Developer Manual

Much of the following was directly taken from the checker framework's manual and adapted slightly for this project. For more information about the checker framework in general, read the manual at
http://types.cs.washington.edu/checker-framework/current/checker-framework-manual.pdf

Our development website can be found at
http://students.washington.edu/gyhughes/cse403/TeamWebsite/

Our repository can be found at
https://github.com/gyhughes/checker-framework

Obtaining Source Code
In order to work with the checker framework, three repositories need to be checked out. In order to avoid setup confusion, we require that anyone obtaining the source code put it in ~/jsr308.

The following commands achieve this:

```
export JSR308=$HOME/jsr308
mkdir -p $JSR308
cd $JSR308
hg clone https://bitbucket.org/typetools/jsr308-langtools jsr308-langtools
git clone https://github.com/gyhughes/checker-framework.git
git clone https://github.com/typetools/annotation-tools.git annotation-tools
```

This requires both git and mercurial to be installed.

Directory Structure
The directory structure of the checker framework is very large, but we are only adding a very small part to it in the form of a new checker. The new source code we are adding can be found in the directory checker-framework/checker/src in the package org.checkerframework.checker.index

The checker's tests can be found in the directory checker-framework/checker/tests. Our tests are located in the folders index-general, index-hierarchy, index-introduction, and index-type

The checker framework's default checkers can be found in the packages adjacent to our index checker. Specifically, in the packages org.checkerframework.checker.<checkerNameHere>. We recommend looking at the classic regex checker as a simple, but nontrivial example of a checker's structure.

If you would like to dive into how the checker itself is working, the checker framework source files can be found in the directory checker-framework/framework/src in the package org.checkerframework

First Time Build Instructions
For first time setup, all three repositories need to be built.

First: build langtools

The checker framework manual tells us this:

“

1. Set the JAVA_HOME environment variable to the location of your JDK 7 or 8 installation (not the JRE installation, and not JDK 6 or earlier). This needs to be an Oracle JDK. (The JAVA_HOME environment variable might already be set, because it is needed for Ant to work.) In the bash shell, the following command *sometimes* works (it might not because java might be the version in the JDK or in the JRE):

   ```
   export JAVA_HOME=${JAVA_HOME:-$(dirname $(dirname $(dirname $(readlink -f
   $(/usr/bin/which java)))))}
   ```

2. Compile the Type Annotations tools:

   ```
   cd $JSR308/jsr308-langtools/make
   ant clean-and-build-all-tools
   ```

3. Add the jsr308-langtools/dist/bin directory to the front of your PATH environment variable. Example command:

   ```
   export PATH=$JSR308/jsr308-langtools/dist/bin:${PATH}
   ```

“

Second: build annotation-tools

We only need to run the ant to build. No special path editing is needed. This is done with:

```
cd $JSR308/annotation-tools
ant
```

Third: build checker-framework

The following instructions have been adapted from the checker-framework manual:

1. Run ant to build the Checker Framework:

   ```
   cd $JSR308/checker-framework/checker
   ant
   ```

2. Once it is built, you may wish to put the Checker Framework’s javac even earlier in your PATH:

   ```
   export
   PATH=$JSR308/checker-framework/checker/bin:$JSR308/jsr308-langtools/dist/bin:${PATH}
   ```

   The Checker Framework’s javac ensures that all required libraries are on your classpath and boot classpath, but is otherwise identical to the Type Annotations compiler. Putting the Checker Framework’s javac earlier in your PATH will ensure that the Checker Framework’s version is used.

3. Set your PATH to use `javac` in the `bin-devel` directory:

   ```
   export
   PATH=$JSR308/checker-framework/checker/bin-devel:$JSR308/jsr308-langtools/dist/bin:${
   PATH}
   ```

Re-Build Instructions

The developer version of `javac` allows you to not have to rebuild the jar files after every code change, in turn allowing you to test your changes faster. Source files can be compiled using command `ant build` in the `checker` directory. Specifically:

```
cd $JSR308/checker-framework/checker
ant build
```

At this point, it should be possible for this to be automatically compiled by an IDE such as Eclipse. This currently has not been tested and not guaranteed to work for our project.

Test Instructions

Tests can be run by going into the checker folder of the checker framework and running ant all-tests. Specifically:

```
cd $JSR308/checker-framework/checker
ant all-tests
```

We currently do not know how to run specific subsets of tests.
To run the index checker, use the command:

javac -processor org.checkerframework.checker.index.IndexChecker <*JavaFileName*>.java

New Version Releases

We currently have no plan for allocating version numbers

Bug Tracking/Resolving

Our issue tracker can be found at https://github.com/gyhughes/checker-framework/issues.
Here, bugs and other issues can be added, tracked, and resolved.