Gem #22: Ада знает много национальных языков

Автор: Emmanuel Briot, Senior Software Engineer, AdaCore

Краткое содержание: Gem Ada #22 - Ада знает много национальных языков. Сколько именно? Все языки, которые можно написать на клавиатуре. Читайте дальше, чтобы узнать больше о данном Gem Ada.

Давайте начнем…

Единственными символами, разрешенными в программе Ada 83 (для строк, комментариев и идентификаторов), были 7-битные символы ASCII. Это раздражает. В Ада 83 мы не могли писать:

```
S : String : = "à la carte";
```

и еще меньше программистов пишут:

```
À_La_Carte : Boolean := False;
```

Это было – как бы это выразиться – “комплексное меню” - представление о сущностей языка :)

Поправка изменила эту ситуацию на 8-битные символы в течение жизни Ada 83. Ada 95 сделала это 8-битное изменение более четким и более официальным, обозначив ISO Latin-1 как набор символов. Таким образом, вышеизложенное является законным в Ada 95. Ada 95 также представила 16-битную поддержку стандарта ISO 10646 в виде Wide_Character. Можно написать в Ada 95:

```
My_Favorite_Pie : String := "п";   --  :)
```

но реализация не должна была позволять 16-разрядные символы в идентификаторах и комментариях, и Ada 95 не передал под мандат принятие следующего:

```
   п : constant :=
3.14159_26535_89793_23846_26433_83279_50288_41971_69399_37511;
   --  Can't eat this one :)
```

несмотря на то, что технология GNAT для Ada 95 позволяет полные 16-разрядные символы в идентификаторах и комментариях.

Ada 2005 является конечной с точки зрения открытости: поддерживается полный 32-битный набор символов ISO-10646, и использование π (например) в идентификаторах, комментариях и строках разрешено в программе Ada 2005. На самом деле пакет Ada.Numerics в Ada 2005 теперь содержит:

```
  Pi : constant :=
3.14159_26535_89793_23846_26433_83279_50288_41971_69399_37511;
  п : constant := Pi;
```

Чтобы продемонстрировать использование 32-битного набора символов ISO-10646 в программах Ada 2005, мы написали несколько программ на английском, русском и китайском языках. Эти программы принимают дату ISO начиная с 1983 по 2019 год и печатают дату в локальном формате. Каждая программа желает вам счастливого нового года, если введенная дата соответствует местной дате нового года. Все имена файлов имеют форму:

```
happy_new_year_{locale}_{lang}.adb
```

где {locale} - последовательность из двух букв, указывающая страну, для которой была написана программа (например, «us» для США, «cn» для Китая, «ru» для России) и {lang} язык, на котором программа (например, «en» для английского, «cn» для китайского, «ru» для русского). Например, «happy_new_year_cn_en.adb» - это программа для Китая, написанная на английском языке, а «happy_new_year_cn_cn.adb» - это программа для Китая, написанная на китайском языке. Наличие программы на английском языке для Китая позволяет не китайским программистам понять, что делает программа, и, возможно, изучать некоторые китайские выражения :)

Чтобы скомпилировать программы Ada, поставляемые с этим Gem, мы предлагаем вам использовать опции -gnatW8 -gnat05.

Наслаждайтесь ... Счастливые праздники и счастливый Новый год :)

Связанный исходный код

Attached Files

- happy_new_year_cn_cn.adb - (6 KB)

```
--     本程序从命令行读取一个标准的时间(ISO date)，如：
--     年 月 日，其中：
--
--      * 年的数值在区间 [1983，2019]内
--      * 月              [1，12]
--      * 日              [1，31]
--
--     经过转换，输出中文显示的时间。
--     如果您输入的时间恰好是当年的春节，该程序还会为您献上新春的祝福。

--     虽然16比特的字符串就可以满足中文格式的需要，但本程序统一使用32比特的字符串。

with Ada.Characters.Conversions; use Ada.Characters.Conversions;
with Ada.Command_Line;
with Ada.Wide_Wide_Text_IO;      use Ada.Wide_Wide_Text_IO;

--     注视：我们在这里使用了"Ada日历格式"（Ada.Calendar.Formatting）。

procedure 春节快乐 is

   抛出异常 : exception;
   --  当命令行输入格式不符合（年 月 日）要求时，抛出异常。

   subtype 中文字符串 is Wide_Wide_String;
   --  重命名Wide_Wide_String为中文字符串。

   function 字符串转换 (待转换串 : String) return 中文字符串
                        renames To_Wide_Wide_String;
   --  将将String格式字符串转换为Wide_Wide_String格式。

   function 数值转字符串 (数值 : Integer) return 中文字符串;
   --  将一个正数值转换为字符串格式。

   function 获得正数值 (字符 : String; 下限, 上限 : Positive) return Positive;
   --  输入一个字符（年，月或日），和一对下限，上限值，返回该字符所对应的正数值。
   --  如果该字符对应的不是一个正数值，或者该正数值不在区间[下限 .. 上限]内，打印
```

```
--  错误信息并抛出异常。
--  (此处用于核对，输入的"年"在 [1983, 2019] 内, "月"在 [1, 12] , "日"在 [1, 31] )

-----------------
-- 数值转字符串 --
-----------------

function 数值转字符串 (数值 : Integer) return 中文字符串 is
begin
   return 字符串转换 (数值'Img);
end 数值转字符串;

---------------
-- 获得正数值 --
---------------

function 获得正数值 (字符 : String; 下限, 上限 : Positive) return Positive is
   数值 : Integer;
begin
   数值 := Integer'Value (字符);

   if 数值 in 下限 .. 上限 then
      return 数值;

   else
      Put_Line ("'" & 字符串转换 (字符) & "': 数值不在规定范围内 " &
                数值转字符串 (下限) & " .. " & 数值转字符串 (上限));
      raise 抛出异常;
   end if;

exception
   when Constraint_Error =>
      Put_Line ("'" & 字符串转换 (字符) & "': 时间格式不正确");
      raise 抛出异常;
end 获得正数值;

subtype 年_数字值  is Positive range 1983 .. 2019;
subtype 月_数字值 is Positive range    1 ..   12;
subtype 日_数字值   is Positive range    1 ..   31;

年 : 年_数字值;
月 : 月_数字值;
日 : 日_数字值;

type 春节 is record
   月 : 月_数字值;
   日 : 日_数字值;
end record;

春节表 : constant array (年_数字值) of 春节 :=
  (1983 => (02, 13),
   1984 => (02, 02),
   1985 => (02, 20),
   1986 => (02, 09),
   1987 => (01, 29),
   1988 => (02, 17),
   1989 => (02, 06),
   1990 => (01, 27),
   1991 => (02, 15),
   1992 => (02, 04),
   1993 => (01, 23),
   1994 => (02, 10),
   1995 => (01, 31),
   1996 => (02, 19),
   1997 => (02, 07),
   1998 => (01, 28),
   1999 => (02, 16),
```

```
      2000 => (02, 05),
      2001 => (01, 24),
      2002 => (02, 12),
      2003 => (02, 01),
      2004 => (01, 22),
      2005 => (02, 09),
      2006 => (01, 29),
      2007 => (02, 18),
      2008 => (02, 07),
      2009 => (01, 26),
      2010 => (02, 14),
      2011 => (02, 03),
      2012 => (01, 23),
      2013 => (02, 10),
      2014 => (01, 31),
      2015 => (02, 19),
      2016 => (02, 08),
      2017 => (01, 28),
      2018 => (02, 16),
      2019 => (02, 05));

begin
   if Ada.Command_Line.Argument_Count /= 3 then
      raise 抛出异常;
   end if;

   年 := 获得正数值 (Ada.Command_Line.Argument (1), 年_数字值'First,  年_数字值'Last);
   月 := 获得正数值 (Ada.Command_Line.Argument (2), 月_数字值'First, 月_数字值'Last);
   日 := 获得正数值 (Ada.Command_Line.Argument (3), 日_数字值'First,   日_数字值'Last);

   --  将输入的标准时间转换为中文格式并对照春节表,
   --  如果输入时间恰好是春节，打印新春祝福。

   Put (数值转字符串(年) & "年 " & 数值转字符串(月)& "月 " & 数值转字符串(日) & "日");
   if 月 = 春节表 (年).月 and 日 = 春节表 (年).日 then
      Put_Line (" - 春节快乐! ");
   else
      New_Line;
   end if;

exception
   when 抛出异常 =>
      Put_Line ("您输入的格式不正确! ");
      Put_Line (" 格式提示： " & "(./)春节快乐" & "  年 月 日");
      Put_Line ("          年: " &
                数值转字符串(年_数字值'First)  & " .. " & 数值转字符串(年_数字值'Last));
      Put_Line ("          月  :    " &
                数值转字符串(月_数字值'First) & " ..   " & 数值转字符串(月_数字值'Last));
      Put_Line ("          日  :    " &
                数值转字符串(日_数字值'First) & " ..   " & 数值转字符串(日_数字值'Last));
end 春节快乐;
```

happy_new_year_cn_en.adb - (5 KB)

```
--  This program takes an ISO date specified in the command line as
--  YYYY MM DD, where:
--
--     * YYYY is a number between 1983 and 2019
--     * MM                          1 and   12
--     * DD                          1 and   31
--
--  and outputs the date in chinese format.
--  The date prints a "happy new year china" message when appropriate.

--  We use 32-bit Wide_Wide_Strings in the following program.
--  16-bit Wide_Strings would have been sufficient.
```

```
with Ada.Characters.Conversions; use Ada.Characters.Conversions;
with Ada.Command_Line;
with Ada.Wide_Wide_Text_IO;      use Ada.Wide_Wide_Text_IO;

procedure Happy_New_Year_Cn_En is

   Print_Usage_And_Exit : exception;
   --  Raised when the input format on the command line is not of the form
   --  YYYY MM DD.

   subtype String_Cn is Wide_Wide_String;
   --  Convenient renaming for a Wide_Wide_String.

   function To_String_Cn (Item : String) return String_Cn
                          renames To_Wide_Wide_String;
   --  Convenient renaming of To_Wide_Wide_String.

   function Img (Value : Integer) return String_Cn;
   --  Given an integer Value, returns the corresponding
   --  String_Cn representation.

   function Get_Value (Chars : String; Lo, Hi : Positive) return Positive;
   --  Given a string of characters Chars, and two positive bounds
   --  Lo and Hi, returns the positive number corresponding to its
   --  textual representation specified in Chars. If Chars does not
   --  correspond to an integer, or is not in the interval [Lo .. Hi]
   --  a message is printed and exception Print_Usage_And_Exit is raised.

   ---------
   -- Img --
   ---------

   function Img (Value : Integer) return String_Cn is
   begin
      return To_String_Cn (Value'Img);
   end Img;


   ---------------
   -- Get_Value --
   ---------------

   function Get_Value (Chars : String; Lo, Hi : Positive) return Positive is
      Value : Integer;
   begin
      Value := Integer'Value (Chars);

      if Value in Lo .. Hi then
         return Value;

      else
         Put_Line ("'" & To_String_Cn (Chars) & "': number not in range " &
                   Img (Lo) & " .. " & Img (Hi));
         raise Print_Usage_And_Exit;
      end if;

   exception
      when Constraint_Error =>
         Put_Line ("'" & To_String_Cn (Chars) & "': Bad number format");
         raise Print_Usage_And_Exit;
   end Get_Value;

   subtype Year  is Positive range 1983 .. 2019;
```

```
   subtype Month is Positive range    1 ..   12;
   subtype Day   is Positive range    1 ..   31;

   Y : Year;
   M : Month;
   D : Day;

   type Year_Start is record
      M : Month;
      D : Day;
   end record;

   Chinese_New_Year : constant array (Year) of Year_Start :=
     (1983 => (02, 13),
      1984 => (02, 02),
      1985 => (02, 20),
      1986 => (02, 09),
      1987 => (01, 29),
      1988 => (02, 17),
      1989 => (02, 06),
      1990 => (01, 27),
      1991 => (02, 15),
      1992 => (02, 04),
      1993 => (01, 23),
      1994 => (02, 10),
      1995 => (01, 31),
      1996 => (02, 19),
      1997 => (02, 07),
      1998 => (01, 28),
      1999 => (02, 16),
      2000 => (02, 05),
      2001 => (01, 24),
      2002 => (02, 12),
      2003 => (02, 01),
      2004 => (01, 22),
      2005 => (02, 09),
      2006 => (01, 29),
      2007 => (02, 18),
      2008 => (02, 07),
      2009 => (01, 26),
      2010 => (02, 14),
      2011 => (02, 03),
      2012 => (01, 23),
      2013 => (02, 10),
      2014 => (01, 31),
      2015 => (02, 19),
      2016 => (02, 08),
      2017 => (01, 28),
      2018 => (02, 16),
      2019 => (02, 05));

begin
   if Ada.Command_Line.Argument_Count /= 3 then
      raise Print_Usage_And_Exit;
   end if;

   Y := Get_Value (Ada.Command_Line.Argument (1), Year'First,  Year'Last);
   M := Get_Value (Ada.Command_Line.Argument (2), Month'First, Month'Last);
   D := Get_Value (Ada.Command_Line.Argument (3), Day'First,   Day'Last);

   --  Convert the ISO date to the chinese format and
   --  check for the beginning of the year.

   Put (Img (Y) & " " & Img (M) & " " & Img (D));
```

```
   if M = Chinese_New_Year (Y).M and D = Chinese_New_Year (Y).D then
      Put_Line (" - Happy New Year China");
   else
      New_Line;
   end if;

exception
   when Print_Usage_And_Exit =>
      Put_Line ("Usage: " & To_String_Cn (Ada.Command_Line.Command_Name) &
                "  YYYY  MM  DD");
      Put_Line ("          YYYY: " &
                Img(Year'First) & " .. " & Img (Year'Last));
      Put_Line ("          MM  :    " &
                Img (Month'First) & " ..   " & Img (Month'Last));
      Put_Line ("          DD  :    " &
                Img (Day'First) & " ..   " & Img (Day'Last));
end Happy_New_Year_Cn_En;
```

happy_new_year_ru_en.adb - (4 KB)

```
--  This program takes an ISO date specified in the command line as
--  YYYY MM DD, where:
--
--     * YYYY is a number between 1983 and 2019
--     * MM                         1 and   12
--     * DD                         1 and   31
--
--  and outputs the date in Julian Calendar (referred to as "Old Style"
--  in Russia, and hence New Year in Old Style is jokingly called "Old
--  New Year" in Russia)
--  The program prints a "happy old new year Russia" message when
appropriate.
--  The algorithm is simple: since in the chosen period (in fact, from 1900
--  until 2100) the difference between Julian and Gregorian calendars is 13
--  days, we simply convert one to the other using Ada.Calendar.Arithmetic.

with Ada.Calendar.Arithmetic; use Ada.Calendar, Ada.Calendar.Arithmetic;
with Ada.Command_Line;
with Ada.Text_IO;      use Ada.Text_IO;

--  Note: we could have also used Ada 2005 Ada.Calendar.Formatting

procedure Happy_New_Year_RU is

   Print_Usage_And_Exit : exception;
   --  Raised when the input format on the command line is not of the form
   --  YYYY MM DD.

   function Get_Value (Chars : String; Lo, Hi : Positive) return Positive;
   --  Given a string of characters Chars, and two positive bounds
   --  Lo and Hi, returns the positive number corresponding to its
   --  textual representation specified in Chars. If Chars does not
   --  correspond to an integer, or is not in the interval [Lo .. Hi]
   --  a message is printed and exception Print_Usage_And_Exit is raised.

   ---------------
   -- Get_Value --
   ---------------

   function Get_Value (Chars : String; Lo, Hi : Positive) return Positive is
      Value : Integer;
   begin
      Value := Integer'Value (Chars);
```

```
      if Value in Lo .. Hi then
         return Value;

      else
         Put_Line ("'" & Chars & "': number not in range " &
                   Lo'Img & " .. " & Hi'Img);
         raise Print_Usage_And_Exit;
      end if;

   exception
      when Constraint_Error =>
         Put_Line ("'" & Chars & "': Bad number format");
         raise Print_Usage_And_Exit;
   end Get_Value;

   subtype Year  is Positive range 1982 .. 2019;
   subtype Month is Positive range    1 ..   12;
   subtype Day   is Positive range    1 ..   31;

   Y : Year;
   M : Month;
   D : Day;
   S : Duration;
   T : Time;

   Julian_Gregorian_Difference : constant := 13;

   type Month_Name is (January, February, March, April, May, June,
                       July, August, September, October, November,
                       December);

begin
   if Ada.Command_Line.Argument_Count /= 3 then
      raise Print_Usage_And_Exit;
   end if;

   Y := Get_Value (Ada.Command_Line.Argument (1), Year'First,  Year'Last);
   M := Get_Value (Ada.Command_Line.Argument (2), Month'First, Month'Last);
   D := Get_Value (Ada.Command_Line.Argument (3), Day'First,   Day'Last);

   --  Convert the ISO date to the Julian calendar and
   --  check for the beginning of the year.

   T := Time_Of (Y, M, D);
   T := T - Julian_Gregorian_Difference;
   Split (T, Y, M, D, S);

   Put (Month_Name'Val (M - 1)'Img & D'Img & "," & Y'Img);
   if M = 1 and D = 1 then
      Put_Line (" - Happy Old New Year Russia");
   else
      New_Line;
   end if;

exception
   when Print_Usage_And_Exit =>
      Put_Line ("Usage: " & Ada.Command_Line.Command_Name &
                "  YYYY  MM  DD");
      Put_Line ("          YYYY: " &
                Year'First'Img  & " .. " & Year'Last'Img);
      Put_Line ("          MM  :    " &
                Month'First'Img & " ..   " & Month'Last'Img);
      Put_Line ("          DD  :    " &
```

```
                Day'First'Img   & " ..   " & Day'Last'Img);
end Happy_New_Year_RU;
```

happy_new_year_ru_ru.adb - (6 KB)

```
--  Эта программа получает в качестве параметров командной строки дату ISO в
--  формате ГГГГ ММ ДД, где:
--
--     * ГГГГ это число в диапазоне от 1983 до 2019
--     * ММ                            1 до   12
--     * ДД                            1 до   31
--
--  и распечатывает соответствующюю ей дату по старому стилю
--  Программа также печатает поздравление со "старым новым годом" в
--  соответствующих случаях.
--  Алгоритм прост: поскольку в выбранный период (если быть точным, в период
--  с 1900 по 2100 годы) разница между старым и новым стилем составляет 13
--  дней, мы просто переводим одну дату в другую используя функции из
--  пакета Ada.Calendar.Arithmetic.

with Ada.Calendar.Arithmetic;    use Ada.Calendar, Ada.Calendar.Arithmetic;
with Ada.Characters.Conversions; use Ada.Characters.Conversions;
with Ada.Command_Line;
with Ada.Wide_Text_IO;           use Ada.Wide_Text_IO;

--  Замечание: можно было бы воспользоваться для печати функции языка
--  Ада 2005 из пакета Ada.Calendar.Formatting

procedure Happy_New_Year_RU_RU is

   Выдать_Подсказку_И_Выйти : exception;
   --  Возбуждается когда входные параметры не соответствуют формату
   --  ДДДД ММ ГГ.

   function Печать_Числа (Число : Integer) return Wide_String;
   --  Возвращает текстовое представление числа

   function Число (Строка : String; Низ, Верх : Positive) return Positive;
   --  На основе строки Строка и двух положительных пределов Низ и Верх,
   --  возвращает положительное число, соответствующее его текстовому
   --  представлению в Строка. Если Строка не содержит числа, или число
   --  не принадлежит интервалу [Низ .. Верх], печатается сообщение и
   --  возбуждается исключение Выдать_Подсказку_И_Выйти.

   ------------------
   -- Печать_Числа --
   ------------------

   function Печать_Числа (Число : Integer) return Wide_String is
   begin
      return To_Wide_String (Число'Img);
   end Печать_Числа;

   -----------
   -- Число --
   -----------

   function Число (Строка : String; Низ, Верх : Positive) return Positive is
      Результат : Integer;
   begin
      Результат := Integer'Value (Строка);

      if Результат in Низ .. Верх then
```

```
         return Результат;

      else
         Put_Line ("'" & To_Wide_String (Строка) &
              "': значение не принадлежит диапазону" &
              Positive'Wide_Image (Низ) & " .." & Positive'Wide_Image(Верх));
         raise Выдать_Подсказку_И_Выйти;
      end if;

   exception
      when Constraint_Error =>
         Put_Line ("'" & To_Wide_String (Строка) &
                   "': неверный формат данных");
         raise Выдать_Подсказку_И_Выйти;
   end Число;

   subtype Год   is Positive range 1982 .. 2019;
   subtype Месяц is Positive range    1 ..   12;
   subtype День  is Positive range    1 ..   31;

   Г : Год;
   М : Месяц;
   Д : День;
   С : Duration;
   Т : Time;

   Разница_Старый_Новый : constant := 13;

   type Наименование_Месяца is (Января, Февраля, Марта, Апреля, Мая, Июня,
                        Июля, Августа, Сентября, Октября, Ноября,
                        Декабря);

begin
   if Ada.Command_Line.Argument_Count /= 3 then
      raise Выдать_Подсказку_И_Выйти;
   end if;

   Г := Число (Ada.Command_Line.Argument (1), Год'First,   Год'Last);
   М := Число (Ada.Command_Line.Argument (2), Месяц'First, Месяц'Last);
   Д := Число (Ada.Command_Line.Argument (3), День'First,  День'Last);

   --  Переводим дату ISO в старый стиль и проверяем на начало года.

   Т := Time_Of (Г, М, Д);
   Т := Т - Разница_Старый_Новый;
   Split (Т, Г, М, Д, С);

   Put (Печать_Числа (Д) & " " &
        Наименование_Месяца'Wide_Image (Наименование_Месяца'Val (М - 1)) &
        Печать_Числа (Г) & " ГОДА");
   if М = 1 and Д = 1 then
      Put_Line (" - со Старым Новым Годом, Россия!");
   else
      New_Line;
   end if;

exception
  when Выдать_Подсказку_И_Выйти =>
   Put_Line ("Параметры: " & To_Wide_String (Ada.Command_Line.Command_Name) &
               "  ГГГГ  ММ  ДД");
      Put_Line ("          ГГГГ: " &
              Печать_Числа (Год'First)   & " .. " & Печать_Числа (Год'Last));
      Put_Line ("          ММ  :    " &
          Печать_Числа (Месяц'First) & " ..   " & Печать_Числа (Месяц'Last));
```

```
      Put_Line ("          ДД  :    " &
           Печать_Числа (День'First)  & " ..   " & Печать_Числа (День'Last));
end Happy_New_Year_RU_RU;
```

happy_new_year_us_en.adb - (3 KB)

```
--  This program takes an ISO date specified in the command line as
--  YYYY MM DD, where:
--
--     * YYYY is a number between 1983 and 2019
--     * MM                         1 and   12
--     * DD                         1 and   31
--
--  and outputs the date in US format.
--  The date prints a "happy new year US" message when appropriate.

with Ada.Command_Line;
with Ada.Text_IO;      use Ada.Text_IO;

--  Note: we could have also used Ada 2005 Ada.Calendar.Formatting

procedure Happy_New_Year_US_En is

   Print_Usage_And_Exit : exception;
   --  Raised when the input format on the command line is not of the form
   --  YYYY MM DD.

   function Get_Value (Chars : String; Lo, Hi : Positive) return Positive;
   --  Given a string of characters Chars, and two positive bounds
   --  Lo and Hi, returns the positive number corresponding to its
   --  textual representation specified in Chars. If Chars does not
   --  correspond to an integer, or is not in the interval [Lo .. Hi]
   --  a message is printed and exception Print_Usage_And_Exit is raised.

   ---------------
   -- Get_Value --
   ---------------

   function Get_Value (Chars : String; Lo, Hi : Positive) return Positive is
      Value : Integer;
   begin
      Value := Integer'Value (Chars);

      if Value in Lo .. Hi then
         return Value;

      else
         Put_Line ("'" & Chars & "': number not in range " &
                   Lo'Img & " .. " & Hi'Img);
         raise Print_Usage_And_Exit;
      end if;

   exception
      when Constraint_Error =>
         Put_Line ("'" & Chars & "': Bad number format");
         raise Print_Usage_And_Exit;
   end Get_Value;

   subtype Year  is Positive range 1983 .. 2019;
   subtype Month is Positive range    1 ..   12;
   subtype Day   is Positive range    1 ..   31;
```

```
   Y : Year;
   M : Month;
   D : Day;

   type Month_Name is (January, February, March, April, May, June,
                       July, August, September, October, November,
                       December);

begin
   if Ada.Command_Line.Argument_Count /= 3 then
      raise Print_Usage_And_Exit;
   end if;

   Y := Get_Value (Ada.Command_Line.Argument (1), Year'First,  Year'Last);
   M := Get_Value (Ada.Command_Line.Argument (2), Month'First, Month'Last);
   D := Get_Value (Ada.Command_Line.Argument (3), Day'First,   Day'Last);

   --  Convert the ISO date to the US format and
   --  check for the beginning of the year.

   Put (Month_Name'Val (M - 1)'Img & D'Img & "," & Y'Img);
   if M = 1 and D = 1 then
      Put_Line (" - Happy New Year US");
   else
      New_Line;
   end if;

exception
   when Print_Usage_And_Exit =>
      Put_Line ("Usage: " & Ada.Command_Line.Command_Name &
                "  YYYY  MM  DD");
      Put_Line ("          YYYY: " &
                Year'First'Img  & " .. " & Year'Last'Img);
      Put_Line ("          MM  :    " &
                Month'First'Img & " ..   " & Month'Last'Img);
      Put_Line ("          DD  :    " &
                Day'First'Img   & " ..   " & Day'Last'Img);
end Happy_New_Year_US_En;
```

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

Обсуждение…

3 ответа на “Gem № 22: Ada Speaks Many Languages”

1. 20-го декабря 2007 в 1:57

Stanislaw Goldstein сказал:

Я очень рад видеть этот Gem, поскольку я боролся в течение длительного времени, чтобы видеть символы неASCII в Ada. Я скомпилировал программы и выполнил их. К сожалению, результаты не очень удовлетворительные, поскольку я не видел ни российского ни китайского текста в окне вывода и/или консоли. Проблема уже решена? (У меня есть академическая версия 2007-2 GPS и компилятора на Windows). Кто-либо был в состоянии видеть вывод, поскольку он должен быть похожим?

2. 27-го декабря 2007 в 12:31

Stefan Bellon сказал:

Я не думаю, что это - проблема с компилятором, а в настройке Вашей консоли. Конечно, Вам нужна консоль, которая способна к отображению закодированного текста UTF-8, и имейть необходимые шрифты в наличии. Я просто настроил gnome-terminal (набор к UTF-8) и uxterm, и happy_new_year_cn_cn выводы приятно отформатировали китайские даты (и новогоднее сообщение на 18.02.2007 и 07.02.2008, таким образом, похоже, что это работает). Не то, чтобы я понимаю китайский язык, конечно, … ;-)

3. 21-го января 2008 в 20:16

Stanislaw Goldstein сказал:

Вы правы, это не проблема компилятора. К счастью, проблема была решена для меня Robert Dewar и Nicolas Setton для окна вывода GPS – оказалось, что нужно установить НАБОР СИМВОЛОВ переменной окружения в UTF-8. У меня все еще есть проблема с консолью окон (запускался с cmd). Alghout, который консоль может показать символам UTF-8, используя Консольный шрифт Lucida (от справедливо ограниченного подмножества Unicode), и можно изменить кодовую страницу на UTF-8 (кодовая страница 65001), программы показывают ошибку периода выполнения. Я предоставлю подробную информацию в билете GB04-005.