Introduction to Software Engineering: Tools and Environments

Session 10

Oded Lachish

Room: Mal 405
Visiting Hours: Wednesday 17:00 to 20:00
Email: oded@dcs.bbk.ac.uk
Module URL:
http://www.dcs.bbk.ac.uk/~oded/Tools2012-2013/Web/Tools2012-2013.html
Last time

1. Documentation tools Doxygen and eUML2

2. “Code generation” tools – eUML2

3. Ant revisited
Tools Side Dish

Environment
Variables
Environment Variables

How did eclipse know where to find java

- Scan the directory structure (bad idea, why?)

- Environment variables 😊
  - Probably all operation systems have environment variables.
  - Environment variables can be accessed by running processes
  - Environment variables are also play a significant role in the operation of the OS
Important - Environment Variables

1. PATH – list of directories to look for executables

2. JAVA_PATH – location of the Java related files

3. ANT_PATH – location of the Ant related files
Setting - Environment Variables Windows

Right click

Left click
Environment Variables -

Left click
Environment Variables -

Left click
Environment Variables -

You can add new system variables and edit existing ones
Installing Ant and Setting Environment Variables

URL to download from

- [http://ant.apache.org/bindownload.cgi](http://ant.apache.org/bindownload.cgi)

Download:

```
apache-ant-1.8.2-bin.zip
```

Store contents in C:\

So `C:\apache-ant-1.8.2` exists
Setting Ant Environment Variables

Set ANT_HOME to

First left click

[Image of System Properties and Environment Variables settings]
Add Ant to Path Environment Variables

Semi colon separates directories in list, add 
“; C:\apache-ant-1.8.2\bin”
at the end of the list

First left click
Java Server and Servlets
Why do we need Javaserver and Servlets

1. We are going to install Jenkins
2. Jenkins is used through a browser
3. Jenkins uses dynamic page generation
4. The standard browser does supply a sufficient environment for this goal
5. Jenkins requires a Javaserver for this goal
6. We shall use Tomcat as our Javaserver and run Jenkins a java servelet (an extension to Tomcat)
Download Tomcat

URL to download from

• http://tomcat.apache.org/download-60.cgi

• Download:

  32-bit/64-bit Windows Service Installer

Execute  (select all components)
Tomcat Installation – User Name and Password

Make sure you remember this we will need it
Tomcat Installation – Java Virtual Machine

Make sure correct
Tomcat Destination

Remember, the destination folder will be useful for installation.
Apache Tomcat

If you're seeing this page via a web browser, it means you've setup Tomcat successfully. C

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at

$CATALINA_HOME/webapps/ROOT/index.html

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you are either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her work done yet, then the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration instructions.

NOTE: For security reasons, using the manager webapp is restricted to users with role "manager". Use $CATALINA_HOME/conf/tomcat-users.xml.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation on developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- users@tomcat.apache.org for general questions related to configuring and using Tomcat
- dev@tomcat.apache.org for developers working on Tomcat

Thanks for using Tomcat!
Apache Tomcat – Server Status

Server Status

Manager
List Applications

Server Information

<table>
<thead>
<tr>
<th>Tomcat Version</th>
<th>JVM Version</th>
<th>JVM Vendor</th>
<th>OS Name</th>
<th>OS Version</th>
<th>OS Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Tomcat/6.0.33</td>
<td>1.7.0-b147</td>
<td>Oracle Corporation</td>
<td>Windows 7</td>
<td>6.1</td>
<td>amd64</td>
</tr>
</tbody>
</table>

JVM
Free memory: 87.86 MB Total memory: 119.68 MB Max memory: 1776.00 MB

Left click
### Tomcat Web Application Manager

**Message:** OK

#### Manager

<table>
<thead>
<tr>
<th>List Applications</th>
<th>HTML Manager Help</th>
<th>Manager Help</th>
<th>Server Status</th>
</tr>
</thead>
</table>

#### Applications

<table>
<thead>
<tr>
<th>Path</th>
<th>Version</th>
<th>Display Name</th>
<th>Running</th>
<th>Sessions</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>None specified</td>
<td>Welcome to Tomcat</td>
<td>true</td>
<td>0</td>
<td>Start</td>
</tr>
<tr>
<td>/docs</td>
<td>None specified</td>
<td>Tomcat Documentation</td>
<td>true</td>
<td>0</td>
<td>Start</td>
</tr>
<tr>
<td>/examples</td>
<td>None specified</td>
<td>Servlet and JSP Examples</td>
<td>true</td>
<td>0</td>
<td>Start</td>
</tr>
<tr>
<td>/host-manager</td>
<td>None specified</td>
<td>Tomcat Host Manager Application</td>
<td>true</td>
<td>0</td>
<td>Start</td>
</tr>
<tr>
<td>/jenkins</td>
<td>None specified</td>
<td>Jenkins v1.441</td>
<td>true</td>
<td>0</td>
<td>Start</td>
</tr>
<tr>
<td>/manager</td>
<td>None specified</td>
<td>Tomcat Manager Application</td>
<td>true</td>
<td>1</td>
<td>Start</td>
</tr>
</tbody>
</table>

This shouldn’t be there at this stage
Next Stage
Until now

We have covered many SDLC tools from a single developer's point of view.

From here on

Project point of view (large project)

- **Local** Individual Developer
- **Integration** Collected work of Individual Developers
- **Testing** QA team Environment
Single Developers Tools

1. “light weight”
2. Easy to learn
3. Integrated with IDE
4. Trivial to install and set
5. Massive number of available tools

Project Scale Tools

1. “HEAVY”
2. Easy to learn?
3. Require their own server
4. Installation may require significant effort
5. Relatively small variety of tools
Continuous Integration
Integration

• Where all the code goes to (and also the unit tests)

• When things go wrong here the price is high

• This is the reason the focus is on minimizing the damage when things go wrong
Continuous Integration

Integrate code as soon as possible. Why?

Each time only a small portion of code is added or changed. Why?

- Problems are detected earlier
- Easier to find the problem
Jenkins
What is Jenkins

A crucial element in continuous integration has two main goals:

1. “Continuously” build and test software projects

2. Monitor externally run jobs
Work Flow with Jenkins

A team member/ members commit code to source control repository

Jenkins detects that new code has been committed

Jenkins uses **Build Tools** to build the project and **Test tools** to test the project

Jenkins uses the resulting files to generate reports

Jenkins send notification if anything went wrong (build, test failed etc.)
Jenkins as Part of the Big Picture

- **Easy installation**: java -jar jenkins.war, or deploy it in a servlet container. No additional install, **no database**.

- **Easy configuration**: can be configured entirely from its friendly web GUI with extensive on-the-fly error checks and inline help. There's no need to tweak XML manually anymore, although if you'd like to do so, you can do that, too.

- **Change set support**: can generate a list of changes made into the build from Subversion/CVS. This is also done in a fairly efficient fashion, to reduce the load on the repository.
Jenkins as Part of the Big Picture

- **Permanent links**: Jenkins gives you clean readable URLs for most of its pages, including some permalinks like "latest build"/"latest successful build", so that they can be easily linked from elsewhere.

- **RSS/E-mail/IM Integration**: Monitor build results by RSS or e-mail to get real-time notifications on failures.

- **JUnit/TestNG test reporting**: JUnit test reports can be tabulated, summarized, and displayed with history information, such as when it started breaking, etc. History trend is plotted into a graph.
Jenkins as Part of the Big Picture

- **Distributed builds**: Jenkins can distribute build/test loads to multiple computers. This lets you get the most out of those idle workstations sitting beneath developers' desks.

- **File fingerprinting**: Jenkins can keep track of which build produced which jars, and which build is using which version of jars, and so on. This works even for jars that are produced outside Jenkins, and is ideal for projects to track dependency.

- **Plugin Support**: Jenkins can be extended via 3rd party plugins. You can write plugins to make Jenkins support tools/processes that your team uses.
Jenkins as Part of the Big Picture

An “easy” to configure tool that is responsible for the following:

1. Make sure that everything that should be executed is executed when and as soon as possible

2. Collect all the information and process it into coherent reports (imagine have 10 different tools each with its own report)
Jenkins

Leg Work
Jenkins – .war file

Jenkins

An extendable open source continuous integration server

Meet Jenkins

Find out what Jenkins is and get started.

Download Jenkins

Java Web Archive (.war)

Latest and greatest (1.492)
changelog | past releases | RC

upgrading from Hudson?

jenkins.war
What to do with the .war file
Jenkins – Nice to Meat You
Jenkins – Plug-ins

Manage Jenkins

- **Configure System**: Configure global settings and paths.
- **Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you want to change a setting that affects all builds.
- **Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- **System Information**: Displays various environmental information to assist trouble-shooting.
- **System Log**: System log captures output from java.util.logging output related to Jenkins.
- **Load Statistics**: Check your resource utilization and see if you need more computers for your builds.
- **Jenkins CLI**: Access/manage Jenkins from your shell, or from your script.
- **Script Console**: Executes arbitrary script for administration/trouble-shooting/diagnostics.

Left click
Jenkins Plugin manager

- Left click
Jenkins Git Plugin Installation Success

**Installing Plugins/Upgrades**

**Preparation**
- Checking internet connectivity
- Checking update center connectivity
- Success

**Git Plugin**
Success

*Restart Jenkins when installation is complete and no jobs are running*
Jenkins

Leg Work

New Project
Jenkins new project

1. Write a new Java project
2. Add an Ant build script to the project
3. Store the project in a Git repository
Jenkins New Job

Welcome to Jenkins! Please create new jobs to get started.

Build Queue
No builds in the queue.

Build Executor Status
<table>
<thead>
<tr>
<th>#</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Idle</td>
</tr>
<tr>
<td>2</td>
<td>Idle</td>
</tr>
</tbody>
</table>

Left click
Jenkins New Job 2

Fill in

Select
Jenkins New Job Top

Project name: ContinuousExample
Description: An example of continuous integration

Fill in
Select
Jenkins New Job Middle

Select

Update to your
  git repository
Jenkins New Job

- Left click
- Update
- Save
Jenkins New Job (actually after)

Left click

Left click
Jenkins Build Page
Jenkins Console Output

Console Output

Started by user anonymous
Checkout: ContinuousExample / C:\jenkins\workspace\ContinuousExample - hudson.remoting.LocalChannel@355bfcf78
Using strategy: Default
Checkout: ContinuousExample / C:\jenkins\workspace\ContinuousExample - hudson.remoting.LocalChannel@355bfcf78
Fetching changes from 1 remote Git repository
Fetching upstream changes from C:\git\E
Seen branch in repository origin/HEAD
Seen branch in repository origin/master
Commencing build of Revision f0c2a242f6e486986d8ab09b830d37ae498c43d5 (origin/HEAD, origin/master)
Checking out Revision f0c2a242f6e486986d8ab09b830d37ae498c43d5 (origin/HEAD, origin/master)
No change to record in branch origin/HEAD
No change to record in branch origin/master
[ContinuousExample] $ cmd.exe /C "ant.bat -file build.xml compile && exit %ERRORLEVEL%"
Buildfile: C:\jenkins\workspace\ContinuousExample\ContinuousExample\build.xml
remove dependencies:
compile:
BUILD SUCCESSFUL
Total time: 0 seconds
Finished: SUCCESS
Jenkins Git Build Data

Git Build Data

Revision: f0c2a242fce486986d8ab09b830d37ae498c43d5
- origin/HEAD
- origin/master

Built Branches
- origin/master: Build #1 of Revision f0c2a242fce486986d8ab09b830d37ae498c43d5
  (origin/HEAD, origin/master)
- origin/HEAD: Build #1 of Revision f0c2a242fce486986d8ab09b830d37ae498c43d5
  (origin/HEAD, origin/master)
What Happened After First Build?

1. Jenkins Cloned the Project
2. Jenkins used the Ant script to build the project
Other Things we Could have Done

Post-build Actions

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Publish JUnit test result report
- Publish Javadoc
- Record fingerprints of files to track usage
- Git Publisher
- E-mail Notification
Jenkins

Leg Work

Triggering a Build
Jenkins – When to Build?

1. Can be done manually (as we already did)
2. Can be done in fixed intervals – in general this is not recommended
3. Polling – build when the project changed
Jenkins – Triggering a Build (project configure)

I think this should get it to poll every minute

Check it out
Jenkins – Git Polling Log

Git Polling Log

Started on 05-Dec-2011 19:41:11
Using strategy: Default
[poll] Last Build : #2
[poll] Last Built Revision: Revision aa5c86fe5acfd66ae875f0817b697d216abb4c84
(origin/HEAD, origin/master)
Fetching changes from the remote Git repositories
Fetching upstream changes from C:\git\E
Polling for changes in
Seen branch in repository origin/HEAD
Seen branch in repository origin/master
Done. Took 19 sec
No changes

Build History

#2 05-Dec-2011 19:37:36
#1 05-Dec-2011 18:35:48
RSS for all RSS for failures

Help us localize this page

Page generated: 05-Dec-2011 19:42:11  Jenkins ver. 1.441
Jenkins – Build History

Build History of Jenkins

Export as plain XML

<table>
<thead>
<tr>
<th>Build</th>
<th>Time Since</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>ContinuousExample #1</td>
<td>1 hr 10 min</td>
<td>stable</td>
</tr>
<tr>
<td>ContinuousExample #2</td>
<td>8 min 12 sec</td>
<td>stable</td>
</tr>
<tr>
<td>ContinuousExample #3</td>
<td>3 min 12 sec</td>
<td>stable</td>
</tr>
</tbody>
</table>

Legend:  
RSS for all  
RSS for failures  
RSS for just latest builds
Jenkins

Leg Work

Other Features
You can use RSS to get updated.
Manage Jenkins

Manage Jenkins

Your container doesn’t use UTF-8 to decode URLs. If you use non-ASCII characters as a job name etc, this will cause problems. See Containers and Tomcat i18n for more details.

- **Configure System**
  Configure global settings and paths.

- **Reload Configuration from Disk**
  Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.

- **Manage Plugins**
  Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

- **System Information**
  Displays various environmental information to assist trouble-shooting.

- **System Log**
  System log captures output from java.util.logging output related to Jenkins.

- **Load Statistics**
  Check your resource utilization and see if you need more computers for your builds.

- **Jenkins CLI**
  Access/manage Jenkins from your shell, or from your script.

- **Script Console**
  Execute arbitrary script for administration/trouble-shooting/diagnostics.

- **Manage Nodes**
  Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

- **About Jenkins**
  See the version and license information

- **Prepare for Shutdown**
  Stops executing new builds, so that the system can be eventually shut down safely.
Configure Jenkins – e-mail notification

You can use e-mail notification to get updated

You can change the URL

Can execute concurrent builds guess where?