Installing the Code

Detailed instructions for installing, configuring and testing the source code on OS X.

These instructions also work with the free demo version of the guide.

1. Create a folder called '**TIJ4-Solutions**' in your home folder.
2. Copy the zip file containing the code that you received, '**TIJ4-Solutions-code**', when you purchased the guide to the folder '**TIJ4-Solutions**'. Unzip the file using the Archive Utility. When you're done, you should see a sub-folder '**TIJ4-Solutions-Code'** and numerous subfolders, including subdirectories corresponding to the chapters in the solution guide.
3. Create a folder called '**jars**' in your recently created folder '**TIJ4-Solutions**'. Place the following files into this directory:
   javassist.jar (download here: http://sourceforge.net/project/showfiles.php?group_id=22866; you may need to search for it).
   swt.jar from the Eclipse SWT library (http://download.eclipse.org/eclipse/downloads/). Click on the most recent build number, then scroll down to "SWT Binary and Source" and select the file corresponding to your platform. Further details about finding the jar file are in *Thinking in Java, 4th Edition*, under the heading "Installing SWT."
   xom-1.1.jar, available from http://www.cafeconleche.org/XOM/.
4. Create a small shell script '**setenv.ksh**' using the pico editor in the folder '**TIJ4-Solutions**' to set the CLASSPATH environment variable . Start a terminal by pressing ⇧⌘U in Finder and selecting '**Terminal**'. Change to the directory '**TIJ4-Solutions**' by typing the instruction '**cd ~/TIJ-Solutions**' and start the pico editor by typing the instruction '**pico setenv.ksh**'. Cut-and-paste the code below into the pico editor and save the file by pressing ctrl-o directly followed by ↵. Leave the pico editor by pressing ctrl-x. You can close the Terminal window by pressing ⌘w.

```
#!/bin/ksh

if [[ $0 != '-bash' ]]
then
        echo "Execute this script using \". $0\""
fi

CLASSPATH=.:..
CLASSPATH=$CLASSPATH:~/TIJ4-Solutions/TIJ4-Solutions-code
CLASSPATH=$CLASSPATH:~/TIJ4-Solutions/jars/javaws.jar
CLASSPATH=$CLASSPATH:~/TIJ4-Solutions/jars/javassist.jar
CLASSPATH=$CLASSPATH:~/TIJ4-Solutions/jars/swt.jar
CLASSPATH=$CLASSPATH:~/TIJ4-Solutions/jars/xom-1.1.jar

export CLASSPATH
echo "CLASSPATH exported"
```

When necessary to set you CLASSPATH variable, e.g. before compiling you java files using javac or ant, execute the '**setenv.ksh**' script by typing '**. ./setenv.ksh**'.

5. Download the Ant 1.7 (or newer) build tool at http://ant.apache.org/. Start a terminal by pressing ⇧⌘U in Finder and selecting '**Terminal**'. Type the instruction '**sudo sh**' After entering the sudo instruction a password is requested, enter the password here when you logon on your mac. Your prompt changes now to something similar like "sh-3.2# " indicating you changed to the root user. Cut-and-past the instructions below into the Terminal to install Ant.

```
cd /usr/local
mv ~/Downloads/apache-ant-1.7.1 /usr/local
chown `who am i | awk '{ print $1 }'` apache-ant-1.7.1/
ln -s apache-ant-1.7.1 ant

cd /etc
grep -q 'export ANT_HOME' bashrc
if [[ $? -ne 0 ]]
then
chmod u+w bashrc
echo '# Following two lines are added by TIJ-Solutions' >> bashrc
echo '# install for OS X' >> bashrc
echo 'export ANT_HOME=/usr/local/ant' >> bashrc
echo 'export PATH=${PATH}:${ANT_HOME}/bin' >> bashrc
chmod u-w bashrc
fi
# Leave root user
exit
# Do also cut-and-paste this line
```

You can close the Terminal window by pressing ⌘w.

6. Test Ant by by opening new terminal by pressing ⌘N and typing '**ant**'. This should print "Buildfile: build.xml does not exist! Build failed" .

Note: **Ant** is required in order to compile the examples in the book. Once you successfully run '**ant build**' in the root directory of the TIJ4-Solutions, you can also compile each example individually (once you have the CLASSPATH set, as described in Step 4) using the **javac** command-line compiler. To compile a file called **MyProgram.java**, you type **javac MyProgram.java**.

7. Probably due to a bug in the build.xml in the TIJ4-Solutions root directory, you have to compile one java application before compiling all other examples using ant. Start a new terminal session and change to the "**TIJ4-Solutions**" directory by typing '**cd ~/TIJ4-Solutions**'. Now set the CLASSPATH variable as described in step 4. Change to the "**typeinfo**" directory by typing '**cd ~/TIJ4-Solutions/TIJ4-Solutions-code/typeinfo**'. Then compile the java file "**E12_CoffeeCount.java**" by typing '**javac E12_CoffeeCount.java**'. If you do not get any errors, the compilation has been successful and a "**E12_CoffeeCount.class**" file has been created.

8. You now can run ant build in the root directory, you can also move into individual chapters and type ant (to compile and execute the code in that chapter) or ant build (to compile the code only). Keep in mind that each time when you start a new terminal session you have to set the CLASSPATH variable as described in step 4, before you can use build or javac. You can compile all examples by executing the following cut-and-paste instructions in a terminal session.

```
cd ~/TIJ4-Solutions
. ./setenv.ksh
cd ~/TIJ4-Solutions/TIJ4-Solutions-code
ant build
# Also cut-and-paste this line
```

The compilation should report a "**BUILD SUCCESSFUL**". If it does not and resports something similar to "**... package generics.coffee does not exist   [javac] import generics.coffee.*; ...**" review and if necessary, redo the actions described in step 7.

9. This code is designed to work without an IDE, but it has also been tested with Eclipse (free at http://www.eclipse.org/); see the following section for instructions on how to use the code with Eclipse.
   If you want to use this code inside other IDEs you might need to make appropriate adjustments. Different IDEs have different requirements and you might find it's more trouble than it's worth right now; instead, you may want to begin with a more basic editor like JEdit (free at http://www.jedit.org/).

10. Note: The output for the programs has been verified for Java 6. Certain programs (primarily those that use hashing) can produce different output from one version to the next.