Gem #18: Предупреждения GNAT

Автор: Bob Duff, AdaCore

Краткое содержание: Gem Ada #18 - Данный «gem» содержит информацию не о языке Ada как таковом, а о наборе элементов GNAT, связанных с предупреждениями. Здесь показано, как наилучшим образом обращаться с предупреждениями, генерируемыми GNAT.

Давайте начнем…

Последние «gems 16-17» о прагме No_Return много говорили о проверках во время компиляции, которые генерируют предупреждения. Действительно, GNAT способен генерировать множество полезных предупреждений. Вот несколько советов о том, как извлечь максимальную пользу из этих предупреждений.

В чем разница между «предупреждением» и «ошибкой»? Во-первых, ошибки обычно являются нарушениями правил языка Ada, как указано в Справочном руководстве Ada; предупреждения являются спецификациями GNAT. Таким образом, другие компиляторы Ada могут не сообщать о тех же вещах, что и GNAT. Во-вторых, предупреждения обычно консервативны; то есть некоторые предупреждения будут ложными тревогами, и программисту необходимо изучить код, чтобы узнать, является ли предупреждение реальной проблемой.

Некоторые предупреждения даются по умолчанию, тогда как некоторые из них даются только в том случае, если их разрешает соответствующий переключатель. Используйте переключатель -gnatwa для включения (почти) всех предупреждений.

Предупреждения бесполезны, если вы ничего не делаете с ними. Если вы дадите члену команды какой-то код, который вызывает предупреждения, как он узнает, представляют ли они реальные проблемы? Довольно скоро люди будут игнорировать предупреждения, и они будут разбросаны по всему коду. Используйте переключатель -gnatwae, чтобы включить (почти) все предупреждения и обработать предупреждения как ошибки. Это заставит вас получить чистую (без предупреждений или ошибок) компиляцию.

Но некоторые предупреждения - ложные тревоги. Используйте Pragma Warnings (Off) для подавления ложных тревог. Лучше всего быть как можно более конкретным: сузить до одной строки кода и одно предупреждающее сообщение. И используйте комментарий, чтобы объяснить, почему предупреждение является ложным сигналом тревоги, если это не очевидно. Следующее, скомпилированное с -gnatwae:

```
package body Warnings_Example is

   procedure Mumble (X : Integer) is
   begin
      null;
   end Mumble;

end Warnings_Example;
```

приведет к тому, что GNAT будет жаловаться:

```
warnings_example.adb:5:22: warning: formal parameter "X" is not referenced
```

Но следующее будет компилироваться чисто (без предупреждений):

```
package body Warnings_Example is

   pragma Warnings (Off, "formal parameter ""X"" is not referenced");
   procedure Mumble (X : Integer) is
   pragma Warnings (On, "formal parameter ""X"" is not referenced");
   --  X is ignored here, because blah blah blah...
   begin
      null;
   end Mumble;

end Warnings_Example;
```

Здесь мы подавили конкретное предупреждающее сообщение на определенной строке.

Если вы получаете много предупреждений определенного типа, и нецелесообразно их исправить, тогда подавляйте этот тип сообщения, поэтому хорошие предупреждения не будут скрыты под кучей поддельных. Переключатель -gnatwaeF отключит предупреждение в первой версии Mumble выше: средство F подавляет предупреждения по необъявленным формальным параметрам, и было бы неплохо, если бы у вас их было много.

Таким образом, я предлагаю включить столько предупреждений, сколько целесообразен для Вашего проекта. Тогда каждый раз, когда Вы видите предупреждающее сообщение, смотрите на код и решаете, реально ли это. Если так, фиксируйте код. Если это - ложный аварийный сигнал, подавите предупреждение. Так или иначе заставьте предупреждение исчезнуть прежде, чем проверить Ваш код в Вашей системе управления конфигурацией.

Детали переключателей и прагм можно найти в Справочном руководстве GNAT и руководстве пользователя.

Обсуждение…