

Pharmadex Application Development Guide



USAID
FROM THE AMERICAN PEOPLE

SIAPS 
Systems for Improved Access
to Pharmaceuticals and Services

The preparation of this manual was made possible by the support of the American people through the United States Agency for International Development (USAID), under the terms of Cooperation Agreement number AID-OAA-A-11-00021 with the Government of Mozambique. The content is the responsibility of Management Sciences and does not necessarily reflect the views of USAID or the American Government.

About SIAPS

The purpose of the System for Improved Pharmaceutical Services Access Program (SIAPS) is to ensure the availability of quality pharmaceuticals and effective pharmaceutical services to achieve desired health outcomes. To achieve this goal, SIAPS outcome areas include improving governance, building capacity for pharmaceutical management and services, addressing information needed for decision-making in the pharmaceutical sector, strengthening funding strategies and mechanisms to improve access to medicines, and increase the quality of pharmaceutical services.

Recommended Quote

This manual may be reproduced if credit is given to SIAPS. Please use the following quotation.

Pharmadex Application Development Guide. 2017. Submitted to the US Agency for International Development by the Systems for Improved Access to Pharmaceuticals and Services (SIAPS) Program, Arlington, VA: Management Sciences for Health.

Systems for Improved Access to Pharmaceuticals and Services
Pharmaceuticals and Health Technologies Group
Management Sciences for Health
4301 North Fairfax Drive, Suite 400
Arlington, VA 22203 USA
Telephone: 703.524.6575
Fax: 703.524.7898
E-mail: siaps@msh.org
Website: www.siapsprogram.org

TABLE OF CONTENTS

Acknowledgments.....	iv
Objective	1
Skill Requirements for Developer	2
Architecture.....	3
The Database.....	4
The application.....	7
Development environment.....	9
Description	9
Tools.....	9
Pharmadex SDK.....	9
How to install and configure the development environment.....	12
Pre-conditions.....	12
How to install the tools.....	12
How to install the SDK	15
Configuration hints	24
Preface	24
Database connection.....	24
Error processing	25
String resources and i18n	26
Authentication and security.....	27
Jasper Report additional configuration	28
Known issues	29
JSF errors.....	29
Absence of dependency.....	31
Jasper reports do not load.....	32
Cleanup of the project	33
Development hints	34
Search for JSF page.....	34
Codes hot swap.....	36
Jasper reports	36
Useful links	38

ACKNOWLEDGMENTS

We are grateful for the leadership, direction, and cooperation of the Head of the Pharmacy Department, Dra. Tânia Sitoie.

This manual would not have been completed without the active contributions of these registration sector staff:

Dra. Sultana Razaco, Head of the Registration Sector

Dr. Joaquim Tomás

Dra. Velma Capote

Dra. Lígia Mabunda

Dra. Dra Nazália Macuvele

Dra. Márcia Lithuri

Dra. Cidália VilanculosMr. Aminaldo Coreia

Mr. Acino Paulo

Dra. Josina João

Dr. Cassiano João

Dr. Benedito Nhaquila

Dra. Ariososa Picane

OBJECTIVE

This guide contains “kick start” information for Pharmadex’s developer such as:

1. Skill Requirements for Developer
2. Application Architecture
3. Database structure
4. Application outline
5. Installation of development environment
6. Additional sources of information

This guide contains only Pharmadex specific information. For deep knowledge of tools used, please refer to relevant product documentation and/or corresponding Community posts. Some useful links can be found in the “Additional information” chapter.

SKILL REQUIREMENTS FOR DEVELOPER

Developer's skill can be assessed by using the following gradations:

Awareness – means that developer can answer what this tool is and where it is used

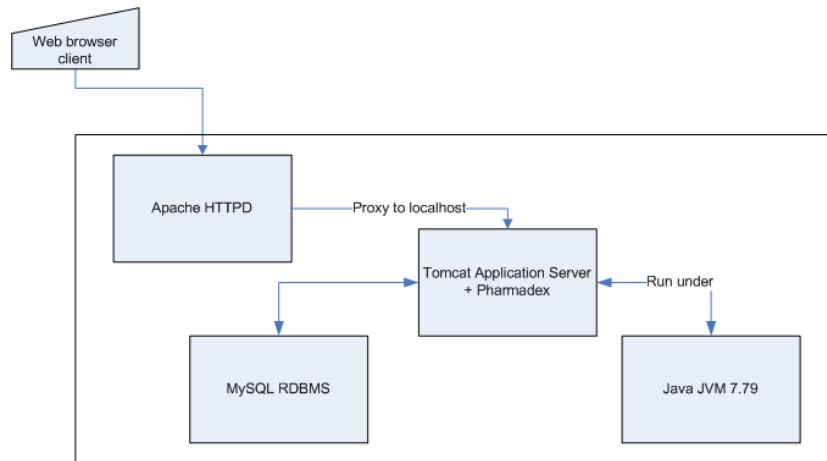
Knowledge – means that developer can point to at least one course taken/passed

Ability – means that developer can prove at least one successful project worked upon that fits the requirement below

- Java EE development - ability
- Maven 2 - knowledge
- Eclipse - ability
- Spring Framework - knowledge
- Jasper report framework - knowledge
- JSF development - ability
- PrimeFaces (implementation of JSF) - knowledge
- Hibernate and SQL – ability
- ERD diagrams and data modeling - ability
- MySQL - knowledge
- Apache Tomcat Application server - knowledge
- Apache HTTPD web server or similar – awareness
- Version control systems, like GIT, SVN - awareness

Please note, the above requirements are only a starting point of assessing a developer's skill. The ability and desire to learn new things is a key factor in the recruitment of a resource.

ARCHITECTURE



Pharmadex is a web application which means that Pharmadex and its functionality can be accessed using a web browser

Apache HTTPD is a web gateway for Pharmadex. The role of this software is to manage web connections from Pharmadex users in the most secure way. All connections for Pharmadex will be passed through (proxy) to the Pharmadex application server. Any other connection from outside will be rejected. Use **only** version 2.2.11 of Apache HTTPD.

Tomcat Application Server is a set of Java libraries (codes and data) that helps run Java web applications. Please, use **only** version 7.0.68 or 7.0.69 of the Tomcat Application Server.

Pharmadex is a web based Java application that can be deployed on any J2EE compliant application server, but for the current purpose it is recommended to use Tomcat Application Server 7. The deployable application is a file named <country_name>.war (for instance mozambique.war for Mozambique). Please, use the latest version of this file.

Both Pharmadex and Tomcat run under **Java Runtime Environment**. Please, use version 7.79 of Java.

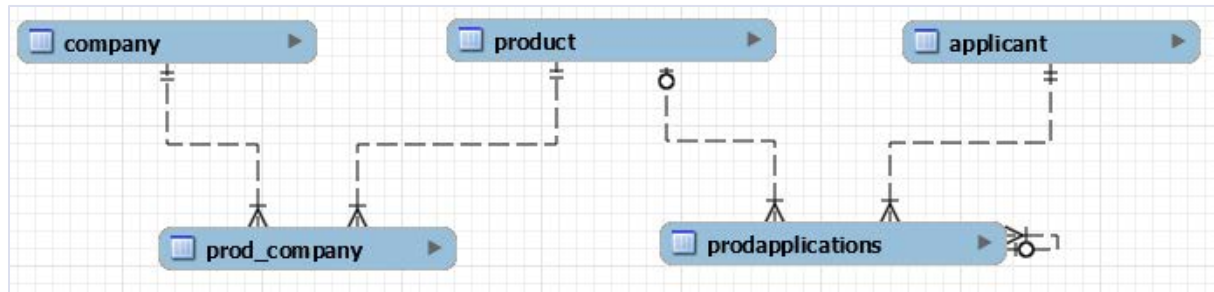
Two Java installations exist – Java SE Virtual Machine (JVM) and Java SE Java Development Kit (JDK). JDK contains the JVM and additional software that helps to manage Java application. Use **only** JDK version 7.79.

MySQL RDBMS is a Database Management System that stores and manages all data that is used by Pharmadex. Use the latest version of MySQL Community Edition.

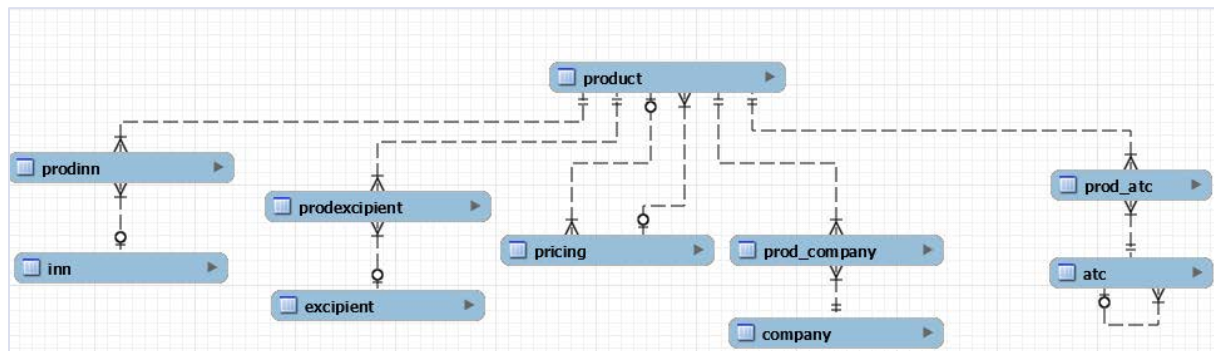
THE DATABASE

The typical database for Pharmadex contains about 70 tables. This chapter provides an outline of the database structure. This outline explains only the most important tables and provides a clue of how to read the whole database structure.

The core of the database is the “product” table that contains information about products. Every product has a product manufacturer and at least one applicant.



Below are the most important tables that make up **product description**. Names of table are almost self – explanatory. Note that “inn” is a table with INN terminology for active ingredients.



User access definition is important part of any application. Pharmadex system uses the database to define user access rights as well as access logging. See below

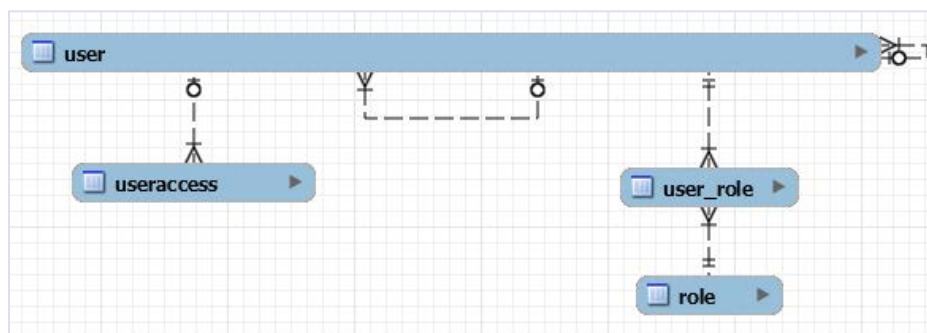
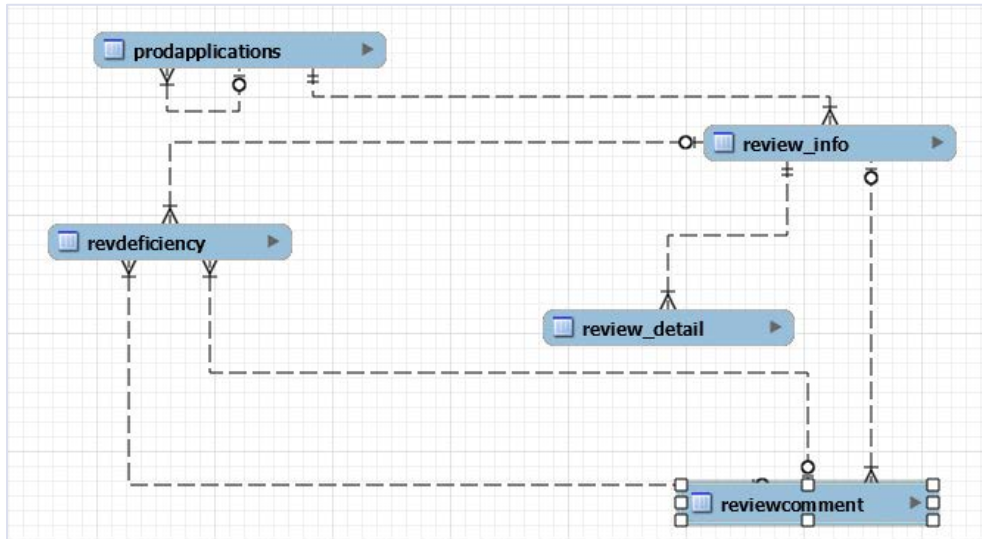


Table “useraccess” is user access log. The current data structure allows that any user may be assigned more than one role and a role may be assigned to more than one user. However, currently Pharmadex actually supports only one role for a given user and many users with the same role.

Most of **Pharmadex’s features** are related to application processing. The core table that connects product and applications is “prodapplications”. The database represents an application as a complex data structure consisting of many sub-models. Generally, a sub-model provides data structures for some sub-processes. Below is a simple example of the sub-model for the review process.



1. Each application may have one or more reviews (table “review_info”)
2. Each review may have
 - One or more review questions (“review_detail”)
 - Zero or more review comments (“reviewcomment”)
 - Zero or more review deficiency letters (“reviewdeficiency”)
3. Any review deficiency letter may be connected to zero or more comments (“reviewcomment”)

Reference data makes Pharmadex flexible enough to use the Pharmadex application software for constantly changing environments and even for different countries. Reference data tables are listed below.

Reference table	Description
admin_route	Route of product administration, e.g. “orally”
applicant_type	Types of Applicant
atc	Tree structure for ATC codes
checklist	Questions for all checklists

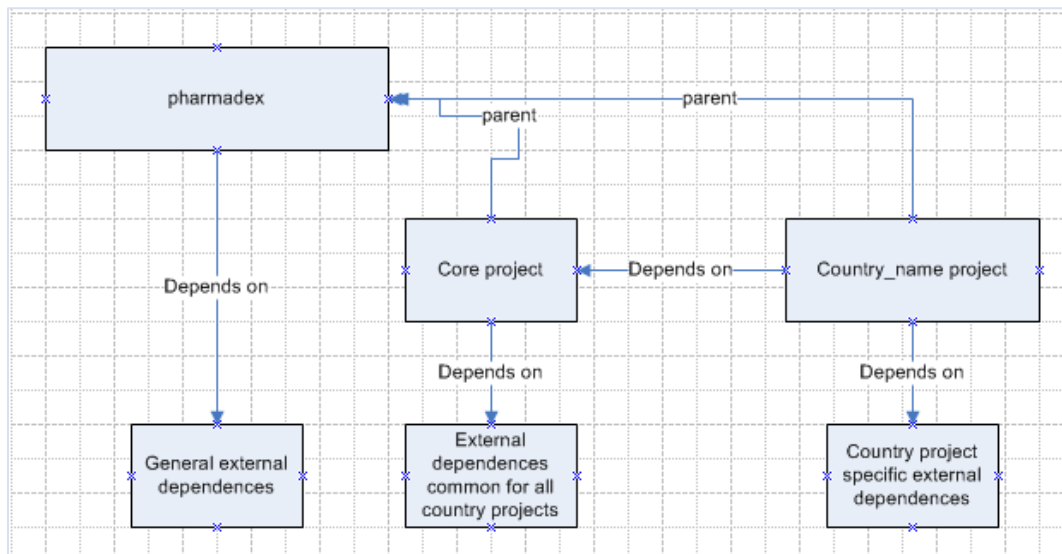
Reference table	Description
country	Countries
dosform	Dosage forms
dosuom	Dosage units
excipient	Excipients
inn	Active ingredients (INN terminology)
pharmclassif	Pharmaceutical classifier
resource_bundle	Set of text resources (only for multi-language installations)
resource_message	Messages and labels for multi-language installations
review_question	Questions for all reviews
role	Possible roles of a user

THE APPLICATION

The application is contained in a single file with name <country_name>.war. For instance mozambique.war.

The application source code is distributed into three Maven projects:

- Pharmadex project (parent)
- Core project that contains common codes/libraries
- Country specific project that contains country specific codes that overrides and extends core's codes



Both core and country specific projects are built upon customized Maven 2 Spring web application archetype. This archetype provides the following parts (recognized by Eclipse):

Project's part	Usage
JAX-RS Web Services	Not used
Deployment descriptor	In use, web.xml
Spring elements	In project "core" Spring root-context.xml can be found here along with dynamically built beans definitions. For the country specific project this part is typically empty
Web Resources	This is a convenient location for all static resources, Beans (Java sources), configurations etc.
JPA Content	Only persistence.xml that defines the name of the persistent unit is located here
Java Resources	All Java sources and related resources. In other words this is the location of WEB-INF/classes in the application

Project's part	Usage
Java Script resources	Not used. All Java scripts are in static web resources (see Web Resources)
src and target	Typical Maven source and target folders. Jasper Report definitions (jrxml) can be found here.

DEVELOPMENT ENVIRONMENT

Description

Pharmadex is a “pure” Maven 2 project. It means that any convenient development tool such as Eclipse, Netbeans, IntelliJ IDEA may be used. This guide details the use of Eclipse.

Pharmadex project uses MySQL database, Community Edition as data storage. Therefore, MySQL tools are needed. This guide explains the usage of native MySQL tool from Oracle.

Pharmadex project uses Jasper Report Community. Therefore, Jasper Report Studio will be needed at a minimum.

Tools

To develop Pharmadex, the following tools need to be installed on the developer’s workstation:

- Eclipse Mars or later. Application has been developed using Mars, later releases of Eclipse have not been tested, and previous releases will not work either.
- Maven 2
- Jasper Report Studio – the latest version¹
- MySQL Community Edition - the latest edition - version 5
- MySQL Workbench - the latest version
- Apache Tomcat server version 7.0.78 or 7.0.79
- Oracle Java JDK version 7.7x2

Pharmadex SDK

Pharmadex SDK is a software that has been developed by MSH. This software is a project with all source codes that can be modified. There are two kinds of Pharmadex SDK:

1. Country specific
2. Generic

¹ It is possible to use Eclipse plugin instead of the standalone studio, but it could not be made to work stably

² In addition the latest JRE may be installed

Country specific Pharmadex SDK contains codes particular to a country. Currently SDKs for the following countries exist:

1. Mozambique
2. Ethiopia
3. Bangladesh
4. Namibia

Generic Pharmadex SDK contains only general-purpose codes that may be used for development of a Pharmadex Application for countries that are not listed above.

The following describes the use of Pharmadex SDK for Mozambique as an example. It is represented as a set of file folders³.

#	Name of file/folder	Description
1	database	Folder for database dumps and additional files
1.1	createUser.bat and createUser.txt	MS Windows script and related to it SQL script for create initial Pharmadex user – name pharm, password pharm
1.2	Dump20161205MZ.sql	Database dump for Mozambique only
1.3	my.ini	Example of my.ini file. For references only.
	Pharmadex.sql	Database dump for the core project.
2	facelets-taglib-jsf20-spring-3	Folder for the additional java library and local Maven 2 installer for it
2.1	0.5	Extra folder, # of version
2.1.1	facelets-taglib-jsf20-spring-3-0.5.jar	The software - binary
2.1.2	facelets-taglib-jsf20-spring-3-0.5.pom	Maven control file
2.1.3	facelets-taglib-jsf20-spring-3-0.5-sources.jar	The software – source codes
2.1.4	install.bat	MS Windows script for install the software and the source codes to a local Maven 2 repository
3	pharmadex	Pharmadex Maven 2 project directory
3.1	core	Folder for core project.
3.1.1	src	Source codes for core project
3.1.1.1	main	Common work codes. It is not a good idea to modify something here.
3.1.1.1.1	jasperreports	Source codes examples for letters and documents. Some ideas from here can be obtained, however do not modify them directly
3.1.1.1.2	java	Common java sources
3.1.1.1.3	resources	Common resources. Mainly examples. Valid only: META-INF/persistence.xml (for tests only), folder “net” with additional fonts for letters and documents, log4j.xml – common logging configuration

³ Hereafter it is possible to distribute Pharmadex SDK from Version Control System such as GIT or SVN. Directions using a version control system will require a separate manual

#	Name of file/folder	Description
3.1.1.1.4	webapp	“View” part of the application and web application related configurations. You can find here JSF files, spring - specific configurations, web.xml.
3.1.1.2	test	Folder that contains examples of test codes. In the SDK provided all test codes are turned off to speed up application build. Some ideas how to implement own Junit tests for real application can be found here.
3.1.1.2.1	java	Test codes
3.1.1.2.2	resources	Test specific configurations and resources
3.1.2	target	Standard Maven 2 target folder. The application cannot be found here, only dependencies for the real project – core.jar, core.war
3.1.3	pom.xml	Maven configuration. The Project Object Model. Valid only for core project. Extends POM file of parent Pharmadex project
3.2	mozambique	Folder for mozambique project
3.2.1	src	Source code for the real project. These source codes extend and/or replace codes from the core project. It is only place to implement country – specific solutions
3.2.1.1	main	Work codes. All software here is intended to be modified
3.2.1.1.1	jasperreports	Real Jasper reports for letters and documents. These can be modified directly
3.2.1.1.2	java	Specific Java sources. Should be placed to the same packages as related core project sources. Will replace and/or extend corresponding core functionality
3.2.1.1.3	resources	Particular resources. All of them are in use, except i18N
3.2.1.1.4	webapp	This folder contains particular views that extend and/or replace core view functionality. Any core software components (templates, beans) can be used for these views. However only if these components are not declared here.
3.2.2	target	Standard Maven 2 target folder. The application can be found here
3.2.3	pom.xml	Maven configuration. The Project Object Model. Extends POM file of parent Pharmadex project
3.3	pom.xml	Maven configuration. The Project Object Model.

HOW TO INSTALL AND CONFIGURE THE DEVELOPMENT ENVIRONMENT

Pre-conditions

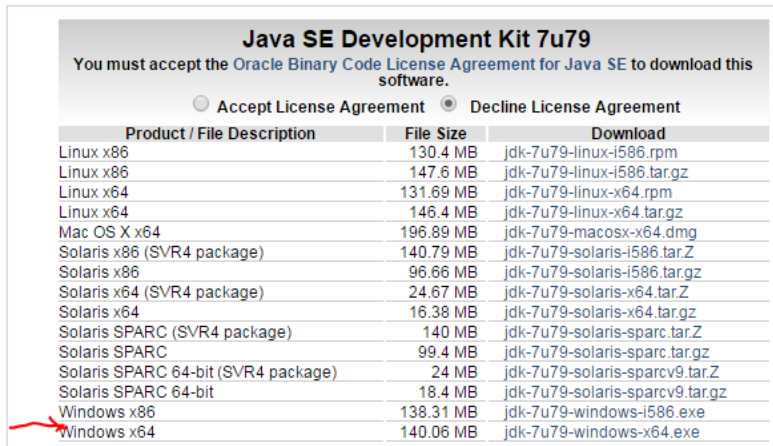
First, ensure that the required hardware is present on the computer. Minimum requirements are:

1. Processor – Intel I3, any generation
2. RAM – 8 GB
3. Disk (HDD, SSD) – 50 GB free
4. OS Windows 7 or above
 - Second, ensure that all tools are available. In addition, it will be nice to refresh knowledge of all components.

How to install the tools

Java

To install Java JDK, download JDK installer for Windows 64 from the <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>

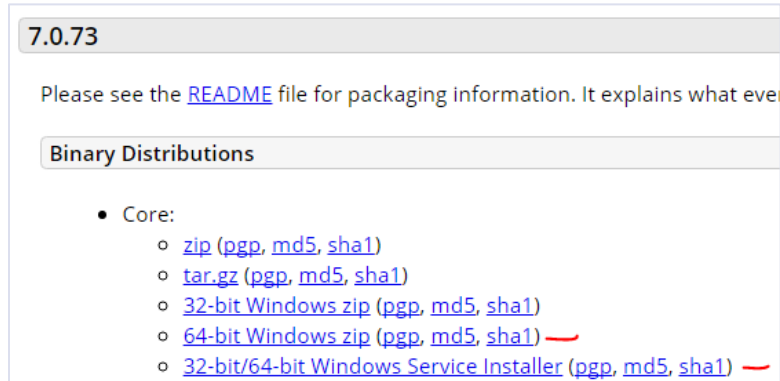


Java SE Development Kit 7u79		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
<input type="radio"/> Accept License Agreement <input checked="" type="radio"/> Decline License Agreement		
Product / File Description	File Size	Download
Linux x86	130.4 MB	jdk-7u79-linux-i586.rpm
Linux x86	147.6 MB	jdk-7u79-linux-i586.tar.gz
Linux x64	131.69 MB	jdk-7u79-linux-x64.rpm
Linux x64	146.4 MB	jdk-7u79-linux-x64.tar.gz
Mac OS X x64	196.89 MB	jdk-7u79-macosx-x64.dmg
Solaris x86 (SVR4 package)	140.79 MB	jdk-7u79-solaris-i586.tar.Z
Solaris x86	96.66 MB	jdk-7u79-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.67 MB	jdk-7u79-solaris-x64.tar.Z
Solaris x64	16.38 MB	jdk-7u79-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	140 MB	jdk-7u79-solaris-sparc.tar.Z
Solaris SPARC	99.4 MB	jdk-7u79-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	24 MB	jdk-7u79-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.4 MB	jdk-7u79-solaris-sparcv9.tar.gz
Windows x86	138.31 MB	jdk-7u79-windows-i586.exe
Windows x64	140.06 MB	jdk-7u79-windows-x64.exe

Follow standard installation steps, nothing special.

Apache Tomcat

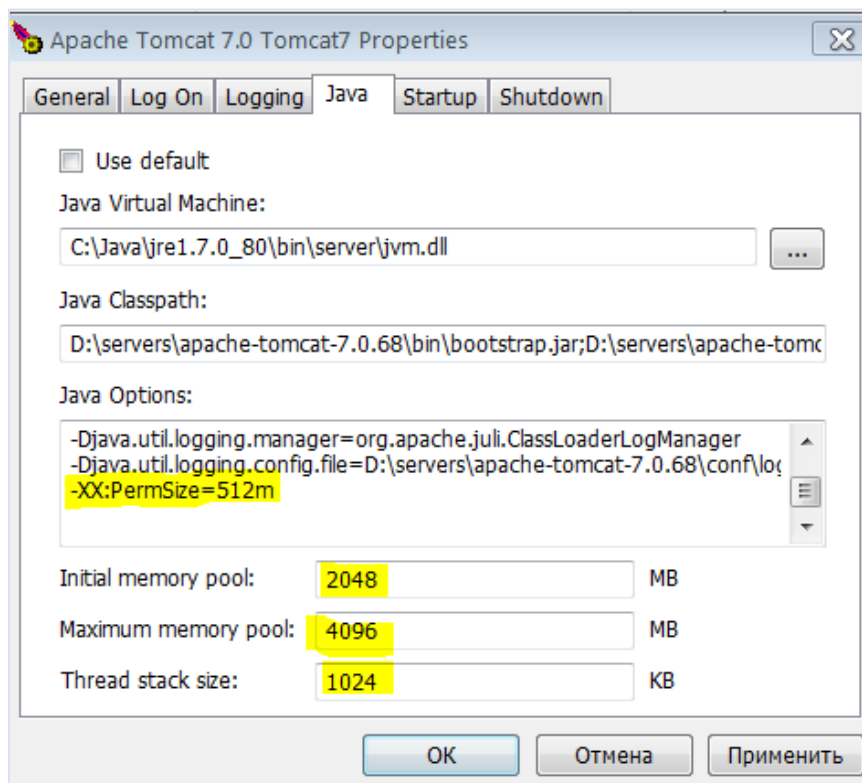
Download installer from <https://tomcat.apache.org/download-70.cgi>.



It will be a good idea to use this installation for final test of the application. Perform additional configuration steps:

Install Apache Tomcat as Windows service as described in the Apache Tomcat documentation (<https://tomcat.apache.org/tomcat-7.0-doc/windows-service-howto.html>)

Increase memory parameters using **tomcat7w.exe**



MySQL

Download MySQL installer from this link <http://dev.mysql.com/downloads/mysql/>.

This installer contains both MySQL Server and MySQL Workbench. After install, do additional configurations:

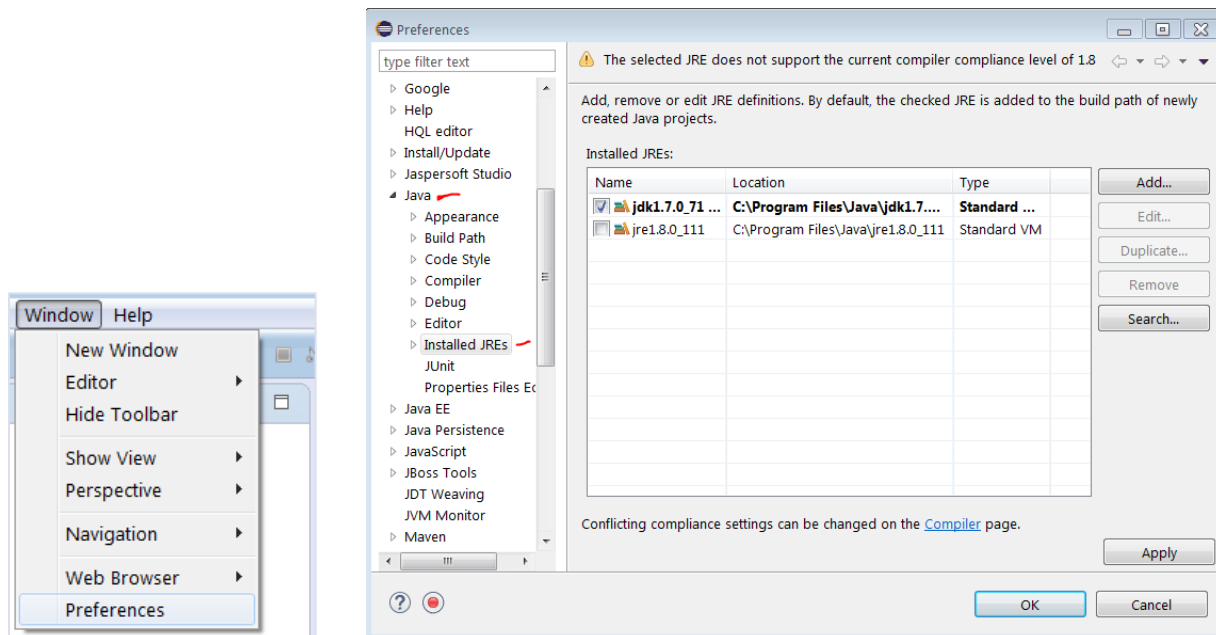
1. Add to all sections ([client] [mysqldump] [mysql] [mysqld]) of file my.ini line parameter **max_allowed_packet=1500M**
2. Create initial user with name **pharm** and password **pharm** using createUser.bat and createUser.txt
3. Create connection to the database in the Workbench

Eclipse

Install Eclipse Mars from <https://eclipse.org/mars/>. Installation process is standard.

Install Spring Tools Suite (STS) plugin. While it is a convenient tool, it may slow down Eclipse.

Ensure, that JDK 7.79 is default runtime environment



Jasper Studio

Install Jasper Studio Community Edition from here <http://community.jaspersoft.com/project/jaspersoft-studio>

Installation process is standard.

How to install the SDK

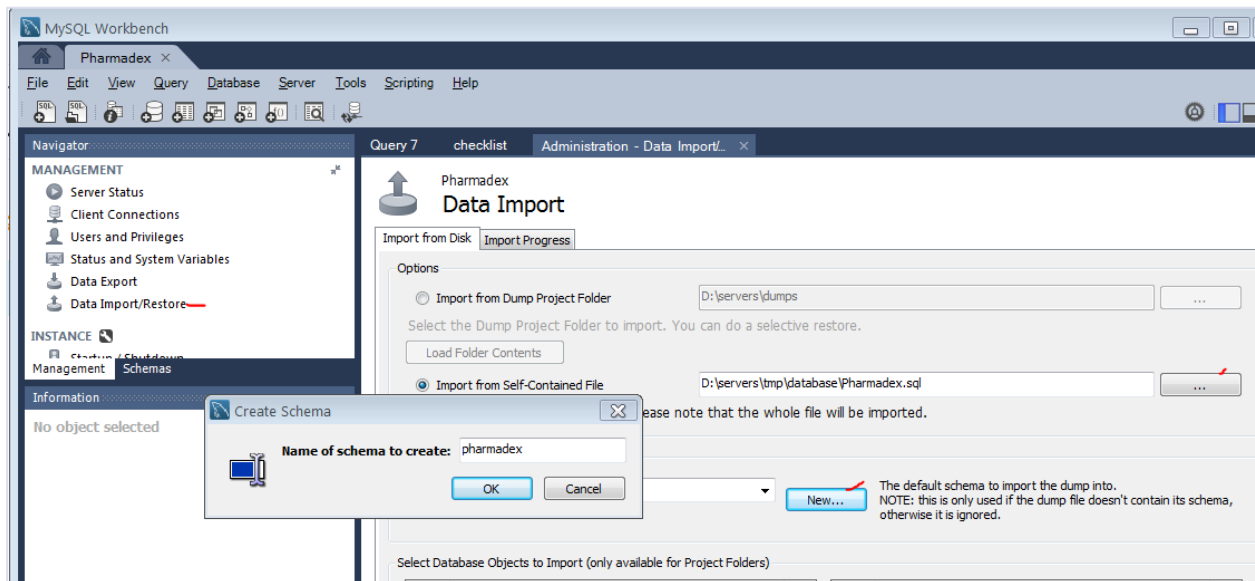
Database

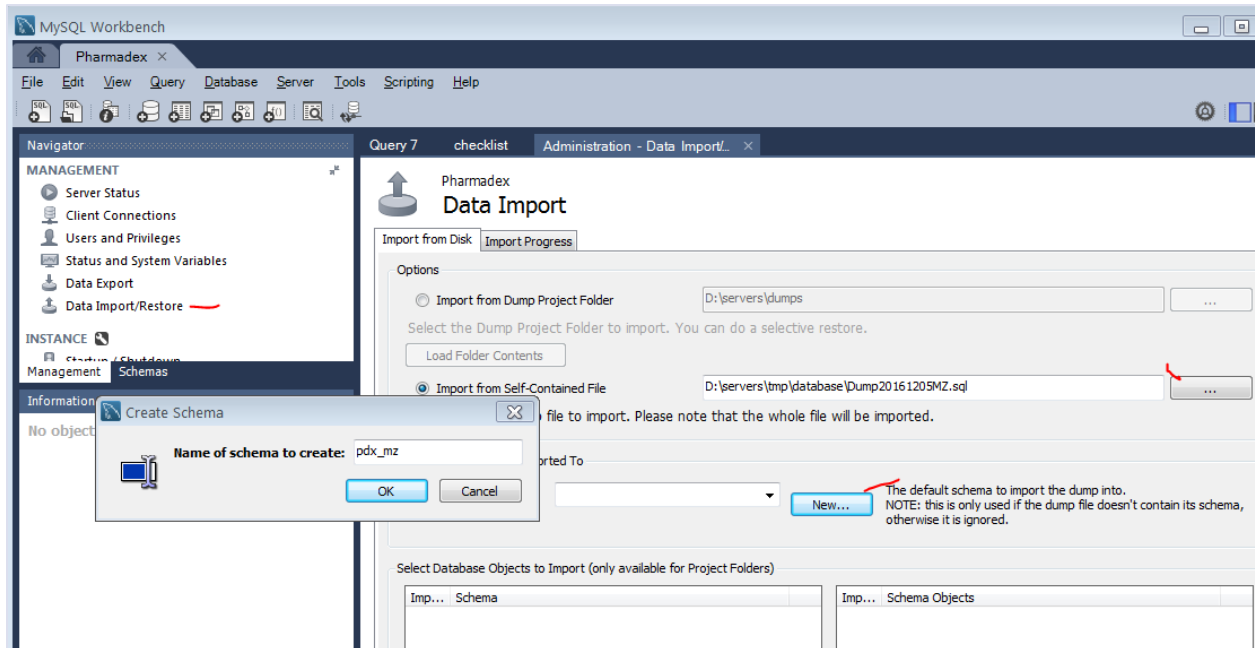
Installation of two databases is needed:

- pharmadex
- pdx_country (for instance pdx_mz)

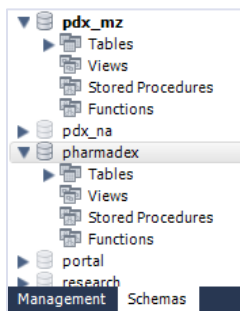
Before database installation, ensure that user with name **pharm** and password **pharm** has already been added as described above and this user has DBA rights to all databases.

Restore both databases using MySQL Workbench





Ensure that restore is completed



Database **pharmadex** is necessary for core project and JUnit tests. This database will never be used for the real application.

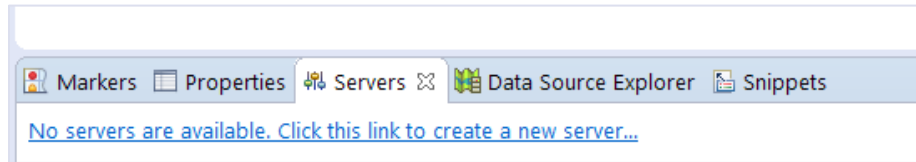
Database **pdx_mz** is a real database for Mozambique country application

Project

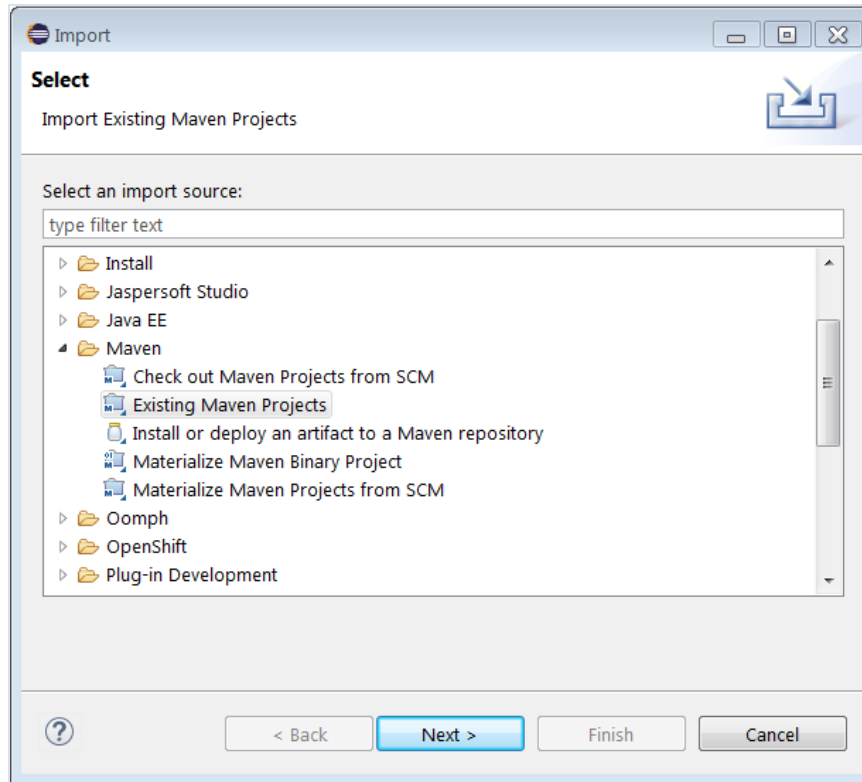
Copy **pharmadex** folder from Pharmadex SDK to somewhere on the disk. The default Eclipse workspace is a good place to copy, however this folder can be copied to any other place. Note that Eclipse will convert this folder to Eclipse Project.

Run Eclipse and switch to J2EE perspective

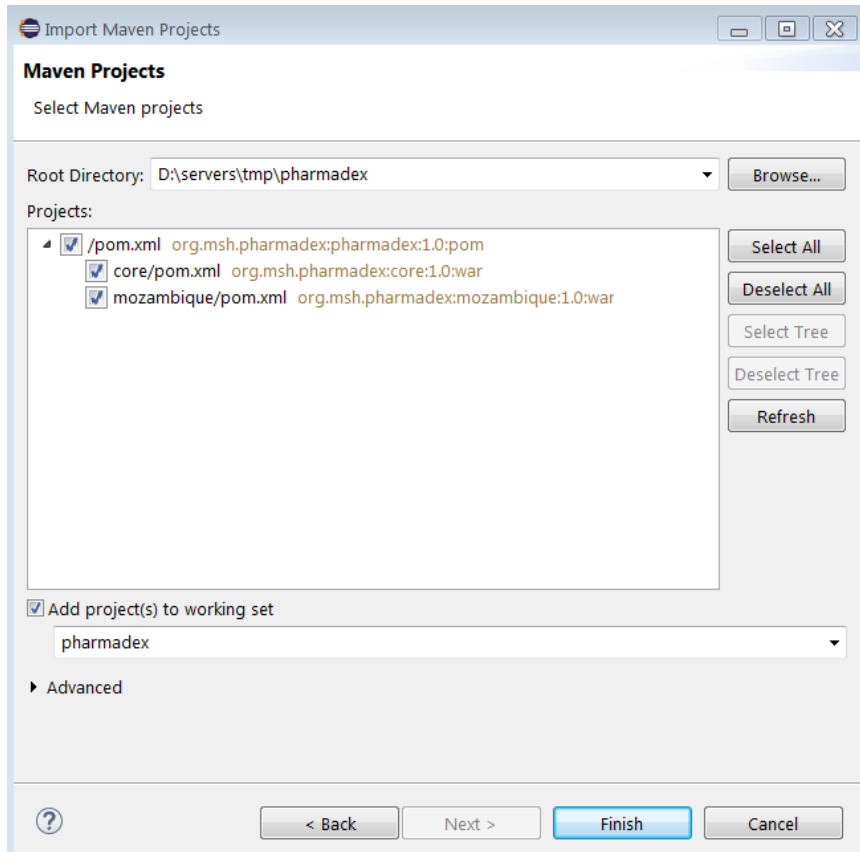
Go to Servers tab and add new Tomcat 7 server⁴. This server should run on 1.7 JDK.



Select to import Pharmadex software as an existing Maven project

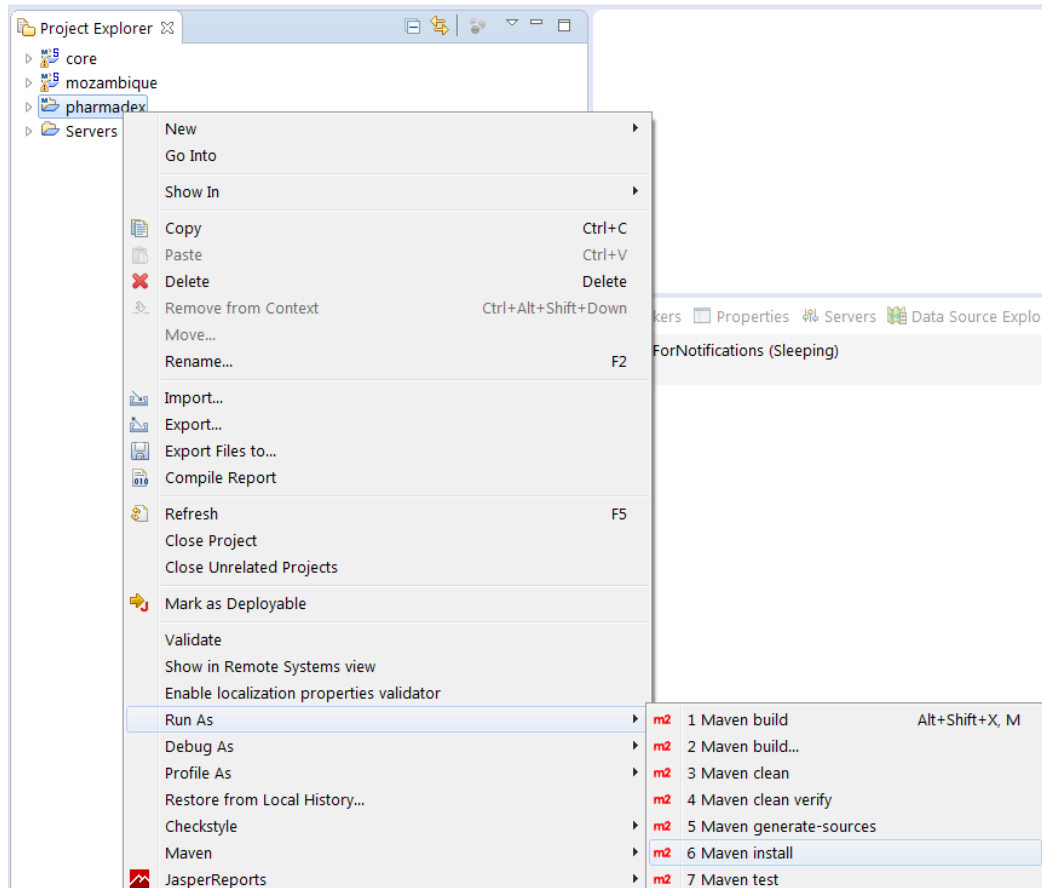


⁴ In case of difficulties refer here <http://stackoverflow.com/questions/14791843/eclipse-add-tomcat-7-blank-server-name>



It may take several minutes for this.

Select pharvadex project and execute Run as – Maven Install

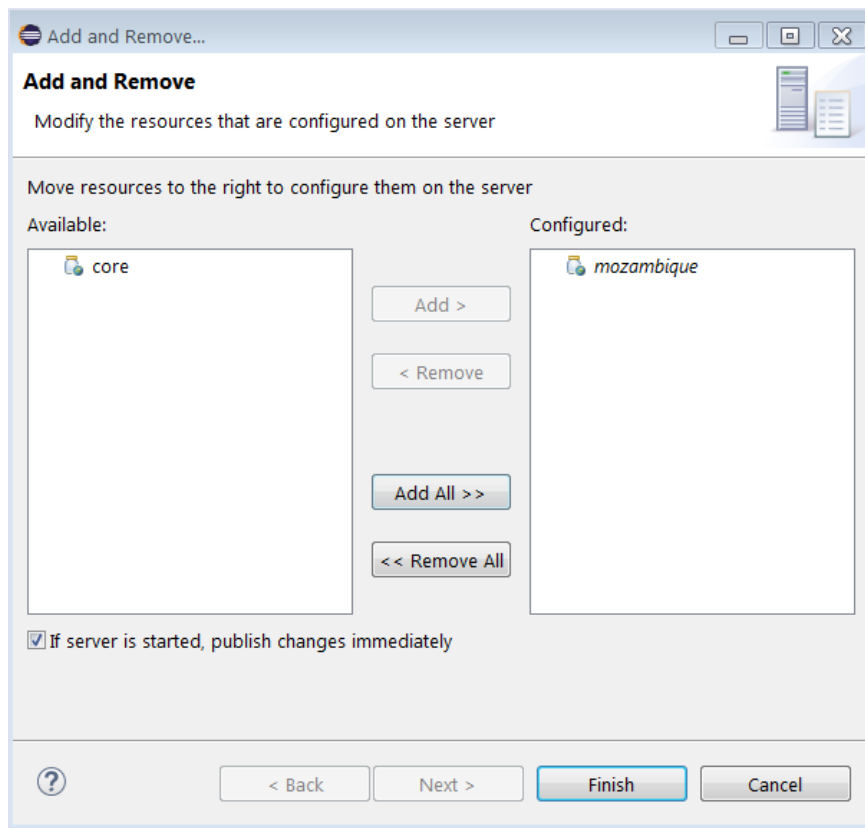
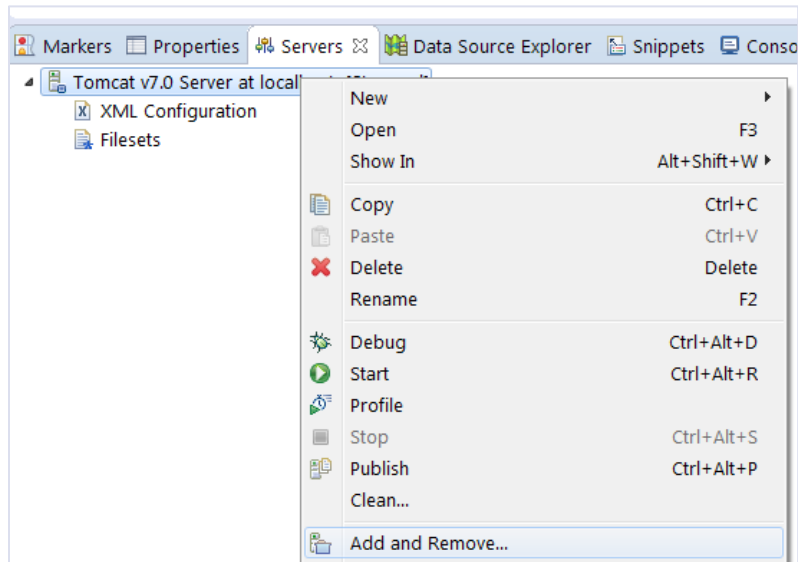


The Result should be as follows:

```
[INFO] --- maven-war-plugin:2.4:war (default-war) @ mozambique ---
[INFO] Packaging webapp
[INFO] Assembling webapp [mozambique] in [D:\servers\tmp\pharvadex\mozambique\target\mozambique]
[INFO] Processing war project
[INFO] Copying webapp resources [D:\servers\tmp\pharvadex\mozambique\src\main\webapp]
[INFO] Processing overlay [ id.org.msh.pharvadex:core]
[INFO] Webapp assembled in [7837 msecs]
[INFO] Building war: D:\servers\tmp\pharvadex\mozambique\target\mozambique.war
[INFO] --- maven-install-plugin:2.4:install (default-install) @ mozambique ---
[INFO] Installing D:\servers\tmp\pharvadex\mozambique\target\mozambique.war to d:\maven\repository\org\mspharvadex\mozambique\1.0\mozambique-1.0.war
[INFO] Installing D:\servers\tmp\pharvadex\mozambique\pom.xml to d:\maven\repository\org\mspharvadex\mozambique\1.0\mozambique-1.0.pom
[INFO] -----
[INFO] Reactor Summary:
[INFO]
[INFO] pharvadex ..... SUCCESS [ 0.628 s]
[INFO] Core Maven Webapp ..... SUCCESS [ 37.883 s]
[INFO] Mozambique Maven Webapp ..... SUCCESS [ 16.605 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 55.618 s
[INFO] Finished at: 2016-12-06T21:00:31+02:00
[INFO] Final Memory: 31M/507M
[INFO] -----
```

In this example, the deployable file of Pharvadex for Mozambique was created. Therefore, the project name is mozambique and application name is mozambique.war.

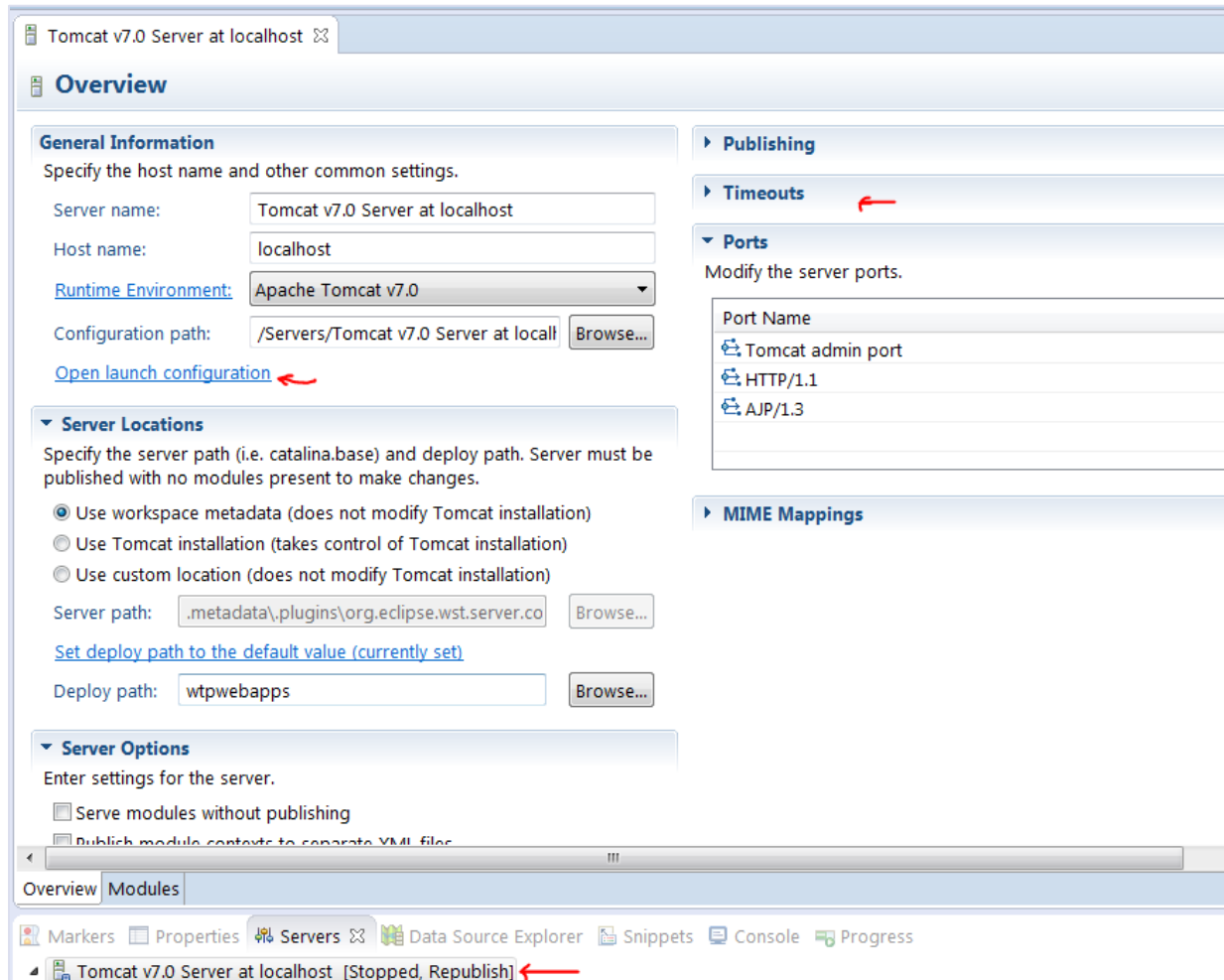
Publish the application to development server. To do this open Servers tab, then right click on Tomcat record



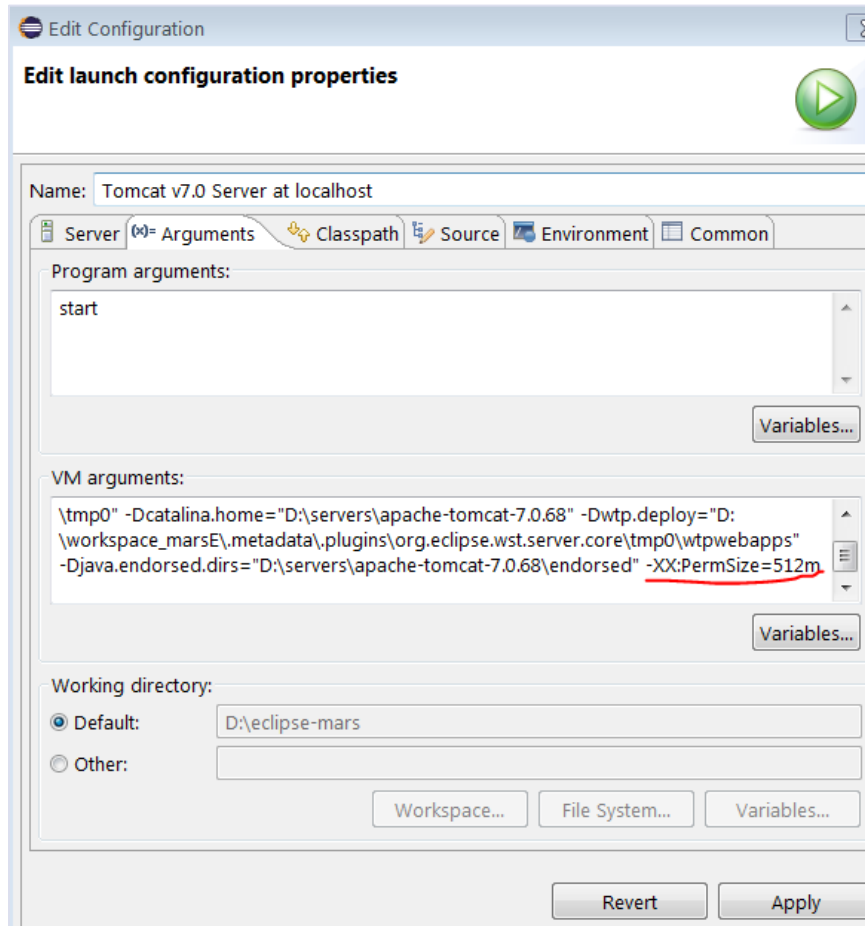
Increase memory settings and timeout for Apache Tomcat run profile. It is necessary because:

- Pharmedex Application requires PermGen at least 512M
- The start of Pharmedex under Apache Tomcat may take up to 1.5 min, depending on current load.

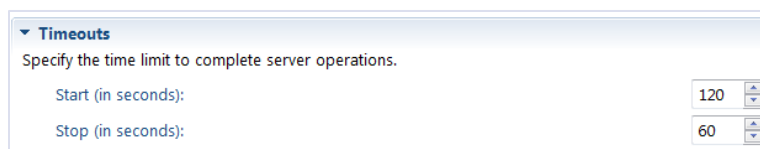
To do this open the tab Servers and click on Tomcat record. Eclipse will open Tomcat configuration screen.



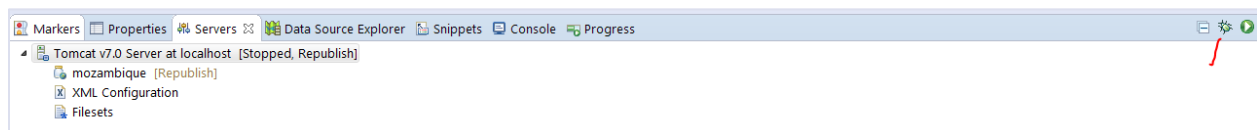
Edit launch configuration parameters to increase PermGen.



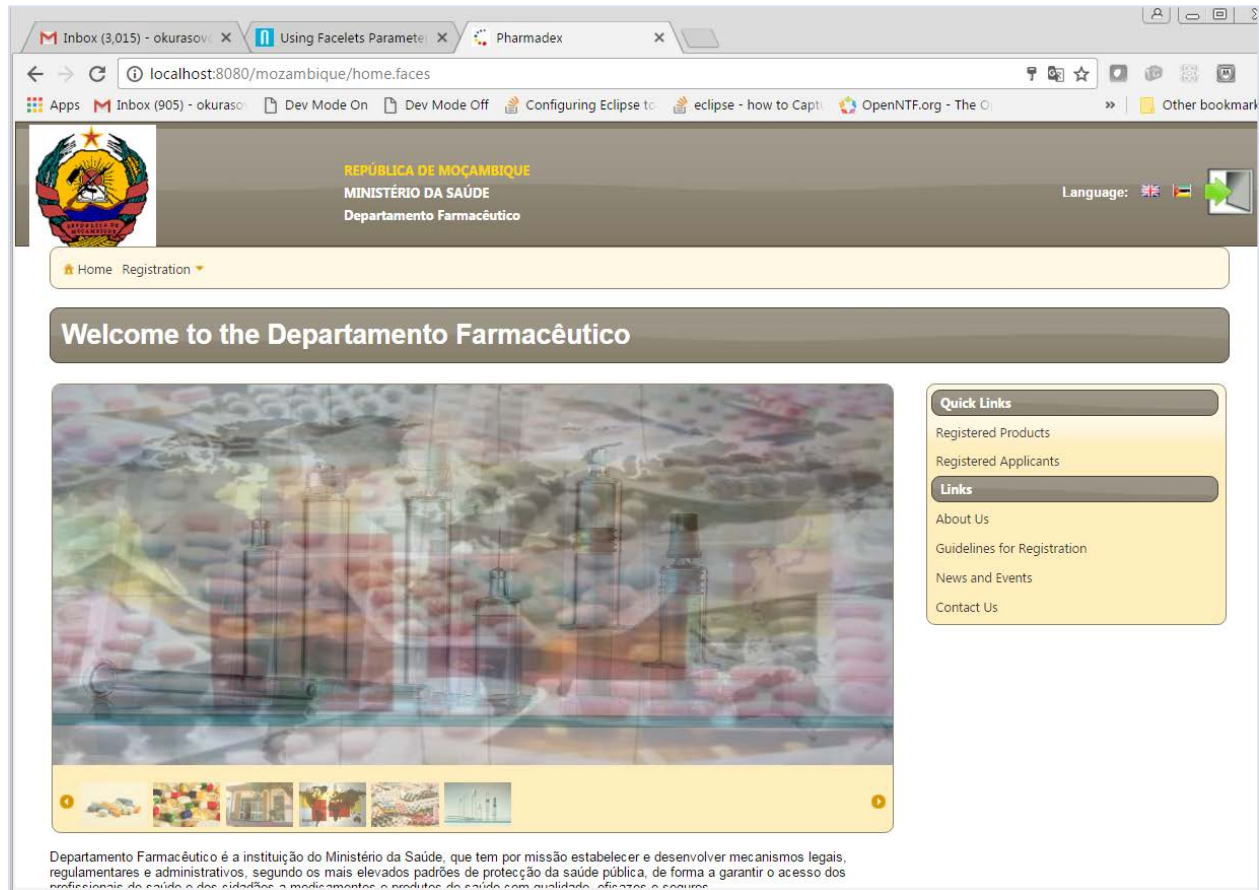
Edit Timeouts section to increase timeouts



After this, run the application in debug mode



Access the application using a web browser. If a home page is displayed as below then the application works!



In case of any difficulties, please check sub-chapter below, then the Community and vendor's documentation.

CONFIGURATION HINTS

Preface

Pharmadex application strictly follows common Spring configuration rules. However, because these rules are so flexible, implementation details are important.

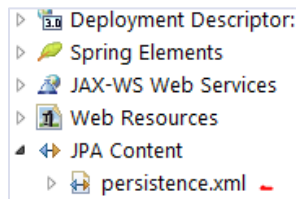
Database connection

There are no specific database configurations for development and deployment. Database configuration is file db.properties. This file can be found under the “resources” folder in src/main of country specific project. Standard Java properties file provides database connection parameters. Example is:

name	value
db.password	pharm
db.username	pharm
db.url	jdbc:mysql://localhost/pdx_mz
db.dialect	org.hibernate.dialect.MySQL5Dialect
db.driver	com.mysql.jdbc.Driver

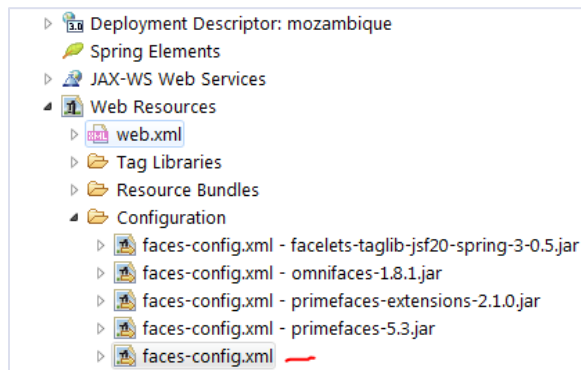
This data is used as defined in web.xml Spring's root-context.xml that can be found under src/main/webapp/WEB-INF folder of core project. Please pay attention that country specific project does not contain root-context.xml. It uses common root-context.xml from core project, however, with project specific connection data from particular db.properties.

In addition, pay attention that each project must have its own unique Hibernate persistence module definition. This can be easily⁵ found here



Error processing

Configuration is in file faces-config.xml. This can be easily found here



There are two exception handlers defined here. First for Ajax calls, second for non- Ajax calls.

1. org.omnifaces.exceptionhandler.FullAjaxExceptionHandlerFactory
2. org.msh.pharmadex.util.ViewExpiredExceptionExceptionHandlerFactory

For Ajax calls, Pharmadex uses standard approach described here

<http://showcase.omnifaces.org/exceptionhandlers/FullAjaxExceptionHandler>

For non-Ajax, Pharmadex uses customized class. The main cause of the customization is to provide error logging to the special database table **error_log**.

Corresponding error pages are defined in web.xml. Pay attention that most errors will redirect the user to the home page. This is because:

1. To continue user's job after error, there is a need to reset Pharmadex to the initial state
2. Error messages with stack traces and programming-specific information are useless for a user
3. All errors are anyways logged to console and database logs

⁵ In addition, it can be found in src\main\webapp\WEB-INF, however Eclipse SRS plugin is very convenient for a Spring application to search for configurations.

String resources and i18n

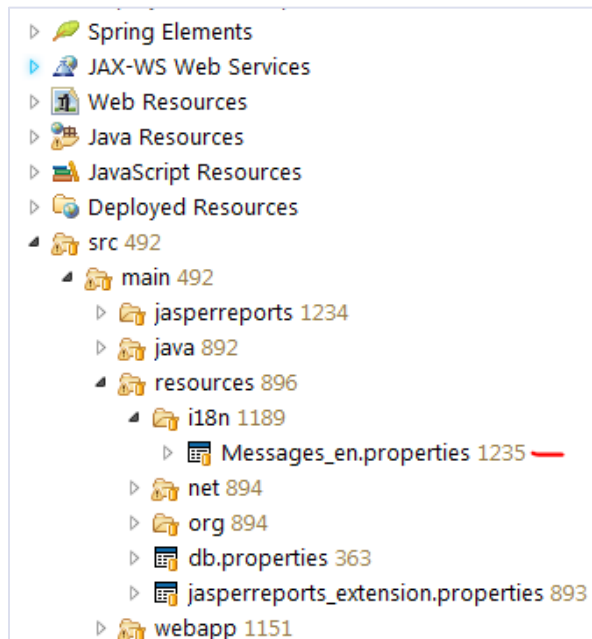
On the first run Pharmadex tries to apply locale from a user's browser. If the browser's locale is unknown for Pharmadex, Pharmadex will not run. For instance, if browser's locale is set to ru_RU, Pharmadex will not run in this browser.

Per the common Spring configuration approach, string resources are defined in faces-config.xml.

Such country specific applications have only en_US locale:

- Bangladesh
- Ethiopia
- Namibia

For these applications, all string resources are in file properties that can be found here



For Mozambique there are two locales, en_US and pt_PT. Internationalization for it is implemented in different way:

- Files for properties exist, but are not used
- Custom resource bundle is defined by class org.msh.pharmadex.auth.DBResourceBundle.
- Control class is org.msh.pharmadex.auth.DBControl.
- Bundles definitions are in a table **resource_bundle**, strings are in a table **resource_message**.

For Generic SDK same approach as for Mozambique will be used.

Internationalization and properties in **core** project are not used.

Authentication and security

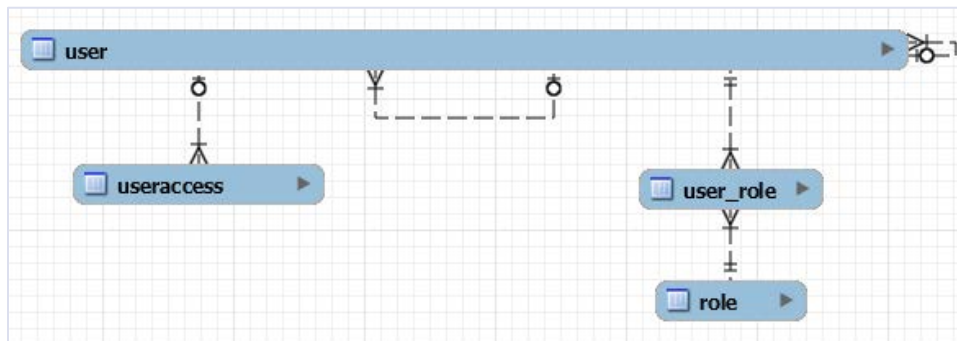
Customized login form can be found in `src/main/webapp/templates/header.xhtml`. It is dialog with backed bean **userAuthHandler** (`org.msh.pharmadex.auth. UserAuthHandler`). This bean dispatches login request to standard Spring's servlet **j_spring_security_check**.

Configuration for this servlet can be found in **core** project, file is

`src/main/webapp/WEB-INF/spring/app/security-context.xml`

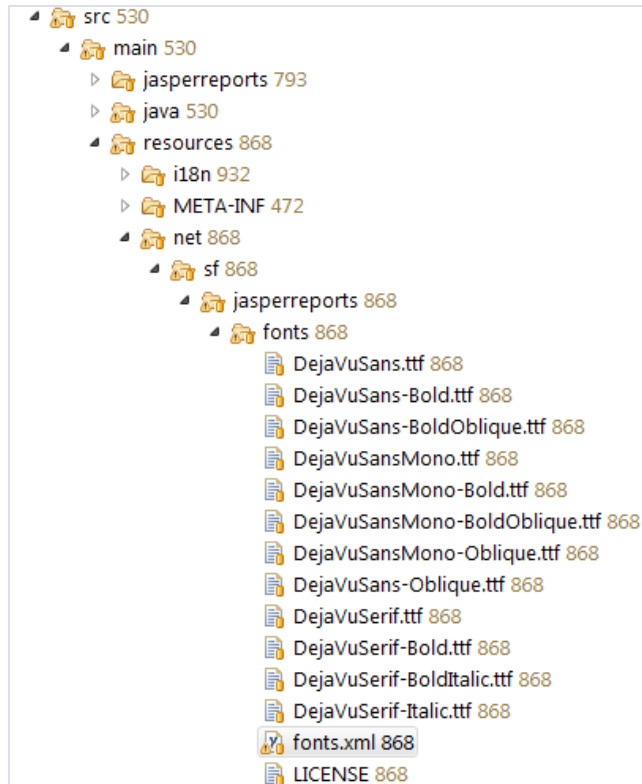
This file is included in `root-context.xml`

As defined in the security-context, authentication manager is implemented by **userDetailsService** (`org.msh.pharmadex.auth. UserDetailsServiceImpl`) service. This service uses user's definitions from the database



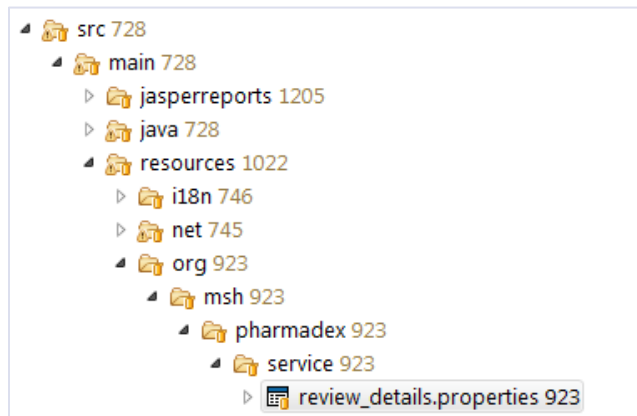
Jasper Report additional configuration

To make letters look better and more formal additional fonts are needed. These can be found here



Refer to the TIBCO Jaspersoft documentation and community.

Review Details report requires additional string resources. These resources are very special and putting them to the common string resources is unwise. These resources can be found in the additional properties file



KNOWN ISSUES

JSF errors

Sometimes fake errors for JSF files may be displayed. For instance for the code below:

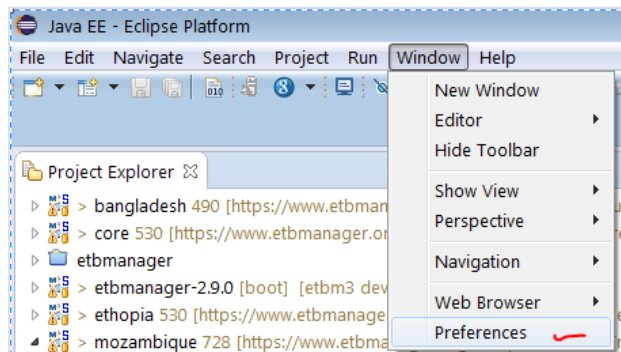
```
<a href="${request.contextPath}/home.faces">   
</a>
```

Error message is

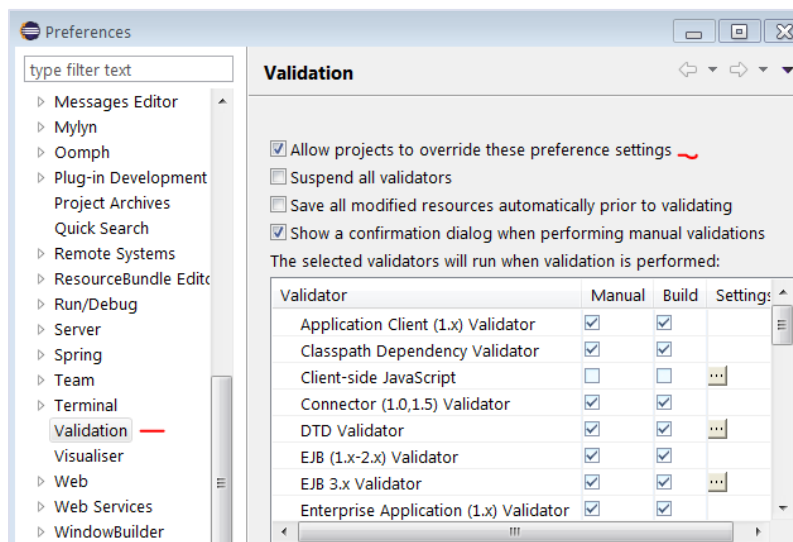
Undefined Image file ("\${request.contextPath}/resources/images/logo_mz.png")

This is not really an error, because variable *request.contextPath* is undefined at compile time. To avoid this kind of message, turn off excess validator:

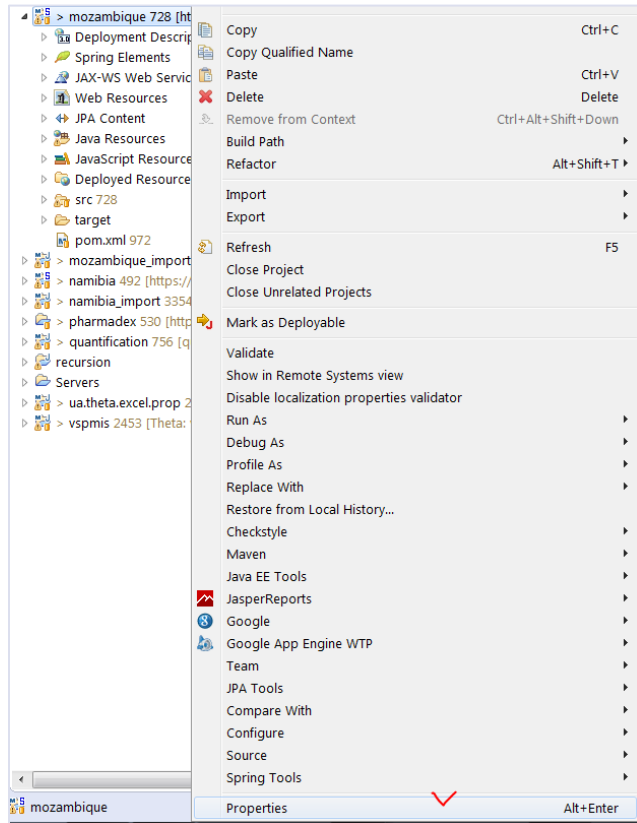
Execute global preferences



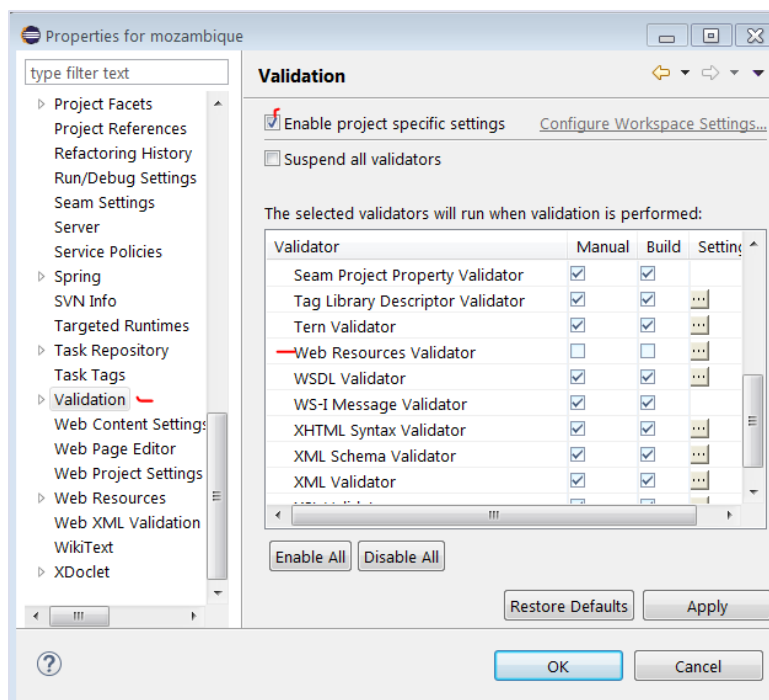
Allow individual validation settings for projects



Select properties for a project



Turn off excess validation



Somewhere on the Internet, there may be recommendations proposing to turn all validators off. Please do not follow it.

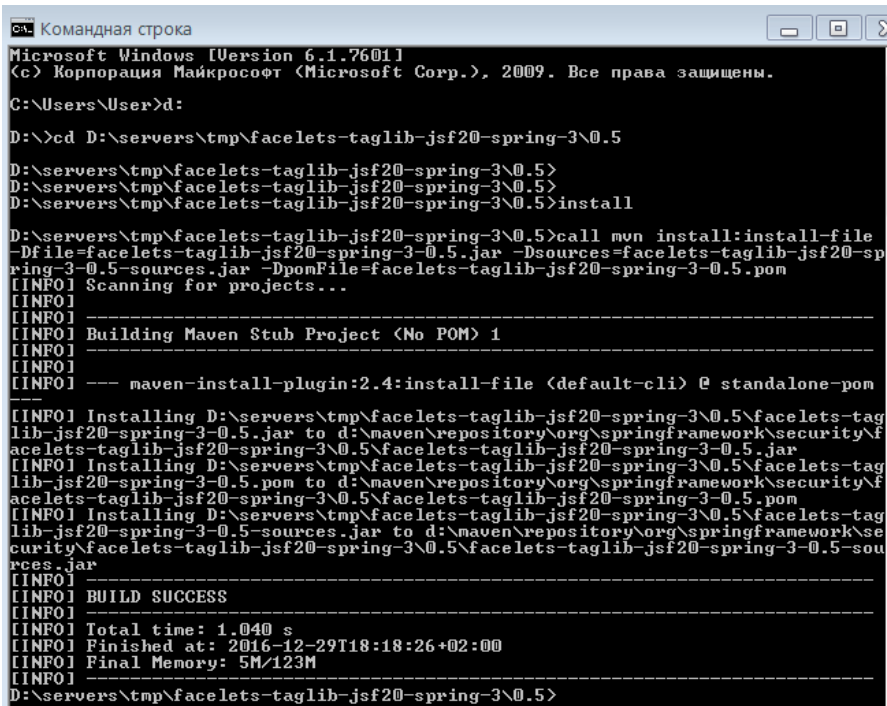
Absence of dependency

Pharmadex depends on the Java library named **facelets-taglib-jsf20-spring-3**. Unfortunately, this library has been recently removed from Maven repository (see <https://github.com/domdorn/spring-security-facelets-taglib/issues/5>).

To fix this issue add the below listed library to the local Maven repository manually. To do this:

1. Ensure that Maven 2 software is installed
2. Copy facelets-taglib-jsf20-spring-3 folder (provided as part of the SDK) to any folder on disk
3. From Command prompt install **this_folder** facelets-taglib-jsf20-spring-3\0.5 as current
4. Run install.bat

Result will be as follows

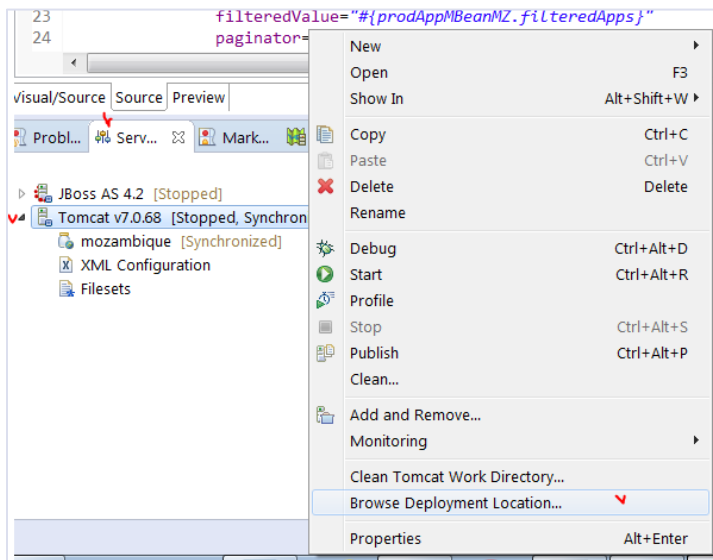


```
Командная строка
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.
C:\Users\User>d:
D:\>cd D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5
D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5>
D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5>
D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5>install
D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5>call mvn install:install-file
-Dfile=facelets-taglib-jsf20-spring-3-0.5.jar -Dsources=facelets-taglib-jsf20-sp
ring-3-0.5-sources.jar -DpomFile=facelets-taglib-jsf20-spring-3-0.5.pom
[INFO] Scanning for projects...
[INFO]
[INFO]
[INFO] Building Maven Stub Project (No POM) 1
[INFO]
[INFO]
[INFO] --- maven-install-plugin:2.4:install-file (default-cli) @ standalone-pom
---
[INFO] Installing D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5\facelets-tag
lib-jsf20-spring-3-0.5.jar to d:\maven\repository\org\springframework\security\fa
celets-taglib-jsf20-spring-3\0.5\facelets-taglib-jsf20-spring-3-0.5.jar
[INFO] Installing D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5\facelets-tag
lib-jsf20-spring-3-0.5.pom to d:\maven\repository\org\springframework\security\fa
celets-taglib-jsf20-spring-3\0.5\facelets-taglib-jsf20-spring-3-0.5.pom
[INFO] Installing D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5\facelets-tag
lib-jsf20-spring-3-0.5-sources.jar to d:\maven\repository\org\springframework\se
curity\facelets-taglib-jsf20-spring-3\0.5\facelets-taglib-jsf20-spring-3-0.5-sou
rces.jar
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 1.040 s
[INFO] Finished at: 2016-12-29T18:18:26+02:00
[INFO] Final Memory: 5M/123M
[INFO]
D:\servers\tmp\facelets-taglib-jsf20-spring-3\0.5>
```

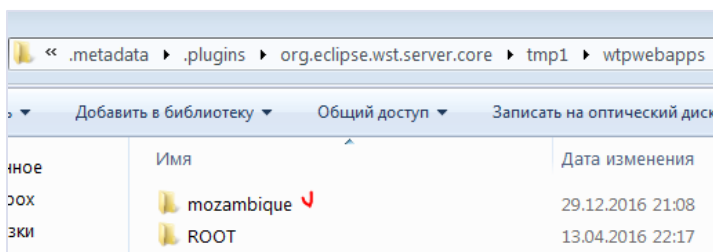
Jasper reports do not load

Sometimes Eclipse does not publish compiled Jasper reports to Tomcat runtime server. To ensure that reports are published, do the following:

Open Servers tab and navigate to Deployment Location



Folder structure as follows will be visible (example for Mozambique application):



Open application folder and go to mozambique\WEB-INF\classes\reports

Should be like this:

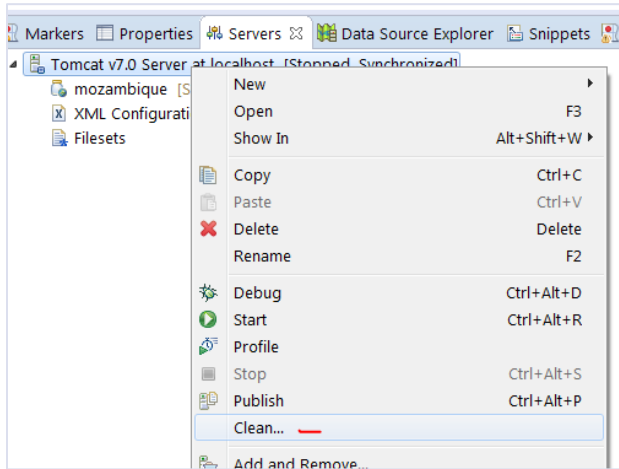
deficiency.jasper	29.12.2016 21:11	Jasper source file
letter.jasper	29.12.2016 21:11	Jasper source file
logo_header.png	29.12.2016 21:08	Рисунок PNG
reg_letter.jasper	29.12.2016 21:11	Jasper source file
rejection_letter.jasper	29.12.2016 21:11	Jasper source file
rev_def_letter.jasper	29.12.2016 21:11	Jasper source file
review_detail_report....	29.12.2016 21:11	Jasper source file
sample_request.jasper	29.12.2016 21:11	Jasper source file

If this folder does not contain *.jasper files, follow next advise “Cleanup of the project”

Cleanup of the project

When any unexpected or unusual issue will rise, it will be a good idea to execute following procedure for cleanup of the project:

1. Stop Tomcat runtime server in Eclipse from Servers tab
2. Run maven clean and maven install commands on the pharmadex project
3. Cleanup deployed codes



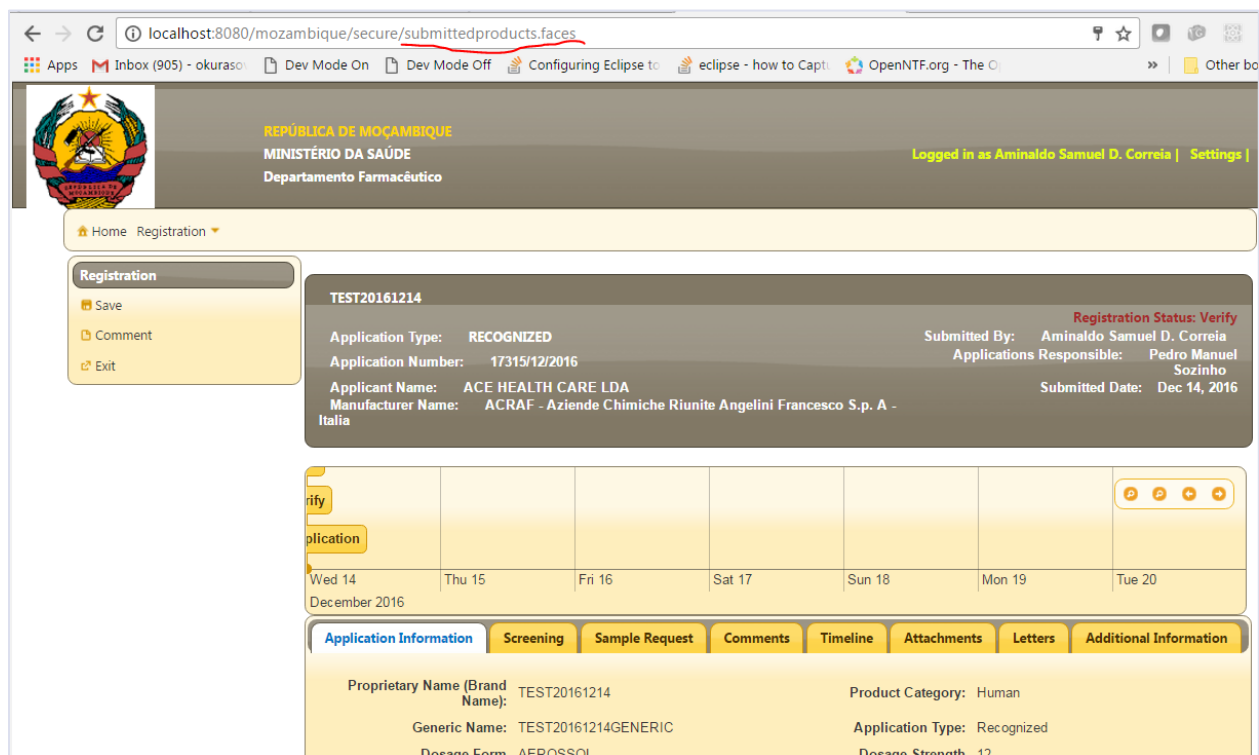
Run Tomcat runtime server in debug mode again and check the issue.

DEVELOPMENT HINTS

Search for JSF page

Usually, the first task for programmer is to figure out right JSF page. It is not a trivial task, because Pharmadex actively uses HTTP POST for navigation. Therefore, address in browser's address line does not always point to a real page on screen. For instance, in address line displays /secure/submittedproducts.faces

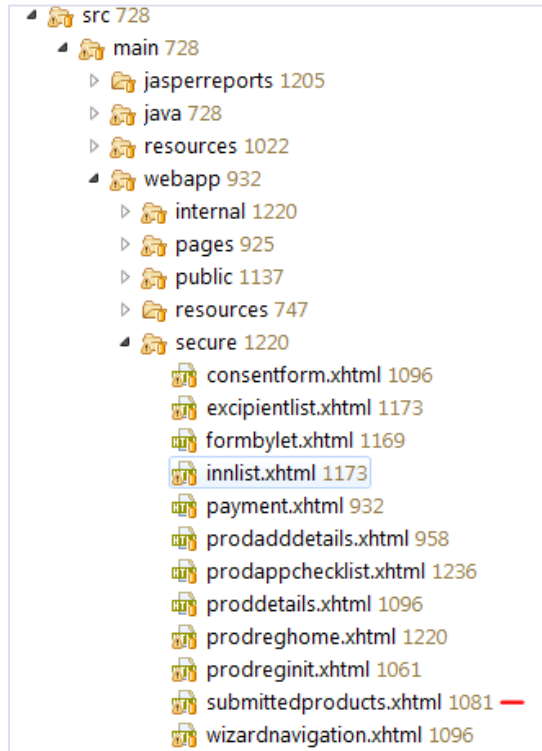
However the real page is different



To determine the real page follow workflow like this:

Search for /secure/submittedproducts.faces in this project, if not found – in core project

Find in this project

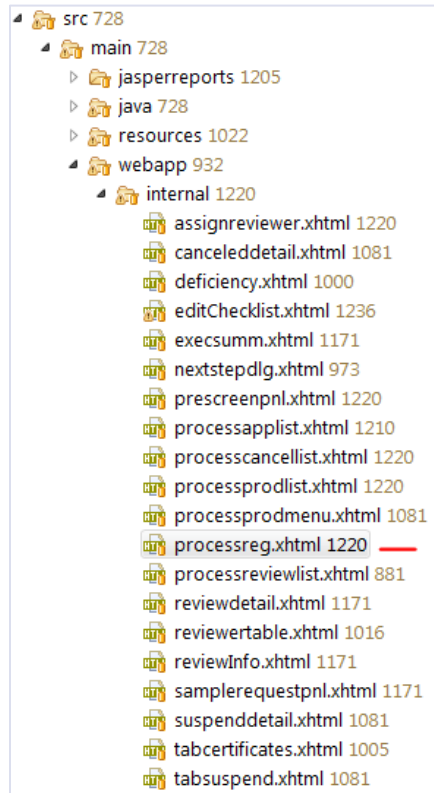


Open /secure/submittedproducts.xhtml and search for right navigation action
It is

```
<p:column sortBy="#{eachprod.product.prodName}"
  headerText="#{msgs.prod_name}" >
  <p:commandLink value="#{eachprod.product.prodName}"
    action="/internal/processreg" ajax="false">
    <f:param name="prodAppID" value="#{eachprod.id}" />
    <f:param name="sourcePage" value="/secure/submittedproducts.xhtml" />
  </p:commandLink>
</p:column>
```

Search for /internal/processreg.xhtml first in this project, if not found here look in **core**.

Find in this project



Codes hot swap

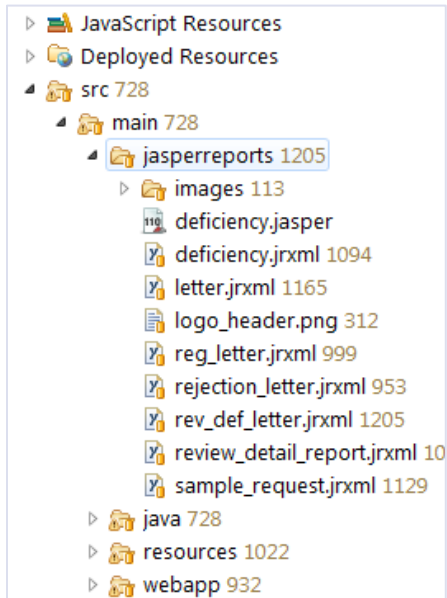
Updated Java code/functionality and/or JSF pages can be deployed when the Tomcat runtime server in Eclipse is running. Typically, “hot swap” works perfectly, except for changes in messages resources and/or configuration files.

Jasper reports

It will be a good idea to have separate Jasper Report Studio software to develop reports. Common workflow is:

1. Copy whole text of jrxml file from the Eclipse project to the Studio
2. Prepare Excel data source for debug purposes and connect it to the Studio
3. Develop and debug in the Studio
4. Copy whole text of result jrxml back to the Eclipse project
5. Perform “Cleanup of the project” from the previous chapter. It is because Eclipse does not automatically compile jrxml files.
6. Finally debug it in Excel environment using the real service and data sources.

All jrxml files can be found here:



Put all images in folder **images**.

Compiled reports will be placed in \WEB-INF\classes\reports folder of the application. Example is:

attachment1.jasper	03.01.2017 17:31
attachment2.jasper	03.01.2017 17:31
deficiency.jasper	03.01.2017 17:31
form8.jasper	03.01.2017 17:31
form9.jasper	03.01.2017 17:31
form13.jasper	03.01.2017 17:31
form16.jasper	03.01.2017 17:31
invoice.jasper	03.01.2017 17:25
letter.jasper	03.01.2017 17:31
logo_header.png	03.01.2017 17:30
logo_header_DGDA.png	03.01.2017 17:30
prod_app_submitted.jasper	03.01.2017 17:25
reg_letter.jasper	03.01.2017 17:31
rejection_letter.jasper	03.01.2017 17:31
report1.jasper	03.01.2017 17:25
rev_def_letter.jasper	03.01.2017 17:31
review_detail_report.jasper	03.01.2017 17:31
sample_request.jasper	03.01.2017 17:31

Refer to Jasper Report Studio documentation for details.

USEFUL LINKS

1. Google search. The most useful (and trusted) source are articles on “stackoverflow.com”
2. <http://www.primefaces.org/> and <http://www.primefaces.org/showcase/>
3. Oracle Java tutorials at <https://docs.oracle.com/javase/tutorial/>
4. www.mysql.com
5. <http://www.jboss.org/projects/> - original documentation portal for Hibernate, Apache tools and a lot of related information
6. Starting point for Jasper Report <https://en.wikipedia.org/wiki/JasperReports>
7. Very good web – related tutorials - <http://www.w3schools.com/>
8. Spring - <https://projects.spring.io/spring-framework/>
9. Maven - <https://maven.apache.org/>