

# TaX Developer Cookbook

## **Author**

Thomas Alexnat  
feedback@alexnat.de  
Truchthari-Anger 28  
81829 München

*Draft — work in progress*  
September 12, 2009

## Disclaimer

Take this document as is. I want to share my experience with the current state of the art in computer programming. This is not a tutorial for the general Java or UNIX newbie. See this document more as my personal side notices. Although most of the screenshots resemble Mac OS X most of the mechanics should work fine on the Windows platform. Anyway there are no warranties made for all this information. Feedback is always welcome.

## Weapon of choice

While developing with the standard Windows operating system at work, I'm using a 2006 generation Mac Pro with dual dual-core Xeon configuration. I prefer to boot Mac OS X Leopard.

## Typography

- Important parts are printed in *italic* font.
- Programcode has this `fixed width` font. Shortened code is escaped with a colon :
- Dynamic attributes or variables are printed with this `CAPITALIZED` font.

## Copyright

This documentation is copyright 2008 by Thomas Alexnat (thomas@alexnat.de).  
All rights reserved;

*There are 10 types of people in the world:  
Those who understand binary and those who don't.*  
— UNKNOWN AUTHOR

# Contents

<b>1</b>	<b>Developer tips for Mac OS X</b>	<b>1</b>
1.1	Shell profile . . . . .	1
1.2	How to remove <code>.DS_Store</code> files . . . . .	1
1.3	English German Dictionary . . . . .	2
1.4	Power Settings . . . . .	2
1.5	Font Antialiasing and Subpixel Rendering . . . . .	2
1.6	Using Java SE 6 on Mac OS X . . . . .	3
1.7	Screensaver on the desktop . . . . .	3
1.8	Remove glass effect from dock . . . . .	3
1.9	Add spacers to the Leopard Dock . . . . .	3
1.10	Kernel Extensions . . . . .	4
1.11	L <sup>A</sup> T <sub>E</sub> X . . . . .	4
1.12	Subversion . . . . .	4
	1.12.1 Subclipse for Eclipse . . . . .	4
<b>2</b>	<b>Java</b>	<b>5</b>
2.1	Eclipse . . . . .	5
2.2	Maven . . . . .	5
	2.2.1 Configuration . . . . .	5
	2.2.2 Installing the Maven plugin for Eclipse . . . . .	6
	2.2.2.1 Download automatically Javadocs and sourcecode . . . . .	6
	2.2.3 Create artifacts . . . . .	7
	2.2.4 Goals . . . . .	7
	2.2.5 Maven Dependencies . . . . .	8
	2.2.6 Archetype for Java Server Faces (JSF) . . . . .	8
	2.2.7 Generating documentation . . . . .	8

<b>3</b>	<b>Continuous Integration</b>	<b>10</b>
3.1	Hudson . . . . .	10
<b>4</b>	<b>Spring Framework</b>	<b>11</b>
4.1	Eclipse integration . . . . .	11
4.2	Maven . . . . .	12
4.3	Hands on Spring in Eclipse . . . . .	12
<b>5</b>	<b>iPhone Webapp development</b>	<b>19</b>
5.1	Building full screen WebApps . . . . .	19
5.1.1	Navigation Elements or fixed Divs . . . . .	19
5.1.2	Detecting orientation changes . . . . .	20
5.1.3	Detecting gestures and touches . . . . .	20
<b>6</b>	<b>Application development for iPhone</b>	<b>21</b>
6.1	Troubleshooting Info.plist . . . . .	21
6.2	Mapkit . . . . .	21
6.3	Objective-C . . . . .	22
6.3.1	Headerfiles . . . . .	22
6.3.2	Singletons . . . . .	22
6.3.3	Properties . . . . .	22
6.3.4	IBOutlet . . . . .	22
6.3.5	Forward Declaration in header files . . . . .	23
6.3.6	Trimming Strings like Java trim() . . . . .	23
6.4	Core Data . . . . .	23
6.5	UIKit . . . . .	23
6.5.1	Abort the current event cycle to get direct UI drawing . . . . .	23
6.5.2	TableViewCell . . . . .	24
6.5.3	Loading NIBs withing the code without the init . . . . .	24
6.5.4	Activity Icon in Navigation Bar . . . . .	24
6.6	Core Location . . . . .	25
6.7	Performance . . . . .	25
6.7.1	Network Latency on Cell Networks . . . . .	25
6.7.2	Resources . . . . .	25

<b>Appendix</b>	<b>26</b>
<b>A Version History</b>	<b>26</b>
Index . . . . .	26

# Chapter 1

## Developer tips for Mac OS X

### 1.1 Shell profile

The `.profile` file is stored in my home directory in `/Users/tax`. I use this profile for my common tasks.

```
1 alias ls="LS -AG"
2 alias ll="LS -ALG"
3 export JAVA_HOME=/System/Library/Frameworks/JavaVM.framework/Home
4 export ANT_HOME=/Applications/java/ant
5 export MAVEN_HOME=/Applications/java/maven
6 export MYSQL_BIN=/usr/local/mysql/bin
7 export SUBVERSION_HOME=/usr/local/bin
8 export PATH=$ANT_HOME:$MYSQL_BIN:$MAVEN_HOME/bin:$PATH
```

Listing 1.1: My shell profile I use on Mac OS X Leopard

### 1.2 How to remove `.DS_Store` files

Ever want to remove all those hidden `.DS_Store` files under some directory? You can do this:

```
1 find <directory> -name .DS_Store -exec rm -f '{}' ';' ;'
```

Or a faster version (thanks to Sean Kelly - the version above does show you how to do arbitrary things to the files though):

```
1 find <directory> -name .DS_Store -delete
```

Just replace `directory` with the directory you want to clean up - every sub-directory will be visited and cleaned up as well.

## 1.3 English German Dictionary

```
1 http://www.tekl.de/deutsch/BeoLingus_Deutsch-Englisch.html
```

## 1.4 Power Settings

With the `pmset` command you can specify the time when a harddisc should spin down when it was idle for some minutes. This is very useful when you have more harddiscs in your system. Due to the fact, that my system has four drives, it is very annoying to hear all that power up and power down all ten minutes which is the OS X default value..

`pmset -g disk` will display settings for all power sources, including those not currently in use.

`pmset disksleep` sets the disk spindown timer in minutes. 0 will disable the countdown. The default value is 10 minutes, but this timerange is too short for my system: I have four harddrives in my Mac Pro and all few minutes a harddrive has to spin up. So I've set the value to 20 minutes and the system runs now more smoothly.

```
1 AC Power:
2 sleep      0
3 displaysleep 0
4 hibernatfile /var/vm/sleepimage
5 ttyskeepawake 1
6 autorestart 0
7 hibernatemode 0
8 powerbutton 1
9 womp      1
10 disksleep 10
```

## 1.5 Font Antialiasing and Subpixel Rendering

OS X Snow Leopard has automatic detection which type of display is connected to the Mac. Unfortunately this seems only to work with native Apple displays and not external monitors on my Mac Pro with DVI input.

Values can be 0 - 2. The 1 works best for my display.

```
1 defaults -currentHost write -globalDomain AppleFontSmoothing -int 2
```

## 1.6 Using Java SE 6 on Mac OS X

Please note that the Java SDK version 6 is only available for Intel Macs with Mac OS X Leopard 10.5 and comes with the 64bit edition. Due to the nature of OS X you can only use 64bit libraries when running 64bit programs. Unfortunately the graphics component library of the Eclipse foundation called *SWT* is built on the 32bit Carbon framework. So currently you cannot use 64bit JVM programs with 32bit carbon parts. In other words: Eclipse wouldn't start in 64bit Java SE 6 mode – but you can use 64bit java programs like Maven to run in Eclipse.

So the current SDK is pointing to version 1.6 of Java, stored in `/System/Library/Frameworks/JavaVM.framework/Versions`

## 1.7 Screensaver on the desktop

```
1 /System/Library/Frameworks/ScreenSaver.framework/Resources/  
   ScreenSaverEngine.app/Contents/MacOS/ScreenSaverEngine -background &
```

## 1.8 Remove glass effect from dock

```
1 defaults write com.apple.dock no-glass -boolean YES
```

## 1.9 Add spacers to the Leopard Dock

If you like to have an organized Dock in OSX, here is a quick hack that will let you organize the applications that live on your dock by placing spacers between them. Here we go.

1. Open Terminal and type (or copy and paste) :

```
1 defaults write com.apple.dock persistent-apps -array-add '{"TILE-TYPE":"' SPACER-TILE";}'
```

2. While still in Terminal we will restart the Dock by typing:

```
1 killall Dock
```

Once the Dock has restarted, you will see a space between some of your icons. You can reposition the space, by clicking on it and dragging it to a new location. You can also delete it by dragging it off your dock. You can even have many spacers by typing in the above commands as often as you like!

## 1.10 Kernel Extensions

Nicht alle installierten kexts sind auch aktiviert. Wenn Sie wissen möchten, welche kexts von Drittanbietern geladen sind, geben Sie den folgenden Befehl ins Terminal ein:

```
1 kextfind -case-insensitive -loaded -not -bundle-id -substring 'COM.APPLE'
```

## 1.11 L<sup>A</sup>T<sub>E</sub>X

Download at: <http://www.tug.org/mactex>

## 1.12 Subversion

The *Coding Monkey* Martin Ott provides the current version 1.5.5 of Subversion for OS X. The package installs Subversion in `/usr/local/` and requires Mac OS X 10.3.9 or later.

<http://homepage.mac.com/martinott/>

Then add Subversion to your path:

```
1 export SUBVERSION_HOME=/usr/local/bin
2 export PATH=$SUBVERSION_HOME:$PATH
```

Add a repository in your local Library:

```
1 svnadmin create ~/Library/Subversion
```

### 1.12.1 Subclipse for Eclipse

Installation Instructions at

<http://subclipse.tigris.org/servlets/ProjectProcess?pageID=p4wYuA>

The Eclipse update site URL: [http://subclipse.tigris.org/update\\_1.4.x](http://subclipse.tigris.org/update_1.4.x)

You may import a project:

```
1 svn import myproject file:///Volumes/data/projects/subversionrepository/
   myproject
```

# Chapter 2

## Java

### 2.1 Eclipse

Currently I'm using Eclipse Web Tools Platform (WTP) to develop my applications. This is a smart set of plugins when developing on the Java EE platform.

### 2.2 Maven

Maven is the current state of the art build system to resolve dependencies and help the programmer to develop faster. You can find it on <http://maven.apache.org>.

To get started there is a nice tutorial on the Maven site:

<http://maven.apache.org/guides/getting-started/index.html>

Please note that Mac OS X Leopard comes with Maven already preinstalled. However this version is 2.0.6 and is installed in `/usr/bin/mvn`. I prefer to use the latest version and installed it myself. So add it to the system path:

```
1 export MAVEN_HOME=/Applications/java/maven
2 export PATH=$MAVEN_HOME/bin:$PATH
```

#### 2.2.1 Configuration

The user specific settings are stored in the the Maven `.m2` directory located in the user home. On my system the Maven directory is located at `/Users/tax/.m2/`. Here you can find:

`settings.xml` This optional file contains proxy settings, locations to different Maven repositories, etc.

`repository/` contains all downloaded artifacts Maven needs to run. Sometimes its a good idea to delete your *local* repository. Maven should download again all the needed dependent artifacts – although this may take some time.

## 2.2.2 Installing the Maven plugin for Eclipse

Visit the Codehaus page on the Maven homepage:

<http://m2eclipse.codehaus.org/>

You can install the Maven Integration for Eclipse by using the following update site from within Eclipse:

<http://m2eclipse.sonatype.org/update/>

Or as a Development version:

<http://m2eclipse.sonatype.org/update-dev/>

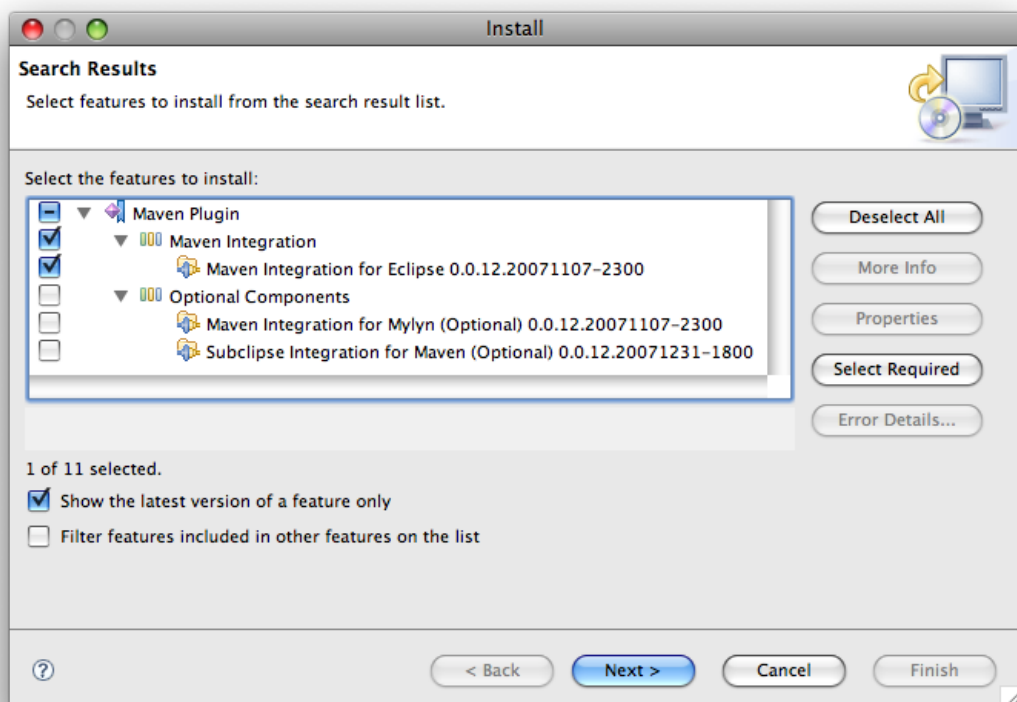


Figure 2.1: Installing the Maven Plugin in Eclipse. Support for Subversion was not needed here and is disabled.

### 2.2.2.1 Download automatically Javadocs and sourcecode

You can turn on the automatic downloading of artifact Javadocs and artifact sources of the referenced dependencies. Select `Eclipse>Preferences...` and turn the switches on.

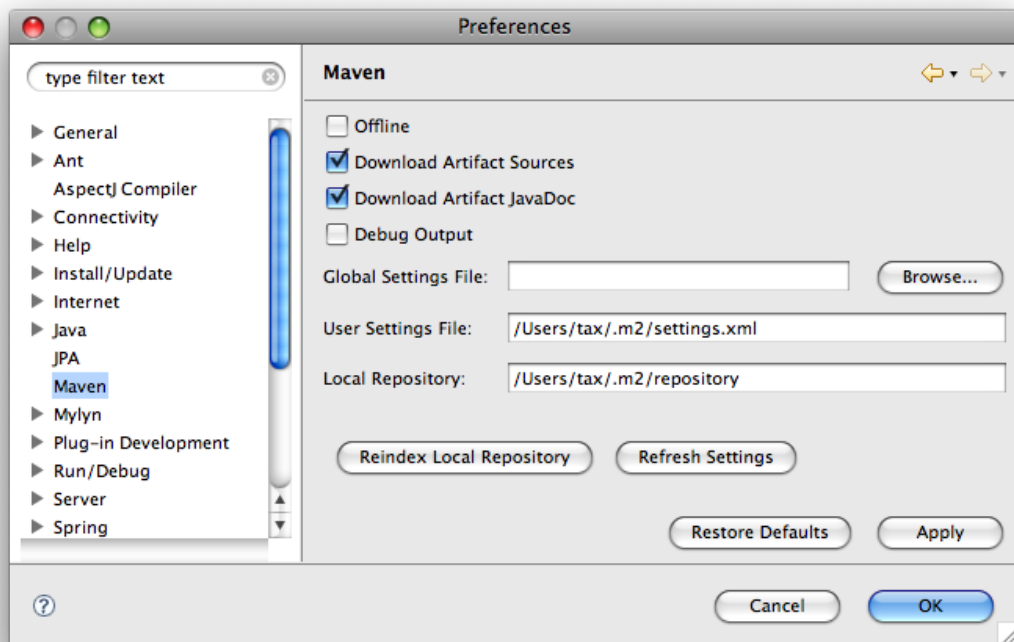


Figure 2.2: Load automatically artifact JavaDocs and artifact sources by Maven

### 2.2.3 Create artifacts

```
1 mvn archetype:create -DgroupId=com.mycompany.app -DartifactId=my-app
```

### 2.2.4 Goals

The classic ant targets are called *Goals* in Maven. Here is a list of the most common goals:

`mvn compile` will compile all the available source files corresponding to the defined `pom.xml` object model. Each command will cause Maven to resolve the dependencies.

`mvn package` Will generate the artifact. Most of the time this is a `.jar`, `.war` or `.ear` file.

`mvn clean` Will delete all the files in the `target/` directory

`mvn build` Compiles the application and generates an artifact like a `.jar` file

`mvn install` Will store the built artifact in your local directory. Usually this is `~/.m2/repository`.

```
1 <!-- ===== -->
2 <!-- Build Settings -->
3 <!-- ===== -->
4 <build>
5     <plugins>
6         <plugin>
7             <groupId>org.mortbay.jetty</groupId>
8             <artifactId>maven-jetty-plugin</artifactId>
9         </plugin>
10    </plugins>
11 </build>
```

Listing 2.1: Add the Jetty-Plugin to your `pom.xml` and start your webapp with the `jetty:run` goal.

`mvn site` is a very powerful command to generate documentation about your code. There are plenty of report plugins just to maintain JavaDocs, test reports and code coverage. In my opinion this adds an extra value to this great tool.

`mvn jetty:run` Starts the artifact in a small servlet container. Very useful to develop small web applications. Sometimes you need to specify a proxy for Jetty. Just use the normal Java system properties to configure the proxy and port: `java -Dhttp.proxyHost=proxyhost -Dhttp.proxyPort=portNumber]`

`mvn dependency:resolve` Displays the dependent Artifacts.

## 2.2.5 Maven Dependencies

The Maven Dependencymanagement Section is for parent POMs and the Dependencies section is for normal module poms dependencies.

## 2.2.6 Archetype for Java Server Faces (JSF)

To create a small JSF project you only need this smart archetype. It can be found on the webpage [http://wiki.rodcoffin.com/index.php?title=JSF\\_Maven\\_Archetype](http://wiki.rodcoffin.com/index.php?title=JSF_Maven_Archetype).

## 2.2.7 Generating documentation

Try out the command `mvn site` and see a generated documentation site in your `target/site/` directory.

```

1 mvn archetype:create
2   -DgroupId=com.my-company.my-project
3   -DartifactId=my-project-web
4   -DpackageName=com.company.project.web
5   -DarchetypeGroupId=com.rfc.maven.archetypes
6   -DarchetypeArtifactId=jsf-maven-archetype
7   -DremoteRepositories=http://maven.rodcoffin.com/repo

```

Listing 2.2: Maven goal `archetype:create` to specify in a commandline a new JSF project

The screenshot shows a web browser window displaying the Maven-generated documentation for the 'springtest' project. The page is titled 'springtest' and includes a navigation menu on the left with options like 'Project Information', 'About', 'Continuous Integration', 'Dependencies', 'Issue Tracking', 'Mailing Lists', 'Project License', 'Project Summary', 'Project Team', and 'Source Repository'. The main content area is divided into sections for 'Project Dependencies', 'Project Transitive Dependencies', and 'Project Dependency Graph'.

**Project Dependencies**

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
org.springframework	spring-context	2.5.1	-	jar	

**Project Transitive Dependencies**

The following is a list of transitive dependencies for this project. Transitive dependencies are the dependencies of the project dependencies.

The following is a list of compile dependencies for this project. These dependencies are required to compile and run the application:

GroupId	ArtifactId	Version	Classifier	Type	Optional
aopalliance	aopalliance	1.0	-	jar	
commons-logging	commons-logging	1.1	-	jar	
org.springframework	spring-beans	2.5.1	-	jar	
org.springframework	spring-core	2.5.1	-	jar	

**Project Dependency Graph**

**Dependency Tree**

- de.tax:springtest1:jar
  - org.springframework:spring-context:jar
    - aopalliance:aopalliance:jar
    - org.springframework:spring-beans:jar
      - commons-logging:commons-logging:jar
  - org.springframework:spring-core:jar

Figure 2.3: Generated documentation by Maven

# Chapter 3

## Continuous Integration

### 3.1 Hudson

My best choice for Continuous Integration (CI) is currently Hudson. Just a few weeks I chose Apache Continuum, but Hudson seems to be the market best practice.

<https://hudson.dev.java.net/>

```
java -jar hudson.war
```

```
--httpPort=$HTTP_PORT
```

```
--javaHome=$JAVA_HOME
```

```
/Users/tax/.hudson
```

```
JAVAHOME for Hudson configuration. /System/Library/Frameworks/JavaVM.framework/Home
```

Basic maven project files are stored in `.hudson/jobs`

# Chapter 4

## Spring Framework

### 4.1 Eclipse integration

The Spring IDE integration supports full refactoring. This means that you can rename your class names and the configuration files are automatically adapted. This saves time and is very convenient.

Download the Spring IDE from <http://springide.org>. Update site is <http://dist.springframework.org/release/IDE>.

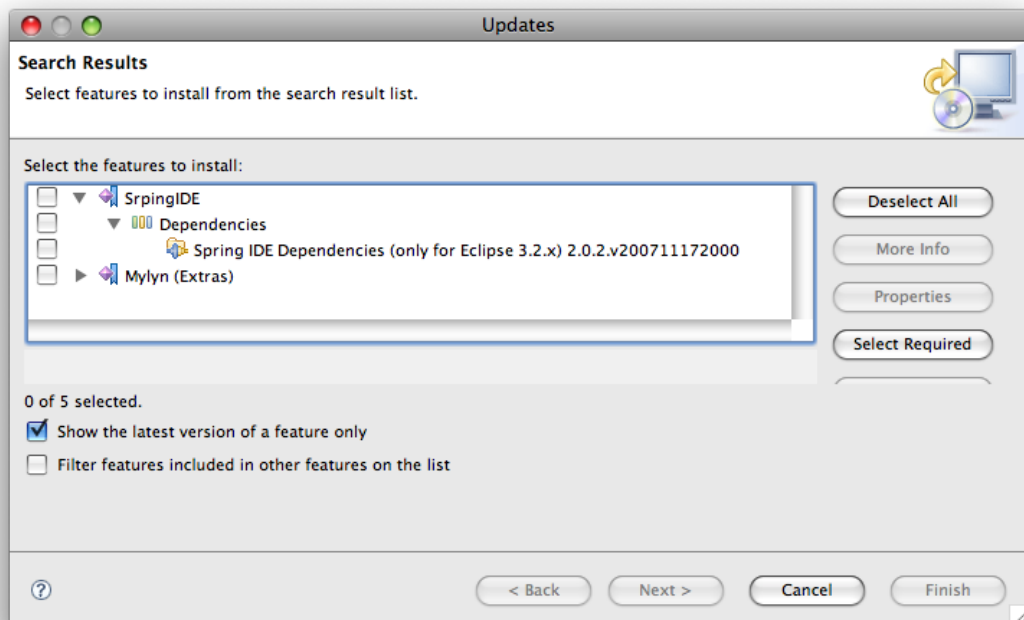


Figure 4.1: Please note, that the Spring dependencies for Eclipse 3.2 have to be disabled in Eclipse 3.3. (The *Srping* typo is my fault.)

## 4.2 Maven

You can find the Spring-Core 2.5.1 at the official Maven repository:

<http://repo1.maven.org/maven2/org/springframework/spring-core>

**Hint:** Please do not get confused with the old style of Maven group IDs. They can be found in the root directory of the Maven2 repository but are no longer maintained.

Ben Hale from Springsource writes:

For any final release (Spring 2.5, Spring Web Flow 2.0, etc.) the Maven artifacts for that release will be uploaded to the Maven Central repository (<http://repo1.maven.org/maven2>). Using this repository requires no effort on your part as Maven will automatically look for artifacts there.

The artifacts in this repository do follow expected repository behaviors and will not (and cannot) be removed.

<http://blog.springsource.com/main/2007/09/18/maven-artifacts-2>

## 4.3 Hands on Spring in Eclipse

In this section you will see how to start a Spring-Core Application in Eclipse with the Maven and Spring plugins. After the `pom.xml` is generated it should be modified to reflect the current version 2.5.1 of Spring.

Afterwards set the project properties for the Eclipse builders right. Otherwise they could interfere with each other.

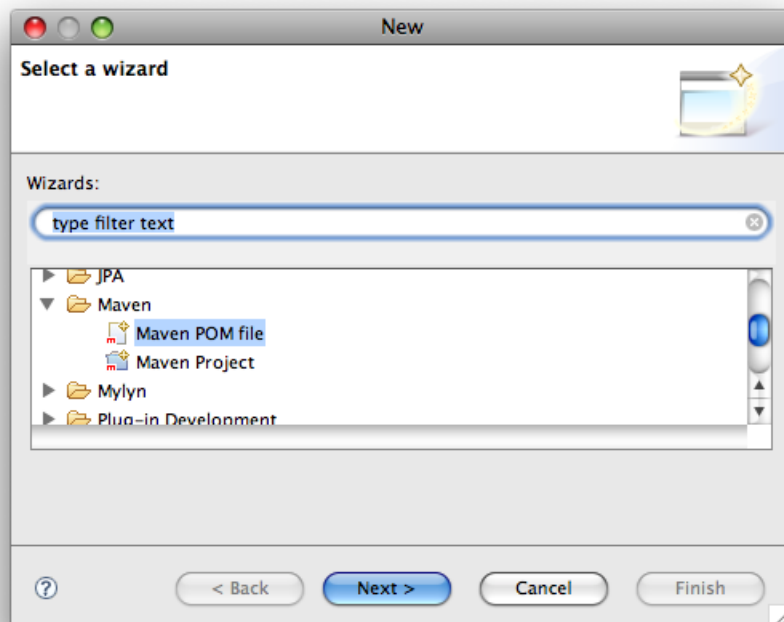


Figure 4.2: Create a new Maven pom.xml project file from the Eclipse menu `File>New>Other...`

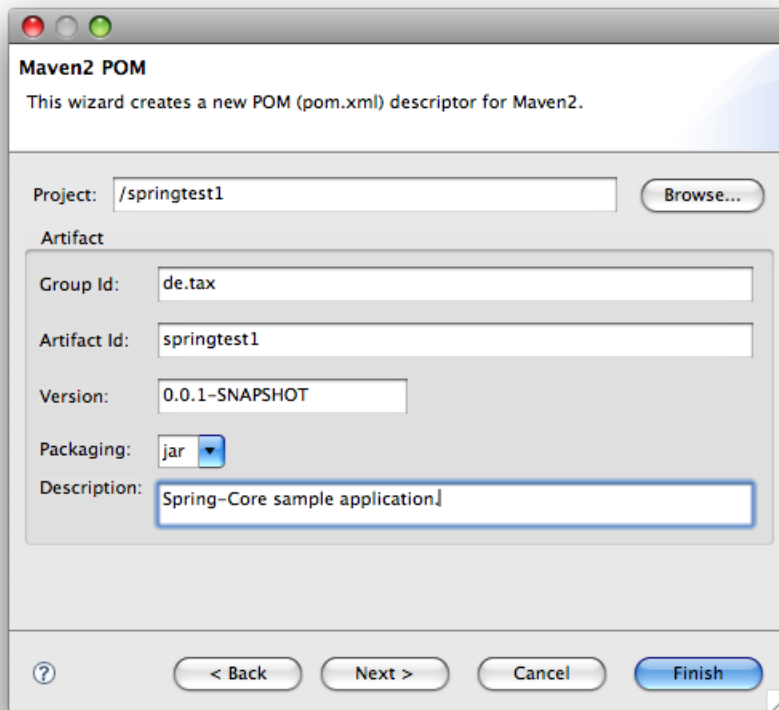


Figure 4.3: We want a normal runnable `.jar` file as artifact. No `.war` for a webapplication needed for now.

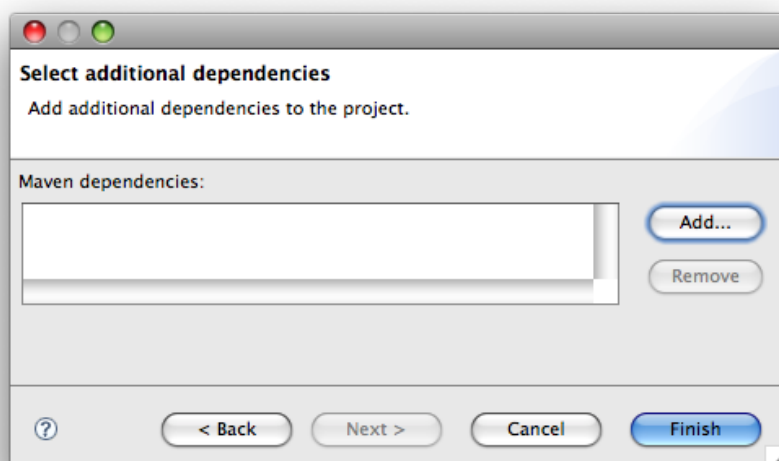


Figure 4.4: Add the Spring-Core as dependency

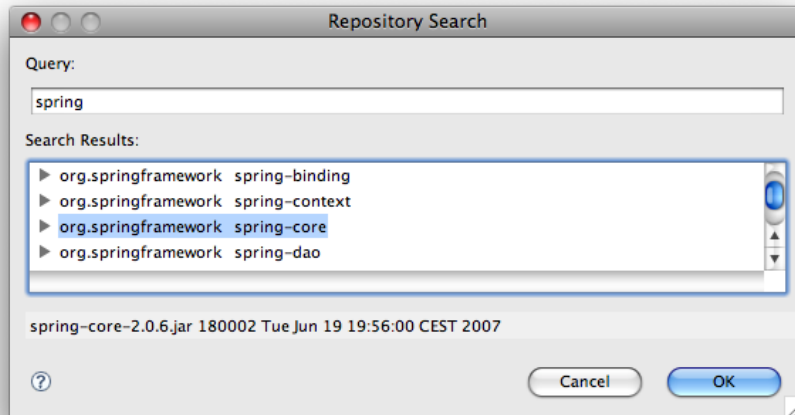


Figure 4.5: **Type ahead** – The Maven-plugin automatically expands your search string *spring* in realtime based on the Maven repository. Select the new Maven group ID scheme `org.springframework spring-core`. I also selected Spring-Context.

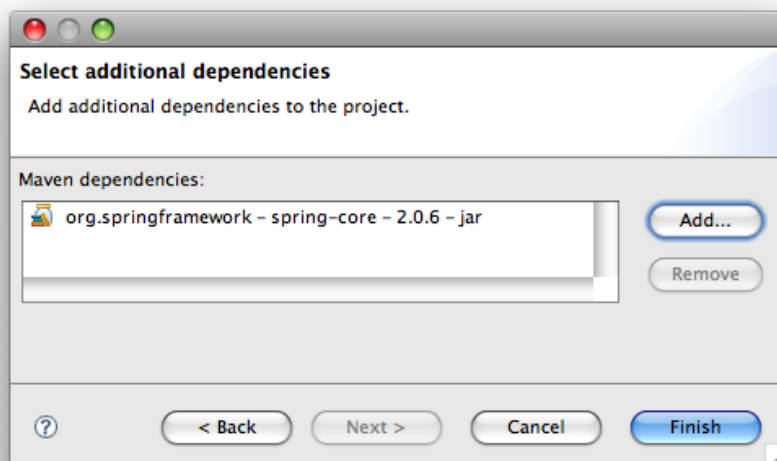


Figure 4.6: Press *Finish* and we are done.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="HTTP://MAVEN.APACHE.ORG/POM/4.0.0"
3     xmlns:xsi="HTTP://WWW.W3.ORG/2001/XMLSCHEMA-INSTANCE"
4     xsi:schemaLocation="HTTP://MAVEN.APACHE.ORG/POM/4.0.0 HTTP://MAVEN.
5         APACHE.ORG/MAVEN-V4_0_0.XSD">
6
7     <!-- ===== -->
8     <!-- The Basics -->
9     <!-- ===== -->
10
11     <modelVersion>4.0.0</modelVersion>
12
13     <groupId>de.tax</groupId>
14     <artifactId>springtest1</artifactId>
15     <version>0.0.1-SNAPSHOT</version>
16     <description>Spring-Core sample application.</description>
17
18     <name>springtest</name>
19     <dependencies>
20         <dependency>
21             <groupId>org.springframework</groupId>
22             <artifactId>spring-core</artifactId>
23             <version>2.5.1</version>
24         </dependency>
25         <dependency>
26             <groupId>org.springframework</groupId>
27             <artifactId>spring-context</artifactId>
28             <version>2.5.1</version>
29         </dependency>
30     </dependencies>
31 </project>
```

Listing 4.1: The generated pom.xml with the modified version 2.5.1

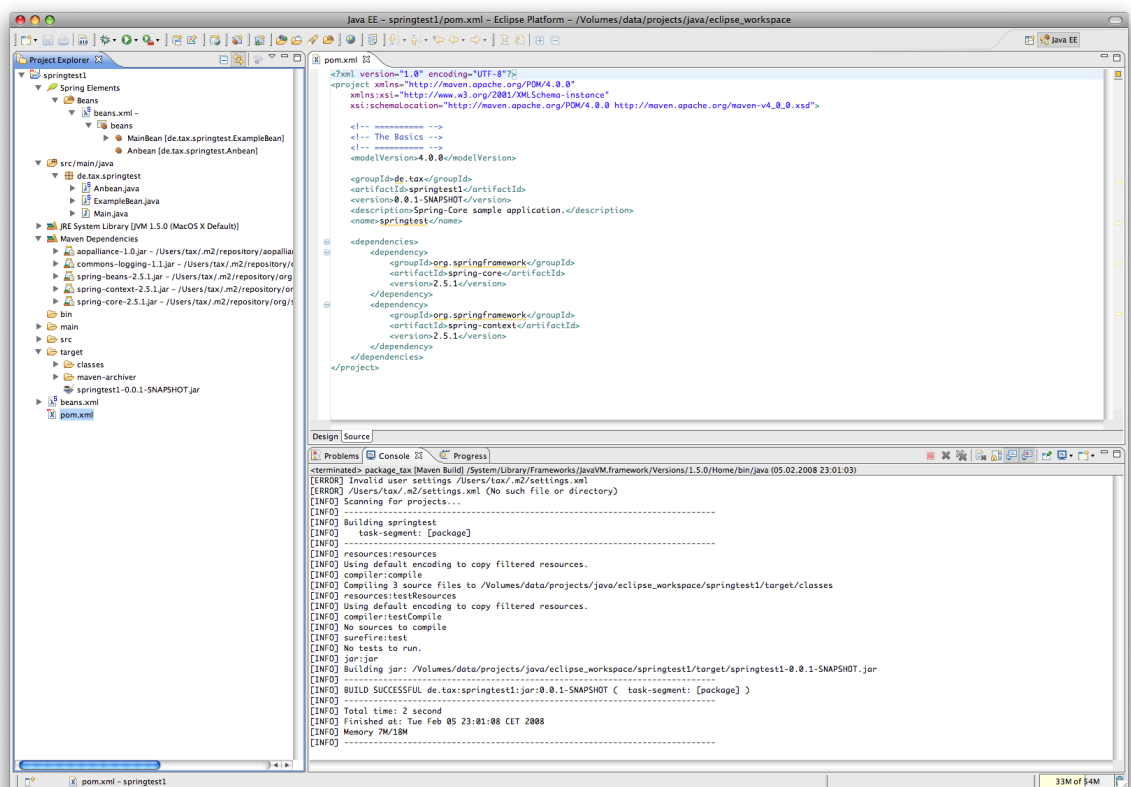


Figure 4.7: Your workbench should look like this.

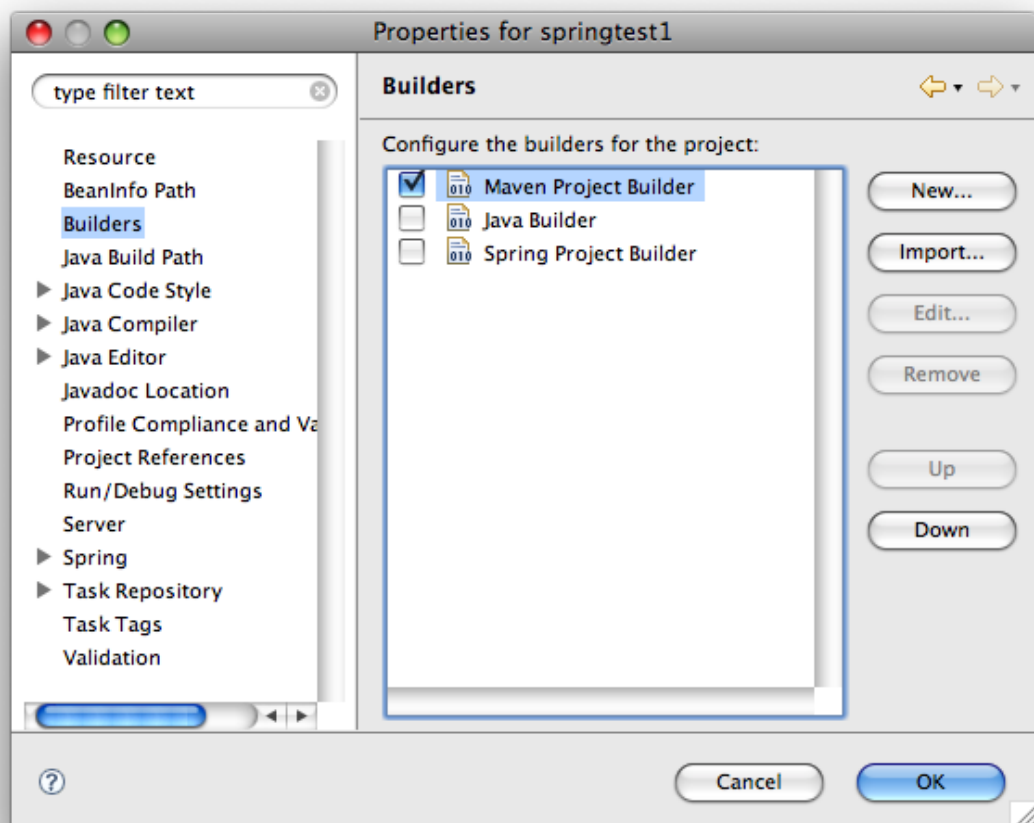


Figure 4.8: Right Mouseclick on the Project file. Now we have three different builders in Eclipse. I just disable Java and Spring and switch to the Maven solution. [TODO] Enabled again the standard Java builder, but the output directory is set to `bin/` and Maven uses `target/`. So each builder should be fine.

# Chapter 5

## iPhone Webapp development

On the iPhone the built in browser Safari is based on WebKit. WebKit has some extension und features for building web application with a better user experience. See the WebKit animation reference on

<http://webkit.org/specs/CSSVisualEffects/CSSTransitions.html>

### 5.1 Building full screen WebApps

To define the current webpage as a full screen webapp add the following meta tags to your `<head>` section of the html page. Please note that it only takes affect, when you store the webapp on your homescreen. A normal bookmark won't work.

**Hint:** The native iPhone resolution is currently `width=320px` and `height=480px`.

#### 5.1.1 Navigation Elements or fixed Divs

The iPhone accepts the `position:fixed` attribute but the rendering appears like a `position:absolute` element. This is problematic for toolbars. There are some solutions on the internet, but I experienced problems with them, when rotating the canvas or entering form elements: In those use cases the pseudo fixed elements moved up or down, and after leaving the iPhone form editor the position did not restore. Perhaps a nice UI-Library will come along. For this time I'm using full screen apps with no fixed elements.

Please note that this is not a bug on the iPhone OS but more a behaviour of the defined viewport in Safari.

1

```
<meta name="APPLE-MOBILE-WEB-APP-CAPABLE" content="YES">
```

Listing 5.1: Add this meta tag to suppress the toolbar and the Safari address bar..

```

1 <meta name="APPLE-MOBILE-WEB-APP-STATUS-BAR-STYLE" content="BLACK-
  TRANSLUCENT">

```

Listing 5.2: You can use this feature only if you place the webpage on the iPhone homescreen. Any other weblink will open another webpage. This simulates a real world iPhone application.

### 5.1.2 Detecting orientation changes

You can register the `onOrientationChange()` event on your body element. Then you can read the `window.orientation` attribute and will get the discrete number

Value	Description
0	Portrait
-90	Landscape right, screen turned clockwise
90	Landscape left, screen turned counterclockwise
180	Portrait upside-down portrait

Table 5.1: `window.orientation` values

**Hint:** Using the iPhone simulator I noticed, that in full screen mode, the iPhone 2.1 OS does not send the `onOrientationChange()` events.

### 5.1.3 Detecting gestures and touches

```

1 <body onTouchMove="DOSOMETHING(EVENT)";>
2 // event.preventDefault();

```

Listing 5.3: Register the touch event with a function

# Chapter 6

## Application development for iPhone

Please see the good Apple Developer Connection for tutorials:

<http://developer.apple.com/iphone/library/documentation/iPhone/Conceptual/iPhone>

When developing with iPhone simulator you find the local installments and app documents in the user library `Application Support/iPhone Simulator/User/Applications/MYAPPID/Documents`.

### 6.1 Troubleshooting Info.plist

**Symptom:** I don't see my newly added logo but I declared it in the info.plist file.

**Answer:** When you add resources or files in the filesystem please be sure that also in the xcode user interface the files are added or updated. Otherwise you may not see in the build and target process the added file. This can be easily verified in the target section, which files are copied.

**Answer variant 2:** Please do not drag and drop files from non-project paths. Xcode will reference only the file and then not copy it to the destination.

**Answer variant 3:** Doublecheck the filename. Preview App showed to me Adobe Photoshop PNG file. Resaving in Preview as PNG solved the issue.

### 6.2 Mapkit

You must include the Mapkit framework when using `MKMapView`

## 6.3 Objective-C

```
1 #define MYCONSTANT @"MYTEXT"
```

Define is used without a = and semicolon ;. I try to remember my old C++ time.

### 6.3.1 Headerfiles

Use the @interface directive for declaration of classes. The definition is done in the .m file which contains the implementation.

```
1 @interface firstappAppDelegate : NSObject <UIApplicationDelegate> {  
2     UIWindow *window;  
3     MyViewController *myView;  
4     SpecialClass *instanceVariable;  
5 }
```

### 6.3.2 Singletons

Please see

```
1 http://cocoawithlove.com/2008/11/singletons-appdelegates-and-top-level.html
```

and

```
1 http://developer.apple.com/mac/library/documentation/Cocoa/Conceptual/  
CocoaFundamentals/CocoaObjects/CocoaObjects.html#//apple\_ref/doc/uid/  
TP40002974-CH4-SW32
```

### 6.3.3 Properties

Add accessor setter/getter methods in the header file.

Please consult also the synthesise property in the implementation context file between the @implementation property.

```
1 @property (nonatomic, retain) MyViewController *myViewController;
```

### 6.3.4 IBOutlet

IBOutlet tags a method for Interface Builder.

### 6.3.5 Forward Declaration in header files

You don't include the header but the compiler gets a promise that the file is there. This avoids circularities. The `@class` compiler declaration must be added before the `@interface` directive.

```
1 // inform the compiler not to search for the header file for MyViewController
2 @class MyViewController;
```

But this is in the implementation .m needed.

```
1 #import "MyVIEWCONTROLLER.H"
```

### 6.3.6 Trimming Strings like Java trim()

```
1 NSString *searchText = [[adressTextField text]
    stringByTrimmingCharactersInSet:[NSCharacterSet
    whitespaceAndNewlineCharacterSet]];
```

## 6.4 Core Data

To use Core Data symbols import the Core Data framework by doubleclicking on the project name in the Xcode target and add it to the linked libraries. The import statement for Core Data is `CoreData/CoreData.h`.

**Symptom:** Console output says: Can't merge models with two different entities named 'myXYZName'.

**Answer:** Perform a Build Clean All command to ensure that all target models are deleted. I renamed my persistent store and Core Data had two model description files in the target app.

## 6.5 UIKit

### 6.5.1 Abort the current event cycle to get direct UI drawing

When Cocoa handles an UI event sometimes calculations or businesslogic can freeze the ui. To prevent this and update the UI directly just add his line after your UI commands and before the calculation intensive code.

```
1 [activityIndicator startAnimating];
2 // fire the current event cycle to update the animation
3 CFRunLoopRunInMode (CFRunLoopCopyCurrentMode(CFRunLoopGetCurrent()),
    0, FALSE);
```

### 6.5.2 TableViewCells

Displaying an image in a TableViewCell should work like this. Please note to crop or size your image to the default size with 43 pixels height and 50 pixels width.

```
1 UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:
    CellIdentifier];
2 if (cell == nil) {
3     cell = [[[UITableViewCell alloc] initWithStyle:
        UITableViewCellStyleSubtitle reuseIdentifier:CellIdentifier]
        autorelease];
4 }
5 [cell.imageView setImage:[UIImage imageNamed:@"ICON.PNG"]];
```

### 6.5.3 Loading NIBs withing the code without the init

When you want to load NIBs from the code you could use the following construct.

```
1 [[ NSBundle mainBundle ] loadNibNamed:@"MY_NIBFILE" owner:self options:nil
    ];
```

Then some connection can be made.

### 6.5.4 Activity Icon in Navigation Bar

This is a simple code snippet that can save you hours of mucking around trying to place your activity indicator (spinning icon) in the Navigation Bar of the iPhone. This example assumes that your view controller has a property of type `UIActivityIndicatorView` called `self.activityIndicator`:

```
1 // Create a 'right hand button' that is a activity Indicator
2 CGRect frame = CGRectMake(0.0, 0.0, 25.0, 25.0);
3 self.activityIndicator = [[UIActivityIndicatorView alloc]
4     initWithFrame:frame];
5 [self.activityIndicator sizeToFit];
6 self.activityIndicator.autoresizingMask =
7     (UIViewAutoresizingFlexibleLeftMargin |
8     UIViewAutoresizingFlexibleRightMargin |
9     UIViewAutoresizingFlexibleTopMargin |
10    UIViewAutoresizingFlexibleBottomMargin);
11
12 UIBarButtonItem *loadingView = [[UIBarButtonItem alloc]
13     initWithCustomView:self.activityIndicator];
14 loadingView.target = self;
15 self.navigationItem.rightBarButtonItem = loadingView;
```

You can then call `[self.activityIndicator startAnimating]` when you want it to start spinning.

## 6.6 Core Location

The MKReverseGeocoderDelegate must implement both methods to work properly.

## 6.7 Performance

### 6.7.1 Network Latency on Cell Networks

The Latency on Cell Networks can be usually about 500[ms].

### 6.7.2 Resources

Please check your website for too many resources. Tipp: Lesser resources in HTML files but use larger image data, e.g. image sprites with CSS background positioning.

# Appendix A

## Version History

- 12. September 2009** Refined UIKit and CoreLocation tips
- 10. August 2009** Added iPhone application development
- 10. Februar 2009** Added continuous integration with hudson
- 7. Februar 2009** Updated Subversion informations.
- 5. November 2008** Added more information about Maven dependency management
- 5. Oktober 2008** Added a new Chapter for the iPhone Webapp development
- 6. Mai 2008** Added JSF Maven archetype
- 4. Mai 2008** Note to Eclipse Web Tools Platform (WTP)
- 2. Mai 2008** Corrected URL to Eclipse Maven plugin
- Winter 2007** Initial release