Gem #61: Сопряжение с конструкторами C++

Автор: Javier Miranda, Arnaud Charlet, AdaCore

Краткое содержание: Gem Ada #61 - В предыдущем Gem, демонстрирующем генерацию связей с заголовочными файлами C++, было кратко упомянуто сопряжение Ada с конструкторами C++ с помощью прагмы CPP_Constructor.

В данном Gem представляются типичные примеры использования конструкторов C++ в GNAT в программах, написанных на смешанных языках, а в следующем Gem будет показано применение нескольких весьма полезных взаимодействий функций Ada 2005 и конструкторами C++.

Давайте начнем…

Предположим, что нам нужно взаимодействовать со следующим классом C ++:

```
class Root {
public:
   int  a_value;
   int  b_value;
   virtual int Get_Value ();
   Root();              // Default constructor
   Root(int v);         // 1st non-default constructor
   Root(int v, int w);  // 2nd non-default constructor
};
```

Используя генератор автоматической привязки или вручную, мы можем получить соответствующую спецификацию пакета:

```
with Interfaces.C; use Interfaces.C;
package Pkg_Root is

   type Root is tagged limited record
       A_Value : int;
       B_Value : int;
   end record;
   pragma Import (CPP, Root);

   function Get_Value (Obj : Root) return int;
   pragma Import (CPP, Get_Value);

   function New_Root return Root'Class;
   pragma Cpp_Constructor (New_Root, "_ZN4RootC1Ev");

   function New_Root (v : int) return Root'Class;
   pragma Cpp_Constructor (New_Root, "_ZN4RootC1Ei");

   function New_Root (v, w : int) return Root'Class;
   pragma Cpp_Constructor (New_Root, "_ZN4RootC1Eii");

end Pkg_Root;
```

На стороне Ada конструктор представлен функцией (имя которой произвольно), которая возвращает тип класса, соответствующий импортированному классу C ++. Тип возвращаемого значения требуется для класса, а не для конкретного типа Root, так что функция не будет рассматриваться как примитивная диспетчерская операция типа. Хотя конструктор описывается как функция, он обычно является процедурой с дополнительным неявным аргументом (инициализируемым объектом) на уровне реализации. GNAT выдает соответствующий вызов, какой бы он ни был, для инициализации объекта.

Конструкторы могут отображаться в следующих контекстах:

- как выражение инициализации объекта типа `T`;

- в качестве выражения инициализации компонента записи типа `T`;

- в ограниченном агрегате;

- в ограниченном агрегате, используемом как выражение оператора `return`.

Обратите внимание, что все вышеперечисленные контексты - это места, где Ada 2005 позволяет отображать ограниченные агрегаты и вызовы функций с ограниченными результатами, и поэтому на самом деле необходимо включить режим компиляции Ada 2005 (-gnat05), чтобы использовать эту функцию.

В объявлении объекта, тип которого является классом, импортированным из C ++, либо конструктор C ++ по умолчанию неявно вызывается GNAT, либо требуемый конструктор C ++ должен быть явно вызван в выражении, которое инициализирует объект. Например:

```
Obj1 : Root;
Obj2 : Root := New_Root;
Obj3 : Root := New_Root (v => 10);
Obj4 : Root := New_Root (30, 40);
```

Первые два объявления эквивалентны: в обоих случаях вызывается конструктор C ++ по умолчанию (в первом случае вызов конструктора является неявным, а в последнем случае вызов явным в объявлении объекта). Obj3 инициализируется конструктором C ++ nondefault, который принимает целочисленный аргумент, а Obj4 инициализируется конструктором C ++ nondefault, который принимает два целых числа.

Стоит отметить, что обычно не разрешается вызывать функцию класса для инициализации объекта определённого типа, и только в случае этих специальных импортированных конструкторских функций компилятор допускает это использование.

В следующем Gem #062 мы рассмотрим, как вывести и расширить импортированные классы C ++ на стороне Ada и покажем использование конструкторов в расширенном `return` Ada 2005 и ограниченных агрегатах.

Связанный со статьёй текст программы

Attached Files отсутствуют

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

Об авторе

Данные об авторе отсутствуют.

Last Updated: 10/13/2017
Posted on: 3/23/2009

Обсуждение…