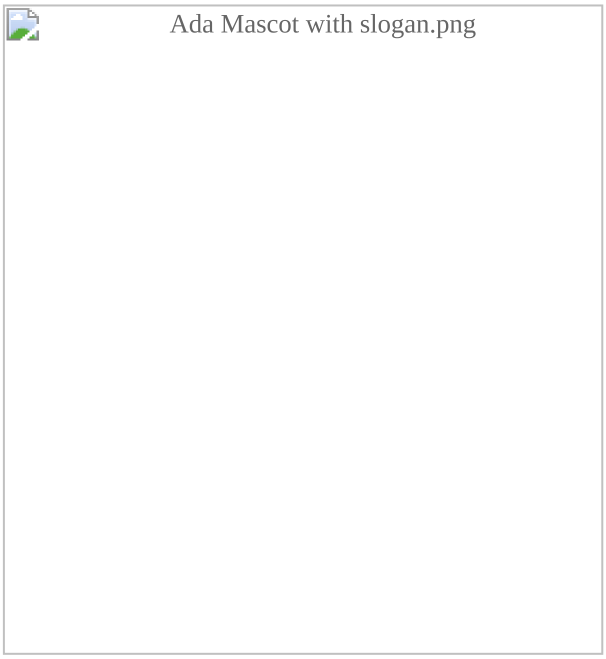
Ada Programming


Ada Mascot with slogan.png

Welcome to the **Ada Programming** tutorial at Wikibooks. This is the first Ada tutorial covering the Ada 2005 and 2012 standards. If you are a beginner you will learn the latest standard — if you are a seasoned Ada user you can see what's new.

Current Development Stage for **Ada Programming** is "". At this date, there are more than 200 pages in this book, which makes **Ada Programming** one of the largest programming wikibooks.[1]

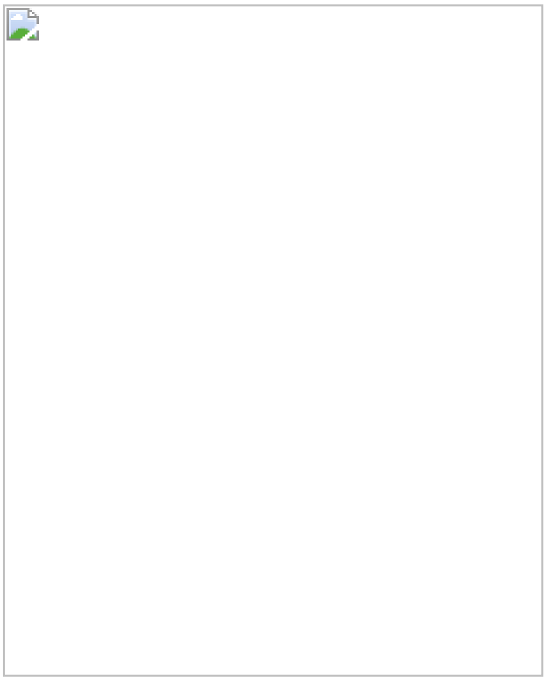
But still there is always room for improvement — do help us to expand **Ada Programming**. Even beginners will find areas to participate.

About Ada

Ada is a programming language suitable for all development needs. It has built-in features that directly support structured, object-oriented, generic, distributed and concurrent programming.

Ada is a good choice for Rapid Application Development, Extreme Programming (XP), and Free Software development.

Ada is named after Augusta Ada King-Noel, Countess of Lovelace.


Augusta Ada King, Countess of Lovelace.

Programming in the large

Ada puts unique emphasis on, and provides strong support for, good software engineering practices that scale well to very large software systems (millions of lines of code, and very large development teams). The following language features are particularly relevant in this respect:

- An extremely strong, static and safe **type system**, which allows the programmer to construct powerful abstractions that reflect the real world, and allows the compiler to detect many logic faults before they become errors.
- **Modularity**, whereby the compiler directly manages the construction of very large software systems from sources.
- **Information hiding**; the language separates interfaces from implementation, and provides fine-grained control over visibility.
- **Readability**, which helps programmers review and verify code. Ada favours the reader of the program over the writer, because a program is written once but read many times. For example, the syntax bans all ambiguous constructs, so there are no surprises, in accordance with the Tao of Programming's Law of Least Astonishment. (Some Ada programmers are reluctant to talk about *source code* which is often cryptic; they prefer *program text* which is close to English prose.)
- **Portability**: the language definition allows compilers to differ only in a few controlled ways, and otherwise defines the semantics of programs very precisely; as a result, Ada source text is very portable across compilers and across target hardware platforms. Most often, the program can be recompiled without any changes.[2]
- **Standardisation**: standards have been a goal and a prominent feature ever since the design of the language in the late 1970s. The first standard was published in 1980, just 3 years after design commenced. Ada compilers all

support exactly the same language; the only dialect, SPARK, is merely an annotated subset and can be compiled with an Ada compiler.

Consequences of these qualities are superior **reliability**, **reusability** and **maintainability**. For example, compared to programs written in C, programs written in Ada 83 contain *"70% fewer internal fixes and 90% fewer bugs"*, and cost half as much to develop in the first place.[3] Ada shines even more in software maintenance, which often accounts for about 80% of the total cost of development. With support for object-oriented programming, Ada 95 may bring even more cost benefit, depending on how objects are used; although no serious study comparable to Zeigler's has been published.

Programming in the small

In addition to its support for good software engineering practices, which are applicable to general-purpose programming, Ada has powerful specialised features supporting **low-level programming** for real-time, safety-critical and embedded systems. Such features include, among others, machine code insertions, address arithmetic, low-level access to memory, control over bitwise representation of data, bit manipulations, and a well-defined, statically provable concurrent computing model called the Ravenscar Profile.

Other features include restrictions (it is possible to restrict which language features are accepted in a program) and features that help review and certify the object code generated by the compiler.

Several vendors provide Ada compilers accompanied by minimal run-time kernels suitable for use in certified, life-critical applications. It is also possible to write Ada programs which require no run-time kernel at all.

It should come as no surprise that Ada is heavily used in the aerospace, defence, medical, railroad, and nuclear industries.

The Language Reference Manual

The Ada Reference Manual (RM) is the official language definition. If you have a problem and no one else can help, you should read the RM (albeit often a bit cryptic for non-language-lawyers). For this reason, all complete (not draft) pages in **Ada Programming** contain links into the appropriate pages in the RM.

This tutorial covers *Ada Reference Manual — ISO/IEC 8652:2012(E) Language and Standard Libraries*, colloquially known as *Ada 2012* or just *Ada*.

You can browse the complete Reference Manual at http://www.ada-auth.org/standards/12rm/html/RM-TTL.html

There are two companion documents:

- The Annotated Reference Manual, an extended version of the RM aimed at compiler writers or other persons who want to know the fine details of the language.
- The Rationale for Ada 2012, an explanation of the features of the language.

The Ada Information Clearinghouse also offers the older Ada 83, 95, and 2005 standards and companion documents.

The RM is a collective work under the control of Ada users. If you think you've found a problem in the RM, please report it to the Ada Conformity Assessment Authority. On this site, you can also see the list of "Ada Issues" raised by other people.

Ada Conformity Assessment Test Suite

Unlike other programming languages, Ada compilers are officially tested, and only those which pass this test are accepted, for military and commercial work. This means that all Ada compilers behave (almost) the same, so you do not have to learn any dialects. But because the Ada standard allows the compiler writers to include some additions, you could learn a cool new feature only to find out that your favourite compiler does not support it...

Programming in Ada

Getting Started

Where to get a compiler, how to compile the source, all answered here:

- Basic Ada — **Read Me First!**
- Finding and Installing Ada
- Building an Ada program
- Ada Development Environment

Language Features

These chapters look at the broader picture, introducing you to the main Ada features in a tutorial style.

- Expressions
- Control Structures
- Type System
- Strings
- Subprograms
- Packages
- Input Output
- Exceptions
- Generics
- Tasking
- Object Orientation
- Contract Based Programming
- Memory Management (Access Types)
- New in Ada 2005
- New in Ada 2012
- Containers
- Interfacing to other Languages
- Coding Standards
- Ada Programming Tips
- Common Programming Errors

Computer Programming

The following articles are Ada adaptations from articles of the Computer programming book. The texts of these articles are language neutral but the examples are all Ada.

- Algorithms
  - Chapter 1
  - Chapter 6
  - Knuth-Morris-Pratt pattern matcher
  - Binary search
- Error handling
- Function overloading
- Mathematical calculations
- Statements
  - Control
- Variables

Language Reference

Within the following chapters we look at foundations of Ada. These chapters may be used for reference of a particular keyword, delimiter, operator and so forth.

- Lexical elements
  - Keywords
  - Delimiters
- Operators
- Attributes
- Aspects
- Pragmas
  - Restrictions

Predefined Language Libraries

This section is a reference of the Ada Standard Library, which is extensive and well structured. It has these four root packages:

- Standard
- Ada
- Interfaces
- System

Besides the Standard Library compilers usually come with a built-in library. This chapter describes the GNAT library in particular:

- GNAT

External Libraries

This section is a reference of third-party Ada libraries which are not part of the compiler predefined environment but are freely available.

- Libraries
  - Multi Purpose
  - Container Libraries
  - GUI Libraries
  - Distributed Objects
  - Database
  - Web Programming
  - Input/Output
- Platform specific
  - Programming Ada in Linux
  - Programming Ada in Windows
  - Programming Ada in Virtual Machines (Java, .NET)

External resources

- Open-source portals
- Web Tutorials
- Web 2.0

Collections

Printable Versions

The following are collection pages. All collection pages are comprised of groups of the already available pages. You can use them for printing or to gain a quick overview. Please note that those pages are partly very long.

Tutorial
Show HTML (1,839 kb) — Download PDF (2,663 kb, 243 pages)
Keywords
Show HTML (470 kb) — Download PDF (290 kb, 59 pages)
Operators
Show HTML (232 kb) — Download PDF (189 kb, 27 pages)

Source Code

The Source from the Book is available for download and online browsing. The latter allows "drill down", meaning that you can follow the links right down to the package bodies in the Ada runtime library

References

1. See wikistats (http://en.wikipedia.org/wikistats/wikibooks/EN/Wikibooks_EN.htm), Category:Book:Ada Programming or /All Chapters
2. Gaetan Allaert, Dirk Craeynest, Philippe Waroquiers (2003). "European air traffic flow management: porting a large application to GNU/linux". *Proceedings of the 2003 annual ACM SIGAda international conference on Ada*. SIGAda'03. pp. 29–37. doi:10.1145/958420.958426. ISBN 1-58113-476-2. http://www.sigada.org/conf/sigada2003/SIGAda2003-CDROM/SIGAda2003-Proceedings/p29-allaert.pdf. Retrieved 2009-01-02. Paper by Eurocontrol (PDF, 160 kB) on portability.
3. Stephen F. Zeigler (1995-03-30). *Comparing Development Costs of C and Ada*. http://archive.adaic.com/intro/ada-vs-c/ada-vs-c.html. Retrieved 2009-01-02. "Our data indicates that Ada has saved us millions of development dollars.".

Further reading

Ada 2005 textbooks

- John Barnes (2006). *Programming in Ada 2005*. Addison Wesley. ISBN 0-32-134078-7. http://www.pearsonhighered.com/educator/academic/product/0,,0321340787,00%2Ben-USS_01DBC.html
- Mordechai Ben-Ari (2009). *Ada for Software Engineers (Second Edition with Ada 2005)*. Springer. ISBN 978-1-84882-313-6. http://www.springer.com/978-1-84882-313-6
- Alan Burns, Andy Wellings (2007). *Concurrent and Real-Time Programming in Ada*. Cambridge University Press. ISBN 9780521866972. http://www.cambridge.org/uk/catalogue/catalogue.asp?isbn=9780521866972
- Nell Dale, John W. McCormick (2007). *Ada Plus Data Structures: An Object Oriented Approach* (2nd Edition ed.). Jones and Bartlett. ISBN 0763737941. http://www.jbpub.com/catalog/9780763737948/.
- John W. McCormick, Frank Singhoff, Jérôme Hugues (2011). *Building Parallel, Embedded, and Real-Time Applications with Ada*. Cambridge University Press. ISBN 9780521197168. http://www.cambridge.org/gb/knowledge/isbn/item5659578/?site_locale=en_GB

Ada 2012 textbooks

- John Barnes (2014). *Programming in Ada 2012*. Cambridge University Press. ISBN 9781107424814. http://www.cambridge.org/gb/academic/subjects/computer-science/software-engineering-and-development/programming-ada-2012?format=PB

Manuals and guides

- Ada Quality & Style Guide: Guidelines for Professional Programmers (wikibook)

- Rationale for Ada 2005, by John Barnes (2007)
- Ada 2005 Reference Manual (2007)

High-Integrity Software

- ISO/IEC TR 15942:2000, *Guide for the use of the Ada programming language in high integrity systems* ISO Freely Available Standards [1]
- ISO/IEC TR 24718:2005, *Guide for the use of the Ada Ravenscar Profile in high integrity systems* ISO Freely Available Standards [2]
- John Barnes (April 2003). *High Integrity Software: The SPARK Approach to Safety and Security*. Addison-Wesley. ISBN 0-321-13616-0.

External links

Resources

- Ada Information Clearinghouse — News and resources
- Ada Answers — Building better software with Ada
- comp.lang.ada (Google groups) — International Usenet newsgroup

Research and user groups

- Journals:
  - *Ada-Letters*
  - *Ada User Journal* (archive)
- International Conferences/Workshops:
  - International Real-Time Ada Workshop (IRTAW) [IRTAW 15]
  - ACM SIGAda international conference on Ada [HILT 2012]
  - 18th International Conference on Reliable Software Technologies — Ada-Europe 2013
  - 17th International Conference on Reliable Software Technologies — Ada-Europe 2012
  - Ada Connection 2011 (videos)
  - Ada "Developers Room" at FOSDEM (2012)
- Local conferences:
  - Ada-Belgium Conference
  - Ada-Spain Technical Day
  - Ada Conference UK (videos: 2009, 2007, 2006)

Associations

- ACM SIGAda — ACM Special Interest Group on Ada
- Ada-Europe
- Ada Germany
- Ada Russia
- Ada-Belgium
- Ada-France
- Ada in Denmark
- Ada Switzerland

Free online books/courses

- Computer-Books.us — Online Ada books
- *The Big Online Book of Linux Ada Programming*
- *Ada Distilled*
- *Ada in Action*
- *Introducing Ada 95*
- *Learn Ada on the Web*
- *Quick Ada*
- *Ada 95: The Craft of Object-Oriented Programming* — Free textbook originally published by Prentice Hall

Authors and contributors

This Wikibook has been written by:

- Martin Krischik (Contributions)
- Manuel Gómez (Contributions)
- Santiago Urueña (Contributions)
- C.K.W. Grein (Contributions)
- Bill Findlay (Contributions)
- B. Seidel (Contributions)
- Simon Wright (Contributions)
- Allen Lew (Contributions)
- John Oleszkiewicz (Contributions)
- Nicolas Kaiser (Contributions)
- Larry Luther (Contributions)
- Georg Bauhaus (Contributions, more contributions)
- Samuel Tardieu (Contributions)
- Ludovic Brenta (Contributions)
- Ed Falis
- Pascal Obry
- Unnamed Hero (Contributions)

If you wish to contribute as well you should read Contributing and join us at the Contributors lounge.

Retrieved from "https://en.wikibooks.org/w/index.php?title=Ada_Programming&oldid=3503742"