

# Populus Guide for Developers

Lars Roe

July 11, 2014

# Contents

<b>I</b>	<b>Overview</b>	<b>4</b>
<b>II</b>	<b>Java Source Code</b>	<b>4</b>
<b>1</b>	<b>Models</b>	<b>4</b>
1.1	Files . . . . .	4
1.1.1	Model . . . . .	4
1.1.2	ModelPacket . . . . .	4
1.1.3	ModelPanel . . . . .	4
1.1.4	ModelOutputPanel . . . . .	4
1.2	Events . . . . .	5
1.3	Adding a Model to the Menus . . . . .	5
1.4	Basic Model . . . . .	5
1.5	Common Variants . . . . .	5
<b>2</b>	<b>Main</b>	<b>6</b>
<b>3</b>	<b>Help</b>	<b>6</b>
<b>4</b>	<b>Preferences</b>	<b>6</b>
4.1	PreferencesFile . . . . .	6
<b>5</b>	<b>GUI Widgets</b>	<b>6</b>
5.0.1	ParameterField . . . . .	6
<b>6</b>	<b>Javadoc</b>	<b>6</b>
<b>III</b>	<b>Installer</b>	<b>6</b>
<b>IV</b>	<b>Web Page</b>	<b>7</b>
<b>V</b>	<b>Test and Verification</b>	<b>7</b>

<b>7</b>	<b>Release Checklist</b>	<b>7</b>
<b>8</b>	<b>Platform</b>	<b>7</b>
8.1	Linux . . . . .	8
8.2	MacOS . . . . .	8
<b>VI</b>	<b>Setting Up New Development Machine</b>	<b>8</b>
<b>9</b>	<b>Development Software</b>	<b>8</b>
9.1	Java Development Kit . . . . .	8
9.2	Git . . . . .	8
9.3	Eclipse . . . . .	8
9.4	TeX . . . . .	9
9.5	Photoshop . . . . .	9
<b>10</b>	<b>Setup</b>	<b>9</b>
10.1	Files . . . . .	9
10.2	Eclipse . . . . .	9
10.2.1	JClass . . . . .	9
10.3	Other Software . . . . .	9

# Part I

## Overview

# Part II

## Java Source Code

### 1 Models

#### 1.1 Files

By convention, each end model (not meant to be inherited from) should be in the package `edu.umn.ecology.populus.model.ModelName`.

##### 1.1.1 Model

A `Model` holds together the basic parts of a model.

##### 1.1.2 ModelPacket

A `ModelPacket` is a simple wrapper for a model so we can refer to one class at a time, and used in making the menus. The menus are created in `initializeMenuPackets()`, and this is manually updated to add or remove models.

##### 1.1.3 ModelPanel

The `ModelPanel` (input window) base files are in `edu.umn.ecology.populus.edwin` (short for editor window, from the Pascal DOS program's naming conventions).

`registerChildren()` looks at all of the components, and sets event listeners where appropriate. Read `Events` for more information.

##### 1.1.4 ModelOutputPanel

The `OutputPanel` (output window) base files are in `edu.umn.ecology.populus.resultwindow`

## 1.2 Events

When changes in the input panel occur, events - or messages - are sent to the output. The `ModelPanel` will call `fireModelPanelEvent()` whenever a change occurs, with a constant such as `CHANGE_PLOT`. If this warrants a new output, `ModelPanel` will be queried for, in the case of Basic Plot, new plot info. Do not assume that `getPlotInfo()` will be called whenever you call `fireModelPanelEvent`. For example, if changing the value of a radio button should disable another parameter, that should be done separately from `getPlotInfo()`. See the method `modelPanelChanged()` to see which events are ignored and which events create a new plot.

Inherited models should not have to worry about when to show the output screen. `registerChildren()` is called after the initialize of the front panel, and this routine looks at all of the components and adds listeners to the ones that should through events. There is a setting in the User Settings so that users can change when to automatically update the output and making decisions on a model-by-model basis will not work with this.

## 1.3 Adding a Model to the Menus

To add a model to the menu, add a `ModelPacket` in Preferences.

Lars - it appears this is done in two different spots, hard-coded. I wanted the ability to be able to dynamically determine files. What is `SelectModelDialog??`

## 1.4 Basic Model

Most models will derive from `BasicPlotModel`, in the plot directory.

## 1.5 Common Variants

Most models extend from `edu.umn.ecology.populus.plot.BasicPlotModel`, which does basic graphing. But you don't have to do this. See `Woozleology` for an example of one that does not extend from this.

## 2 Main

`main` is found in `edu.umn.ecology.populus.core.PopRun`. The `DesktopWindow` is the primary GUI background to the application.

## 3 Help

When we click the Help button on a model or the main `DesktopWindow`, we LARS - TODO The help system was changed dramatically in 5.5, by modifying the local help file to use the language specified by the user's configuration.

## 4 Preferences

### 4.1 PreferencesFile

The file for keeping state is stored as `userpref.po` in the user's home directory (as of Populus 5.4). It is loaded during initialization. By default, it is in the user's home directory – not in Populus's – because we aren't guaranteed write permission for all systems. This can be overridden by the startup command - see `README.config`.

## 5 GUI Widgets

### 5.0.1 ParameterField

The `ParameterField` was originally concocted as a spinner. But then we added the variable name, and variable information to the parameter.

## 6 Javadoc

I wish the code were better documented. But you can still use `javadoc` to generate documentation for the files.

## Part III

# Installer

Populus Splash Screen. We have a file called Populus\*.\*.psd which is a photoshop file describing the title screen. For a new release, we probably want a change in version number, so make a new .psd file with the new version, and then export it to gif format (calling it PopulusSplashScreen.gif) and replace the one in edu/.../core/ with the new gif.

## Part IV

# Web Page

This should all be handled by the UMN Web team. They now use Drupal (a content management system). For 5.5, I just gave them a new JAR file.

## Part V

# Test and Verification

## 7 Release Checklist

Check that help works on all different platforms.

current issues for troubleshooting help file: on mac os x: the populus parameter field arrows are dim screen resolution can cause windows to be smaller than they should be - just resize on pc:

## 8 Platform

It's a good idea to test on different platforms.

## 8.1 Linux

LiveCD SLAX can boot up Linux on an otherwise Windows computer. There are other options now too.

## 8.2 MacOS

You really just need a Mac for this. The UofM computer team have testers to help with this.

# Part VI

# Setting Up New Development Machine

## 9 Development Software

### 9.1 Java Development Kit

Download and install Java SE (Standard Edition) from [oracle.com](http://oracle.com). I like to use JDK 1.7 (aka 7) and 1.6 (aka 6) for backwards compatibility.

### 9.2 Git

Git on the command line should be default for OS X and Linux. You don't have to install more, but <http://git-scm.com/downloads/guis> has some nice GUIs. I used GitX-dev (rowanj) for OS X, which seems good.

For Windows, I like git for Windows: <http://msysgit.github.io/>, which includes the command line tools and GUI.

### 9.3 Eclipse

Download the Eclipse Standard from <https://www.eclipse.org/downloads/>. (You can actually install any version that has Java support.) The "installer" is just a zip file that you extract somewhere. You'll run it by running the executable in there.



## 9.4 TeX

We don't use LaTeX for any externally-facing file, but it is used for modifying this document. I use MiKTeX for Windows. MacTeX and livetex are recommended for OS X and Linux respectively.

## 9.5 Photoshop

Use Photoshop to make the pictures for, say, the Web page. There are saved .psd files around that contain the source image to work from with its Layers.

# 10 Setup

## 10.1 Files

Everything used is in Git now. To get the source, run: `gitclone .git pop` where pop is the directory you want the source to go into.

## 10.2 Eclipse

Now run Eclipse. For the workspace, choose the `pop` directory, or whatever you used to extract the files from git.

Go to File  $\gg$  New  $\gg$  Java Project

For the project name, choose `PopulusE`. Eclipse should know that this is an existing project, and don't set any more options.

Click the green run button. You want to run this as a Java Application. The main class is `PopRun` (`edu.umn.ecology.populus.core.PopRun`).

### 10.2.1 JClass

JClass includes the chart software for Java that we use. The Manifest file in the JAR file they included has some bogus `dependson` lines that give warnings when you try to run. I manually deleted these, and just keep this new version around. JClass keeps switching companies. We have an old version of their product, and I don't have any reason for upgrading.

## 10.3 Other Software