Tutorial Form

Title: No Pointers, Great Programs. How to stay on the value semantics side of the Ada way with the help of container libraries.

Presenter : Mário Amado Alves
Contact name: Mário Amado Alves
Contact address: LIAAC, room G. 1.2, Rua Campo Alegre 823, 4150-180 PORTO, Portugal
Contact phone: 351-226078830, ext. 172
Contact fax: 351-226003654
Contact email: amado.alves@netcabo.pt
Requested level: beginner
Background: The tutorial is directed primarily at newcomers to Ada—perhaps making the transition from C, C++, or Java. In any case a background in programming is desirable. The tutorial is also appropriate for experienced Adaists in the pointer idiom who want to try a different approach.

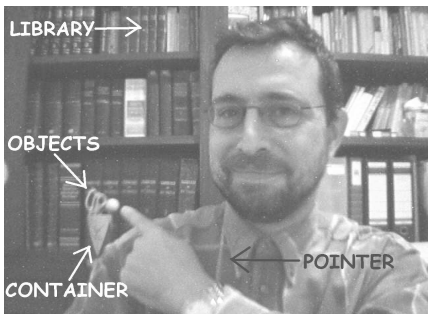
Abstract

How would you like writing complex object-oriented programs without the words **new**, **access**, **null**, or **Free**? Well, there is way—the Ada way—, and that's what we'll learn in this tutorial.

Pointers are evil. Although Ada has succeeded in averting many of their dangers (in relation to other languages), pointers in Ada (access types and values) are still considered a nuisance to say the least in many situations. For example, heterogeneous arrays, ragged arrays, recursive types: you cannot write these things strictly in Ada without using pointers. But you can have them *encapsulated*, inside the proper container type—and then forget about them and get on with your life of writing great programs. We'll study several of these magical containers, starting with the wonderful Unbounded_String—which we'll take as our paradigmatic example—, and moving on to fully fledged container libraries like Charles. We'll also review the Ada foundations for value semantics (e.g. unconstrained types), strict Ada 95 containers (arrays, strings, files), generics, and class-wide programming.

Presenter summary

Mário Amado Alves is currently with the University of Porto, doing PhD research on adaptive hypertext. He has degrees in Linguistics and Informatics Engineering. He has thaugth Ada, Software Engineering and other courses at the New University of Lisbon, Open University of Portugal, and other places. He has been a software developer since the mid 1980's and an Adaist since the mid 1990's. He has been on the Ada Standard Container Library Working Group since its inception in the Ada-Europe 2002 workshop. Lately he has been a contributor to Ada Issue 302 for the creation of such a library for Ada 2005, where he has focused on indefinite elements and persistence.



Why you should participate in this tutorial? Because pointers are evil, and although they are safer in Ada than in other languages, they are still a nuisance in many situations (see abstract). Fortunately, we can escape the pointer tyranny with the help of properly designed container libraries and a focus on value semantics--an area where Ada also excels. If you're a newcomer to the Ada way, you'll benefit from starting your journey on the value semantics side of it. If you're an experienced Adaist, unsatisfied with the pointer idiom, you'll want to try a new approach. In any case: because you want to write great programs without the words new, access, null or free --and therefore with increased understandability and safety.