Experience in 40 Years of Teaching Ada

Jean-Pierre Rosen
Adalog
www.adalog.fr
The Students

- Teaching for industrials
  - All students are working in a company
  - From fresh-out-of-school to experienced
  - Mostly developers, sometimes Q/A or certification people

- No beginners
  - They know at least one other programming language
  - ... and they may have bad habits
Teaching the language

• General syntax is easy
  • Remove these damn () from the condition in if statements!

• But use the special features to stress safety
  • Completeness of case, safety of for loop...
  • Every feature has a goal

• Show the consistency of the language

• Evolution of past experience of students changes the difficulties...
  • Exceptions are now well understood
  • Various levels of experience with concurrency
  • ... but some have difficulties understanding a variable exists even if there is no new!
Teaching How To Use The Language

• Don’t teach how to use Ada for programming
  • Teach how to program *in* Ada

• Stress :
  • Information hiding
  • Specifying before implementing (and checking the specs)
  • Paying attention to the choice of data structures
  • Formal "proofs"

• Tell to use the compiler often to check the code
  • Compile every 10 lines
  • The compiler is your friend!
    • Read error messages
    • Say "thank you" to error messages
    • Don’t try to remove error messages; understand how to fix your code
Examples Of Wrong Reactions

D : Duration;
begin
  D := 1;

Expected type Standard.Duration, found an integer type

D : Duration;
begin
  D := Duration(1);

Wrong!

D : Duration;
begin
  D := 1.0;

Wrong!

if Data_Valid then ...
end ...
Hard Points

• Ada has a precise, but special vocabulary
  • Need to explain (especially in error messages)
    • Subtype mark, subprograms, elaboration...

• Features without equivalent in other languages
  • Discriminants
  • Rendezvous
  • Class-wide types

• Things with similar features in other languages
  • Hard to "unlearn"
  • People without experience understand better sometimes!
The Choice of Exercises (1)

- Avoid "Hello World" exercises
  - Not enough time...

- First exercise:
  - General features of the language
  - Write a package, stress specification vs implementation
  - Opportunity to define proper types, chase usage of Integer!

- Exercise on generics
  - Nobody really understands generics until the exercise!

- Exercise on access types
  - Implicit dereference is easily adopted
  - Oh! you can write a linked list without any core dump!
The Choice of Exercises (2)

• OOP
  • Stress what a class is, class-wide types and class-wide operations

• Tasking with rendezvous
  • Students have difficulties understanding that accept is a sequential statement

• Tasking with protected types/requeue

• Distribution demo
Software Engineering?

- **Strong typing**
  - If there is no point, it is an Integer, if there is a point it is a Float

- **Modularity?**

- **2/3 of the students have difficulties with writing a simple linked list.**

- **Many change the (agreed upon) specification when developing the body**

There is no problem with teaching Ada as a language