Contents

[Introduction 1](#_Toc277165764)

[GNAT 2010 2](#_Toc277165765)

[Download GNAT 2010 2](#_Toc277165766)

[Open TAR file 3](#_Toc277165767)

[Installing GNAT 2010 4](#_Toc277165768)

[Verify Installation of GNAT 2010 6](#_Toc277165769)

[PolyORB 7](#_Toc277165770)

[Installing PolyORB 7](#_Toc277165771)

[Download Cygwin 7](#_Toc277165772)

[Verify Cygwin Installation 10](#_Toc277165773)

[Make 3.80 11](#_Toc277165774)

[Opening PolyORB 12](#_Toc277165775)

[Makefile 16](#_Toc277165776)

[PolyORB\_Config.gpr 17](#_Toc277165777)

[Make 17](#_Toc277165778)

[Verify PolyORB 17](#_Toc277165779)

[Distributed Hello World 18](#_Toc277165780)

# Introduction

This documentation is written because I think there is a need for some supporting text to people that wish to experiment with Ada and Distributed Annex E / PolyORB on Windows 7. The installation of GNAT 2010 Ada compiler is very similar to how you install any other software on Windows platforms, but the “Distributed Anned E” is not simple, because you need to learn a lot of additional tools in the process which can make it really difficult.

The document will (try☺ ) to make a note on all those small tricks that are useful to go through the installation, and also to make you able to ask questions in forum / mailing lists in a way to which participant there can contribute.

This document is based on an installation on Windows 7 laptop and on stationary PC.

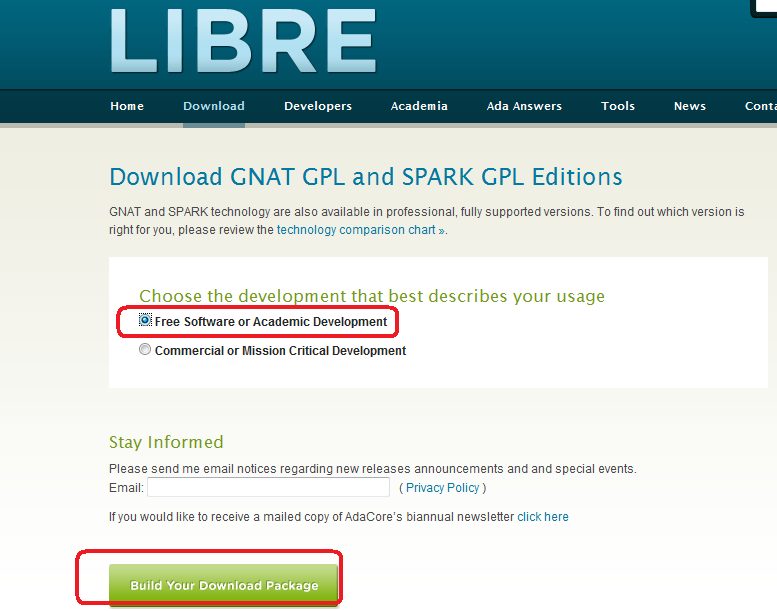
# GNAT 2010

## Download GNAT 2010

Go to AdaCore site to download GNAT 2010 for students <http://libre.adacore.com/libre/>



Click on the encircled area on the screen above. This will give you a student version of GNAT that can not be used for commercial development.

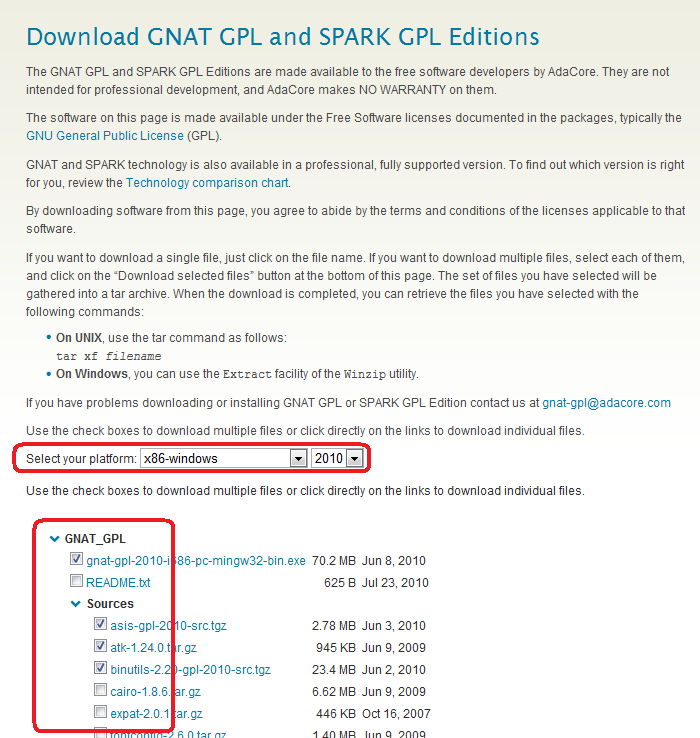


Chose “Free Software or Academic Development” and press “Build Your Download Package”

Now you get to a page where you need to select the pieces of Ada that you need.

Since we are thinking to use Windows 7, we chose the “x86-windows” platform, and 2010.

Secon encircled area below must be expanded, and you chose “gnat-gpl-2010-ix86-pc-mingw32-bin.exe” – in my download I chose three packages for source code, more or less by accident, I don’t think you need them (you can include them for safety☺ )



In addition expand the paragraph for PolyOrb and and XMLAda chose these.



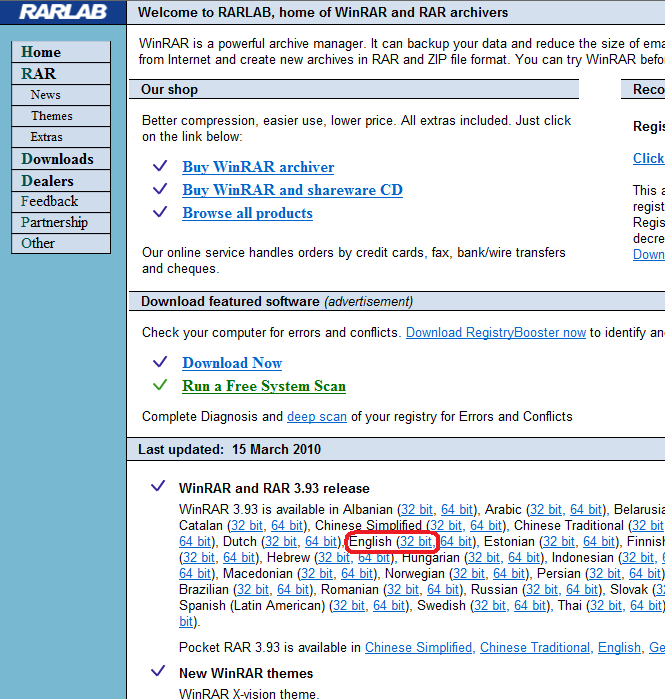
You are done selecting the download, and can press “Download selected files”.

You get the familiar download dialogue box, this will indicate that you will get a .TAR file – TAR file is a file format that keeps the directory structure. Save this to any folder.

## Open TAR file

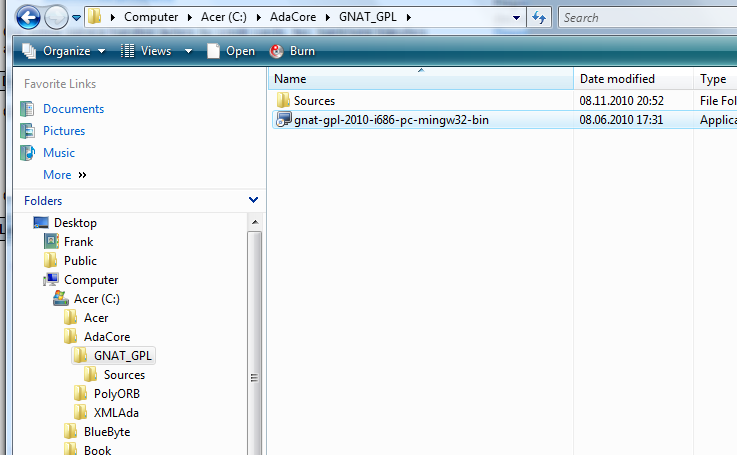
I use WinRAR to unpack these downloads. This can be downloaded from <http://www.rarlab.com/>

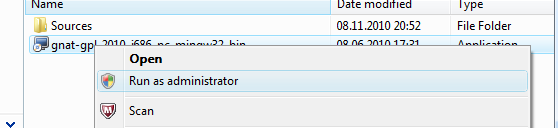
Chose the link in the encircled area below. WinRAR is installed in the normal Windows way.



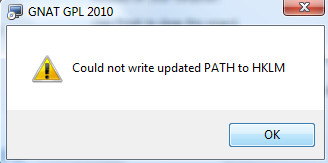
Use WinRAR to extract the downloaded file we saved earlier, to a folder called C:\AdaCore.

## Installing GNAT 2010



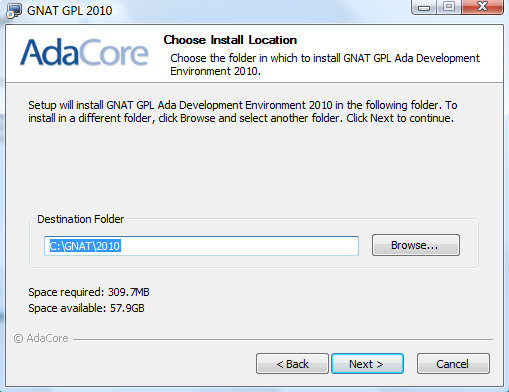
Right-click on the file in C:\AdaCore\GNAT\_GPL and chose “Run as administrator”. 

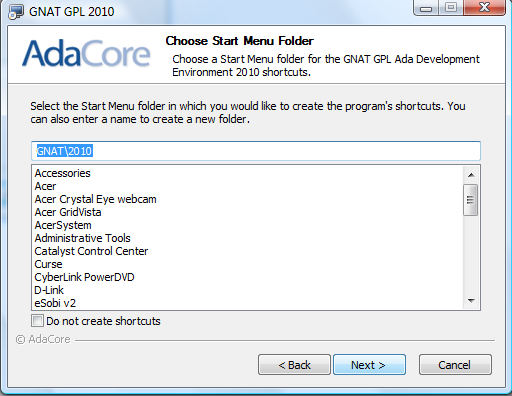
If you don’t do this, you will get an error box at the end of the installation.

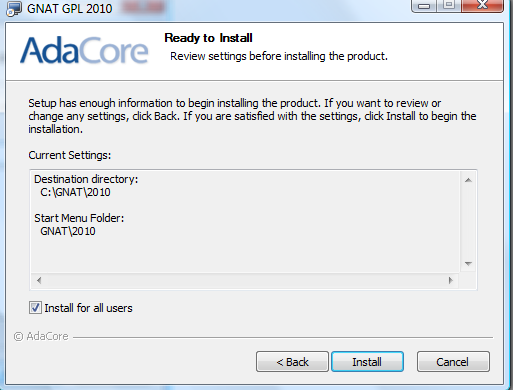


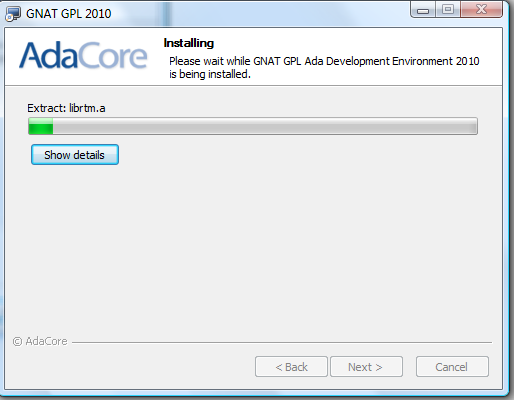
Installation will no proceed in a very familiar way (to Windows users ☺ ) Just press “Next” on all this to be able to use this document













GNAT 2010 GPS and compiler should now be installed.

## Verify Installation of GNAT 2010

You should make a simple Ada program to verify the installation.

Create a text document and copy/paste the following simple Ada program into it:

With Text\_IO;  
Procedure Hello\_World is  
begin  
Text\_IO.Put\_Line(“Hello, World!”);  
end;

Save this to a folder of your choice, with the name “Hello\_World.adb”. The file name must match the procedure in the text file so this is not the moment to be creative. Open “Command Prompt” in Windows and go to this folder and execute:

Gnatmake Hello\_World.adb

Then execute:

Hello\_World

# PolyORB

## Installing PolyORB

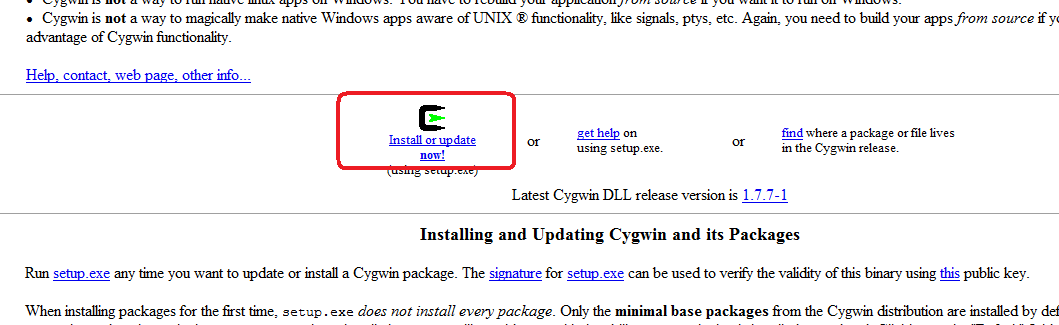
It is now getting trickier. PolyORB is not supplied as a binary package, only as source code. This means you need to compile and build the source code. This must be done in a Unix-like environment because the tools used in this process are from Unix-world.

To do this in Windows 7, we need to install a Linux-for-Windows tool. The tool I use is Cygwin. This is a big piece of software on it’s own, and it pretty much feels like Unix when you are using it.

## Download Cygwin

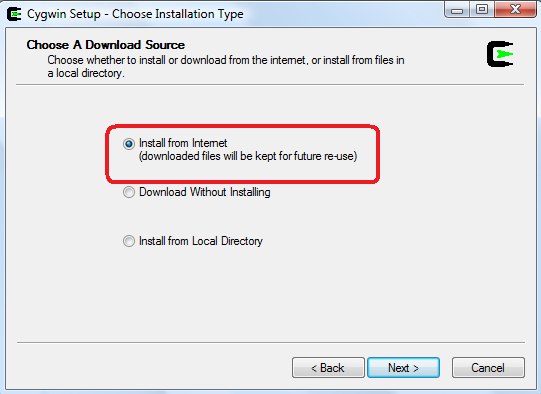
Cygwin can be downloaded by going to <http://www.cygwin.com/>

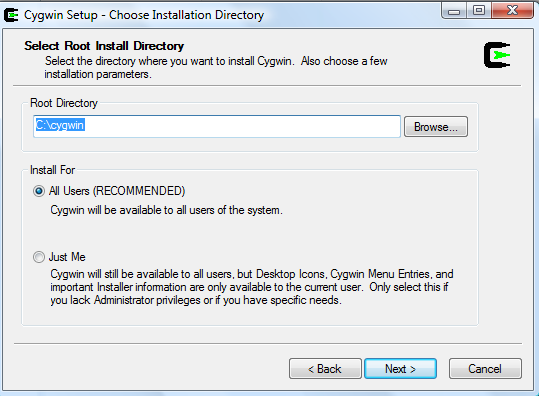
A bit down the page there is a black and green icon “Install and update now!”. Right-click there, and save “Setup.exe” to some folder.

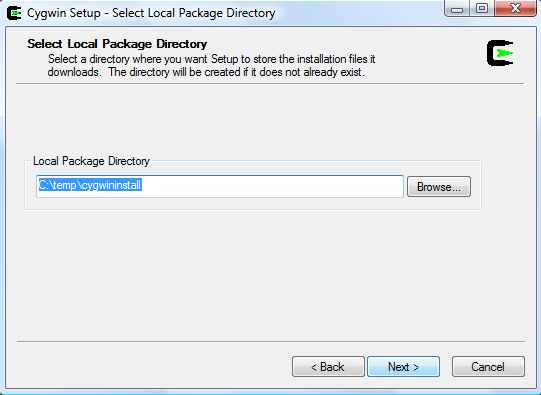


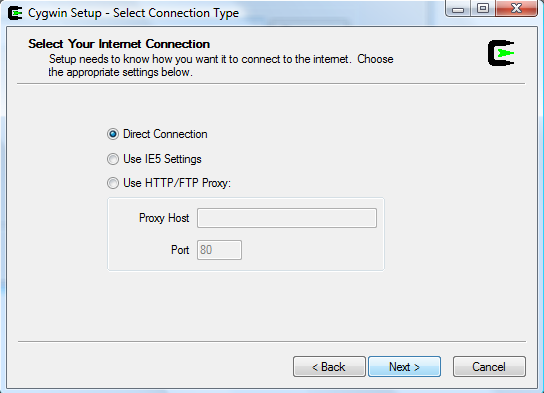
Run this new setup.exe, as administrator.

Some dialogue boxes appear, chose default values. When I saved Setup.exe, I saved it to a folder on C:\temp that’s is why the screenshot below proposes the folder to be c:\temp\cygwininstall.

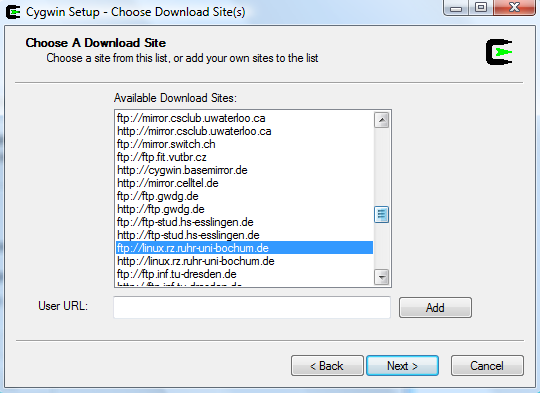




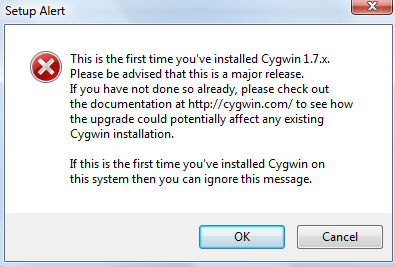




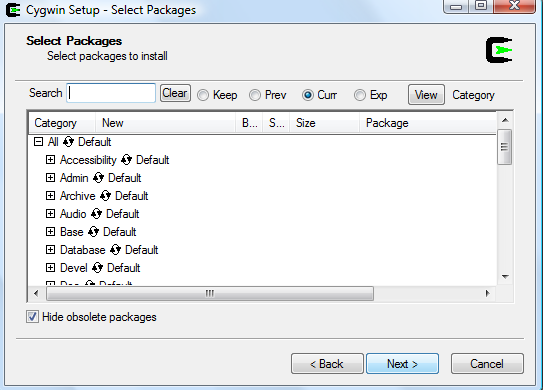
In dialogue box below, chose any site – if one seems to be slow or not operating try anther one by restarting the Setup.exe.



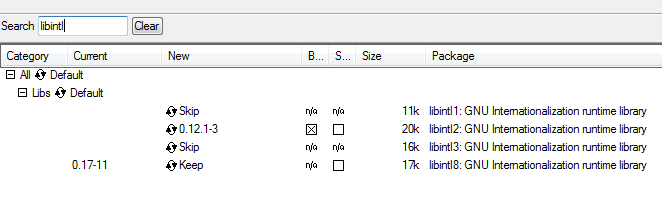
The information box below did not relate to my case so I ignored it, by pressing “OK”.



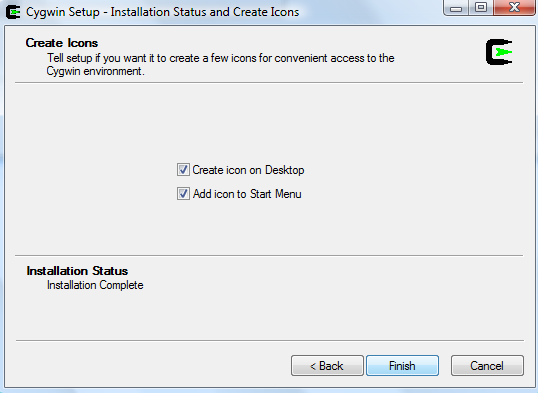
Now setup will download some information and in the end present you with choices to what you can install.



In this dialogue, enter “libintl” in the search field and expand the nodes as displayed below, and chose libintl2 by clicking that line. This is related to Make-3.80 which we will return to later.



Press ok to this and wait.

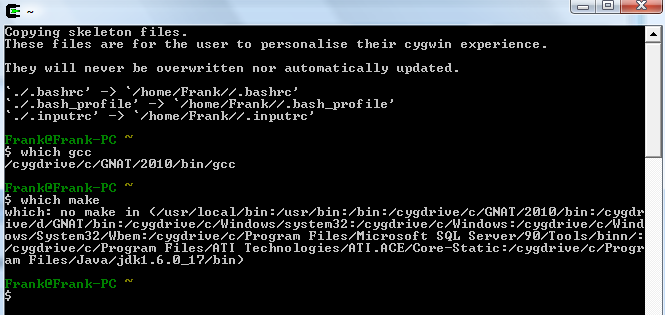


In dialogue above, chose “Finish”.

## Verify Cygwin Installation

From Windows 7, start meny, chose Cygwin-> Cygwin Bash Shell

I window like the following should appear.



It is now useful to check if this environment can see any version of GNAT 2010 that we installed earlier. To check this, type:

Which gcc

Which gnatmake

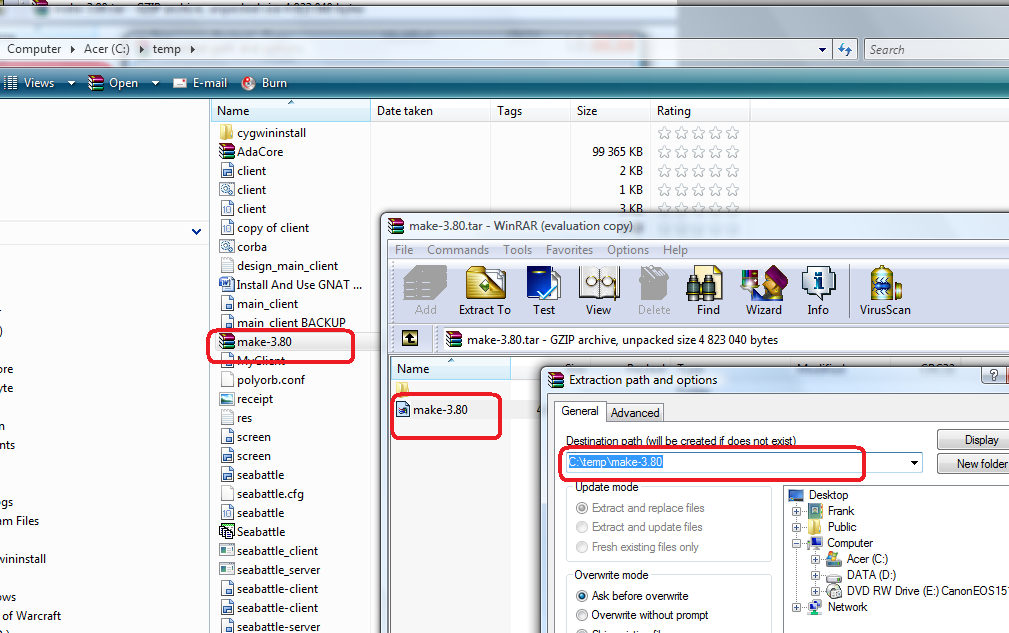
The response from these two, should be referring to /cygdrive/c/GNAT/2010/gcc and gnatmake (see screenshot above). This means that in this environment, as it is configured now, you will execute gcc and gnatmake from this installation. If these tests shows a different result – it might be because there are more than one installation of GNAT 2010 on your PC. You should look into this, by investigating the PATH environment variable in Windows7, e.g by making sure your last installation occur first among the paths there. If this doesn’t get right you will most probably get error messages and problems that no one really can explain or help you to solve.

## Make 3.80

Another tricky issue, is a tool called make. There is a version of this tool available from Cygwin, but this version contains a problem that makes it impossible to ‘make’ PolyORB. We need to get an older version, I use ‘make-3.80’. We already prepared for this step while downloading cygwin, by chosing the libintl2

Go to the following site to download http://www.go-mono.com/archive/helper/make-3.80-1.tar.bz2

With WinRAR extract the make-3.80 file

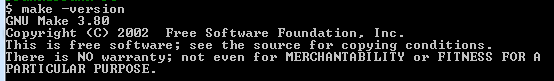


Copy make-3.80 into c:/cygwin/bin

Rename it to “make”

In Cygwin Bash Shell:

Make -version



Should show GNU Make 3.80.

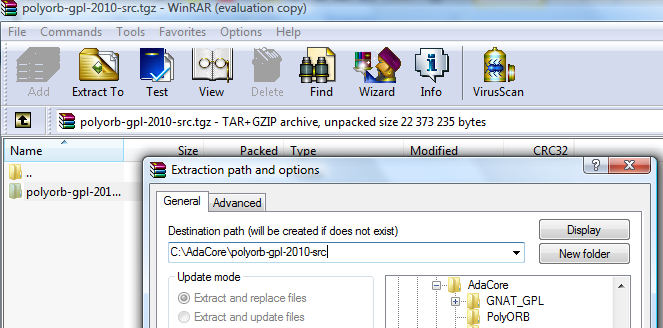
If you find some other version still you maybe have copied into wrong folder earlier, or you have more versions floating around. Try

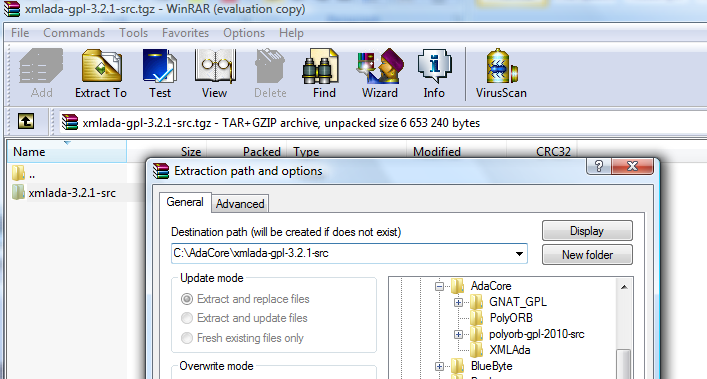
Which make

To see what is going on.

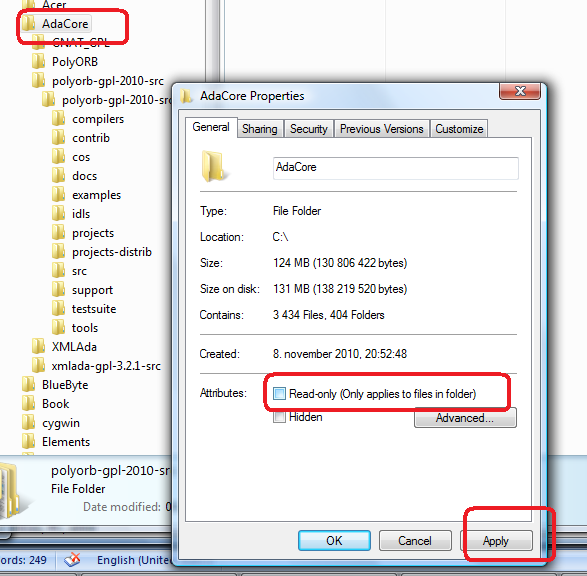
## Opening PolyORB

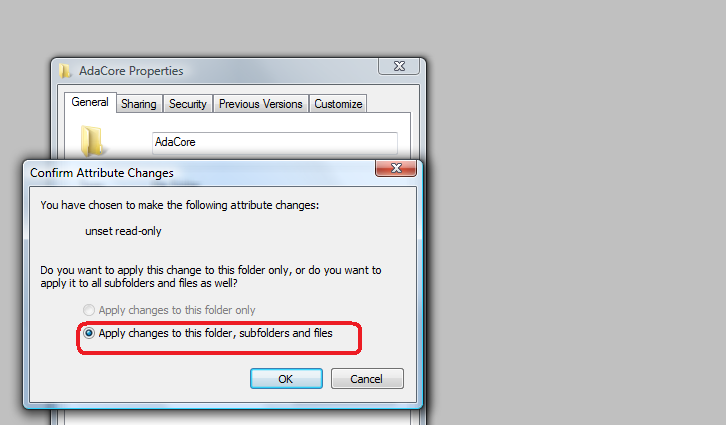
Extract and set folder as below.





Make sure the folders we need to work in are all read/write before continuing:





In Cygwin Bash Shell move to the folder of ployorb source code:

cd /cygdrive/c/adacore/polyorb-gpl-2010-src/polyorb-gpl-2010-src

A very useful thing to do, is to mount /cygdrive/c/AdaCore to “C:/AdaCore”. This helps to the problem that Unix-tools doesn’t always understand Windows-paths (e.g C: and D: and so on)

We do the similar thing with /cygdrive/c/gnat/2010.

umount "C:/AdaCore"  
umount "C:/GNAT/2010"

mount -f -o noacl "C:/AdaCore" /cygdrive/c/AdaCore  
mount -f -o noacl "C:/GNAT/2010" /cygdrive/c/gnat/2010

Another thing, do:

export TEMP=/tmp

**Be aware** that these things you need to do every time you start a new Cygwin Bash Shell.

Next do:

./configure --prefix=C:/GNAT/2010 --with-appli-perso="moma corba dsa" --with-corba-services="event naming"

This sets the environment and arrange some files. There are plenty of things that can go wrong here. For me it was an issue related to gcc compiler that created problem. We have already solved that by setting the TEMP environment variable earlier but I bring the experience as an illustration to what you may need to resolve on your own.

|  |
| --- |
| An early attempt to do ./configure    This is not promising ☹ I open configure in a text editor, and search for the text ”checking for C compiler default output file name”  I find the following code snippet:    This is not so easy to understand if you are not used to Unix. However, there is another file in the folder called config.log this file is a log of what happened during ./configure. In this file we can find the same error message.    I chose to create a small test with this c-program and these options:  int main () {  ;  return 0; }  Compiling this with gcc:    I more or less type this into google and get the following forum discussion where several has had this problem, and many have succeded by setting the environment variable TEMP.  <http://stackoverflow.com/questions/2151257/cygwin-compiling-error-this-application-has-requested-the-runtime-to-terminate>  export TEMP=/tmp |

Problems during this phase generally originates from some tool is not found -> check with “which” and try to search for it, or it finds an inadequate version of the tool (e.g a windows version when it should have found the cygwin version or just wrong version of the tool

## Makefile

The ./configure has constructed a file called Makefile that is now tailored to our needs.

We need to do a small change in this Makefile.

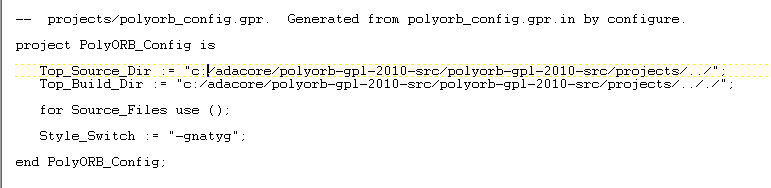
Open the Makefile and go to some lines that looks like below (in my case lines 60, 61):

Comment the generated lines for top\_srcdir and top\_builddir, and create two new lines, using the Windows7-like paths (we prepared for this by the mounting earlier).

#top\_srcdir := /cygdrive/c/adacore/polyorb-gpl-2010-src/polyorb-gpl-2010-src  
#top\_builddir := /cygdrive/c/adacore/polyorb-gpl-2010-src/polyorb-gpl-2010-src  
top\_srcdir := **C:/adacore/**polyorb-gpl-2010-src/polyorb-gpl-2010-src  
top\_builddir := **C:/adacore/**polyorb-gpl-2010-src/polyorb-gpl-2010-src

## PolyORB\_Config.gpr

Open PloyORB\_Config.gpr and edit the two paths there as show below.



## Make

You are now perhaps☺ ready to make and install PolyORB.

In Cygwin Bash Shell perform:

make

make install

This process if lasting for several minutes.

## Verify PolyORB

Quite a few executables and libraries are built during this process:

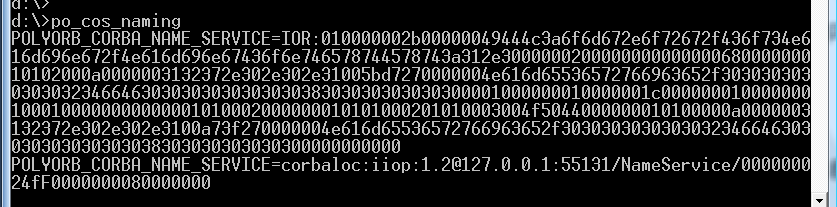
In C:\GNAT\2010\bin you should find:

gnatprfh.exe  
iac.exe  
po\_gnatdist.exe  
po\_cos\_naming.exe  
po\_cos\_naming\_shell.exe  
ir\_ab\_names.exe  
po\_ir.exe  
po\_catref.exe  
po\_createref.exe  
po\_names.exe  
po\_dumpir.exe

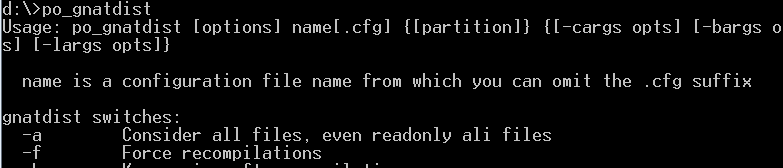
In C:\GNAT\2010\lib you should find a folder called “polyorb”

Plus many more ☺

Try to open a Command Prompt in Windows 7 and type Po\_cos\_naming. You should get the following output. Use CTRL-C to stop this (nothing more will happen at this point).



Another test is po\_gnatdist. This is the program that performs the build of a distributed application.



# Distributed Hello World

We will create a small distributed solution to “Hello World”. We will have a server that supplies a text string to a client. The client will print this text to the console and the server will print to its console what he sent to the client.

## Source Code

This is the main program for the client. It is calling a function in a package, Server.Get\_Text. This function will return a text string based on the number that is passed into the function.

|  |
| --- |
| with Server;  with Text\_IO; use Text\_IO;  procedure Main\_Client is  Text\_From\_Server : String(1..11);  begin  Text\_From\_Server := Server.Get\_Text(1);  Text\_IO.Put\_Line("Yes, server said:" & Text\_From\_Server);    Text\_From\_Server := Server.Get\_Text(87);  Text\_IO.Put\_Line("Yes, server said:" & Text\_From\_Server);  Text\_From\_Server := Server.Get\_Text(4);  Text\_IO.Put\_Line("Yes, server said:" & Text\_From\_Server);  end Main\_Client; |

The next source code shows the main program for the server. Since our server doesn’t do anything except sitting there, we just present a text to show that it is executing and then delay for 10 seconds.

|  |
| --- |
| with Server;  with Text\_IO;  procedure Main\_Server is  begin  Text\_IO.New\_Line;  Text\_IO.New\_Line;  Text\_IO.Put\_Line("Hello World server by Frank J Jorgensen 2010");  Text\_IO.Put\_Line("Developed by using Ada 2005 / GNAT 2010");  delay 10.0;  end Main\_Server; |

In the server we place an Ada package. The package is specified with “Remote\_Call\_Interface”. This set some limitation to what you can do in the specification part but it allows us to tell the compilator that this package can be accessed by other partitions in the distributed application. In this case it means that the client code (which will be a partition on its own) can call this package.

|  |
| --- |
| package Server is  pragma Remote\_Call\_Interface;  function Get\_Text (pNumber : in Integer) return String;  end Server; |
| with Text\_IO;  package body Server is  function Get\_Text (pNumber : in Integer) return String  is  sRes : String(1..11);  begin  case pNumber is  when 1 =>  sRes := "Hello World";  when 4 =>  sRes := "Another day";  when others =>  sRes := "Don't know ";  end case;  Text\_IO.Put\_Line("Server received pNumber:" & pNumber'Img & " and will return:" & sRes);  return sRes;  end Get\_Text;  end Server; |

To make the compilator build this application distributed as we want it to be we need to create a file to configure what pieces of the source code makes the server and what makes the client.

|  |
| --- |
| configuration helloworld is  pragma Version (False);  pragma Starter (None);  helloworld\_client : Partition;  helloworld\_server : Partition := (Server);  procedure main\_server is in helloworld\_server;  procedure main\_client;  for helloworld\_client'Main use main\_client;  for helloworld\_server'Host use "localhost";  end helloworld; |

To compile and build this, open a Command Prompt and type:

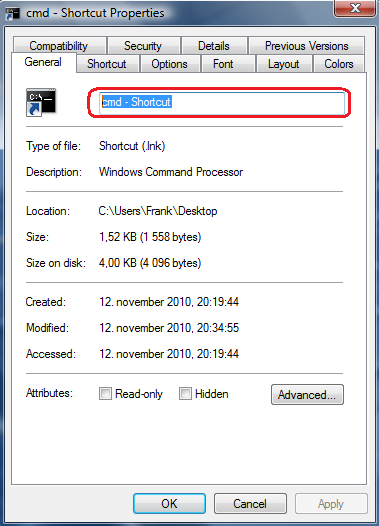
po\_gnatdist helloworld.cfg

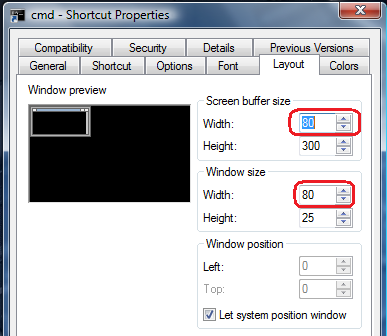
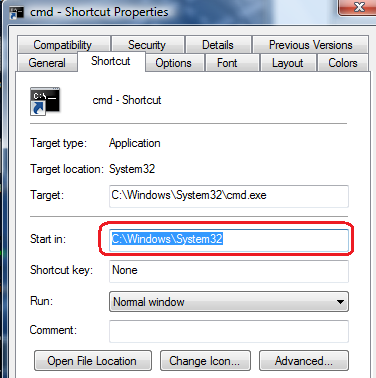
After this is done (takes a few seconds) you will find two executables in your folder helloworld\_client.exe and helloworld\_server.exe.

## Execution on one PC

You need to arrange a bit before executing the application. It is very useful to create some scripts to simplify this work a little. First we will look at the fully manual approach.

1. Create a shortcut for Command Prompt on your desktop
2. Open properties for the shortcut
3. Change the title of the shortcut to “HelloWorld”



1. Go to the Layout-tab  
   
2. Change Screen Buffer Size and Window Size to 150
3. Go to the Shortcut-tab  
   
4. Change the “Start in” to be in the folder where your HelloWorld-project is.
5. From this shortcut, start three Command Prompts. We will call them A, B and C.
6. In A you type po\_cos\_naming
7. Po\_cos\_naming will return an output. This output is needed to make the server and client to get in contact with CORBA process. There are actually two strings in the output that can be used to identify the CORBA process. The simplest is to use corbaloc. Left-click in top-left corned of Command Prompt, chose Edit-Mark from meny there. Mark the string that looks like:  
   POLYORB\_CORBA\_NAME\_SERVICE=corbaloc:iiop:1.2@192.168.0.199:49211/NameService/000000024fF0000000080000000  
   Then press Enter and the string is on clipboard and you can paste it in somewhere.
8. Open a text editor. In this editor, paste the string.
9. In the text editor, construct a statement similar to this, using the string you got from po\_cos\_naming:  
   set POLYORB\_DSA\_NAME\_SERVICE=corbaloc:iiop:1.2@192.168.0.199:49211/NameService/000000024fF0000000080000000
10. Copy and paste this statement into Command Prompt B and C.
11. Now in Commend Prompt B, execute helloworld\_server
12. In Command Prompt C, execute helloworld\_client