Programming Languages mini–HOWTO

Table of Contents

Programming Languages mini−HOWTO....1
Risto S. Varanka, <risto.varanka@helsinki.fi>....1
1.Introduction....1
2.Programming Languages....1
3.GUI Toolkits....1
1.Introduction....2
1.1 Copyright....2
1.2 License....2
Requirements of Modified Works....2
1.3 Disclaimer....3
1.4 Author....3
1.5 Credits....3
1.6 Links....3
2.Programming Languages....4
2.1 Concepts in the Table....4
2.2 Major Languages....5
2.3 Shell Programming....6
2.4 Other Languages....6
2.5 Links....6
3.GUI Toolkits....6
3.1 Concepts in the Table....6
3.2 Major GUI Toolkits....7
3.3 Links....8

Programming Languages mini–HOWTO

Risto S. Varanka, <risto.varanka@helsinki.fi>

v, 6 January 2000

A brief comparison of major programming languages for Linux and major libraries for creating graphical user interfaces (GUIs) under Linux

---

1.Introduction

- 1.1 Copyright
- 1.2 License
- 1.3 Disclaimer
- 1.4 Author
- 1.5 Credits
- 1.6 Links

2.Programming Languages

- 2.1 Concepts in the Table
- 2.2 Major Languages
- 2.3 Shell Programming
- 2.4 Other Languages
- 2.5 Links

3.GUI Toolkits

- 3.1 Concepts in the Table
- 3.2 Major GUI Toolkits
- 3.3 Links

1.Introduction

Linux is a fascinating operating system because it lets any user participate in its development. The variety of available languages, however, can be confusing to beginning Linux developers. This document lists the most common options for everyday development and states some key facts about them. (Well, ``most common'' and ``key'' as I perceive them.) My aim is neither to review the languages nor to determine which one is the best. Each language is a tool that fits some jobs and some tastes. You can get further (often conflicting) information easily, if you ask around or keep your ears open. The Links sections in this document will give you some pointers for your own research.

There is a plethora of languages and libraries for Linux, so this document only covers the most common languages and GUI toolkits at the moment. Also note that the languages and GUI toolkits are seen from the Linux point of view, their features on other platforms are not discussed or implied.

This document is a recent addition to the LDP, so there has not been opportunity for much community feedback. However, it is released in hopes that it will prove useful for people interested in programming under Linux. A question mark in the tables indicates lack of information. If you can fill it in, please contact the author.

1.1 Copyright

Copyright (c) 2000 Risto Varanka.

1.2 License

The following license terms apply to all LDP documents, unless otherwise stated in the document. The LDP documents may be reproduced and distributed in whole or in part, in any medium physical or electronic, provided that this license notice is displayed in the reproduction. Commercial redistribution is permitted and encouraged. Thirty days advance notice via email to the author(s) of redistribution is appreciated, to give the authors time to provide updated documents.

Requirements of Modified Works

All modified documents, including translations, anthologies, and partial documents, must meet the following requirements:

1. The modified version must be labeled as such.
2. The person making the modifications must be identified.
3. Acknowledgement of the original author must be retained.
4. The location of the original unmodified document be identified.
5. The original author's (or authors') name(s) may not be used to assert or imply endorsement of the resulting document without the original author's (or authors') permission.

In addition it is requested that:

1. The modifications (including deletions) be noted.
2. The author be notified by email of the modification in advance of redistribution, if an email address is provided in the document.

As a special exception, anthologies of LDP documents may include a single copy of these license terms in a conspicuous location within the anthology and replace other copies of this license with a reference to the single copy of the license without the document being considered ``modified'' for the purposes of this section.

Mere aggregation of LDP documents with other documents or programs on the same media shall not cause this license to apply to those other works.

All translations, derivative documents, or modified documents that incorporate any LDP document may not have more restrictive license terms than these, except that you may require distributors to make the resulting document available in source format.

1.3 Disclaimer

THIS DOCUMENT COVERS A LARGE AND CONSTANTLY CHANGING DOMAIN. THEREFORE, THE INFORMATION CONTAINED IN THIS DOCUMENT MAY BE INCORRECT OR OUTDATED. ALL USE OF THIS DOCUMENT AND ALL INFORMATION CONTAINED IN IT IS AT YOUR OWN RISK. THE AUTHOR DOES NOT GIVE ANY WARRANTY OR GUARANTEE, EITHER EXPLICIT OR IMPLIED.

1.4 Author

You are welcome to send feedback to the author at: risto.varanka@helsinki.fi.

Author's web site can be found at http://www.helsinki.fi/~rvaranka/.

1.5 Credits

I am thankful to several people who commented on language issues. These conversations have given me a better view of the different languages, and I hope future conversations will allow this mini−HOWTO to mature over time. Especially I would like to thank the people at the IRCNet channel #linux: Morphy, Bluesmurf, Vadim, Zonk^, Rikkus and others whose names I have forgotten. Thanks go also to Stig Erik Sandoe for helpful comments.

1.6 Links

Exhaustive lists of Linux development libraries and tools:

- Freshmeat
- Linux Development Tools
- linuxprogramming.com

The Hacker FAQ by Eric S. Raymond is another interesting text for novice Linux developers. It concentrates on some cultural and psychological aspects of open source development.

Other LDP documents covering general programming subjects include the Reading List HOWTO and the Linux Programmer's Guide − several more have been written on specific subjects.

2.Programming Languages

2.1 Concepts in the Table

Language

A common name of the language.

Beginner

Indicates how well suited the language is for people with little programming experience. A language marked with ``yes'' should be viable for a beginner's first programming language.

Performance

How fast your applications are likely to run when you put them into production use. Performance depends more on your algorithmic programming skills than the actual language. As a rule of thumb, C, C++ and Fortran are sometimes necessary because they can offer better performance than other languages − at other times they might be unwieldy for the desired purpose. (One idea for ``benchmarking'' the languages would be to implement a simple sorting algorithm in all of them and compare running times. Anybody want to help me with this?)

OOP, Object−Oriented Programming vs. other paradigms

Object−oriented programming is an important programming paradigm that is gaining popularity. In object oriented programming, data structures and algorithms are integrated into units, often called classes. OOP is often contrasted with procedural programming (which uses separate algorithms and data structures). It is not strictly dependent on language: you can do OOP in languages not listed as such (C for example), and program in the procedural style in languages that are listed as OOP. I've listed as OOP languages that have special features or add−ons to facilitate OOP. Functional languages (Lisp for example) are a bit different breed − among other things, functional programming is a superset of OOP. Logic programming (Prolog), also called declarative programming, on the other hand, is not related to the other types of programming in a similar sense.

RAD, Rapid Application Development

More dependent on the tools you are using than the actual language. There is a HOWTO on GUI development tools for Linux, although it's out of date. With a good graphical tool you can do RAD. Sometimes RAD is based on code reuse as well, so free software could provide a good starting point.

Examples

Mentions fields of programming the language is most often used in. Other good (and bad) uses exist, but they are less typical.

Comments

Additional information on the language, like capacities and dialects.

2.2 Major Languages

```
PERL
Beginner: Yes - OOP: Yes
Examples: Scripting, sysadmin, www
Comments: Powerful for handling text and strings, very popular

Python
Beginner: Yes - OOP: Yes
Examples: Scripting, application scripting, www
Comments:

TCL
Beginner: Yes - OOP: No
Examples: Scripting, sysadmin, applications
Comments:

PHP
Beginner: Yes - OOP: Yes
Examples: Www
Comments: Popular for web databases

Java
Beginner: Yes - OOP: Yes
Examples: Cross-platform applications, www
Comments:

Lisp
Beginner: Yes - OOP: Functional
Examples: Emacs modes (for elisp), AI
Comments: Variants Elisp, Clisp and Scheme

Fortran
Beginner: No  - OOP: No
Examples: Mathematical applications
Comments: Variants f77 and f90/95

C
```

```
Beginner: No  - OOP: No
Examples: System programming, applications
Comments: Very popular

C++
Beginner: No  - OOP: Yes
Examples: Applications
Comments:
```

2.3 Shell Programming

Shells are an important programming environment, too. I haven't covered them because I don't understand the field very thoroughly yet. Knowledge of shells is important for anyone who works on Linux regularly, more so for system administrators. There are similarities between shell programming and scripting – often they can achieve the same goals, and you have the option of choosing between native shell and a scripting language. Among the most popular shells are bash, tcsh, csh, ksh and zsh. You can get basic information on your shell with the *man* command, *man bash* for example.

2.4 Other Languages

Other languages of note: AWK, SED, Smalltalk, Eiffel, ADA, Prolog, assembler, Objective C, Logo, Pascal (p2c converter)

2.5 Links

- A general info site on programming languages, lots of info and opinions
- TCL
- PERL
- Python
- PHP
- Java
- clisp

3.GUI Toolkits

3.1 Concepts in the Table

Library

Common name or abbreviation of the toolkit.

Beginner

Whether the toolkit is suitable for a newbie programmer.

License

Different licenses for different GUI toolkits have practical significance. GTK+ and TK licenses allow you to develop both open source and closed source applications without paying for a license. Motif license requires payment, while the QT license requires payment only if you write closed source programs.

Language

The language that is most often used with the toolkit.

Bindings

Other languages which can use the toolkit.

Examples

Applications that use the toolkit.

Comments

Additional information on the toolkit.

3.2 Major GUI Toolkits

Library	Beginner	License	Language	Bindings	Examples	Comments
TK	Yes	Free	TCL	PERL, Python, others	make xconfig, TKDesk	
GTK+	No	Free (LGPL)	C	PERL, C++, Python, many others	GNOME, Gimp	
QT	No	Free for open source	C++	Python, PERL, C, others?	KDE	
Motif	No	Non−free	C/C++	Python, others?	Netscape, Wordperfect	

3.3 Links

- TK
- GTK+
- QT
- Motif