions

5 Details of installation for Windows NT

5.1 Installation commands

Installation for Windows NT may be realized by command "make" implemented via files make.bat situated in few different subdirectories of `<project_root>` or by the command "make.cut" implemented via file make.cut.bat located in the `<project_root>`.

5.1.1 Command "make"

1. There are five copies of make.bat located in next five places:

   (a) in the `<project_root>` folder

   (b) in the `<project_root>\ProgOptions` folder

   (c) in the `<project_root>\CORBA\ServiceOptions` folder

   (d) in the `<project_root>\Logger` folder

   (e) in the `<project_root>\Threading` folder

   The choice of each of these locations to run of "make" means the scope of command will be cutted:

   - `<project_root>` - the treatment will be applyed to whole package Gen
   - `<project_root>\ProgOptions` - the treatment will be applyed to ProgOptions only
   - `<project_root>\CORBA\ServiceOptions` - the treatment will be applyed to ServiceOptions only
   - `<project_root>\Logger` - the treatment will be applyed to Logger only
   - `<project_root>\Threading` - the treatment will be applyed to Threading only

   Thus, if you want to get a single component instead of whole Gen, move out to directory concerned and run "make" command from it.

2. Use make.bat as follows: type

   ```
   make <target>
   ```

   with the `<target>` selected from the list below:

<target> *value*	*command action*
build	compilation of [sub]package (i.e. of package or its single component)
install	installation of [sub]package
uninstall	deinstallation of [sub]package
clean	cleaning of [sub]package (i.e. removing of all automatically generated files from <project_root> and folders below)

3. There are two features of "make" usage:

   (a) command "make" with empty target is permitted and is equal to "make build".

   (b) use "make install" without prior call of "make build" command is permitted and leads to automatical compilation of part of code needed to making of entities to be installed.

5.1.2 Command "make.cut"

Command "make.cut" is similar to command "make" runned from root directory with the exclusion of limited scope of action: using "make.cut" brings to uniform processing of three non-CORBA subpackeges (ProgOptions, Logger and Threading) instead of whole package Gen as in the case of "make". Correspondingly, using "make.cut" is analogous to using "make" : you must type

```
make.cut <target>
```

with target takes on a value being one of following: build, install, uninstall, clean. Actions connected with values listed and features of these actions is analogous to ones for "make"

5.2 Dependences

There is a dependence between ServiceOptions and ProgOptions subpackages. Namely, compilation of tests for ServiceOptions depends on library ProgOptions.lib. Generation of ProgOptions.lib arises in two next cases:

1. at the time of compilation of ProgOptions by using "make" with target "build";

2. at the time of installation of ProgOptions if "make build" command has not been used before.

Thus, if you want to compile ServiceOptions without compilation of all Gen code, use "make build" or "make install" for Progoptions previously.

6 Changes

14.02.2001 - added detailed information about ORB configuration options

04.09.2000 - first revision