Gem #126 : Aggregate Library Projects

by Pascal Obry—EDF R&D

Let's get started...

A common scheme for building large Ada applications is to create multiple modules. Each module (or subsystem) comes with a project file, not only to support building it, but also to facilitate editing the code via GPS. This has the advantage of permitting developers to focus only on the code for the module of interest. On the whole, this decreases the complexity of an application by modularizing it.

Each module comes with a library project used for building a library (shared or static) containing the object code for the module. For an application, the final step is just linking together the libraries for all modules. So far, so good.

However, if the goal is not to build an application, but rather to deliver a library (containing API sources and objects), this approach is not really convenient. Indeed, all libraries have to be copied into the installation directory, and a project file for each library must be distributed for the final application to satisfy the required dependencies. This is cumbersome and somewhat tedious, and it is even more tedious when the library is to be used in an application built with C or C++, because we impose the complexity on the client of the API by requiring linking with multiple libraries.

In such cases it would be far more convenient to distribute a single library containing all the code.

Perhaps you're now thinking that it's a bad idea to create modules and develop big

applications using a flat project structure? You're right, and that's where the aggregate library projects come into play.

Aggregate library projects enable the possibility of creating a single library (shared or static) out of a set of libraries or projects.

For example, let's assume the project contains two libraries:

```
library project MyLib1 is
   for Source_Dirs use ("src1");
   for Object_Dir use "obj1";
   for Library_Name use "mylib1";
   for Library_Kind use "static";
   for Library_Dir use "lib1";
end MyLib1;

library project MyLib2 is
   for Source_Dirs use ("src2");
   for Object_Dir use "obj2";
   for Library_Name use "mylib2";
   for Library_Kind use "static";
   for Library_Dir use "lib2";
end MyLib2;
```

Project mylib1.gpr and mylib2.gpr can be used separately to create mylib1.a and mylib2.a.

But by using an aggregate library it's possible to create a single library (say fulllib.a) combining both of the projects:

```
aggregate library project MyLib is
   for Project_Files use ("mylib1.gpr", "mylib2.gpr");
   for Library_Name use "fulllib";
   for Library_Kind use "static";
   for Library_Dir use ".";
end MyLib;
```

Note that an aggregate library project description is very close to a library project. The only new required attribute is Project_Files, which must list all of the projects that will be aggregated.

Note also that there's no need for an Object_Dir attribute, as aggregate libraries do not have object code by themselves.

The following command will build pck1.o and pck2.o (from src1/pck1.ads and src2/pck2.ads) using their respective projects, and will then aggregate the resulting object code in a single library named "libfulllib.a":

```
$ gprbuild -gnat05 -p mylib.gpr
```

The contents of the aggregate library can be verified with:

```
$ nm libfulllib.a
```

And, of course, it's also possible to do the same with shared libraries.

In a forthcoming installment we'll discuss encapsulated libraries, which are quite useful for multilanguage development.

code.zip (http://www.adacore.com/uploads_gems/code.zip)

Last Updated: 10/13/2017

Posted on: 5/28/2012

Help us understand your development needs and get a quote or an evaluation »

(/get-a-quote)

Products (/products)

GNAT Pro (/gnatpro)

CodePeer (/codepeer)

SPARK Pro (/sparkpro)

QGen (/qgen)

Services (/services)

Support (/support)

Industries (/industries)

Automotive (/industries/automotive)

Avionics (/industries/avionics)

Rail (/industries/rail)

Air Traffic (/industries/atm)

Space (/industries/space)

Defense (/industries/defense)

Resources (/resources)

Books (/books)

Tech Papers (/tech-papers)

Documentation (/documentation)

Webinars (/webinars)

Devlog (/devlog)

Company (/company)

About AdaCore (/company/about)

Careers (/company/careers)

Contact (/company/contact)

Customer Support (/support)

Login to GNAT Tracker (/login)

Expert Support (/support)

Contact Us (/company/contact)

Pricing (/get-a-quote)

News (/news)

Press Releases (/press)

AdaCore in the Press (/in-the-press)

Events (/events)

Inside AdaCore (/newsletter)

Community (/community)

Download (/download)

Getting Started (/get-started)

About Ada (/about-ada)

About SPARK (/about-spark)

Academia (/academia)

Overview (/academia)

Projects (/academia/projects)

Universities (/academia/universities)

GAP Login (/login?mode=gap)

Other AdaCore Sites

The AdaCore Blog (http://blog.adacore.com)

Learn.adacore.com (https://learn.adacore.com)

Make with Ada (http://makewithada.org)

(http://twitter.com/AdaCoreCompany)

 (https://www.linkedin.com/company/39996)
 (https://www.youtube.com/user/AdaCore05)
 (https://github.com/AdaCore)  (/rss)