Pavlov For Developers FAQ

External Tools

These are some of the tools that are used in and around the Pavlov project. You don't necessarily need any of them to be an active developer, although Ant and a CVS client are pretty close to necessary.

> **Note:**
> The mention of any software in this document is not meant to imply that that software's creators endorse Pavlov. This document is instructional, not promotional in nature. This note addresses an item in some versions of the BSD and Apache licenses.

Apache Ant

Apache Ant Project

Ant simplifies compiling Java files to class files, and a host of other, similar tasks. It's basically a Java-centric replacement of the UNIX "Make" command. Ant uses an XML file, usually called "build.xml" to get rules for compiling a whole tree of files at once.

Apache Forrest

Apache Forrest Project

Apache Forrest is used to maintain the Pavlov web site. All the pretty HTML and links and stuff is generated automatically by Forrest. The web site documents, like this one, are generated from XML documents which focus on content rather than form.

Apache Log4j

Apache Log4j Site

Log4J is an open-source system for logging, i.e. reporting errors, warnings, informational messages, etc. It replaces "System.out.println" for debugging code, while adding a great deal of extra functionality.

CVS

CVS For New Users

Jim Blandy's Intro To CVS

sourceforge.net's CVS info (See section F)

CVS stands for "Concurrent Versioning System." It allows several people to edit a file or set of files at the same time and minimizes "collisions."

In the old days, people used to use RCS (Revision Control System), where they would check out a file, work on it, and then check it in. Nobody else could edit the file while it was checked out. People would go on holiday with a file checked out and return to find their house burnt down by a mob of angry co-contributors.

CVS takes a little getting used to, and is by no means the easiest program in the world to get installed. It is, however, one of the most useful programs in the world to use for collaborative development.

emacs, jedit, vim, joe, pico, xedit, yadda yadda yadda

GNU Emacs Site

jEdit Site

These are text editors you'll hear referred to every once in a while. While it's possible to edit a Java program file, XML document, HTML document, etc in a commercial word processor, text editors such as these are geared towards doing it more efficiently and cleanly. Developers get extremely attached to their text editors. Few people know this, but the siege at Troy was started when Paris told Menelaus that vim was better than emacs. Zeus, the thunder maker and supreme god, knew that emacs was best and gave victory to the Greeks.

ICQ (I seek you)

ICQ Site

ICQ (say it out loud) is a free communication tool, that includes functionality like instant messaging. ICQ has been vital and free for at least 8 years. ICQ is recommended for project members.

JUnit

JUnit.org

JUnit is a framework for writing unit tests. Avoiding formality, a unit test simply tests a chunk of code, usually a method, to determine if it functions correctly in a variety of situations. JUnit has been found to be extremely useful when multiple developers are working on a program, and is considered by many to be a cornerstone of eXtreme Programming. Pavlov's unit tests, such as they are, are located in the test subdirectory.

UMLGraph

UMLGraph Site

UMLGraph generates UML diagrams from doclet tags. If you see a doclet tag in Pavlov code that you're not familiar with, it's likely a UMLGraph tag. It's a back-burner project to get good UML diagrams from UMLGraph.

VAInstall

VAInstall Site

VAInstall is a GPL-ed program that creates installers, including all of Pavlov's installers. The configuration files that specify how the Pavlov installers are created are located in the pavlov/vai directory in the CVS repository.

PMD, JavaStyle, CheckStyle, Jalopy, and So Forth

PMD Site

JavaStyle Page

CheckStyle Site

I'm grouping these together as "code improvement" tools. They analyze your code and either reformat it or make recommendations for improving it. I'm not aware of any similar tool that works with JDK1.5 constructs, and many have problems with 1.4 code (which is to say asserts). The picosecond that any of these tools starts to work with 1.5 code, I will start using it again.

Bundled API's

It's a hallmark not only of laziness, but of good design to not reinvent the wheel. Since Pavlov is distributed under the GNU General Public License, it can leverage software distributed under several open source licenses. Here is a quick intro to them.

Note:

The mention of any software in this document is not meant to imply that that software's creators endorse Pavlov. This document is instructional, not promotional in nature. This note addresses an item in some versions of the BSD and Apache licenses.

jCharts

jCharts Project

jCharts is an Apache-Licensed API for plotting line-graphs, scatter-plots, pie charts, and so forth. It's used by several Pavlov pluglets. It's lightweight, quick, and the output is very pretty. jCharts replaced Pavlov's "EasyGraph" module around January 2004. jCharts is a trademark of Nathaniel G. Auvil.

JavaHelp/jh.jar

JavaHelp Site at Sun

JavaHelp is software written by Sun Microsystems to provide a familiar help system in Java programs. Click "Help Topics" in Pavlov or Bee to see it in action.

sillyview

sillyview site

sillyview is an API that spun off from Pavlov. It uses Velocity (cf below) to provide a model-view-controller framework based on template files.

steelme

steelme site

steelme is another API spun off from Pavlov. In short, it provides the Themes submenu of the Prefrences menu. In shorter, it lets Pavlov use pretty colors and fonts (or ugly colors and fonts).

Apache Jakarta's Velocity

Velocity Site

Velocity is a very, very pretty piece of software created by the folks at the Apache Jakarta project. It's hard to put velocity in a nutshell, but here's my try: you start with a text "template

file." It could be XML, HTML, RTF, PDF, or any sort of text file -- it doesn't matter.

Now, say there are some "tokens" or string values in the template file you want to replace with a set of values determined in. Velocity lets you do this in a very cool way. You can pass any sort of Java object "into" the template, evaluate methods, do loops and evaluate conditionals.

Pavlov uses HTML templates in many places. The main quiz screen is specified by a HTML template. So is the library view on the left side of the main screen. So is the login widget, the "Export Quiz" widget, and so forth. Velocity Pluglets are simply HTML templates that are given information about the user's current state every time he answers a question.

Templates are converted to Swing entities by the sillyview package. sillyview can also render these templates to other environments, such as servlets.

Apache Xerces2 Java (xercesImpl.jar/xml-apis.jar)

Xerces2 Java Site

Pavlov uses XML heavily. Xerces is a free API for parsing (and doing other stuff to) XML files. Hence, Pavlov uses Xerces heavily. QED.

What the Heck?!?

You just started looking through the code, and it looks more like C++ than Java? See a "cat" or an "rb" and wonder what it is?

What's a "cat" and why is it everywhere?

cat is the standard name for a log4j Category or Logger, an object that can create error, warning, info, debug messages and so forth.

What are asserts all about?

JDK 1.4 and later provides assertions. These are checks, usually of preconditions and postconditions in methods, that can be enabled or disabled at runtime. People spend a lot of time arguing about best-practices with assertions. It's generally agreed on that they shouldn't be used to check that method parameters are valid, but you may find places in Pavlov where they're used this way. If you do stumble upon an assert used to check parameters, tag it with a FIXME.

What's an "rb" and why is it everywhere?

ResourceBroker is a wrapper with some convenience methods for the java.util.ResourceBundle class. It allows Strings to be stored in properties files instead of being hardcoded into the Java program. This makes it easy to a) tweak strings, and b) internationalize the program.

What's this HashMap<String, Object> stuff?

As of JDK1.5, Java allows strong typing in its Collections framework. This is great if you have concerns that somebody might put a CurryChicken object in your vector of IceCream objects. I think we all know what havoc can ensue when you're expecting IceCream and get CurryChicken instead.

What's this for(String x : vec) stuff?

Another JDK1.5 feature. Say you have a Vector<String> vec, and you want to loop through it. You could make an Iterator, an Enumeration, loop through it with elementAt(), and so forth. (Furthermore, in 1.4 you'd have to cast your loop variable to String). This construct does all that for you, decreasing the amount of code and the possibility of little mistakes.

What are FIXME's for?

A FIXME comment is used to point out that something is implemented poorly, bug-prone, inefficent, or so forth. Developers can search for the string FIXME when looking for something to fix. If you see suspicious code, feel free to plop a FIXME with a brief description of the problem.

You'll also see FIXED comments, which should replace the FIXME when the problem is corrected. FIXED comments should last "for a while" but should be removed sooner or later. Later is probably better than sooner.

Miscellany

The Coding Standards seem confusing/draconian/etc...

The Coding Standards document is a mix of simple and advanced topics, which is probably not a good idea. You don't have to understand the Model-View-Controller pattern to tweak some code. CS should not make you want to jump out of a window or be nervous about contributing. It should provide guidance, especially when you're getting close to committing some changes.

If you are nervous about breaking the coding standards, you may take comfort in the fact that

the chunk of code that was lambasted at length to illustrate the standards was written by the project manager. Nobody's perfect.

If there's something in the CS that's patently confusing, ask about it on the pavlov-devel mailing list.

I Can't Get CVS to Check Out Pavlov

There are a buzillion CVS clients out there, and I use precisely one (cvs/openssh for cygwin/linux). I think, especially if you use jEdit, the GruntsPud CVS Client is the second easiest one in the world to set up. The key is to get the CVSROOT and CVS_RSH variables set up right.

If, while you're setting up, you see an option named "pserver," it shouldn't be there. In short, pserver is bad, ext is good. Everybody join hands and repeat: "pserver is bad, ext is good." You have to have some sort of secure socket or secure shell support to use CVS at sourceforge if you plan on checking something in. Many clients include some ssh support, you just have to figure out how to make it work.

That said, here's what a CVS session looks like in CygWin and Linux:

```
$CVSROOT = :ext:YOURSFNAME@cvs.sourceforge.net:/cvsroot/pavlov
export CVSROOT
$CVS_RSH = /usr/bin/ssh
export CVS_RSH

then, you'd check out the repository once:

mkdir ~/foo
cd ~/foo
cvs co .  ( "period" for everything -- cvs co net for just core source)

then edit a file (say, put a space in a comment or something)
cd ~/foo
vi net/sourceforge/pavlov/event/AnswerEvent.java

then checkin
cd ~/foo
cvs ci

then (say the next day) apply diffs from the server to your copy of the code
cd ~/foo
cvs update

or, you could just update the event directory:
cd ~/foo/net/sourceforge/pavlov/event
cvs update
```

What's tiger?

Tiger is slang for JDK1.5X/JRE1.5X.

Where do I find stuff to do?

Here's some pointers:

- Coding Standards
- Contributing I
- Contributing II
- Current Feature Requests
- Mailing List Archives
- Join the Pavlov Mailing List

I'm philosophically against "assigning projects," it stifles innovation and lowers morale. But, sooner or later you will get a message like "hey, can you take a look at X?" that's just a sign that the other developer thinks you have good eyes. If you're still having trouble finding something to work on, send an email to the list and ask if anybody needs help with something.

Can a GPL Project use Apache-Licensed Modules?

According to Apache, yes. There are rumblings that GNU doesn't agree, but I've yet to see anything concrete. There's never been any beef with using BSD modules in a GPL project, so I can't really see what the issue would be. I'm a rabid GPL zealot, but I really can't see what the issue would be.

Shouldn't it be called Skinner?

Maybe skinlov or something. Pavlov makes sense from a meta-viewpoint: conditioning a positive response to studying is Pavlovian. The operant conditioning gets the user to learn questions, the Pavlovian aspect conditions the user to respond in a positive manner to the stimulus of "study time."

Is it "BEE" or "Bee"

Yes, it is "BEE" or "Bee." BEE is, more or less, an acronym for "Book Editing Environment," so, it should be capitalized, but is often not in practice.

Philosophy

What's the goal of the Pavlov Project?

To provide a comfortable, customizable environment for efficient study.

The development of the emacs text editor has deeply affected my goals with Pavlov. My dog could write a text editor -- if there aren't 10,000 text editors out there, I'll eat my hat. What makes emacs special is that it is, and has been, so customizable. As an end user, without touching the core source code, without writing a line of C code, you can dramatically affect how emacs looks and how it works. The result is that the user community adapts emacs to solve the problems they have with it. They share their adaptations, and emacs en-grande develops organically.

This isn't meant to detract in any way from the amazing way that the emacs core has grown with it's staggering network of coders. I'm just saying that the magic bit is the customizability.

Here are some ways that we've strived to make Pavlov customizable:

- themes (color/font schemes) with a easy-to-use theme editor
- skins HTML/VTL documents that specify the user interface
- book XML files: standardized question content
- feedback pluglets: little java programs that respond to the users progress
- velocity feedback pluglets: feedback pluglets without Java
- strategy pluglets: little java programs that choose which questions to ask
- quiz export: VTL documents to export quizzes to any sort of text file
- tool pluglets: pluglets for other sorts of operations

A lot of effort is going into releasing a Java servlet version of Pavlov. This will decouple using the program from installing it, and should increase the user base by orders of magnitude.

Why write another "flashcard" program?

There wasn't one that did what I wanted. I'd say 25% of the ones out there have their questions hardcoded. Another 25% can't handle media, like pictures and sounds, embedded in their questions. I'm not aware of another one that supports pluggable feedback mechanisms, pluggable question selection strategies, skins, and so forth.

Get to Know Everybody

I know I'm repeating myself here, but it bears repeating. A lot of the joy of collaborative development comes from talking to other developers, discussing ideas, sharing victories and defeats, and socializing between builds. Coaching a new developer can be particularly rewarding. 'nuff said.

Being Nice

As computer scientists, engineers, mathematicians, and so forth, we tend to be a bit abrupt when someone asks a question. This isn't conducive to creating a vital community. There is no place in the Pavlov community for flaming users or other developers. Don't do it.

Directory Structure

Pavlov has many, many directories. A current developer directory tree, discounting build directories, CVS directories, and Javadocs has 106 directories. This document is almost guaranteed to not keep up with changes in the directory structure, but should provide a good basic idea of what is where.

- PAVLOV: startup scripts, Licenses, Readme, ant's build.xml
  - BUILD: where ant creates a build image
  - DIAGRAMS: question images
  - FOR: website content in Forrest format
  - LIB: holds jars for bundled API's
  - LIBRARY: holds Book XML files -- where the questions live
  - PLUGLETS: where feedback, strategy, tool, and velocity pluglets live.
    - FEEDBACK:feedback pluglet classes/jars
    - STRATEGY: question selection strategy classes/jars
    - TOOLS: tool pluglet classes/jars
    - PLUGSRC: pluglet source code
    - BIN: images, sounds, etc for pluglets before deployment in Jar files
    - VELOCITY: velocity pluglet template files
  - RESOURCES: application-level image files, Icons, About, etc.
    - AUDIO: files for random audio pluglet
    - BEEHS: bee JavaHelp files
    - HCTEMPLATES: velocity templates for quiz export
    - HS: pavlov JavaHelp files
    - ICONS: icon images for Bee and Pavlov.
    - PROPERTIES: original ResourceBundles. Copied by ant into pavlov.jar.
    - RANDOM: images for RandomImage pluglet
    - SEQUENCE: images for Sequenced Image (aka Filmstrip) pluglet
    - SKINS: velocity templates and resources for skins

    - THEMES: installed steelme themes
    - VIEWS:Deprecated, functionality will be replaced by skins/default
- TEST: holds JUnit test suite
- USERS: holds per-user history files.
- VAI: holds config files and scripts for VAInstaller
- NET: top of source tree. See JavaDocs for source tree descriptions.