# Spring PetClinic Sample Application

[Build Status](https://travis-ci.org/spring-projects/spring-petclinic/)

## Understanding the Spring Petclinic application with a few diagrams

See the presentation here

## Running petclinic locally

git clone https://github.com/spring-projects/spring-petclinic.git  
 cd spring-petclinic  
 ./mvnw tomcat7:run

You can then access petclinic here: http://localhost:9966/petclinic/

## In case you find a bug/suggested improvement for Spring Petclinic

Our issue tracker is available here: https://github.com/spring-projects/spring-petclinic/issues

## Database configuration

In its default configuration, Petclinic uses an in-memory database (HSQLDB) which gets populated at startup with data. A similar setup is provided for MySql in case a persistent database configuration is needed. Note that whenever the database type is changed, the data-access.properties file needs to be updated and the mysql-connector-java artifact from the pom.xml needs to be uncommented.

You may start a MySql database with docker:

docker run -e MYSQL\_ROOT\_PASSWORD=petclinic -e MYSQL\_DATABASE=petclinic -p 3306:3306 mysql:5.7.8

## Working with Petclinic in Eclipse/STS

### prerequisites

The following items should be installed in your system: \* Maven 3 (http://www.sonatype.com/books/mvnref-book/reference/installation.html) \* git command line tool (https://help.github.com/articles/set-up-git) \* Eclipse with the m2e plugin (m2e is installed by default when using the STS (http://www.springsource.org/sts) distribution of Eclipse)

Note: when m2e is available, there is an m2 icon in Help -> About dialog. If m2e is not there, just follow the install process here: http://eclipse.org/m2e/download/

### Steps:

1. In the command line

* git clone https://github.com/spring-projects/spring-petclinic.git

1. Inside Eclipse

* File -> Import -> Maven -> Existing Maven project

## Looking for something in particular?

<th width="300px">Java Config</th><th width="300px"></th>

<td>Java Config branch</td>  
<td>  
 Petclinic uses XML configuration by default. In case you'd like to use Java Config instead, there is a Java Config branch available <a href="https://github.com/spring-projects/spring-petclinic/tree/javaconfig">here</a>. Thanks to Antoine Rey for his contribution.   
</td>

<th width="300px">Inside the 'Web' layer</th><th width="300px">Files</th>

<td>Spring MVC - XML integration</td>  
<td><a href="/src/main/resources/spring/mvc-view-config.xml">mvc-view-config.xml</a></td>

<td>Spring MVC - ContentNegotiatingViewResolver</td>  
<td><a href="/src/main/resources/spring/mvc-view-config.xml">mvc-view-config.xml</a></td>

<td>JSP custom tags</td>  
<td>  
 <a href="/src/main/webapp/WEB-INF/tags">WEB-INF/tags</a>  
 <a href="/src/main/webapp/WEB-INF/jsp/owners/createOrUpdateOwnerForm.jsp">createOrUpdateOwnerForm.jsp</a></td>

<td>Bower</td>  
<td>  
 <a href="/pom.xml">bower-install maven profile declaration inside pom.xml</a> <br />  
 <a href="/bower.json">JavaScript libraries are defined by the manifest file bower.json</a> <br />  
 <a href="/.bowerrc">Bower configuration using JSON</a> <br />  
 <a href="/src/main/resources/spring/mvc-core-config.xml#L30">Resource mapping in Spring configuration</a> <br />  
 <a href="/src/main/webapp/WEB-INF/jsp/fragments/staticFiles.jsp#L12">sample usage in JSP</a></td>  
</td>

<td>Dandelion-datatables</td>  
<td>  
 <a href="/src/main/webapp/WEB-INF/jsp/owners/ownersList.jsp">ownersList.jsp</a>   
 <a href="/src/main/webapp/WEB-INF/jsp/vets/vetList.jsp">vetList.jsp</a>   
 <a href="/src/main/java/org/springframework/samples/petclinic/PetclinicInitializer.java">PetclinicInitializer.java</a>  
 <a href="/src/main/resources/dandelion/datatables/datatables.properties">datatables.properties</a>

<td>Thymeleaf branch</td>  
<td>  
 <a href="http://www.thymeleaf.org/petclinic.html">See here</a></td>

<td>Branch using GemFire and Spring Data GemFire instead of ehcache (thanks Bijoy Choudhury)</td>  
<td>  
 <a href="https://github.com/bijoych/spring-petclinic-gemfire">See here</a></td>

<th width="300px">'Service' and 'Repository' layers</th><th width="300px">Files</th>

<td>Transactions</td>  
<td>  
 <a href="/src/main/resources/spring/business-config.xml">business-config.xml</a>  
 <a href="/src/main/java/org/springframework/samples/petclinic/service/ClinicServiceImpl.java">ClinicServiceImpl.java</a>  
</td>

<td>Cache</td>  
 <td>  
 <a href="/src/main/resources/spring/tools-config.xml">tools-config.xml</a>  
 <a href="/src/main/java/org/springframework/samples/petclinic/service/ClinicServiceImpl.java">ClinicServiceImpl.java</a>  
</td>

<td>Bean Profiles</td>  
 <td>  
 <a href="/src/main/resources/spring/business-config.xml">business-config.xml</a>  
 <a href="/src/test/java/org/springframework/samples/petclinic/service/ClinicServiceJdbcTests.java">ClinicServiceJdbcTests.java</a>  
 <a href="/src/main/java/org/springframework/samples/petclinic/PetclinicInitializer.java">PetclinicInitializer.java</a>  
</td>

<td>JdbcTemplate</td>  
<td>  
 <a href="/src/main/resources/spring/business-config.xml">business-config.xml</a>  
 <a href="/src/main/java/org/springframework/samples/petclinic/repository/jdbc">jdbc folder</a></td>

<td>JPA</td>  
<td>  
 <a href="/src/main/resources/spring/business-config.xml">business-config.xml</a>  
 <a href="/src/main/java/org/springframework/samples/petclinic/repository/jpa">jpa folder</a></td>

<td>Spring Data JPA</td>  
<td>  
 <a href="/src/main/resources/spring/business-config.xml">business-config.xml</a>  
 <a href="/src/main/java/org/springframework/samples/petclinic/repository/springdatajpa">springdatajpa folder</a></td>

<th width="300px">Others</th><th width="300px">Files</th>

<tr>  
 <td>Spring Boot branch</td>  
 <td>  
 <a href="https://github.com/spring-projects/spring-petclinic/tree/springboot">See here</a></td>  
</tr>

<td>Gradle branch</td>  
<td>  
 <a href="https://github.com/whimet/spring-petclinic">See here</a></td>

## Interaction with other open source projects

One of the best parts about working on the Spring Petclinic application is that we have the opportunity to work in direct contact with many Open Source projects. We found some bugs/suggested improvements on various topics such as Spring, Spring Data, Bean Validation and even Eclipse! In many cases, they've been fixed/implemented in just a few days. Here is a list of them:

<th width="300px">Name</th>  
<th width="300px"> Issue </th>

<td>Spring JDBC: simplify usage of NamedParameterJdbcTemplate</td>  
<td> <a href="https://jira.springsource.org/browse/SPR-10256"> SPR-10256</a> and <a href="https://jira.springsource.org/browse/SPR-10257"> SPR-10257</a> </td>

<td>Bean Validation / Hibernate Validator: simplify Maven dependencies and backward compatibility</td>  
<td>  
 <a href="https://hibernate.atlassian.net/browse/HV-790"> HV-790</a> and <a href="https://hibernate.atlassian.net/browse/HV-792"> HV-792</a>  
 </td>

<td>Spring Data: provide more flexibility when working with JPQL queries</td>  
<td>  
 <a href="https://jira.springsource.org/browse/DATAJPA-292"> DATAJPA-292</a>  
 </td>

<td>Eclipse: validation bug when working with .tag/.tagx files (has only been fixed for Eclipse 4.3 (Kepler)). <a href="https://github.com/spring-projects/spring-petclinic/issues/14">See here for more details.</a></td>  
<td>  
 <a href="https://issuetracker.springsource.com/browse/STS-3294"> STS-3294</a>  
</td>

# Contributing

The [issue tracker](https://github.com/spring-projects/spring-petclinic/issues) is the preferred channel for bug reports, features requests and submitting pull requests.

For pull requests, editor preferences are available in the [editor config](https://github.com/spring-projects/spring-petclinic/blob/master/.editorconfig) for easy use in common text editors. Read more and download plugins at <http://editorconfig.org>.