

Ludovic Rousseau's blog

My activities related to smart card and Free Software (as in free speech).

Friday, August 20, 2010

PCSC sample in Ada

Here is the PCSC sample in Ada language I promised in PC/SC sample in different languages.

Installation

The PCSC/Ada project is hosted at <http://www.nongnu.org/pcscada>.

PCSC/Ada is available as package in Debian [testing/unstable](#) and Ubuntu [10.04](#).

Debian squeeze (testing at time of writing this article)

To install the PCSC/Ada development package type:

```
$ sudo apt-get install libpcscada1-dev
```

Debian Lenny (stable)

If you use Debian stable (Lenny), an unofficial backport is also available from [code-labs.ch](#). To install the backport package, perform the following steps:

Add this line to `/etc/apt/sources.list`:

```
deb http://www.code-labs.ch/debian lenny-backports main contrib non-free
```

Then, execute the following commands:

```
$ sudo apt-get update
$ sudo apt-get install debian-code-labs-archive-keyring
$ sudo apt-get update
```

To install the PCSC/Ada development package type:

```
$ sudo apt-get install libpcscada1-dev
```

If you want to compile PCSC/Ada from source, see the distributed README file for details.

API

The API documentation is available online at <http://www.nongnu.org/pcscada/api/index.html>.

Source code

Makefile

```
all: ada_sample

ada_sample:
    @gnatmake -p -Ppcscada_sample
```

Google+ Badge

Ludovic Rousseau blog



Follow

Blog Archive

- 2017 (33)
- 2016 (49)
- 2015 (51)
- 2014 (61)
- 2013 (38)
- 2012 (27)
- 2011 (46)
- ▼ 2010 (55)
 - December (5)
 - November (5)
 - October (9)
 - September (1)
 - ▼ August (8)
 - RAM and CPU improvements in pcsc-lite 1.6.x
 - PCSC sample in Ada
 - How to help my projects?
 - pcsc-lite 1.6.x breaks some programs at compilatio...
 - ATR not (yet) in my database
 - PCSC API spy for GNU systems
 - My blog messages license
 - New CCID 1.4.0 and card movement notification mech...
 - July (1)
 - June (10)
 - May (6)
 - April (10)

Search This Blog

Subscribe To

- Posts ▼
- Comments ▼

Google+ Followers

```
clean:
@rm -rf obj
```

pcscada_sample.gpr

```
with "pcscada";

project PCSCAda_Sample is

  for Object_Dir use "obj";
  for Source_Dirs use (".");
  for Main use ("ada_sample.adb");

  Compiler_Switches := ("-gnaty3aAbcdefhiIklnprStuxM80o",
                        "-gnatVa",
                        "-gnat05",
                        "-gnatwal",
                        "-gnatf",
                        "-fstack-check",
                        "-gnato",
                        "-g");

  package Compiler is
    for Default_Switches ("ada") use Compiler_Switches;
  end Compiler;

  package Binder is
    for Default_Switches ("ada") use ("-E");
  end Binder;

end PCSCAda_Sample;
```

ada_sample.adb

```
with Ada.Text_IO;

with PCSC.SCard.Utils;

use PCSC;

procedure Ada_Sample is

  package SCU renames SCard.Utils;

  My_Context : SCard.Context;
  First_Card : SCard.Card;
  Readers    : SCard.Reader_ID_Set;
  Recv_PCI   : SCard.IO_Request;
  Recv_Len   : Natural := 0;

  APDU_Select : constant SCard.Byte_Set :=
    (16#00#, 16#A4#, 16#04#, 16#00#, 16#0A#,
     16#A0#, 16#00#, 16#00#, 16#00#, 16#62#,
     16#03#, 16#01#, 16#0C#, 16#06#, 16#01#);
  APDU_Command : constant SCard.Byte_Set :=
    (16#00#, 16#00#, 16#00#, 16#00#);

begin

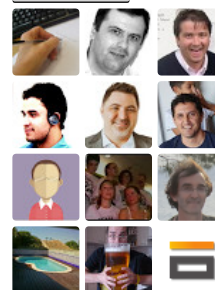
  -- Establish context

  SCard.Establish_Context (Context => My_Context,
                           Scope  => SCard.Scope_System);

  -- List readers
```

Ludovic Rousseau b...

Follow



336 have us in circles

[View all](#)

```

Readers := SCard.List_Readers (Context => My_Context);
Ada.Text_IO.Put_Line ("Readers found:");
SCU.For_Every_Reader (Readers => Readers,
                     Call  => SCU.Print_ReaderID'Access);

-- Connect with the card in first reader

SCard.Connect (Context => My_Context,
               Card    => First_Card,
               Reader  => Readers.First_Item,
               Mode    => SCard.Share_Shared);

-- Send APDUs

Send_Select_Applet_APDU :
declare
  Recv_Buffer : SCard.Byte_Set (1 .. 128);
begin
  SCard.Transmit (Card      => First_Card,
                  Send_Buffer => APDU_Select,
                  Recv_Pci   => Recv_PCI,
                  Recv_Buffer => Recv_Buffer,
                  Recv_Len   => Recv_Len);
  Ada.Text_IO.Put_Line
    ("Select applet: " & SCU.To_Hex_String
     (Given => Recv_Buffer,
      Len  => 2 * Recv_Len));
end Send_Select_Applet_APDU;

Send_Command_APDU :
declare
  Recv_Buffer : SCard.Byte_Set (1 .. 32);
begin
  SCard.Transmit (Card      => First_Card,
                  Send_Buffer => APDU_Command,
                  Recv_Pci   => Recv_PCI,
                  Recv_Buffer => Recv_Buffer,
                  Recv_Len   => Recv_Len);
  Ada.Text_IO.Put_Line
    ("Command      : "
     & SCU.To_String (Given => Recv_Buffer (1 .. Recv_Len)));
end Send_Command_APDU;

-- Disconnect the card

SCard.Disconnect (Card  => First_Card,
                  Action => SCard.Unpower_Card);

-- Release context

SCard.Release_Context (Context => My_Context);

exception
when others =>
  Ada.Text_IO.Put_Line ("OOPS - got an exception:");
  if SCard.Is_Valid (Context => My_Context) then
    SCard.Release_Context (Context => My_Context);
  end if;

  raise;
end Ada_Sample;

```

Compilation

Just type make.

```

$ make
object directory "/home/rousseau/blog/pcscada_sample/obj" created

```

```
gcc-4.4 -c -gnaty3aAbcdefhiIklnprStuxM80o -gnatVa -gnat05 -gnatwal -gnatf -fstack-  
check -gnato -g -I- -gnatA /home/rousseau/blog/pcscada_sample/ada_sample.adb  
gnatbind -shared -E -I- -x /home/rousseau/blog/pcscada_sample/obj/ada_sample.ali  
gnatlink /home/rousseau/blog/pcscada_sample/obj/ada_sample.ali -shared-libgcc -L/usr/l  
ib -lpcscada -o /home/rousseau/blog/pcscada_sample/obj/ada_sample
```

The generated binary is `./obj/ada_sample`.

Output

```
$ ./obj/ada_sample  
Readers found:  
Gemalto GemPC Twin 00 00  
Select applet: 9000  
Command      : Hello world!◆
```

Conclusion

I do not use Ada myself so I can't really say more. This wrapper should do the job if you do use Ada.

Thanks a lot to Reto Buerki, the author of the Ada wrapper, for writing the sample code for me.



Labels: [code](#), [pcsc-lite](#)

[Newer Post](#)

[Home](#)

[Older Post](#)

Bitcoin



License: [by-nc-sa](#)



This blog by [Ludovic Rousseau](#) is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License](#).

Simple theme. Powered by [Blogger](#).