Models de desenvolupament i Gestió de projectes
Source Code Management

Carles Mateu & Jordi Planes

Departament d'Informtica
Universitat de Lleida

Curs 2010/2011
Scheme

1. Markup languages

2. Code Documentation
1 Markup languages
   - Man pages
   - \TeX\ and \LaTeX\
   - Texinfo
   - HTML
   - DocBook
   - Wiki

2 Code Documentation
Mark-up languages, as Troff, XML, LaTeX, HTML, etc. are old computer text editing systems available.

XML descends from SGML (Standard Generalized Markup Language) which was created in 1986. It uses angle brackets to start and end tags.

HTML is mostly a publishing-oriented instance of SGML.

SGML descends from GML (Generalized Markup Language), created by Goldfarb, Mosher, and Lorie in the 1960s. It uses double colon ‘:’ to start tags and dot ‘.’ to end tags.
Markup languages

Mark-up languages ensure:

- Data interchangeability (almost indefinitely)
- Easy porting
- Easy rendering
- Easy automation
- Technology neutrality (if well designed)
- Purpose neutrality.
Ye olde man page

- Short for Manual Pages.
- It is a set of macros for Troff (Terminal RunOff) with additions for the manuals.
- The most popular version of Troff was created by Brian Kernighan.
Ye olde man page

- Short for Manual Pages.
- It is a set of macros for Troff (Terminal RunOff) with additions for the manuals.
- The most popular version of Troff was created by Brian Kernighan.

Example

To get the following formula with Troff:

\[ \sum (\alpha \times \beta) \rightarrow \infty \]

you need to type the following line:

( S (( a (mu( b ) ( > (if}
Ye olde man page

- Short for Manual Pages.
- It is a set of macros for Troff (Terminal RunOff) with additions for the manuals.
- The most popular version of Troff was created by Brian Kernighan.
- It documents configuration files, system commands, system calls, C functions, etc.

Example

It is easily accessible from the command line:
$ man man
Ye olde man page

Divided in sections (Original Unix Sections):

1. Commands available to users
2. Unix and C system calls
3. C library routines for C programs
4. Special file names
5. File formats and conventions for files used by Unix
6. Games
7. Word processing packages
8. System administration commands and procedures
Ye olde manpage

Actual linux sections:

1. User commands that may be started by everyone.
2. System calls, that is, functions provided by the kernel.
3. Subroutines, that is, library functions.
3p. Subroutines, that is, library functions, from PERL.
4. Devices, that is, special files in the /dev directory.
5. File format descriptions, e.g. /etc/passwd.
6. Games, self-explanatory.
7. Miscellaneous, e.g. macro packages, conventions.
8. System administration tools that only root can execute.
9. Another (Linux specific) place for kernel routine documentation.

n,o,l (Deprecated) New, old or local documentation that may be moved to a more appropriate place.
Creating a man page

- Man pages are written using *Troff* formatting tools.
- Troff is an ancient formatting tool for Unix, Troff files are ASCII plain textual files with embedded commands.
- To create a simple man page: Create a file ending with the section number (e.g. man.1).
- You can compress a man page, so the name could be: man.1.gz or man.1x
apropos(1)

NAME
  apropos - search the whatis database for strings

SYNOPSIS
  apropos keyword ...

DESCRIPTION
  apropos searches a set of database files containing short descriptions
  of system commands for keywords and displays the result on the standard
  output.

AUTHOR
  John W. Eaton was the original author of man. Zeyd M. Ben-Halim
  released man 1.2, and Andries Brouwer followed up with versions 1.3
  thru 1.5p. Federico Lucifredi <flucifredi@acm.org> is the current
  maintainer.

SEE ALSO
  whatis(1), man(1).

September 19, 2005
Aspect of a man page

Sections of a man page:

NAME Unique mandatory section. (a .SH troff commands). Mandatory format: name1,name2,name3,name4 — Short description

SYNOPSIS Short description.

DESCRIPTION Longer and more thorough description.

OPTIONS Command line options and effect of that options.

FILES Which files uses the command/daemon/service. Configuration or data files.

ENVIRONMENT Which environment variables affect command.

DIAGNOSTICS Overview of the most common errors we can find.

BUGS Should document known bugs (ideally none).

AUTHOR Who wrote this?

SEE ALSO Other man pages or interesting documentation.
.TH apropos 1 "September 19, 2005"
.LO 1
.SH NAME
apropos \- search the whatis database for strings
.SH SYNOPSIS
.BI apropos
keyword ...
.SH DESCRIPTION
apropos searches a set of database files containing short descriptions
of system commands for keywords and displays the result on the
standard output.
.SH AUTHOR
John W. Eaton was the original author of
.BR "man" .
Zeyd M. Ben-Halim released man 1.2, and Andries Brouwer followed up with version
s 1.3 thru 1.5p.
Federico Lucifredi <flucifredi@acm.org> is the current maintainer.
.SH "SEE ALSO"
whatis(1), man(1).
Installing a man page

Installation

The installation of a man page is simple: copy the man page (e.g., foobar.3) to an appropriate directory (one in $MANPATH):

$ cp foobar.3 /usr/share/man/en/man3/ (it can be gzip it!)

Compilation (to PostScript, DVI (DeVice Independent), etc.)

Manual compilation on Linux:

$ groff -mandoc -Tps foobar.3 > foobar.ps
TEX

• TEX is a typesetting system designed and mostly written by Donald Knuth (first memo in 1977).

• Together with the METAfont language for font description and the Computer Modern family of typefaces,

• it was designed with two main goals in mind:
  1. to allow anybody to produce high-quality books using a reasonable amount of effort, and
  2. to provide a system that would give exactly the same results on all computers, now and in the future.

• The output could be PostScript, DVI, PDF, HTML, . . .

• The changes to the text take only effect in a block (starting with { and ending with }).

Example

The quadratic formula is $-b \pm \sqrt{b^2 - 4ac} \over 2a$ \bye
\LaTeX is intended to provide a high-level language that accesses the power of TeX.
\LaTeX essentially comprises a collection of TeX macros and a program to process \LaTeX documents.
Because the TeX formatting commands are very low-level, it is usually much simpler for end-users to use \LaTeX.
\LaTeX was originally written in the early 1980s by Leslie Lamport
It has become the dominant method for using TeX
It splits the source document between the preamble and the body.
Example

\documentclass[12pt]{article}
\title{\LaTeX}
\author{Jordi}
\date{\today}
\begin{document}
\maketitle
\LaTeX\ is a document preparation system for the \TeX\ typesetting program.
\[m = \frac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}}.\] \end{document}
\LaTeX

Jordi

November 9, 2010

\LaTeX{} is a document preparation system for the \TeX{} typesetting program. \[ m = \frac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}} \]
Texinfo

- Created by Richard Stallman in 1980s, Texinfo is a typesetting syntax used for generating documentation.
- It was inspired in Bolio, BoTeX and Info.
- It can create documentation in formats as dvi, html, pdf, ... and its own hypertext format, info

Example

It is accessible from the command line:
$ info info

Example

In order to create a texinfo page:
$ makeinfo page $ makeinfo --format page
@ifnottex
@node Top
@top Short Sample
@insertcopying
@end ifnottex

@menu
  * First Chapter:: The first chapter is the only chapter in this sample.
  * Index:: Complete index.
@end menu
HTML

- HTML (HyperText Markup Language) is the predominant markup language for web pages.
- A markup language is a set of markup tags.

Example

```html
<HTML>
<HEAD>
<TITLE>Main page</TITLE>
</HEAD>
<BODY>
<P>Some text.</P>
</BODY>
</HTML>
```
DocBook

Created in 1991 by HAL Computer Systems and O’Reilly Associates. DocBook is used, among others, by

- Linux Documentation Project
- reference manuals of GNOME and GTK+ APIs
- Linux kernel documentation
- Solaris man pages
DocBook

Created in 1991 by HAL Computer Systems and O’Reilly Associates. DocBook is used, among others, by

- Linux Documentation Project
- reference manuals of GNOME and GTK+ APIs
- Linux kernel documentation
- Solaris man pages

Main purpose of DocBook

- How to efficiently write documentation
- How to make this documentation available in a variety of formats
- How to exchange document information with other code or content

DocBook is XML

- Docbook is a mark-up language based on XML.
- It can be seen as a style (a.k.a. a DTD) to XML.
<article lang="en">
  <title>Latex Slides Howto</title>
  <articleinfo>
    <author>
      <firstname>Carles</firstname>
      <surname>Mateu</surname>
    </author>
  </articleinfo>
  <section id="intro">
    <title>Introduction</title>
    <para>
      This document will explain how to make really nice looking \LaTeX Slides. It uses beamer, available at: <ulink url="http://latex-beamer.sf.net/"></ulink>.
    </para>
  </section>
</article>
Any text editor (VI, Scribes, EMACS/PSGML, etc.) or special XML editor:
We can use two tools: xmlto or docbook2* tools. With xmlto, we can convert DocBookXML to:

- dvi
- html
- javahelp
- man
- pdf
- ps
- txt
- xhtml

among others. As in:

```latex
xmlto pdf article .xml
```
With docbook2* we can convert DocBookXML to:

- docbook2dvi
- docbook2html
- docbook2man
- docbook2pdf
- docbook2ps
- docbook2rtf
- docbook2tex
- docbook2texi
- docbook2txt

among others. As in:

docbook2pdf article.xml
Wiki Wiki Web

WikiWikiWeb was created by Ward Cunningham in 1994, inspired by HyperCard. Wiki means ‘quick’ in Hawaiian.

Wiki

Collaboratively edited web page. Created with a special mark-up language (wiki).

Wikis are becoming one of the most used dynamic documentation tools. They allow easy contribution, dynamic reorganization of information and revision/history control.
Lots of projects are using wikis for documentation: Supported hardware (DVB) in video4linux:

Whatever you can contribute will be highly appreciated, no matter if it’s a HOWTO, help and explanations how to write a driver, how to install some a driver or whether you can explain how that beast called DVB is working.

Have fun, enjoy -- The LinuxTV WikiTeam

The basic plan is to collect facts about available hardware, linux-dvb experiences, HOWTOs, encyclopedia-like information and explanations about how DVB, MPEG-2 and MHP work, what an EPG is and so on, to make this a place useful for everybody who is doing anything related to analog or digital TV. Save and share knowledge.

Getting Started:
- Supported Hardware
- How to install DVB device drivers
- Testing your DVB device (PCI, USB, ...)
- Example setups and Use cases by country
- FAQ

Software and Usage
- Commented software list, including:
  - Kaffeine

Technical Background
- Analog TV
- Digital TV
  - DVB Standard
- Modulation Schemes used for Analog and Digital TV
- Interface chipsets used for DVB devices
- MPEG2 Standard
- Hardware or Software Decoder?
- i2c Protocol
- DSM-CC ObjectCarousel Protocol
Wiki for what?

API documentation in mozilla:

**Toolkit: Home Page**

The Mozilla "toolkit" includes all kinds of glue and bootstrap code shared between XUL applications, built on top of Gecko.

- [Specification of Language Pack UI for Applications and Extensions](#)

**See also**

- [Plugins](#)
- [VRML](#)
Wikis for what?

Trac: integrated wiki+svn+bugtracking+simple project management:

WordPress Trac

Trac is the place to follow along with the development of WordPress. You can track changes in the Timeline section of this site. There is also an RSS feed and a mailing list for those interested.

Bugs reports are also kept on Trac. If you are looking to submit a bug report, please head on over.

You may login to this site using your WordPress.org username and password. If you don't already have one, sign up at the forums.

Contributing

There are many ways to contribute to WordPress. If you want to get started quickly, grab a bug and see if you can reproduce the problem it describes.

The Life of a Ticket

Reporting Bugs in the WordPress Codex tells how to report a bug obviously, what all the keywords we use in Trac mean, and more.

Notification

If you would like to receive notification when a ticket you have submitted or are interested in changes then please configure your email address in trac after you have logged in by going to the settings page.

If you then add your user name to the cc on a ticket you will be notified of changes on the ticket.

How to Submit Patches

Check out this great article by Owen on the subject of creating patches and using trac.
Wikis for what?

OLPC: everything.

The OLPC Wiki

(Redirected from Home)

This page is maintained by the OLPC team. Please use the discussion tab for your comments.

Welcome to the OLPC Wiki, home to collaborative notes about the One Laptop per Child project and related projects and communities. We currently have 1,677 pages and over one-thousand registered contributors; please join us and share your ideas.

What's new

- Latest release: The new stable build, Build 406, along with a new firmware release, Q2C11, are now available. Please update your machines: the autoreinstallation image makes this easy; you'll find many improvements in virtually all system functions (Release notes).
- The first pre-B3 machines have been built and are now operational.
- Localization: The Spanish translation of www.laptop.org is now on line, thanks to the tireless efforts of Xavier Alvarez. The templates for adding additional languages can be found at Localization/www.laptop.org. The Simplified Chinese translation is also on line, thanks to Scott Templer. A Chinese translation of the rest of the site is expected.

Submit new content
Creole: Wikitext language or wiki markup

The first version was released in 2007.
Summary of the syntax:

//emphasized// (e.g., italics)
**strongly emphasized** (e.g., bold)

* Bullet list  
# Numbered list

Link to [[wikipage]]

= Extra-large heading

---- Horizontal line

{{Image.jpg|title}}
==Etymology and meaning==
The word ’’wiki’’ is a [[Hawaiian language|Hawaiian]] word meaning "quick". <ref> AnswerNote Dictionary on term "wiki". </ref>
Wiki systems are therefore designed so that their content can be made available in a quick and uncomplicated manner.<ref> Richard Heigl, Markus Glaser, Anja Ebersbach(2006), p.11. </ref>

==History==
[[Image:HNL Wiki Wiki Bus.jpg|thumb|Wiki Wiki bus at [[Honolulu International Airport]]]] [[WikiWikiWeb]] was the first such site to be called a wiki. <ref name="Britannica"/> [[Ward Cunningham]] started developing WikiWikiWeb in 1994 and installed it on [[Domain name|Internet domain]] [http://c2.com c2.com] on [[March 25]], [[1995]]. It was named by Cunningham, who remembered a Honolulu International Airport counter employee telling him to take the so-called "[[Wiki Wiki Shuttle|Wiki Wiki]]" [[Chance RT-52]] shuttle [[Bus|bus line]] that runs between the airport’s terminals.
Wiki article on Wiki: Etymology and meaning

The word *wiki* is a Hawaiian word meaning "quick".[6] Wiki systems are therefore designed so that their content can be made available in a quick and uncomplicated manner.[7]

History

*WikiWikiWeb* was the first such site to be called a wiki.[4] Ward Cunningham started developing WikiWikiWeb in 1994 and installed it on Internet domain [c2.com][3] on March 25, 1995. It was named by Cunningham, who remembered a Honolulu International Airport counter employee telling him to take the so-called *Wiki Wiki* Chance RT-52 shuttle bus line that runs between the airport's terminals. According to Cunningham, "I chose wikiwiki as an alliterative substitute for 'quick' and thereby avoided naming this stuff quick-web."[1][8] *Wiki Wiki* is a replication of *wiki*, a Hawaiian-language word for fast. The word wiki is a shorter form of *wiki wiki* (IPA /wiːkiː/).

The word is sometimes interpreted as the "backronym" for *what I know is*, which describes the knowledge contribution, storage, and the exchange function.[9]
Scheme

1. Markup languages

2. Code Documentation
   - Javadoc
   - PerlPOD
   - Python
   - C/C++
Java includes an automatic documentation extraction tool from code, called javadoc.

Java uses formatted information included in source code comments to extract API-like documentation from source code:

- Easy to maintain documentation: it’s on the code.
- Easy to synchronize documentation.
- Convenient access.
Javadoc uses function/class comments (comments right before function or class definitions) to store HTML formatted descriptions for functions. It uses special tags to document code, most used are:

- **parameters** @param is used to tag input parameters
- **return** @return is used to tag return values
- **see** @see is used to reference other classes, functions, etc.
- **exceptions** @throws @exception used to document thrown exceptions, etc.
- **authorship** @author used to tag authorship of code
/**
 * Returns an Image object that can then be painted on the screen.
 * The url argument must specify an absolute \{@link URL\}. The name
 * argument is a specifier that is relative to the url argument.
 * <p>
 * This method always returns immediately, whether or not the
 * image exists. When this applet attempts to draw the image on
 * the screen, the data will be loaded. The graphics primitives
 * that draw the image will incrementally paint on the screen.
 * <p>
 * @param url an absolute URL giving the base location of the image
 * @param name the location of the image, relative to the url argument
 * @return the image at the specified URL
 * @see Image
 */

public Image getImage(URL url, String name) {
    try {
        return getImage(new URL(url, name));
    } catch (MalformedURLException e) {
        return null;
    }
}
getImage

public Image getImage(URL url,
        String name)

Returns an Image object that can then be painted on the screen. The url argument must specify an absolute URL. The name argument is a specifier that is relative to the url argument.

This method always returns immediately, whether or not the image exists. When this applet attempts to draw the image on the screen, the data will be loaded. The graphics primitives that draw the image will incrementally paint on the screen.

Parameters:
        url - an absolute URL giving the base location of the image
        name - the location of the image, relative to the url argument

Returns:
        the image at the specified URL

See Also:
        Image
## Javadoc

### Javadoc example result

#### Packages

- `javax.jdo`
- `javax.jdo.datatstore`
- `javax.jdo.identity`
- `javax.jdo.listener`
- `javax.jdo.spi`

#### Packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>javax.jdo</code></td>
<td>This package contains the JDO specification interfaces and classes.</td>
</tr>
<tr>
<td><code>javax.jdo.datatstore</code></td>
<td>This package contains the JDO specification datastore interfaces.</td>
</tr>
<tr>
<td><code>javax.jdo.identity</code></td>
<td>This package contains the JDO specification identity interfaces and classes.</td>
</tr>
<tr>
<td><code>javax.jdo.listener</code></td>
<td>This package contains the JDO specification listener interfaces and classes.</td>
</tr>
<tr>
<td><code>javax.jdo.spi</code></td>
<td>This package contains the interfaces and classes used by JDO implementations.</td>
</tr>
</tbody>
</table>

#### Overview

Copyright © 2005-2006 Apache Software Foundation. All Rights Reserved.
PerlPOD

PerlPOD

Perlpod, the Plain Old Documentation format, is an embedded documentation toolset for PERL.

It is a markup for including documentation inside source code as a mark-up language.
Can be converted to HTML, plain text, man pages, etc.
Perl API documentation is written with perlpod.
PerlPOD uses commands (=word) at begin of line to mark comments.
Some common commands are:

**Headings**  
=head<n> Heading of size n (1,2,3,4).

**Lists**  
=over,=item,=back Creates indented lists.

**Formats**  
=begin format,=end marks blocks to be processed as format (html, etc.)

**End block**  
=cut ends POD processing.
The megafunction() function will eat all your memory, while doing nothing useful at all.

=cut back to the compiler

sub megafunction($) {
    my $param = shift;
    ..........}

chown

Changes the owner (and group) of a list of files. The first two elements of the list must be the *numeric* uid and gid, in that order. A value of -1 in either position is interpreted by most systems to leave that value unchanged. Returns the number of files successfully changed.

```perl
$cnt = chown $uid, $gid, 'foo', 'bar';
chown $uid, $gid, @filenames;
```

On systems that support fchown, you might pass file handles among the files. On systems that don't support fchown, passing file handles produces a fatal error at run time.

Here's an example that looks up nonnumeric uids in the passwd file:

```perl
print "User: ";
chomp($user = <STDIN>);
print "Files: ";
chomp($pattern = <STDIN>);

($login,$pass,$uid,$gid) = getpwnam($user)
  or die "$user not in passwd file";

@ary = glob($pattern);
  # expand filenames
chown $uid, $gid, @ary;
```
Python

Python provides short in-code documentation and there’s also a tool: `pythondoc` very similar to javadoc.

Unfortunately, both tagging languages are incompatible (!)
PythonDoc

PythonDoc uses python comments to mark comments and tags similar to those of javadoc.

```python
##
# This comment provides documentation for the following function.
# @param argument An argument ...
def function(argument):
    pass

##
# This comment provides documentation for the following variable.
variable = value
```
Python in-code documentation

Python provides mark-up to tag documentation just below definitions.

```python
def methodToAdd( arg1=0.0, arg2=0.0):
    """add two numbers
    Keyword arguments:
    arg1 --- a number (default 0.0)
    arg2 --- another number (default 0.0)
    """
    return arg1+arg2
```

Carles Mateu & Jordi Planes (DIEI) Desenvolupament i Gestió de FOSS Curs 2010/2011 45 / 53
Python in-code documentation

All this documentation is accessible using __doc__

```python
>>> def methodToAdd(arg1=0.0, arg2=0.0):
...     '''add two numbers
...     Keyword arguments:
...     arg1 -- a number (default 0.0)
...     arg2 -- another number (default 0.0)
...     '''
...     return arg1 + arg2
...
>>> methodToAdd.__doc__
'add two numbers
Keyword arguments:

arg1 -- a number (default 0.0)
arg2 -- another number (default 0.0)
' 

>>> methodToAdd
```
DOC++

DOC++ is a documentation generator for C, C++, IDL and Java. Generates HTML and $\TeX$ (for PDF and paper generation). Documentation is extracted from header files.

- Structured documentation (Hierarchy)
- Java Applet for Class graph generation
- Cross references
- High end formatting support
/** A derived class.
   Here we show multiple inheritance from two docified classes. This example shows how to structure the members of a class, i.e.

   This is how this documentation has been generated:
*/

class Derived_Class : public CommonBase, protected Intermediate {
  public:
  /** @name parameters */
  //@{
    /// the first parameter
    double a;
    /// a second parameter
    int b;
  //@}

  /** @name methods */
  //@{
    /// constructor
    /// This constructor takes two arguments, just for the sake of demonstrating how documented members are displayed by DOC++
    @param a this is good for many things
    @param b this is good for nothing
  @}
class Derived Class

A derived class.

Inheritance:

Diagram of inheritance relationships.

Public Fields

- parameters

Public Methods

- methods

Inherited from CommonBase:

Public Methods

- const Derived Class & getB(const Intermediate & c) const

Protected Fields

- double variable
DoxyGen

DoxyGen uses javadoc, C++, C (and QT ) comment documentation and supports Python Comment Documentation.

/**
   * This is an example documentation block
*/

/**!
 * This is another documentation block QT style
*/
//! A simple class.
/*! 
   A class is born!
*/
class Simple
{
  public:

    //! A constructor.
    /*! 
      A log description of the constructor.
    */
    Simple();

    //! Member function with two arguments and integer return
    /*! 
      \param i an integer
      \param s a char pointer.
      \return something..
      \sa Simple(), ~Simple(), and publicVar()
    */
    int runMe(int i, char *s);
}
A test class. More...

List of all members.

**Public Types**

```cpp
enum TEnum { TVal1, TVal2, TVal3 }
```

An enum. More...

**Public Member Functions**

- `Test()`: A constructor.

- `~Test()`: A destructor.

- `int testMe(int a, const char *s)`: A normal member taking two arguments and returning an integer value.

- `virtual void testMeToo(char c1, char c2)=0`: A pure virtual member.
### deparse_context Struct Reference

#### Data Fields

<table>
<thead>
<tr>
<th>StringInfo</th>
<th>buf</th>
</tr>
</thead>
<tbody>
<tr>
<td>List *</td>
<td>namespaces</td>
</tr>
<tr>
<td>int</td>
<td>prettyFlags</td>
</tr>
<tr>
<td>int</td>
<td>indentLevel</td>
</tr>
<tr>
<td>bool</td>
<td>varprefix</td>
</tr>
</tbody>
</table>

#### Detailed Description

Definition at line 77 of file `ruleutils.c`.

#### Field Documentation

**StringInfo deparse_context::buf**

Definition at line 79 of file `ruleutils.c`. 