A Fresh Look at Building & Deploying uPortal

Bruce Tong & Drew Wills

Jasig Conference Denver, May 23, 2011

UNICON

My OHIO Portal

- OHIO University
  - Main campus in Athens, Ohio
  - ~20k students
  - ~2.2k faculty
  - ~3.5k staff
- My OHIO Portal
  - Work began in May, 2010
  - Based on uPortal 3.2.4 + a few recent patches
  - Applicants & students: Fall, 2010
  - Faculty, staff, others: light content

1. svn:externals + Overlay
2. “Puppet Master” Build Script
3. Maven Filters
4. RPMs

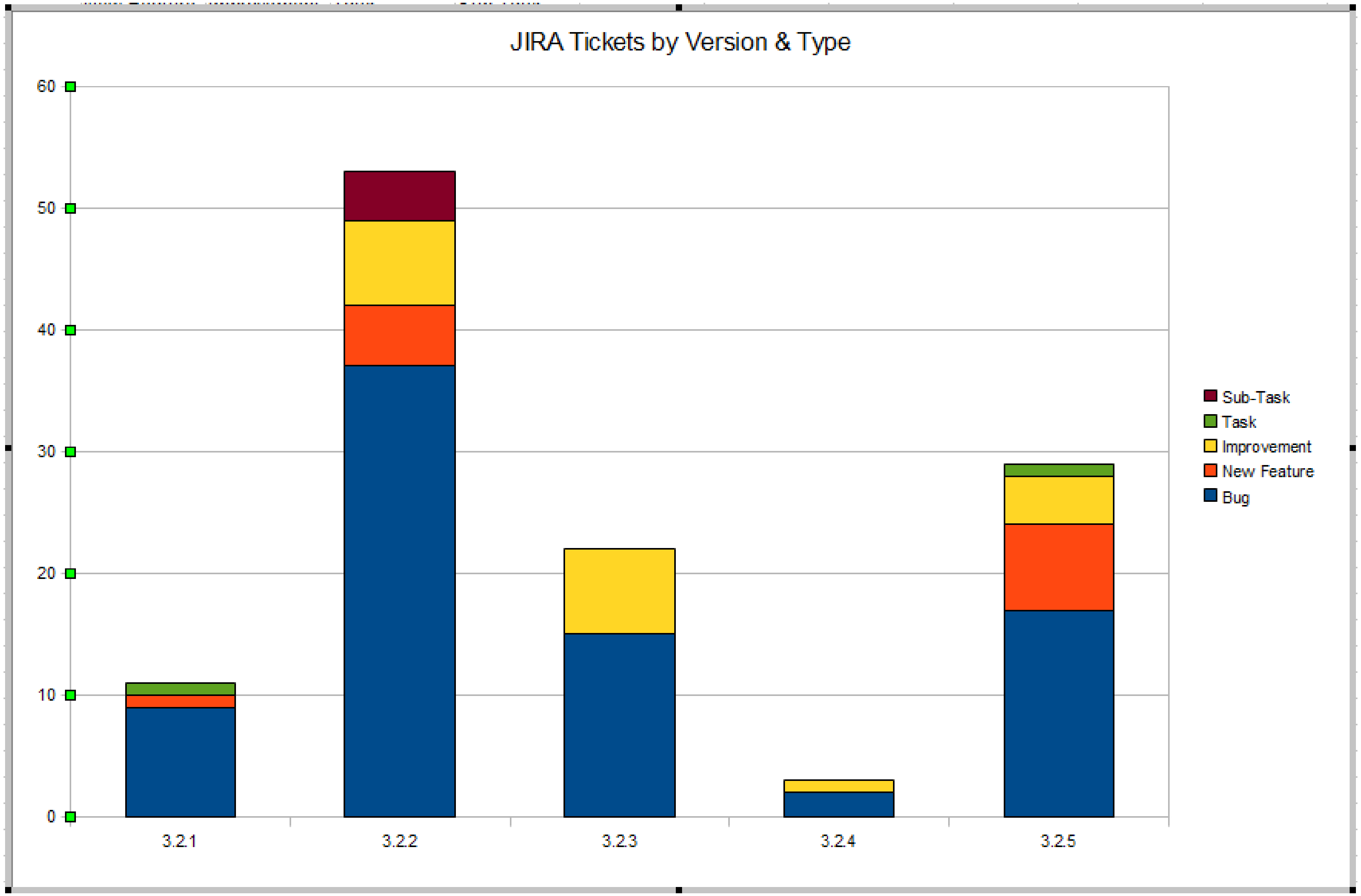
svn:externals + Overlay

How do I handle the fact that our source & Jasig project code change independently?

Keeping Up With The Joneses

- A lot of energy goes into uPortal & Jasig portlets, contributed by leading schools and talented professionals
- One individual – even a small team – can't compete with the pace of innovation
- Even if you could, why would you want to?
- Wouldn't it be better to benefit both from your own efforts and community contributions *continuously... at the same time?*

JIRA Summary 3.2.1 - 2011/05/01



Coordinated Practices

- **OHIO University** is not interested in duplicating that work (though they want to have it!)
- Nor does it particularly want to spend cycles cutting & applying patches, resolving conflicts, tracking down files that moved, *etc.*
- We've adopted a coordinated set of practices that make integrating the ongoing work from Jasig with the ongoing work at **OHIO** both **simple** and **quick**

More Than One Way

- We don't have a monopoly on good ideas
- The practices we use aren't the only good ones out there
- What works for us may not be perfect for you
- In particular, consider Vendor Branching
  - Popular, well-documented industry practice
  - Leverages diff tools to reconcile your changes with Jasig's automatically, where possible

svn:externals

- We use Externals Definitions to pull in Jasig source code
- This feature allows you to compose a working copy from separate, aggregated checkouts
- It even works across (SVN) repositories!
- Allows you to *develop on the original project* and commit patches directly!
- Which is a key tactic we use to keep local customizations to a minimum

.externals File

- Using an .externals (text) file to track changes is a popular convention

```
uPortal -r 23430 https://source.jasig.org/uPortal/branches/rel-3-2-patches/
email-preview -r 23391 https://source.jasig.org/portlets/email-preview/trunk/
FeedbackPortlet -r 23025 https://source.jasig.org/sandbox/FeedbackPortlet/trunk/
AnnouncementsPortlet -r 22710 https://source.jasig.org/portlets/AnnouncementsPortlet/trunk/
CalendarPortlet -r 23325 https://source.jasig.org/portlets/CalendarPortlet/trunk/
JasigWidgetPortlets -r 20743 https://source.jasig.org/sandbox/JasigWidgetPortlets/trunk/
SimpleContent -r 22815 https://source.jasig.org/portlets/SimpleContentPortlet/trunk/
TabbedSearch -r 22623 https://source.jasig.org/portlets/TabbedSearchPortlet/trunk/
WeatherPortlet -r 23340 https://source.jasig.org/portlets/WeatherPortlet/trunk/
```

- It's easier to make changes to a file
- You can view it in a web browser
- Always be certain to specify a revision number for each external item

svn:externals Setup

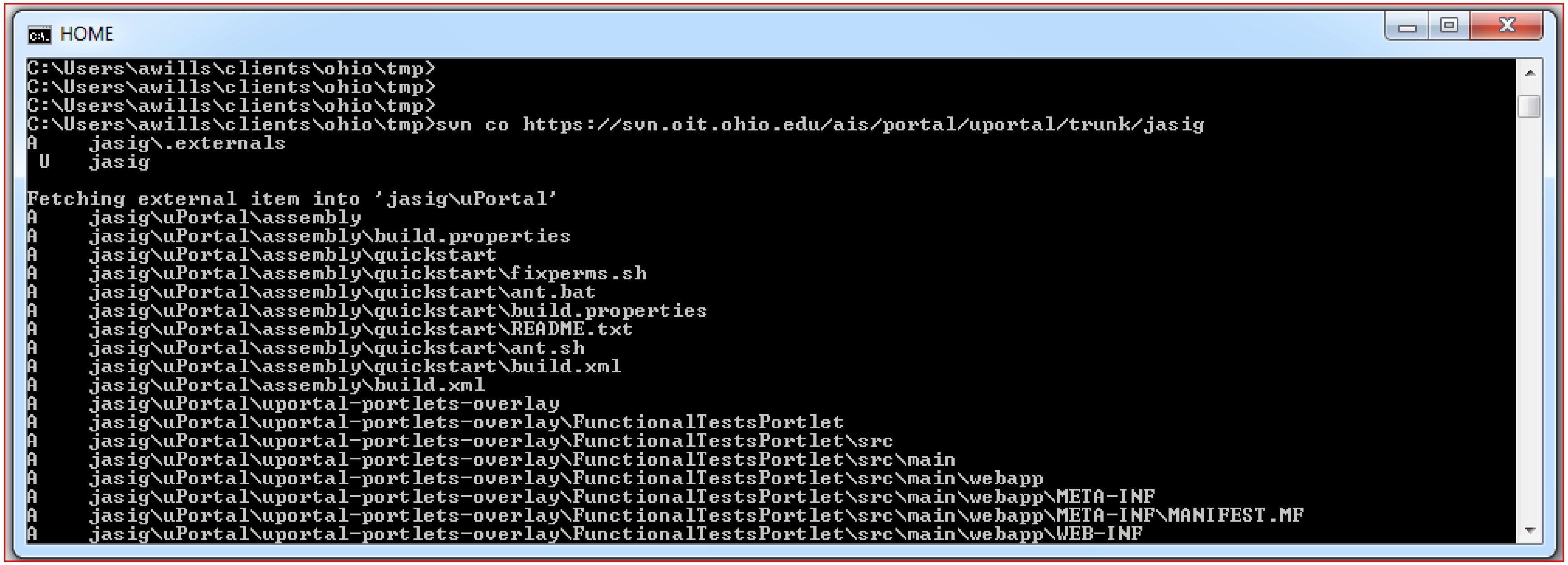
- Create the file first, then use svn propset -F

```
>svn add .externals

>svn propset svn:externals -F .externals .

>svn commit -N . .externals -m "Incorporating external checkouts"
```

- Commit the property & the file at the same time, *atomically*
- To change the version of an external dependency, just edit the file & repeat the process

svn:externals Checkout

- When you checkout your project, Subversion automatically includes external directories where you place them

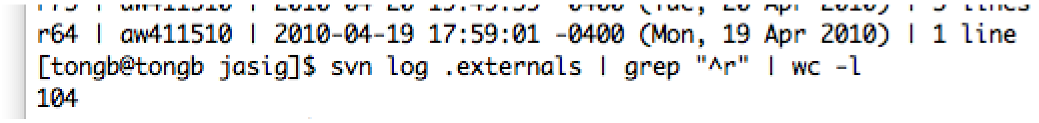

HOME

```
C:\Users\awills\clients\ohio\tmp>
C:\Users\awills\clients\ohio\tmp>
C:\Users\awills\clients\ohio\tmp>
C:\Users\awills\clients\ohio\tmp>svn co https://svn.oit.ohio.edu/ais/portal/uportal/trunk/jasig
A    jasig\.externals
 U   jasig

Fetching external item into 'jasig\uPortal'
A    jasig\uPortal\assembly
A    jasig\uPortal\assembly\build.properties
A    jasig\uPortal\assembly\quickstart
A    jasig\uPortal\assembly\quickstart\fixperms.sh
A    jasig\uPortal\assembly\quickstart\ant.bat
A    jasig\uPortal\assembly\quickstart\build.properties
A    jasig\uPortal\assembly\quickstart\README.txt
A    jasig\uPortal\assembly\quickstart\ant.sh
A    jasig\uPortal\assembly\quickstart\build.xml
A    jasig\uPortal\assembly\build.xml
A    jasig\uPortal\uportal-portlets-overlay
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp\META-INF
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp\META-INF\MANIFEST.MF
A    jasig\uPortal\uportal-portlets-overlay\FunctionalTestsPortlet\src\main\webapp\WEB-INF
```

- If you change the revision, Subversion automatically changes the external directory on update

Changes to .externals @ OHIO

- svn:exterlas for Jasig projects have been updated 104 times since April 2010



```
r64 | aw411510 | 2010-04-19 17:59:01 -0400 (Mon, 19 Apr 2010) | 1 line
[tongb@tongb jasig]$ svn log .externals | grep "^r" | wc -l
104
```

svn:externals & Local Changes

- svn:externals does not support keeping your config, skinning, and local customizations together with Jasig source code
- (FYI, Vendor Branching *does do this)*
- Consider *overlaying* local customizations on top of Jasig source

Do-It-Yourself Overlays

- We use a work/ directory to combine original source files with **OHIO** customizations

```
>mkdir work

>copy original/uPortal work/uPortal

>copy overlay/uPortal work/uPortal

>cd work/uPortal

>ant initportal
```

- Works with any type of project and build system
- Consider also Maven Overlays, which work with Maven <packaging>war</packaging> projects

“Puppet Master” Build Script

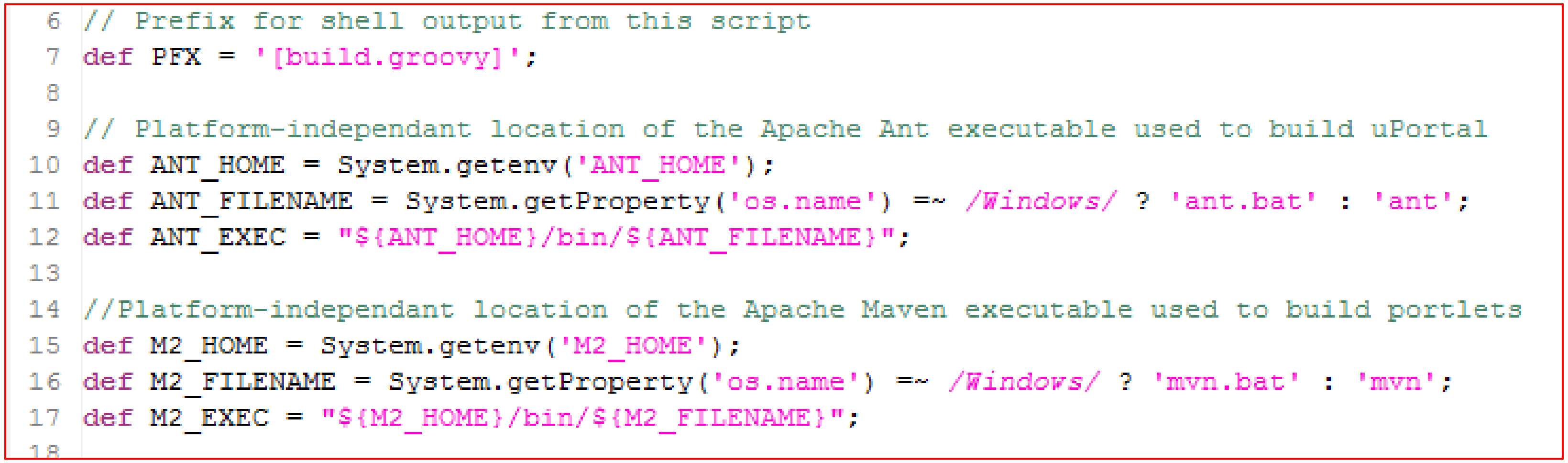
Managing the overlay process, aggregating portal & portlet builds

"Puppet Master" Build

- The inner-workings of uPortal & Jasig portlet build systems sometimes change in confusing ways
- But the way(s) you invoke them generally don't
- So you can safely *aggregate* the builds of uPortal and related projects
- For this purpose, we use Groovy
  - Java-based syntax
  - Platform-independent

Ant & Maven

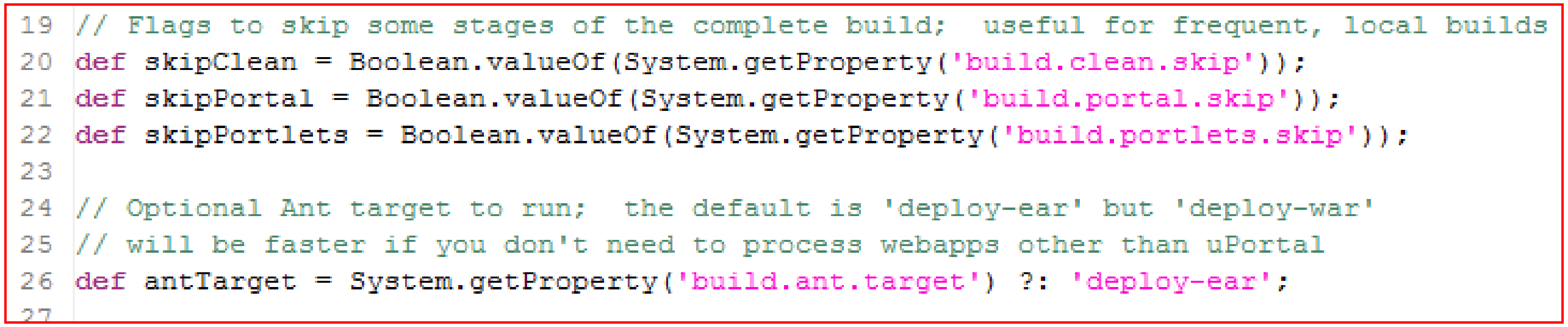
- Use the same installations of Ant & Maven as building from the command line



```
// Prefix for shell output from this script
def PFX = '[build.groovy]';

// Platform-independant location of the Apache Ant executable used to build uPortal
def ANT_HOME = System.getenv('ANT_HOME');
def ANT_FILENAME = System.getProperty('os.name') =~ /Windows/ ? 'ant.bat' : 'ant';
def ANT_EXEC = "${ANT_HOME}/bin/${ANT_FILENAME}";

//Platform-independant location of the Apache Maven executable used to build portlets
def M2_HOME = System.getenv('M2_HOME');
def M2_FILENAME = System.getProperty('os.name') =~ /Windows/ ? 'mvn.bat' : 'mvn';
def M2_EXEC = "${M2_HOME}/bin/${M2_FILENAME}";
```

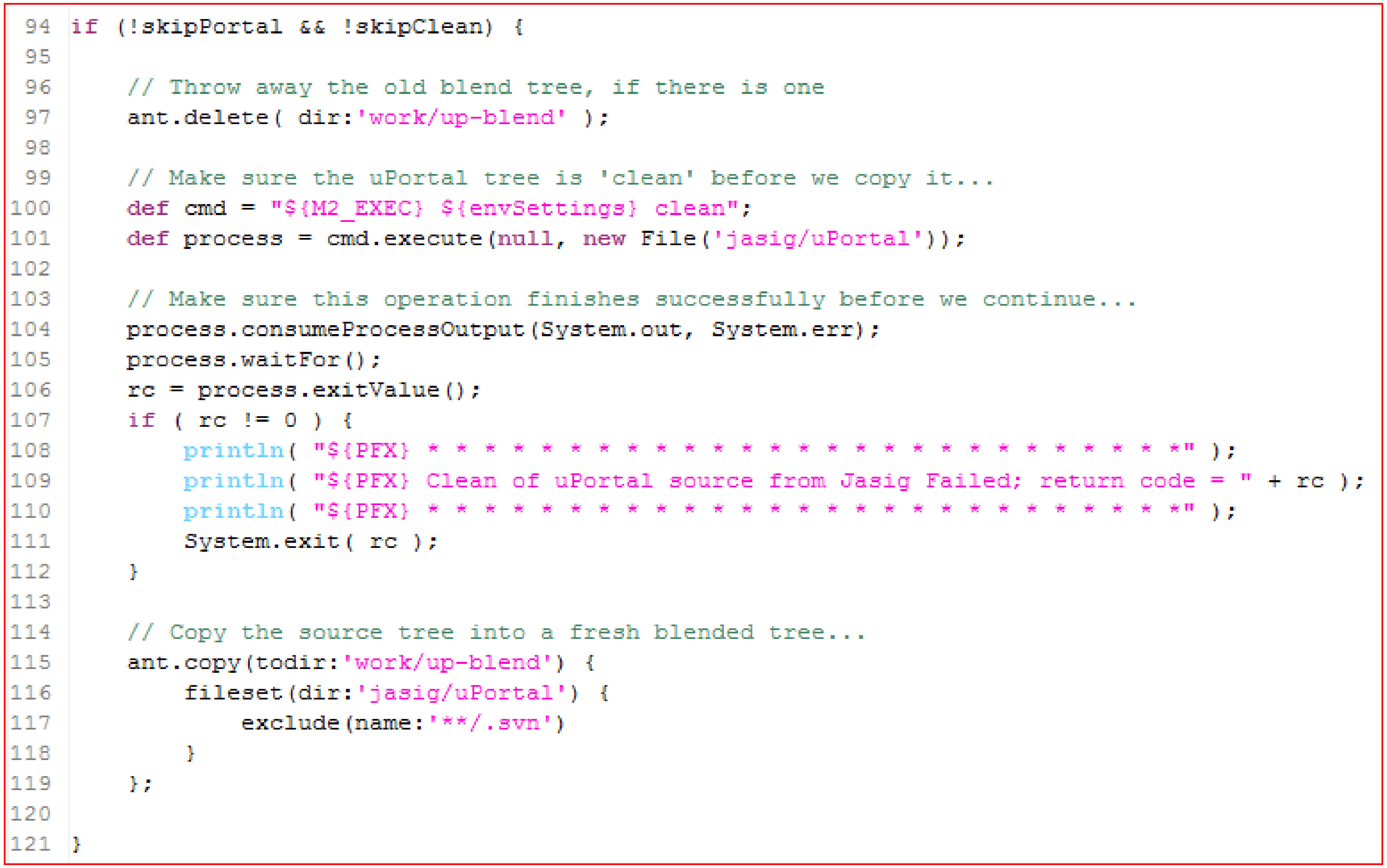
Build Parameters

- Make the default behavior *“build everything from scratch”*
- But allow users to skip parts of the process by passing special parameters



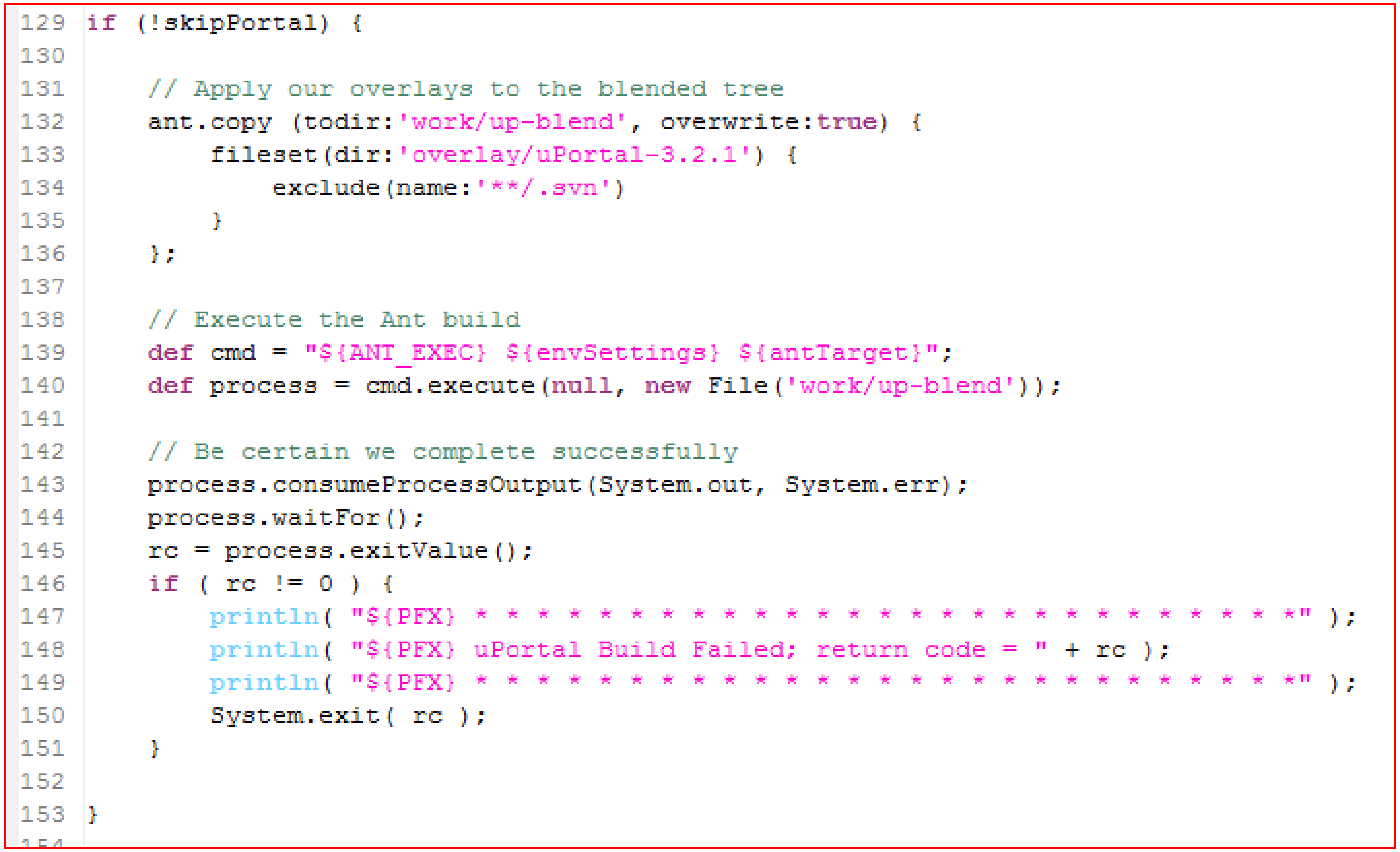
```
// Flags to skip some stages of the complete build;  useful for frequent, local builds
def skipClean = Boolean.valueOf(System.getProperty('build.clean.skip'));
def skipPortal = Boolean.valueOf(System.getProperty('build.portal.skip'));
def skipPortlets = Boolean.valueOf(System.getProperty('build.portlets.skip'));

// Optional Ant target to run;  the default is 'deploy-ear' but 'deploy-war'
// will be faster if you don't need to process webapps other than uPortal
def antTarget = System.getProperty('build.ant.target') ?: 'deploy-ear';
```

Reset the Portal Build



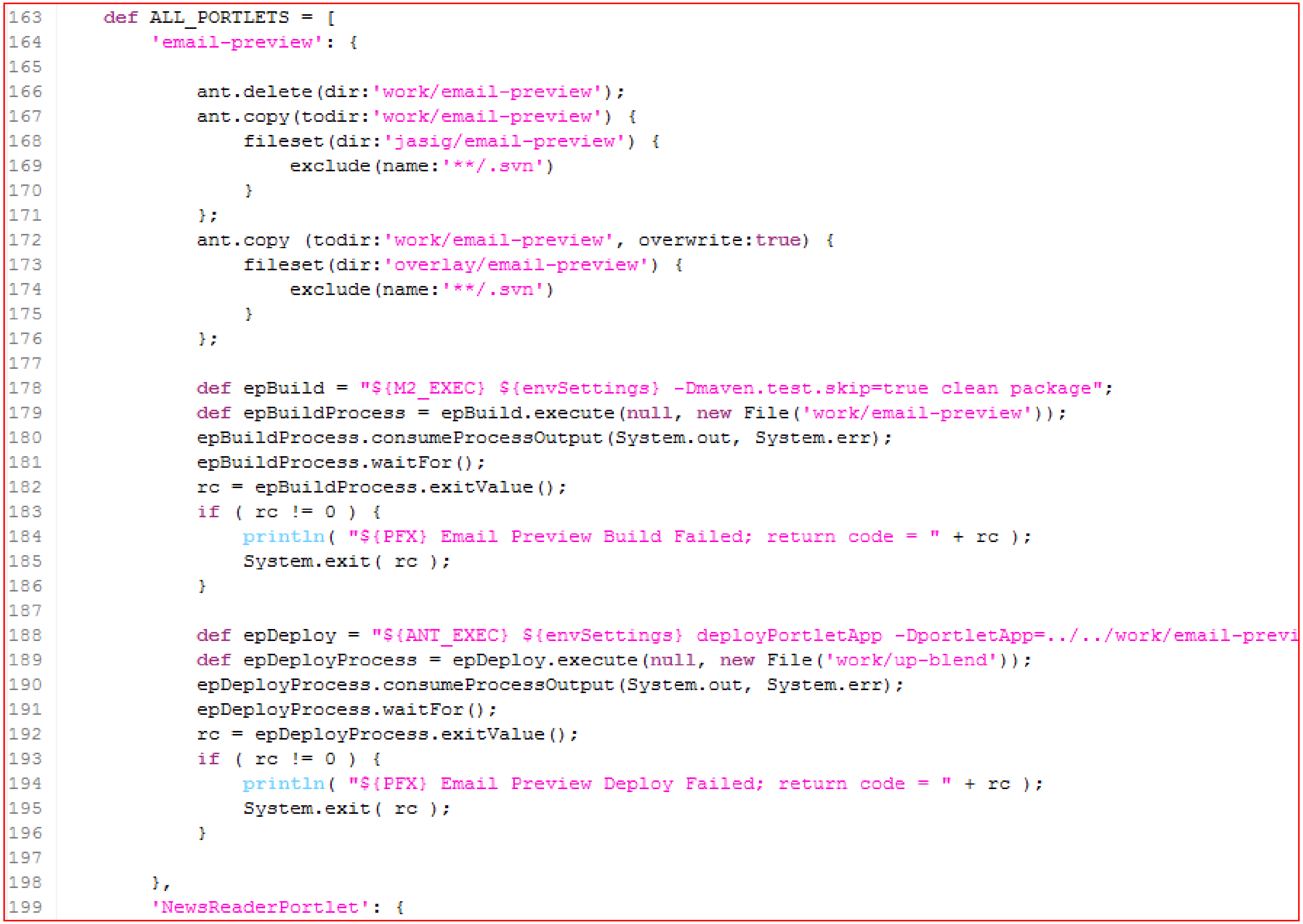
```groovy
if (!skipPortal && !skipClean) {

    // Throw away the old blend tree, if there is one
    ant.delete( dir:'work/up-blend' );

    // Make sure the uPortal tree is 'clean' before we copy it...
    def cmd = "${M2_EXEC} ${envSettings} clean";
    def process = cmd.execute(null, new File('jasig/uPortal'));

    // Make sure this operation finishes successfully before we continue...
    process.consumeProcessOutput(System.out, System.err);
    process.waitFor();
    rc = process.exitValue();
    if ( rc != 0 ) {
        println( "${PFX} * * * * * * * * * * * * * * * * * * * * * * * * * * *" );
        println( "${PFX} Clean of uPortal source from Jasig Failed; return code = " + rc );
        println( "${PFX} * * * * * * * * * * * * * * * * * * * * * * * * * * *" );
        System.exit( rc );
    }

    // Copy the source tree into a fresh blended tree...
    ant.copy(todir:'work/up-blend') {
        fileset(dir:'jasig/uPortal') {
            exclude(name:'**/.svn')
        }
    };

}
```

Overlay & Build the Portal



```
if (!skipPortal) {

    // Apply our overlays to the blended tree
    ant.copy (todir:'work/up-blend', overwrite:true) {
        fileset(dir:'overlay/uPortal-3.2.1') {
            exclude(name:'**/.svn')
        }
    };

    // Execute the Ant build
    def cmd = "${ANT_EXEC} ${envSettings} ${antTarget}";
    def process = cmd.execute(null, new File('work/up-blend'));

    // Be certain we complete successfully
    process.consumeProcessOutput(System.out, System.err);
    process.waitFor();
    rc = process.exitValue();
    if ( rc != 0 ) {
        println( "${PFX} * * * * * * * * * * * * * * * * * * * * * * * * * * *" );
        println( "${PFX} uPortal Build Failed; return code = " + rc );
        println( "${PFX} * * * * * * * * * * * * * * * * * * * * * * * * * * *" );
        System.exit( rc );
    }

}
```

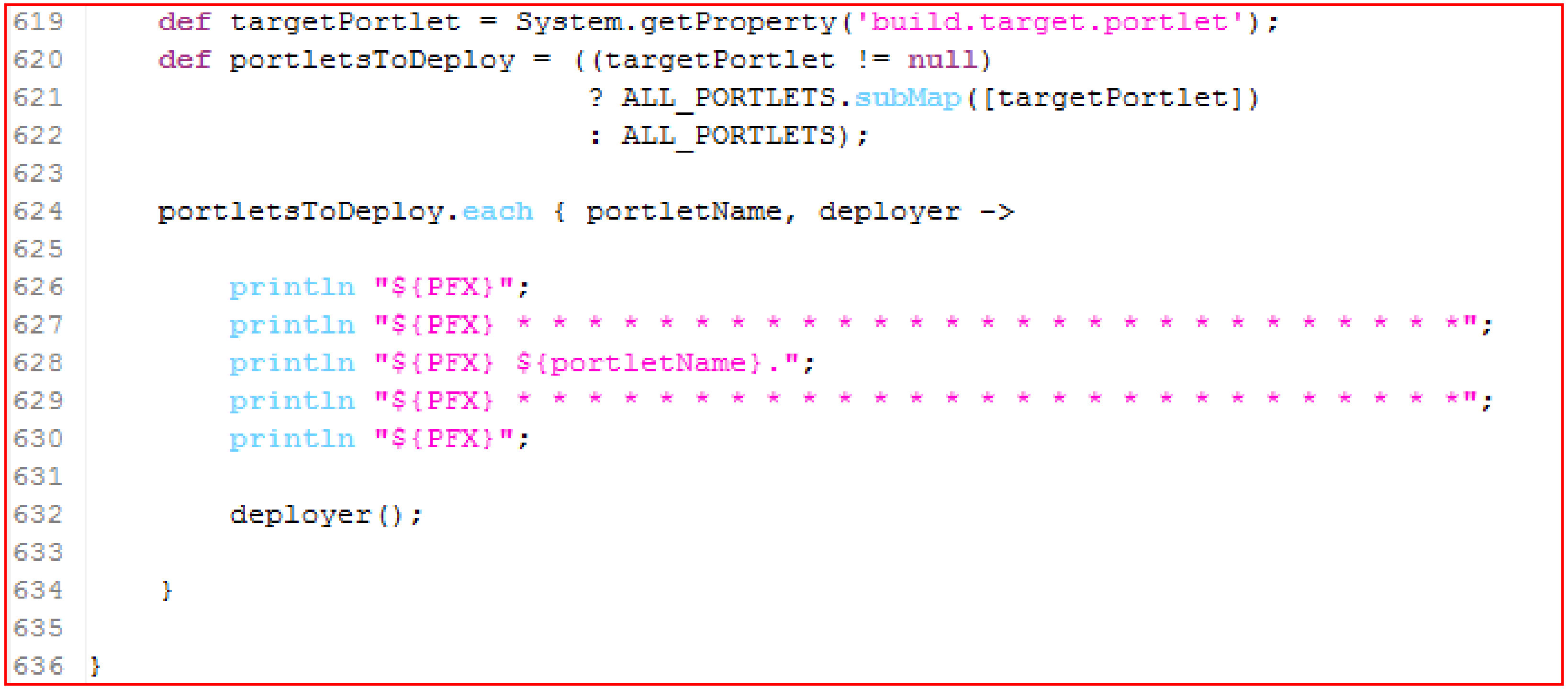
Put Portlet Builds in a Map of Closures



```
    def ALL_PORTLETS = [
        'email-preview': {

            ant.delete(dir:'work/email-preview');
            ant.copy(todir:'work/email-preview') {
                fileset(dir:'jasig/email-preview') {
                    exclude(name:'**/.svn')
                }
            };
            ant.copy (todir:'work/email-preview', overwrite:true) {
                fileset(dir:'overlay/email-preview') {
                    exclude(name:'**/.svn')
                }
            };

            def epBuild = "${M2_EXEC} ${envSettings} -Dmaven.test.skip=true clean package";
            def epBuildProcess = epBuild.execute(null, new File('work/email-preview'));
            epBuildProcess.consumeProcessOutput(System.out, System.err);
            epBuildProcess.waitFor();
            rc = epBuildProcess.exitValue();
            if ( rc != 0 ) {
                println( "${PFX} Email Preview Build Failed; return code = " + rc );
                System.exit( rc );
            }

            def epDeploy = "${ANT_EXEC} ${envSettings} deployPortletApp -DportletApp=../../work/email-previ
            def epDeployProcess = epDeploy.execute(null, new File('work/up-blend'));
            epDeployProcess.consumeProcessOutput(System.out, System.err);
            epDeployProcess.waitFor();
            rc = epDeployProcess.exitValue();
            if ( rc != 0 ) {
                println( "${PFX} Email Preview Deploy Failed; return code = " + rc );
                System.exit( rc );
            }

        },
        'NewsReaderPortlet': {
```

Invoke Portlet Deployer(s)

- Choose a portlet with -Dbuild.target.portlet or build them all



```
    def targetPortlet = System.getProperty('build.target.portlet');
    def portletsToDeploy = ((targetPortlet != null)
                            ? ALL_PORTLETS.subMap([targetPortlet])
                            : ALL_PORTLETS);

    portletsToDeploy.each { portletName, deployer ->

        println "${PFX}";
        println "${PFX} * * * * * * * * * * * * * * * * * * * * * * * * * * *";
        println "${PFX} ${portletName}.";
        println "${PFX} * * * * * * * * * * * * * * * * * * * * * * * * * * *";
        println "${PFX}";

        deployer();

    }

}
```

Maven Filters

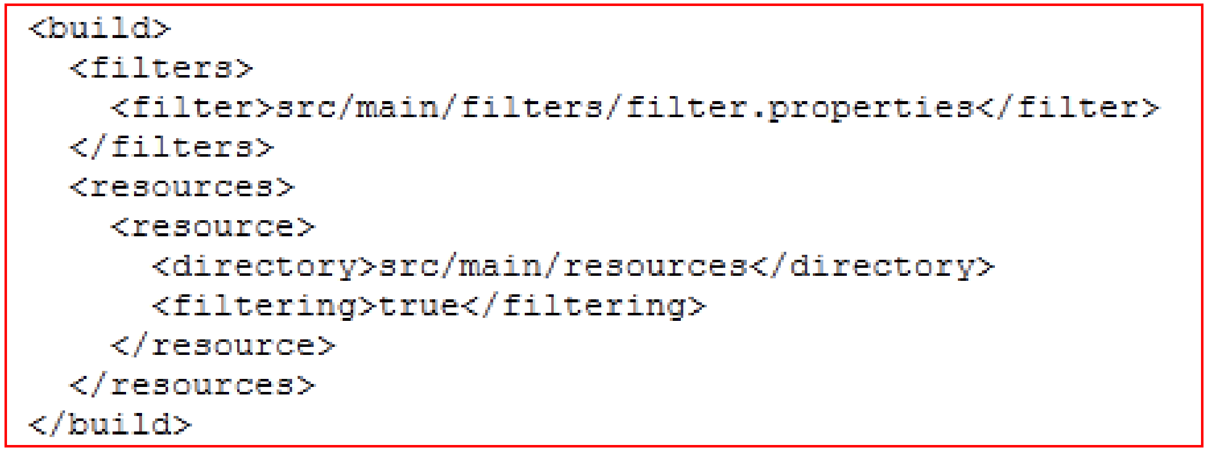
Manage different config & data for different environments

Maven Filters

- Allows project files to contain values that will be supplied at build time
- These values can come from several sources:
  - The pom file (*e.g.* ${pom.version})
  - The settings file (*e.g.* ${settings.localRepository})
  - Pom <properties> (*e.g.* ${my.custom.value})
  - -D parameters (*e.g.* mvn -Dfoo=bar install)
  - A filters file

Filters Files

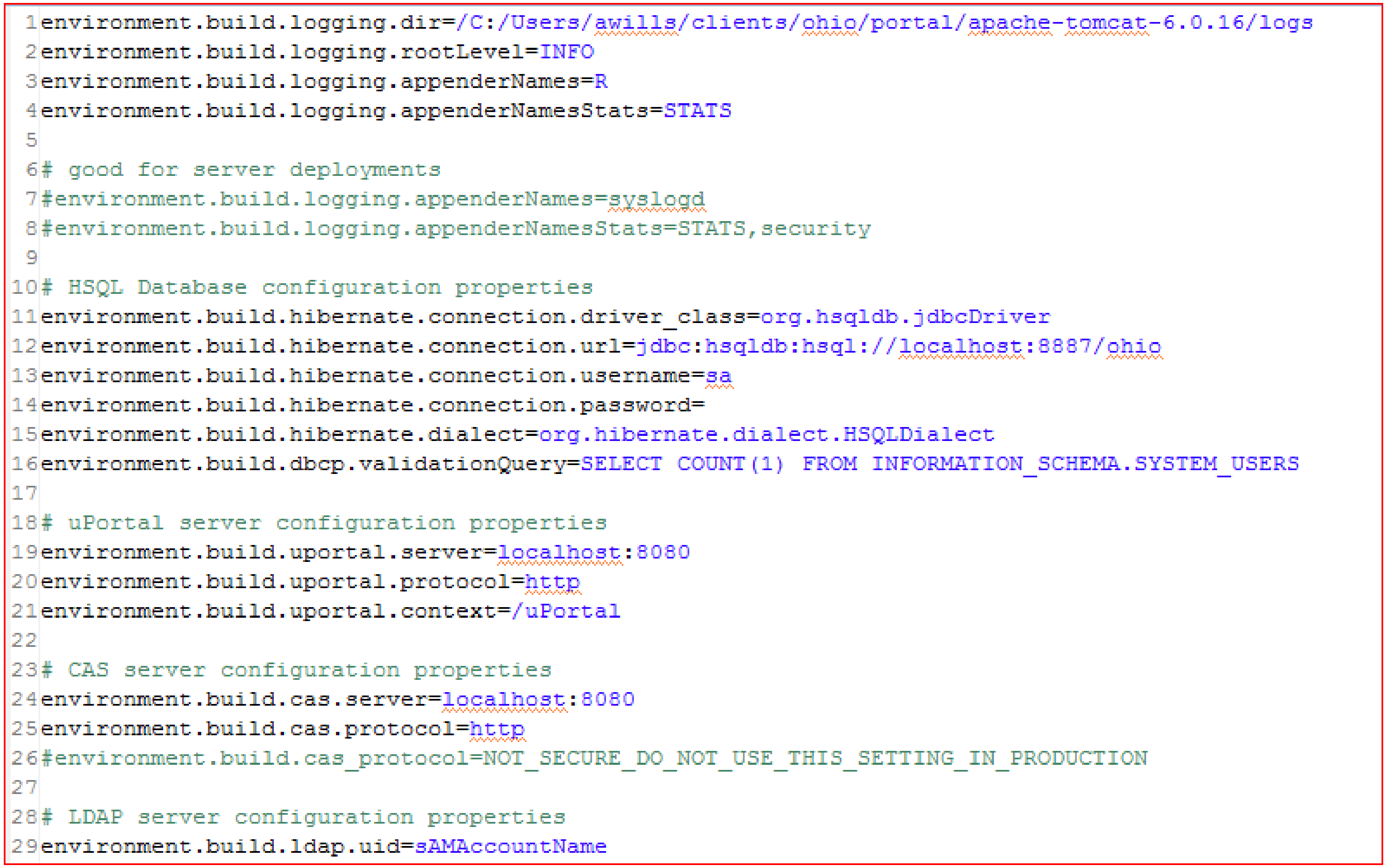
- Use Maven filters files to gather values for filters into one file



```
<build>
  <filters>
    <filter>src/main/filters/filter.properties</filter>
  </filters>
  <resources>
    <resource>
      <directory>src/main/resources</directory>
      <filtering>true</filtering>
    </resource>
  </resources>
</build>
```

- Use a different file for each environment!
- WARNING: Never filter binary files

local.properties



```
environment.build.logging.dir=/C:/Users/awills/clients/ohio/portal/apache-tomcat-6.0.16/logs
environment.build.logging.rootLevel=INFO
environment.build.logging.appenderNames=R
environment.build.logging.appenderNamesStats=STATS

# good for server deployments
#environment.build.logging.appenderNames=syslogd
#environment.build.logging.appenderNamesStats=STATS,security

# HSQL Database configuration properties
environment.build.hibernate.connection.driver_class=org.hsqldb.jdbcDriver
environment.build.hibernate.connection.url=jdbc:hsqldb:hsql://localhost:8887/ohio
environment.build.hibernate.connection.username=sa
environment.build.hibernate.connection.password=
environment.build.hibernate.dialect=org.hibernate.dialect.HSQLDialect
environment.build.dbcp.validationQuery=SELECT COUNT(1) FROM INFORMATION_SCHEMA.SYSTEM_USERS

# uPortal server configuration properties
environment.build.uportal.server=localhost:8080
environment.build.uportal.protocol=http
environment.build.uportal.context=/uPortal

# CAS server configuration properties
environment.build.cas.server=localhost:8080
environment.build.cas.protocol=http
#environment.build.cas_protocol=NOT_SECURE_DO_NOT_USE_THIS_SETTING_IN_PRODUCTION

# LDAP server configuration properties
environment.build.ldap.uid=sAMAccountName
```

Maven Filters in uPortal 3.2

- We set up filtering in uPortal & Jasig portlet pom.xml files to insert these values in the appropriate places
- But (especially in uPortal) there's a lot of custom logic & sophistication baked in the build...
  - Web server deployment
  - Unit tests & static analysis
  - Pluto-fication
  - yuicompressor & resource-aggregator
- So it's not much fun to maintain local deltas to build files
- But thankfully...

Maven Filters in uPortal 4

UP-2813: *Add hooks for Maven filters to uPortal poms to support multi-environment builds*

- Use build.properties itself as a filters file
- Or use build.${env}.properties for multiple environments if you want to keep them in the same place
- Or choose your own location by specifying the filters.file property in build.properties

RPMs

Bundling uPortal & portlets for RedHat Linux

RPMs

- Evolution of Software Deployments
  - Manual → Scripted → **Packaged** → Automated
- Repeatable in all Environments
  - Dev → Test → QA → Prod
- Auditor Friendly
  - Allows Separation of Engineering and Operations
- ITIL Friendly
  - Clean Separation of Release and Change Mgmt

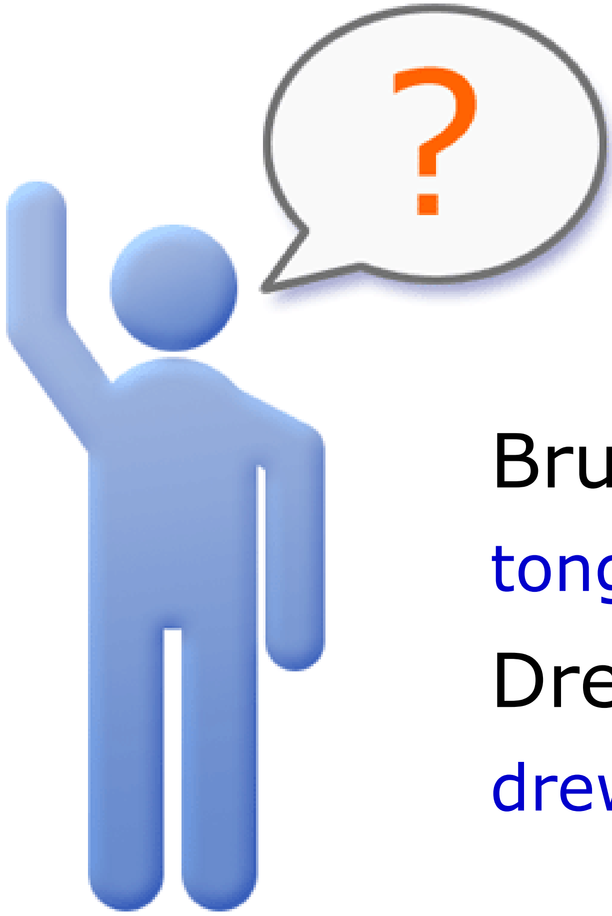
RPM Contents

- Software
  - uPortal and Portlet WARs
- System Files
  - Service Initialization Scripts
  - Configuration Files (logrotate, cron, etc...)
- RPM Specification
  - File List
  - Deployment Event Scripts, if needed

RPM Commands

- Install
  - rpm -i uportal-prod-2011-05-13-14:20:05.rpm
- Update
  - rpm -U uportal-prod-2011-05-15-09:55:35.rpm
- Remove
  - rpm -e uportal-prod
- Query
  - Version, File List, MD5 Checksums, more...

Questions?



Bruce Tong

tongb@ohio.edu

Drew Wills

drew@unicon.net