Gem #46: Несовместимость Ada 83 и Ada 95

Автор: Robert Dewar, AdaCore

Краткое содержание: Gem Ada #46 - Часть 1, Неограниченные массивы в обобщённых типах (`generic`).

Давайте начнем…

По большей части Ada 95 полностью совместим с Ada 83, что означает, что правильно написанный портируемый код Ada 83 может быть скомпилирован с помощью компилятора Ada 95, и никаких изменений не требуется.

Однако есть некоторые случаи несовместимости, которые необходимо устранить при работе с устаревшим кодом. Конечно, вы можете компилировать в режиме Ada 83 с помощью pragma Ada_83 или эквивалентного переключателя компилятора "`-gnat83`" (`/83`, если вы используете VMS). Однако это означает, что Вы не можете использовать какие-либо функции Ada 95 по мере продолжения разработки, поэтому лучше исправить эти несовместимости, если это возможно.

В этой серии GEM мы рассмотрим эти несовместимости. Некоторые из них действительно легко исправляются, другие сложнее. В этом первом выпуске мы рассмотрим один из наиболее распространённых случаев.

Рассмотрим следующую декларацию,

```
generic
   type T is private;
package GP is
   ...
end GP;
```

В Ada 83 вы можете создать экземпляр этого родословного типа с неограниченным типом, например:

```
package NP is new GP (String);
```

Это больше не разрешено в Ada 95, так как это представляет собой так называемое «нарушение контрактной модели». Если вы попытаетесь скомпилировать экземпляр в режиме Ada 95, вы получите сообщение об ошибке:

```
8.    package NP is new GP (String);
                                |
      >>> actual for "T" must be a definite subtype
```

`String` - это неопределённый подтип, означающий, что он является неограниченным. Напротив, определённый подтип означает подтип, границы которого ограничены, так что-то вроде

```
subtype S50 is String (1 .. 50);
package NP is new GP (S50);
```

было бы законно, но неограниченная строка не может быть использована.

Модель контракта для универсальных шаблонов (`generic`) гласит, что если `generic` компилирует без ошибок, то все возможные экземпляры должны компилироваться без ошибок. Но если вы разрешаете вышеуказанный экземпляр, то является ли это законным или нет, зависит от того, имеет ли созданный объект объявление внутри себя переменной типа `T`. Если это так, то экземпляр является незаконным, так как у вас не может быть переменных типа `String` без заданных границ.

Предположим, что в устаревшем коде нет незаконных экземпляров. Это означает, что в случае, подобном приведённому выше, на самом деле нет объявлений переменных типа `T`. В этом случае все, что вам нужно сделать, это изменить `generic` следующим образом:

```
generic
   type T(<>) is private;
package GP is
   ...
end GP;
```

Использование (< >) здесь означает, что создание экземпляра с неограниченным подтипом разрешено, и, следовательно, `generic` проверит, чтобы гарантировать, что никакие переменные типа `T` не объявлены в шаблоне, как вы видите из этого примера:

```
1. generic
2.    type T (<>) is private;
3. package GP is
4.    Var : T;
            |
   >>> unconstrained subtype not allowed (need initialization)

5. end;
```

Таким образом, следуя правилам Ada 95, нарушение контрактной модели теперь избегается. Либо `generic` допускает неограниченные подтипы, либо нет, и `generic` правильно проверяется при компиляции.

Таким образом, у нас здесь есть реальная несовместимость между Ada 83 и Ada 95, но это несовместимость исправляет реальную дыру в языке Ada 83, и легко исправить старый код Ada 83, чтобы соответствовать новым правилам Ada 95.

Связанный исходный

Attached Files отсутствуют

Файлы примеров Ada Gems распространяются AdaCore и могут быть использованы или изменены для любых целей без ограничений.

Об авторе

Robert Dewar (1945-2015)
http://www.adacore.com/press/adacore-president-robert-dewar-1945-2015/
AdaCore

Д-р Роберт Дьюар являлся соучредителем, президентом и генеральным директором AdaCore и заслуженным профессором компьютерных наук в Нью-Йоркском университете. Сфокусировавшись на разработке и внедрении языка программирования, д-р Дьюар был основным вкладчиком в Ada на протяжении всей своей эволюции и являлся главным разработчиком технологии AdaCore GNAT Ada. Он совместно разработал компиляторы для SPITBOL (SNOBOL), Realia COBOL для ПК (в настоящее время продается Computer Associates) и Alsys Ada, а также написал несколько операционных систем реального времени для Honeywell Inc. Д-р Дьюар поставлял документы и

презентации по вопросам языка программирования и сертификации по безопасности, а также как эксперт по компьютерам и законодательству, его часто приглашали на конференции, чтобы поговорить о программном обеспечении с открытым исходным кодом, вопросах лицензирования и смежных вопросах.

Last Updated: 10/13/2017
Posted on: 9/29/2008

Обсуждение…