Pragmatic Project Automation

Mike Clark
Clarkware Consulting, Inc.
www.clarkware.com
Road Map

One-Step Builds
- Compile
- Test
  - (On Command)

Installation & Deployment
- Deliver
- Auto-Update
- Install
- Test
  - (Monthly)

Monitoring
- Cell Phone / Pager
- log4j
- Visual Devices
- RSS
  - (Continuously)

Scheduled Builds
- Checkout
- Compile & Test
- Email
  - (Hourly)

Push-Button Releases
- Branch
- Test
- Package
- Release
  - (Weekly)
Types of Automation

- Scheduled Automation
- Triggered Automation
- Commanded Automation
Why Should You Automate?

- You’ve got better things to do
- Improves consistency and repeatability
- Minimizes risk and errors
- Reduces need for documentation
- Saves time and money
Why Should You Automate?

- Gets you off the treadmill
- Puts your computer to work
- More time to do exciting stuff
- You go home earlier!
When Do You Automate?

- Manual task is boring
  - risk making errors
- Procedure needs to be consistent
  - machines are better at repetitive tasks
- Once, twice...
  - third time won’t be the last
- Might forget to do a chore
What Do You Need?

- Version control
  - synchronize work
- Automated tests
  - gain confidence
- Scripting
  - train the computer
- Monitoring devices
  - get timely feedback
One-Step Builds

Build Process

- Build File
- Production Code
- Test Code
- Vendor Libraries
- Documentation

Outputs:
- Documentation
- Scripts / Executables
- Your Libraries
- Vendor Libraries
CRISP Builds

- **Complete**
  - recipe lists all ingredients
- **Repeatable**
  - version control time machine
- **Informative**
  - radiate valuable information
- **Scheduleable**
  - complete and repeatable
- **Portable**
  - machine-independent
Beware the Compile Button!

- Not everybody uses the same IDE
  - even when they do, configurations differ
- Not schedulable
- Externalize the canonical build process
  - modern IDEs can use standard build systems
Project Directory Structure

(Template at http://pragmaticautomation.com)
Commanded Build

$ javac -classpath vendor/lib/lucene-1.3.jar:
  vendor/lib/Tidy.jar -d build/prod
  src/com/pragauto/dms/*.*.java

$ sh compile.sh
Ant

- Ubiquitous
- Portable
- Tracks file dependencies
- Comprehensive set of tasks
<project name="dms" default="compile" basedir=".">

<property name="build.dir" location="build"/>
<property name="build.prod.dir" location="${build.dir}/prod"/>
<property name="build.test.dir" location="${build.dir}/test"/>
<property name="doc.dir" location="doc"/>
<property name="index.dir" location="index"/>
<property name="src.dir" location="src"/>
<property name="test.dir" location="test"/>
<property name="vendor.lib.dir" location="vendor/lib"/>

<path id="project.classpath">
    <pathelement location="${build.prod.dir}"/>
    <pathelement location="${build.test.dir}"/>
    <fileset dir="${vendor.lib.dir}">
        <include name="*.jar"/>
    </fileset>
</path>
</project>
<target name="prepare">
  <mkdir dir="${build.prod.dir}"/>
  <mkdir dir="${build.test.dir}"/>
</target>

<target name="compile" depends="prepare">
  <javac srcdir="${src.dir}" destdir="${build.prod.dir}">
    <classpath refid="project.classpath" />
  </javac>
</target>
<target name="compile-tests" depends="compile">
  <javac srcdir="${test.dir}" destdir="${build.test.dir}"
    <classpath refid="project.classpath" /></javac>
</target>

<target name="test" depends="compile-tests">
  <junit haltonfailure="true">
    <classpath refid="project.classpath" />
    <formatter type="brief" usefile="false" />
    <batchtest>
      <fileset dir="${build.test.dir}"
        includes="**/*Test.class" />
    </batchtest>
    <sysproperty key="index.dir" value="${index.dir}" />
  </junit>
</target>

</project>
Running the Build

$ ant test
void compile() {
    prepare()
    ant.javac(srcdir: srcDir, 
        destdir: buildProdDir, 
        classpath: projectClasspath())
}

void instrumentJARs() {
    scanner = ant.fileScanner {
        filesystem(dir: vendorLibDir)
    }
    scanner.each { instrument(it) }
}

void instrument(file) {
    println("Instrumenting ${file}"$
    // insert code here
    
}
Scheduled Builds

- No more “But it works on my machine!”
- Continuous integration and nightly builds
- Detects problems early
  - integration (compile time) problems
  - failing test (run time) problems
- Finding and fixing problems is easier
Scheduling with cron or at

0 2 * * * $HOME/work/dms/build.sh

at 02:00 /every: c:\work\dms\build.bat
CruiseControl

- Open source continuous integration framework
- Extensible with plugins
- Supportive community
- Scales up to big projects
- Simple things easy; difficult things possible
Choosing a Build Machine

- You can’t afford **not** have a dedicated build machine

**Check me out:** Unit test seeks code for ongoing relationship. I know I can be demanding and try, but let me check you out—you’ll feel better for it. 555-8017

**Building Relationships:** I’m a single-processor, middle-age bit twiddler (SPMABT) seeking a long-lasting relationship with a stable provider of data who can handle change. My friends say I’m resourceful because I can access version control repositories. Unlike those jet-setting laptops, I enjoy staying at the office 24 hours a day. The last time I was rebooted was more than 7 days ago, but I bounced right back. I look forward to meeting you face to face or we can chat discreetly over one of my many remote interfaces.

**Ideal Lover:** Seek TV for fun and romance. Likes daytime television, reality shows, and anything involving animals eating things. Remote relationship preferred. Call, but only when there’s nothing worth watching. 555-6527
Build Workspace

(builds/)

- cc-build.xml
- config.xml

(checkout/)

- dms/

(logs/)

(Template at http://pragmaticautomation.com)
<project name="cc-build" default="build" basedir="checkout">

  <target name="build">
    <delete dir="dms" />
    <cvs command="co dms" />
    <ant antfile="build.xml" dir="dms" target="test" />
  </target>

</project>
<cruisecontrol>
    <project name="dms" buildafterfailed="false">
        <bootstrappers>
            <currentbuildstatusbootstrapper
                file="logs/dms/currentbuildstatus.txt" />
        </bootstrappers>
        <modificationset quietperiod="60">
            <cvs localworkingcopy="checkout/dms" />
        </modificationset>
        <schedule interval="60">
            <ant buildfile="cc-build.xml" target="build" />
        </schedule>
        <log dir="logs/dms">
            <merge dir="checkout/dms/build/test-results" />
        </log>
        <publishers>
            <currentbuildstatuspublisher
                file="logs/dms/currentbuildstatus.txt" />
        </publishers>
    </project>
</cruisecontrol>
Push-Button Releases

- Releases pay the bills
  - generating a release should be easy

- A release consists of:
  - unique name and version number
  - set of features
  - complete set of files
  - installation script or utility
1. Test the mainline directory
2. Create a release branch
3. Check out the release branch
4. Build and test the release branch
5. Create a distribution file
6. Test the distribution file
7. Tag the release
8. Hand off the distribution file to QA
Release Branch
Packaging the Distribution File

$ ant -buildfile package.xml -Dversion=1.0
Testing the Release

$ ant test
  -Dbuild.prod.dir=/testinstall/dms-1_0/lib \
  -Dvendor.lib.dir=/testinstall/dms-1_0/vendor/lib
Automating the Release Procedure

Steps 1-4

$ sh release_branch.sh 1_1

Steps 5-8

$ sh release_generate.sh 1_1

0 2 * * * $HOME/work/dms/release_daily.sh
Installation & Deployment

• Delivering the goods to end-users
  • copy or ftp the distribution file
  • download link on a web site
  • ship CDs
Installers

• Many free and commercial Java installers
  • IzPack - http://www.izforge.com/izpack

• Nullsoft Scriptable Install System (NSIS)
  • Windows-based
  • Scriptable and customizable
  • http://nsis.sourceforge.net
Deployment

- Ant tasks
  - jar, war, ear
- Hot deploy
  - copy, scp, ftp
- Server support
  - Tomcat Ant tasks
  - WebSphere wsadmin with Jython
<target name="deploy">
  <deploy url="${tomcat.manager.url}"
    username="${tomcat.manager.username}" 
    password="${tomcat.manager.password}" 
    path="/${webapp.name}" 
    war="file:${dist.dir}/${webapp.name}.war"/>
</target>

$ ant deploy 
  -Dtomcat.manager.url=http://qaserver:8080/manager
- Automating the tech support checklist
$ sh bin/selftest.sh
.F.F
Time: 0.092
There were 2 failures:

1) testJavaVersion(com.pragprog.dms.selftest.InstallTests)
   junit.framework.AssertionFailedError:
   Incompatible Java version. Requires version 1.4.x, but found 1.3.1

2) testIndexDirectory(com.pragprog.dms.selftest.InstallTests)
   junit.framework.AssertionFailedError:
   Directory not readable: /Users/somebody/dms-1_0/index

FAILURES!!!
Tests run: 2, Failures: 2, Errors: 0
Auto-Update

- Custom updaters
  - More prevalent now
  - Reusable solutions may emerge
- Java Web Start
  - Swing or SWT applications
  - requires JRE
  - “sandbox” constraints
Monitoring

• Eyes in the back of your head
• Timely and accurate feedback
• Synchronization points for dislocated teams
• Take these techniques as suggestions
  • and discover new ones!
Builds Results at the Beach

<email mailhost="your.smtp.host"
       returnaddress="cruisecontrol@yourdomain.com"
       buildresultssurl="">

    <map alias="fred" address="303551212@mobile.att.net" />
    <failure address="fred" reportWhenFixed="true" />

</email>
Visual Devices!
X10 Devices

<x10publisher port="COM1"
    passDeviceCode="A2"
    failDeviceCode="A1" />

(Software at http://pragmaticautomation.com)
Other Monitoring Techniques

- Screen scraping
- Process monitoring
- Watching log files
- Trip wires
- Ask a web server
log4j.logger.sales=INFO, rss
log4j.appender.rss=com.pragauto.dms.RSSAppender

Logger logger = Logger.getLogger("sales");
logger.info("Joe (joe@xyz.com) ordered 10 books!");
Post-Mortem Analysis

- Script that creates a crash report
  - application, OS, and JVM version
  - last x log messages
  - environment variables and system properties
- Automatically submit report over the web
  - updates an issue-tracking system
Other Monitoring Devices

(Have you spiced up your project?)
Summary

• Automation frees up your time
• Consistency, accuracy, repeatability
• Make it fun!
• **Pragmatic Project Automation**
  Mike Clark
  The Pragmatic Bookshelf (July 2004)
  http://pragmaticprogrammer.com/sk/auto

• **PragmaticAutomation.com**
  News, stories, tools, tips, and tricks!
Other Automation Books

- **Pragmatic Unit Testing**
  Andrew Hunt and Dave Thomas
  The Pragmatic Bookshelf (2003)
  http://pragmaticprogrammer.com

- **Pragmatic Version Control**
  Dave Thomas and Andrew Hunt
  The Pragmatic Bookshelf (2003)
  http://pragmaticprogrammer.com

- **Java Development with Ant**
  Erik Hatcher and Steve Loughran
  Manning (2002)
Free Tools

- PragAuto Software - http://pragmaticautomation.com
- Ant - http://jakarta.apache.org/ant
- CVS - http://cvshome.org
- Groovy - http://groovy.codehaus.org
- JUnit - http://junit.org
Customized Training and Mentoring

http://clarkware.com
mike@clarkware.com
Thanks, and have fun!