# Control Room Accelerator Physics

## Day 1

## Installing Open XAL at Command Line

# Plan

- Get Open XAL from Repository
- Build Open XAL from the command line
- Tour of project
- Review build options
- Configuration for Running Applications and Scripts

# Requirements

- Git 1.7.5
- Ant 1.8
- Java J2SE 7 with JDK
- Terminal for command line

# Preparation

- Create a directory where you want to install Open XAL alongside other related files
  - We will refer to this directory as the “Project Directory”
  - It will contain source code and other files such as documents and configuration files
  - The following path could be considered:

```
~/Projects/OpenXAL
```

# Download the Code

1. Open a Terminal
2. Change Directory to the Project Directory
3. Type the following to fetch the code:

- `git clone http://git.code.sf.net/p/xaldev/openxal`
- The resulting openxal directory will be referred to as “Open XAL Home”

# Building Open XAL

1. Open a Terminal
2. Change Directory to the Open XAL Home directory
3. Type the following to build everything:

- `ant`

# Get and Build Open XAL
# Sample Session

```
> cd
> mkdir -p Projects/OpenXAL
> cd Projects/OpenXAL
> git clone http://git.code.sf.net/p/xaldev/
openxal
Cloning into 'openxal'...
....
> cd openxal
> ant
Buildfile: /Users/t6p/Scrap/OpenXAL/openxal/
build.xml
....
```

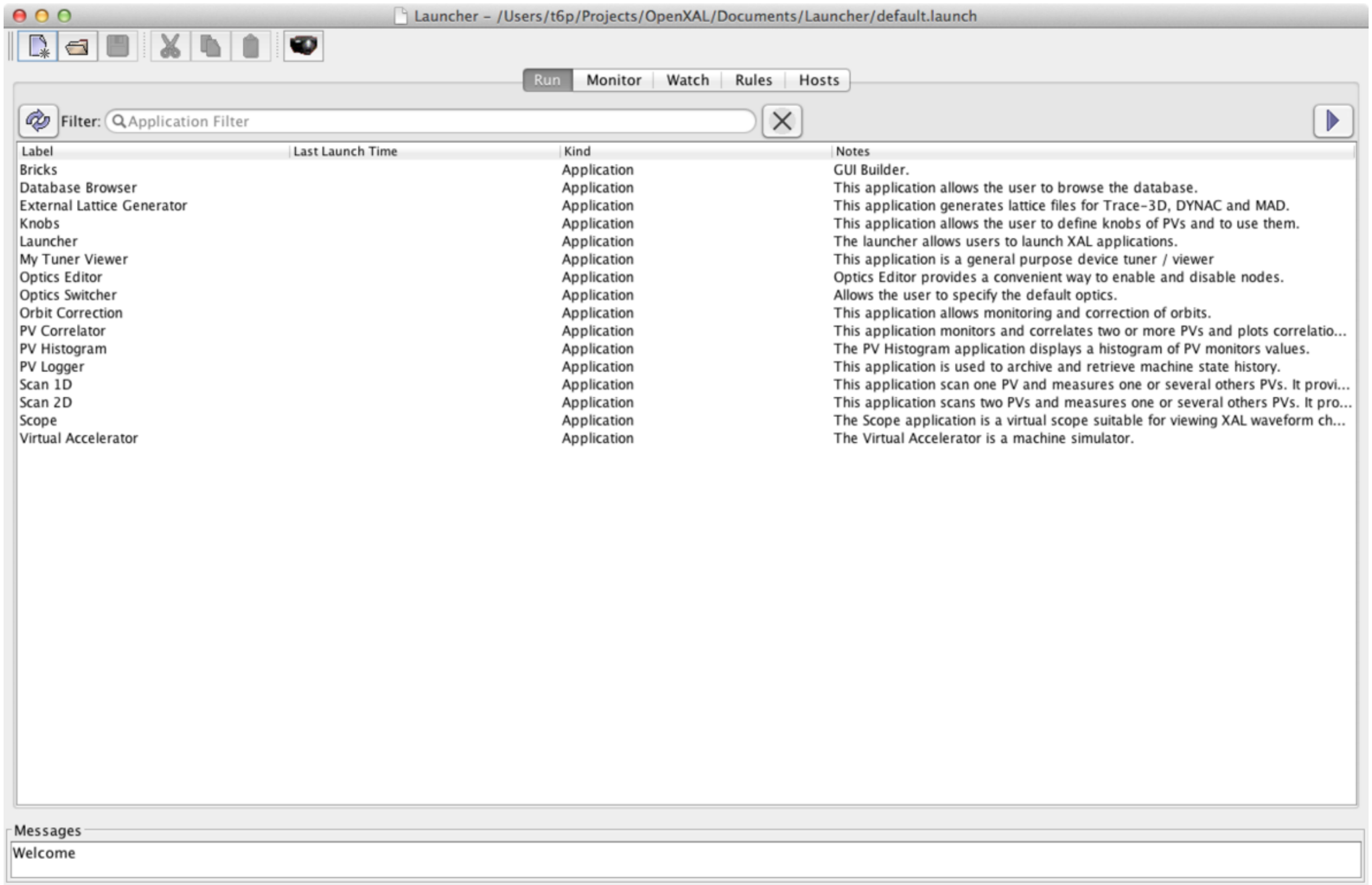
# Build Products
# Under build/products

| Subdirectory | Contents |
|---|---|
| apps | Java Applications |
| lib | Shared Library |
| scripts | JRuby and Jython scripts |
| services | Java Services |

# Running an Application
# Example - Run Launcher

```
> cd
> cd Projects/OpenXAL/openxal
> java –jar build/products/apps/launcher.jar
```

# Launcher
# Launches Applications and Scripts



# Configuration

- Channel Access configuration for locating EPICS servers
- CLASSPATH to Open XAL
  - JRuby and Jython scripts need to know the path to the Open XAL shared library
  - Java applications don't need this as it is already baked into the jars
- Documents Directory
- Default Accelerator

# Channel Access Support

- Provides configuration for Java Channel Access (JCA) both native (JNI) and pure Java (CAJ)
  - JNI requires EPICS client libraries to be installed
  - CAJ is self contained
  - We will use CAJ for this course

# Install Channel Access Support

Copy the JCALibrary.properties file to: ~/.JCALibrary/JCALibrary.properties

# JCALibrary.properties

```
# define the location of the epics shared libraries and caRepeater executable for Mac OS X
gov.aps.jca.jni.epics.darwin-x86.library.path = /Library/EPICS/Base/lib/darwin-x86
gov.aps.jca.jni.epics.darwin-x86.caRepeater.path = /Library/EPICS/Base/bin/darwin-x86

# define the location of the epics shared libraries and caRepeater executable for Linux
gov.aps.jca.jni.epics.linux-x86.library.path= /usr/share/epics/baseR3.14.4/lib/linux-x86
gov.aps.jca.jni.epics.linux-x86.caRepeater.path= /usr/share/epics/baseR3.14.4/bin/linux-x86

# define the location of the epics shared libraries and caRepeater executable for Windows
gov.aps.jca.jni.epics.win32-x86.library.path= c:/epics/baseR3.14.4/bin/win32-x86
gov.aps.jca.jni.epics.win32-x86.caRepeater.path= c:/epics/baseR3.14.4/bin/win32-x86
```

```
# define default values for both JNI_THREAD_SAFE and JNI_SINGLE_THREADED contexts.
gov.aps.jca.jni.JNIContext.preemptive_callback = true

# Channel Access address list for JNI context
# test network:
gov.aps.jca.jni.JNIContext.addr_list = localhost

gov.aps.jca.jni.JNIContext.auto_addr_list = false
gov.aps.jca.jni.JNIContext.connection_timeout = 30.0
gov.aps.jca.jni.JNIContext.beacon_period = 15.0
gov.aps.jca.jni.JNIContext.repeater_port = 5065
gov.aps.jca.jni.JNIContext.server_port = 5064
gov.aps.jca.jni.JNIContext.max_array_bytes = 5000000

# define default values only for JNI_SINGLE_THREADED context
gov.aps.jca.jni.SingleThreadedContext.event_dispatcher = gov.aps.jca.event.DirectEventDispatcher

# define default values only for JNI_THREAD_SAFE context
gov.aps.jca.jni.ThreadSafeContext.event_dispatcher = gov.aps.jca.event.QueuedEventDispatcher
gov.aps.jca.jni.ThreadSafeContext.priority = 5
```

# JCALibrary.properties
# JCA/JNI and JCA/CAJ Config

```
# define default values for QueuedEventDispatcher components
gov.aps.jca.event.QueuedEventDispatcher = 5

# Channel Access address list for CAJ context
com.cosylab.epics.caj.CAJContext.addr_list = localhost

com.cosylab.epics.caj.CAJContext.max_array_bytes = 5000000
com.cosylab.epics.caj.CAJContext.auto_addr_list = false
com.cosylab.epics.caj.impl.reactor.lf.LeaderFollowersThreadPool.thread_pool_size = 30

# generic address list support (only seems to apply to CAJ)
#gov.aps.jca.Context.auto_addr_list = false
gov.aps.jca.Context.addr_list = localhost
```

# Install CLASSPATH Support

1. Copy the script snippet into your ~/.bashrc file

# CLASSPATH for Open XAL

```
# setup the OPENXAL and related environment variables
if [ -z "$OPENXAL_SETUP" ] ; then
    # Open XAL HOME
    export OPENXAL_HOME=${HOME}/Projects/OpenXAL/openxal

    # setup CLASSPATH
    OPENXAL_LIBROOT=${OPENXAL_HOME}/build/products/lib
    OPENXAL_CLASSPATH=${OPENXAL_LIBROOT}/xal-shared.jar
    classpath=${XAL_CLASSPATH}:${OPENXAL_HOME}/build/products/lib/xal-shared.jar
    if [ -z "$CLASSPATH" ] ; then
        export CLASSPATH=${classpath}
    else
        export CLASSPATH=${CLASSPATH}:${classpath}
    fi


    # indicate that this file has been executed
    export OPENXAL_SETUP="true"
fi
```

# Configure Documents Directory

1. Run the Launcher
2. Select File -> Open...
3. Press “Default Folder”
4. Select “Yes” to specify a default folder.
5. Navigate to where you want it to reside (e.g. your Project Directory)
6. Create a new folder (e.g. named Documents)
7. Select the Documents directory
8. Select the “Make Default” button

# Configure Default Accelerator

1. Copy the accelerator optics files to a location of your choosing (e.g. the “optics” directory under the Project Directory.
2. Launch the Optics Switcher application
3. Press the “Browse” button
4. Navigate to the “main.xal” file in the optics directory
5. Press “Make Default”