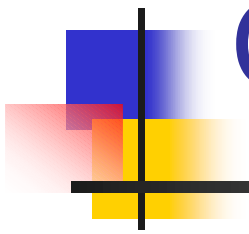


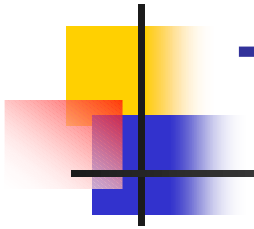
CVS, Logging, Development

Shane Cantrell

Zach Crisman

What is a "Versioning System"?

- Records the history of files
- Shares code within a group
- Allows multiple people to edit the same files
- Merges changes from different people
- Goes back in time

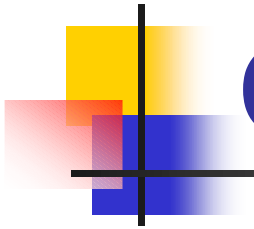


The Alternative

- Saving every version of every file you have ever created
- Saving current state of project every so often
- Wasting disk space by duplicating unchanged portions of code
- Communicating manually to prevent coding conflicts
- Forgetting where the most recent version is or having group members "accidentally" choose an old version

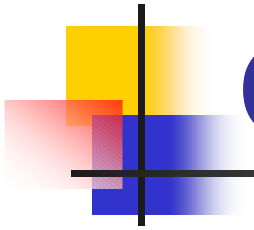
Ways to Access CVS

- Command Line
  - Cygwin
  - SSH Secure Shell Client 3.2.2 (full)
- TortoiseCVS (www.tortoisecvs.org)
- WinCVS (www.wincvs.org)



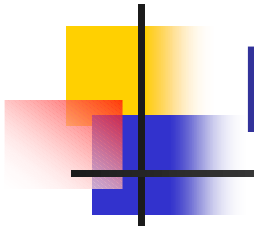
CVS Setup

- Environment Variables:
  - CVSROOT = <location of CVS repository>
  - CVS_RSH = <remote shell>
  - EDITOR = <default editor>
- Windows Examples:
  - CVSROOT = :ext:shanec@vole.cs.washington.edu:/homes/iws/shanec
  - CVS_RSH = ssh2.exe
  - EDITOR = notepad.exe
  - PATH = %PATH%;C:\cvs;C:\Programs\SSH



Common Commands

- Checkout module (get most recent version)
  - cvs co <module-name>
- Update files (incorporate recent changes)
  - cvs update <file-list>
- Commit files (publish your current files)
  - cvs commit <file-list>
- Add new files to the module
  - cvs add <file-list>
  - cvs add –ko <binary file-list>
- Questions? Try ‘man cvs’



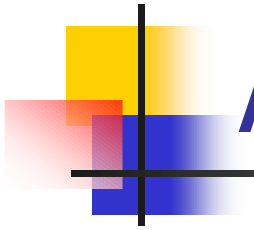
Making a Repository

- Make your CVS directory
- Set CVSROOT to your CVS directory
- *cvs init*

- Tutorial

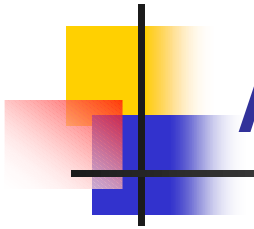
http://www.cs.washington.edu/orgs/acm/tutorials/dev-in-unix/cvs.html

(Typed commands are in italics.)



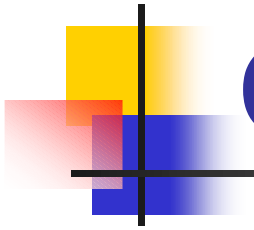
Adding a Module

- *cvs checkout CVSROOT*
- Edit the *./CVSROOT/modules* file
  - Add the line describing your module and location
  - (eg. *project project/*)
- *cvs commit CVSROOT*
- *cd $CVSROOT*
- *mkdir project* (make the project directory)
- Set the file permissions (if necessary) with *chgrp, chown*, and *chmod*.



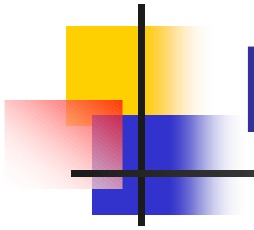
Adding files

- Checkout the module
- Add the new directories using *cvs add*
- Add the new files using *cvs add*
- Commit the new files



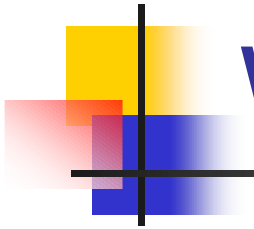
CVS Output Key

U – the file was brought up to date
P – the file was brought up to date via a patch
A – the file has been added
R – the file has been removed
M – the file has not changed in the repository or it has changed in the repository but it was successfully merged
C – the is a conflict between the repository version and your version
? – file note in repository, CVS does not know what to do with it



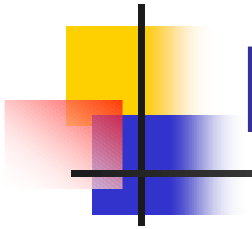
Message Logging in Java

- Chapter 13 of Tomcat
- J2SE 1.4 - java.util.logging
  - standard logging library
- Jakarta Log4j
  - previous “standard”
- System.out.println()
  - lazy man’s technique



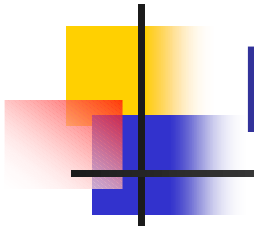
Why not System.out?

- No way of switching logging on or off at runtime
- No way of specifying logging priority or message severity apart from the message text
- Lacks special functionality (like e-mailing an administrator)
- Must be redirected into a file
- Must be removed when the product is released



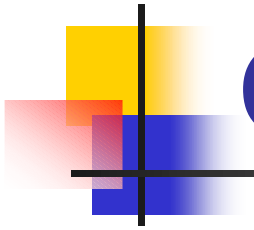
Logging Levels

- Level.SEVERE (highest value)
- Level.WARNING
- Level.INFO
- Level.CONFIG
- Level.FINE
- Level.FINER
- Level.FINEST (lowest value)



Logging Classes

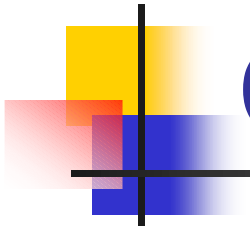
- java.util.logging.*
  - Logger
  - Handler Classes
    - ConsoleHandler
    - FileHandler
    - SocketHandler
  - Formatter Classes
    - SimpleFormatter
    - XMLFormatter



Getting the Logger Object

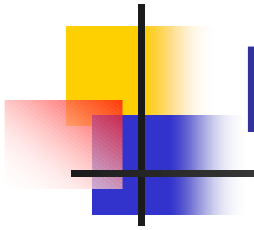
- class Logger
  - public static Logger getLogger(String name)

- Creates a new Logger object if one does not already exist for *name*.
- If package type naming is used, then sub names inherit logging levels
  - For example, “net.hydrus.test” would inherit the logging level setting from “net.hydrus”



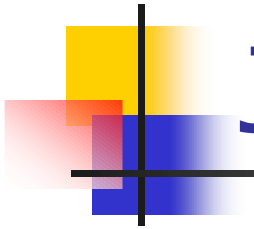
Configuring the Logger

```
...

convLogger.setLevel(Level.INFO);
try {
    FileHandler logfile = new FileHandler("F:/Tomcat/logs/conv.log");
    logfile.setLevel(Level.INFO);
    logfile.setFormatter(new BasicFormatter());
    convLogger.addHandler(logfile);
    convLogger.setUseParentHandlers(false);
}
catch (IOException e) {
    convLogger.warning("Failed to set up logfile");
}

...
```



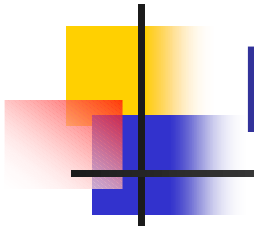
Logging Messages

- class Logger
  - public void severe(String msg)
  - public void warning(String msg)
  - public void info(String msg)
  - public void config(String msg)
  - public void fine(String msg)
  - public void finer(String msg)
  - public void finest(String msg)



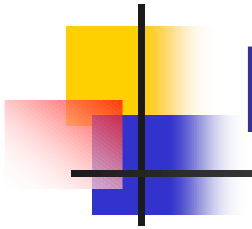
Jakarta Libraries

- http://jakarta.apache.org/
- Cactus (unit testing)
  - http://jakarta.apache.org/cactus
- Apache XML Project
  - http://xml.apache.org/
- Regular Expressions (Regexp)
- Text Processing (ORO)
- Text Search Engine (Lucene)



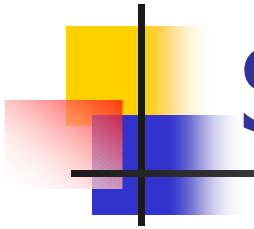
LCO1 Considerations

- Be creative
- Midlet Considerations
  - built-in GUI
  - custom GUI
  - phone specific libraries
- Server Considerations
  - servlet
  - traditional server
  - database



LCO1 Reminders

- What is it?
- What does it do for us?
- How is it supposed to work?
- Is it possible?
  - What is needed?
  - How do you intend to go about making it?
- Who is it for? Are there support people involved in its functionality?
- Be sure to ask the questions throughout and clarify when in doubt of clarity.



Sneak Preview (Next Time)

- Bug Tracking
  - elementool.com
  - fogbugz.com
- Unit Testing
  - jUnit (junit.sourceforge.net)
  - Jakarta Cactus
- Other Tools...