ImageJ Plugins
Maven, GIT and Eclipse

Olivier Burri, Romain Guiet
June 2016, EPFL BIOP
BioImaging & Optics Platform
Goal

• Simplify **ImageJ Plugin creation** as much as possible. **BUT ONLY WHEN IT IS USEFUL**
  – Multiple dependencies (Other plugins, external libraries)
  – Collaborating with other developers
  – Need to keep up to date with other libraries

  – **You want to play nice with the rest of the Fiji community**
Eclipse

- Is an IDE (Integrated Development Environment)

Syntax highlight, error checking, project management, one click compile...
• Is an SCM (Source Control Manager)
  – Keeps a history of who did what
  – Makes code development easier when collaborating
  – Makes source code sharing A LOT more convenient

SOME KEYWORDS

Repository: Your project
Branch: A ‘version’ of your project
Clone: Copying a repository
Commit: Adding changes to a branch
Merge: Combining branches together
Push: Updating a remote repository from your local one
Pull: Updating your local repository from a server

Code sharing, development collaboration, commits allow for easy debugging of larger projects.
Maven

- Is a Build Automation Tool for Java Projects
  - Based on conventions, so **only exceptional cases need to be written**

**ANT Buildfile**

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project basedir="." default="create-jar" name="jangles-distribution">
  <import file="build.xml"/>
  <property name="janglesdist.location" value="dist"/>
  <target name="distrib-init">
    <mkdir dir="dist"/>
  </target>
  <target name="distrib-clean">
    <delete dir="dist"/>
  </target>
  <target name="create-jar" depends="distrib-init">
    <jar destfile="${janglesdist.location}/jangles.jar" basedir="bin"/>
    <echo message="in jar!"/>
  </target>
</project>
```

**Maven POM**

```xml
<project xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd" xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.gsm.app</groupId>
  <artifactId>namaste</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>namaste</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

**How to compile**

Just info about your project. Handles compiling, cleaning, distributing **IMPLICITLY**

**Where to compile**

More lines for dependencies
One **target** per action, all must be written (albeit just once)

**Project Name**

Maven

**Version**

ANT Buildfile

**Dependencies**

Maven POM

**Project name**

**Version**

**Dependencies**
Summary

To automatically compile the code, resolve dependencies and stay updated with the latest Fiji developments

To easily get, share and track code

To put it all together in a nice interface
Step-By-Step: JDK Installation

Install the latest JDK

- JDK stands for Java Development Kit

Note: The standard Java installation contains only the JRE: or Java Runtime Environment

http://www.oracle.com/technetwork/java/javase/downloads/
Step-By-Step: GIT Installation

https://git-scm.com/
Side-Step: Ensure Windows **PATH** and **JAVA_HOME** are set

- This step ensures that everything is going to work without complaints
Side-Step: Ensure Windows **PATH** and **JAVA_HOME** are set

- Add path to java *bin* folder at the beginning of the Variable Value field followed by a ";" 
  eg. C:\Program Files\Java\jdk1.8.0_45\bin;
Side-Step: Ensure Windows PATH and JAVA_HOME are set

- Add path to java jre folder
  eg. C:\Program Files\Java\jdk1.8.0_45\jre;

- Create the JAVA_HOME variable if it does not exist
Step-By-Step: Eclipse Installation

• Install Eclipse for Java Developers

http://www.eclipse.org/downloads/

1. Download ZIP File
2. Drag and Drop `eclipse` folder somewhere
Prepare Maven-GIT Integration
Prepare Maven-GIT Integration

And restart Eclipse...
Import Maven Project from GIT

Use File > Import...

Make sure ‘git’ is available from SCM URL
And provide url where the project can be taken