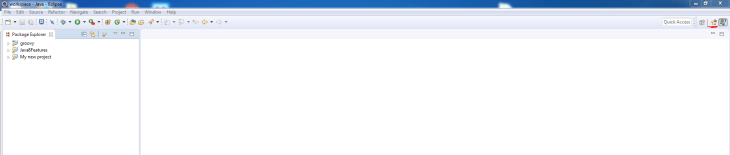
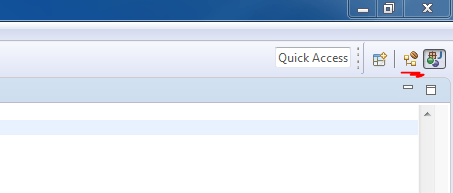
Простой сайт на Java

[Май 26, 2017](https://java-master.com/%d0%bf%d0%b8%d1%88%d0%b5%d0%bc-%d0%b2%d0%b5%d0%b1-%d0%bf%d1%80%d0%b8%d0%bb%d0%be%d0%b6%d0%b5%d0%bd%d0%b8%d1%8f-%d0%bd%d0%b0-java/)  [adminhq](https://java-master.com/author/adminhq/)

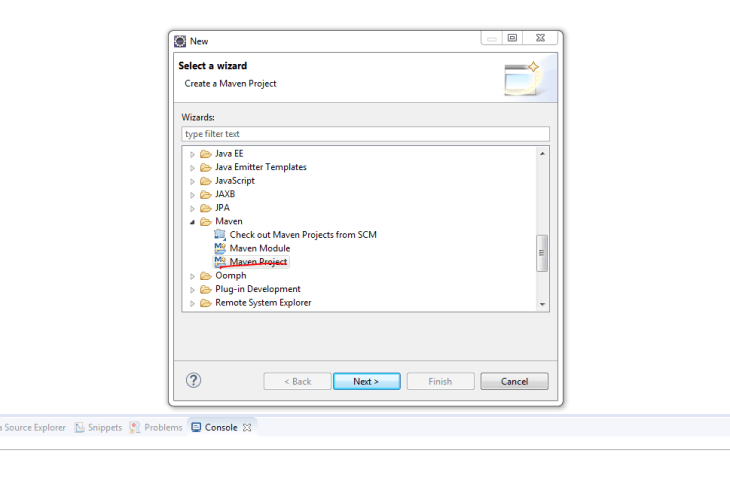
Этой заметкой мы начнем цикл статей посвященных теме веб приложения на языке Java. Здесь Вы узнаете, как написать сайт на Java и как его потом выложить на бесплатный сервис, чтобы Вашим творением могли полюбоваться товарищи.

Для того, чтобы без проблем освоить материал Вам нужны знания [Java core](http://java-master.com/category/java-%D0%B4%D0%BB%D1%8F-%D0%BD%D0%BE%D0%B2%D0%B8%D1%87%D0%BA%D0%B0/) — это основы джава; базовые понятия фронтенда HTML, CSS, JS. Если есть проблемы с фронтендом — не волнуйтесь. Выучите постепенно с джавой)).

Для начала, нам нужно установить [стандартный набор для Java](http://java-master.com/%D0%BD%D0%B0%D1%87%D0%B0%D1%82%D1%8C-%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D1%82%D1%8C-%D0%BD%D0%B0-java/). В Eclipse нужно выбрать перспективу с Java на Java EE в правом верхнем углу.  


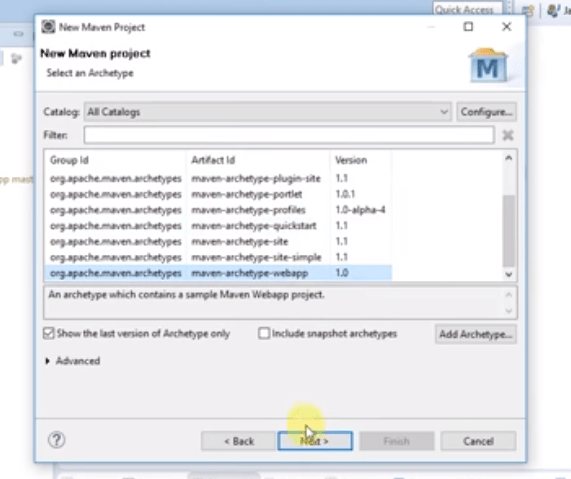


Теперь у Вас в приложении появилось больше выбора для создания файлов. Теперь нужно создать новый Maven проект. File -> new -> other… -> maven -> Maven Project



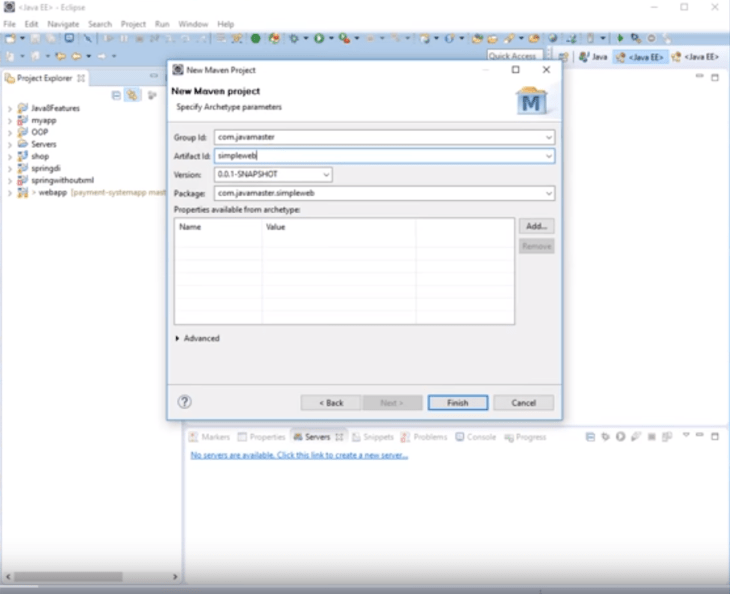
Нажимаем Next и далее нас перебрасывает в окно выбора так называемого архетипа (archetype). Для чего это нужно? Да просто для удобства. Мы используем Maven как инструмент сборки потому, что он очень удобен. Нам не нужно больше искать дополнительные jar библиотеки в Интернете, качать их и подключать. Maven позволяет нам подключать дополнительные библиотеки в специальном файле настроек, который называется pom.xml. Несмотря на его внешнюю запутанность, он очень прост и удобен.

При выборе архетипа нужно выбрать maven-archetype-webapp:

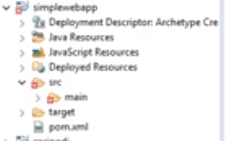


Можно было создать простое java приложение и потом добавить pom.xml, необходимые папки, web.xml; но зачем, если за нас это может сделать выбор архетипа)).

Далее появиться окно ввода Group id, Artifact id. В строку Group id введите com.javamaster, а в строку Artifact id можно ввести например simplewebapp.

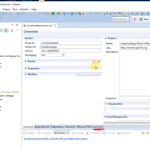


Нажимаем Finish и видим, что в панели проектов нам создало новый проект с названием нашего Artifact id. Если Вы откроете проект, то увидите, что программа создала много непонятных папок и несколько непонятных файлов. Но, это все нам пригодиться.



Не успели мы создать приложение, как уже выдает ошибку. Это не серьезная ошибка и приложение все равно запуститься без нее. Но так как я не люблю красные крестики в проекте нужно добавить одну зависимость в наш проект, чтобы ошибка исчезла. Так мы и познакомимся с pom.xml.

Откройте файл pom.xml, перейдите на вкладку кода: (на картинку можно нажать)

**[](https://java-master.com/%d0%bf%d0%b8%d1%88%d0%b5%d0%bc-%d0%b2%d0%b5%d0%b1-%d0%bf%d1%80%d0%b8%d0%bb%d0%be%d0%b6%d0%b5%d0%bd%d0%b8%d1%8f-%d0%bd%d0%b0-java/editing-pom/)**

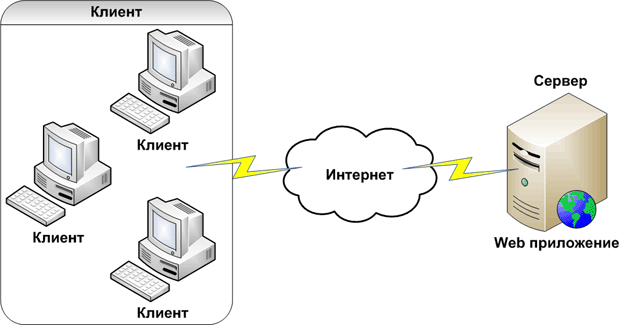
Далее в раздел dependencies (зависимости) нужно добавить:

|  |  |
| --- | --- |
| [Код](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/" \l "codesyntax_1" \o "Нажмите, чтобы показать или скрыть блок кода) | [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/code.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_1) [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_1) [https://i1.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/info.gif?w=730&ssl=1](https://java-master.com/wp-content/plugins/wp-synhighlight/About.html) |

1. **<dependency>**
2. **<groupId>**javax**</groupId>**
3. **<artifactId>**javaee-web-api**</artifactId>**
4. **<version>**6.0**</version>**
5. **<scope>**provided**</scope>**
6. **</dependency>**

Этим действием мы добавили новую библиотеку к нашему проекту. Ничего не нужно скачивать: добавление нескольких строчек в pom.xml укажет вашей программе автоматически загрузить данные библиотеки. Сохраните файл и проблема с ошибкой проекта должна исчезнуть.

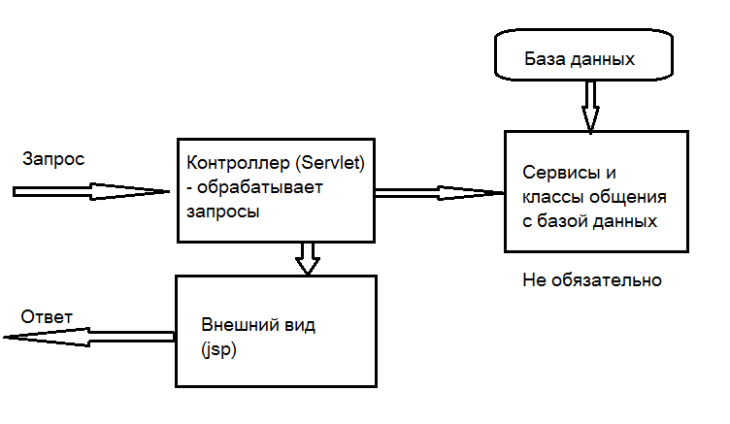
Теперь необходимо рассмотреть как работает веб и в частности, веб приложения.



На рисунке выше представлена схема работы интернет приложений: клиент, то есть Ваш браузер, посылает запрос по определенному адресу и порту; через Интернет запрос доходит до Вашего сервера, на котором находится приложение; обработанный запрос возвращается клиенту в виде ответа; браузер интерпретирует ответ с сервера в понятный для человека вид картинок или текст. Если Вам интересно, что именно передается по сети, советую скачать приложение Fiddler, который поможет Вам увидеть запросы и ответы с Вашего компьютера.

Теперь нужно посмотреть, как обрабатываются запросы самим веб приложением. Мы будем рассматривать пример java приложения, но схема будет работать и для других языков программирования.

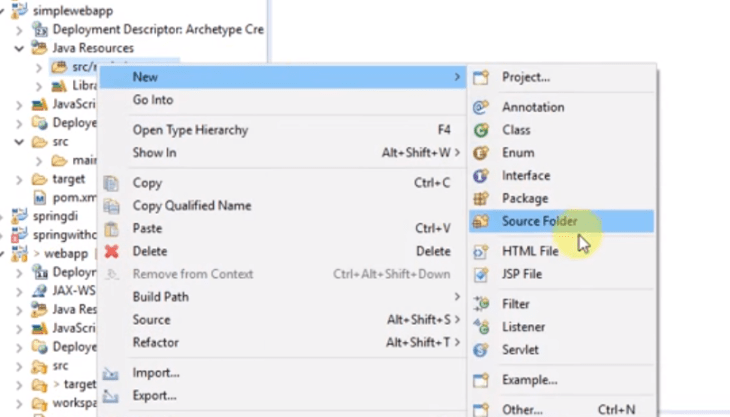
Как мы уже поняли, на сервер приходит запрос далее он должен как-то оповестить о запросе наше приложение и наше приложение должно распознать этот запрос, проделать действия и отослать ответ.



Не самая красивая, но достаточно точная схема работы веб программы. Для распознавания запросов в Java есть такой механизм как сервлет (Servlet) — он может определить строку запроса и перенаправить его на jsp — технология, которая позволяет динамически генерировать веб страницы. По сути jsp очень похож на HTML с тем отличием, что в нем можно запускать Java код. Мы этого делать не будем, так как эта технология считается не самой лучшей. Детальнее о jsp будет в следующих статьях.

Сервлет — это класс, который унаследован от HttpServlet. В классе HttpServlet есть несколько методов по обработке запросов. Нас пока будут интересовать doGet и doPost, который обрабатывают соответственно гет и пост запросы. Если не знаете, что это такое — советую загуглить.

В нашем проекте нужно найти папку src/main/java. Если таковой нет, нужно ее создать: правой кнопкой на src -> new -> source folder.



Далее нужно добавить в эту папку новый [пакет](http://java-master.com/%D0%BF%D0%B0%D0%BA%D0%B5%D1%82%D1%8B-packages-in-java/)(Вы же не хотите помещать все классы в одной папке). Я назвал у себя: com.javamaster.controller

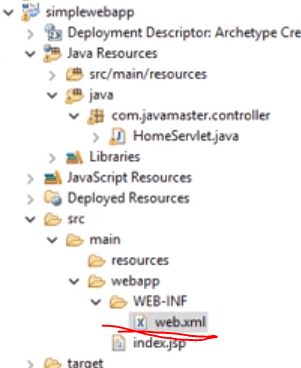
Теперь есть две возможность создать сервлет: создать класс и унаследовать его от HttpServlet и потом переопределить нужные нам методы или создать сервлет (на картинке выше можно увидеть, что система сама нам предлагает создать servlet). Разницы нет никакой. Если Вы выберете автоматическое создание, то получите то же самое. Вот мой сервлет, который я назвал HomeServlet (можно выбирать любое название).

|  |  |
| --- | --- |
| [Код](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/" \l "codesyntax_2" \o "Нажмите, чтобы показать или скрыть блок кода) | [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/code.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_2) [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_2) [https://i1.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/info.gif?w=730&ssl=1](https://java-master.com/wp-content/plugins/wp-synhighlight/About.html) |

1. ackage com.javamaster.controller;
3. **import** java.io.IOException;
5. **import** javax.servlet.ServletException;
6. **import** javax.servlet.http.HttpServlet;
7. **import** javax.servlet.http.HttpServletRequest;
8. **import** javax.servlet.http.HttpServletResponse;
10. **public** **class** HomeServlet **extends** HttpServlet {
12. @Override
13. **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws**ServletException, [IOException](http://www.google.com/search?hl=en&q=allinurl%3Aioexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) {
15. }
16. }

В нем только один метод: doGet на данном этапе этого достаточно.

Теперь нужно дать нашей программе понять, что класс HomeServlet является контроллером и что все запросы нужно отправлять на него.

Для этого у нас есть web.xml — файл настроек нашей программы. Его можно найти в папке WEB-INF.

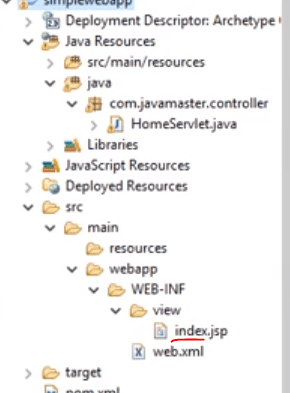
В моем случае системой был сгенерирован немного не такой файл, как мне нужно. Я замели его следующим содержимым:

|  |  |
| --- | --- |
| [Код](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/" \l "codesyntax_3" \o "Нажмите, чтобы показать или скрыть блок кода) | [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/code.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_3) [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_3) [https://i1.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/info.gif?w=730&ssl=1](https://java-master.com/wp-content/plugins/wp-synhighlight/About.html) |

1. **<?xml** version="1.0" encoding="UTF-8"**?>**
2. **<web-app** xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3. xmlns="http://java.sun.com/xml/ns/javaee"
4. xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_3\_0.xsd"
5. version="3.0"**>**
6. **<display-name>**Archetype Created Web Application**</display-name>**
8. **<servlet>**
9. **<description></description>**
10. **<display-name>**HomeServlet**</display-name>**
11. **<servlet-name>**HomeServlet**</servlet-name>**
12. **<servlet-class>**com.javamaster.controller.HomeServlet**</servlet-class>**
13. **</servlet>**
14. **<servlet-mapping>**
15. **<servlet-name>**HomeServlet**</servlet-name>**
16. **<url-pattern>**/**</url-pattern>**
17. **</servlet-mapping>**
18. **</web-app>**

Если присмотреться, то тут все достаточно просто: мы указали что передаем управление урл путями нашему классу HomeServlet.

Теперь осталось добавить файл, который будет отвечать за внешний вид. Система уже создала нам файл index.jsp и проделала всю работу за нас.

Добавьте новую папку view в папку WEB-INF. Когда в нас появиться много файлов, чтобы не запутаться мы поместим все файлы, который отвечают за внешний вид в папку view. Переместите файл index.jsp в только что созданную папку.

Теперь осталось отловить запрос и перенаправить его на нашу страницу index.jsp. Для этого в сервлете метод doGet нужно немного изменить.

|  |  |
| --- | --- |
| [Код](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/" \l "codesyntax_4" \o "Нажмите, чтобы показать или скрыть блок кода) | [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/code.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_4) [https://i2.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/printer.png?w=730&ssl=1](https://java-master.com/%D0%BF%D0%B8%D1%88%D0%B5%D0%BC-%D0%B2%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D1%8F-%D0%BD%D0%B0-java/#codesyntax_4) [https://i1.wp.com/java-master.com/wp-content/plugins/wp-synhighlight/themes/default/images/info.gif?w=730&ssl=1](https://java-master.com/wp-content/plugins/wp-synhighlight/About.html) |

1. **protected** **void** doGet(HttpServletRequest request, HttpServletResponse response) **throws**ServletException, [IOException](http://www.google.com/search?hl=en&q=allinurl%3Aioexception+java.sun.com&btnI=I%27m%20Feeling%20Lucky) {
2. [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) path = request.getServletPath();
3. **if** (path.equals("/")){
4. request.getRequestDispatcher("/WEB-INF/view/index.jsp").forward(request, response);
5. }
6. **else** **if** (path.equals("/welcome")){
7. request.getRequestDispatcher("/WEB-INF/view/welcome.jsp").forward(request, response);
8. }
9. }

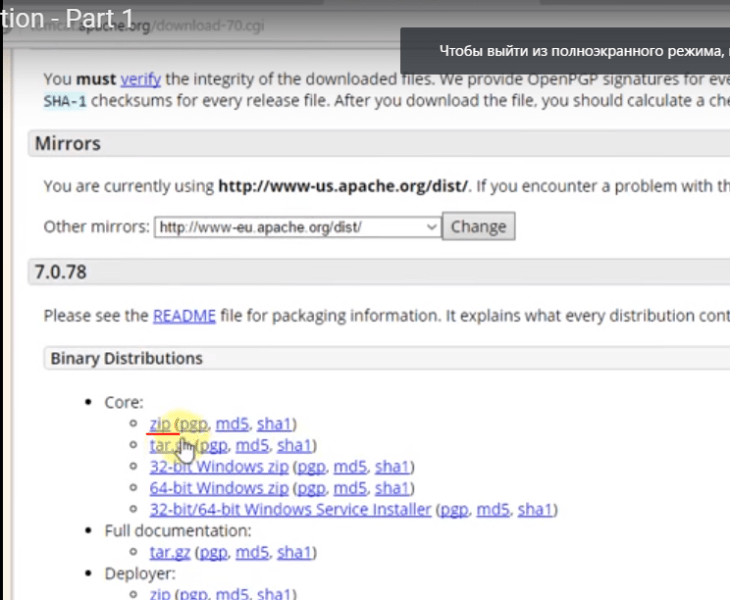
Думаю, код интуитивно понятен: если запрос равен /, тогда указываем путь, куда перенаправить запрос: на страницу индекс, если же запрос равен /welcome тогда, перенаправить на страницу приветствия. О HttpServletRequest подробнее поговорим в следующей части.

Теперь осталось только запустить наше приложение. Потерпите осталось недолго.

Нам нужен контейнер сервлетов, который сможет управлять нашей программой и который реализует Java Servlets и Java Server Pages (jsp) спецификации. Для наших нужд отлично подойдет Apache Tomcat — он бесплатный и очень удобный в использовании.

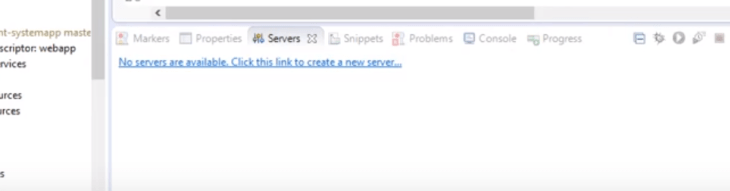
Apache Tomcat нужно скачать с официального сайта, распаковать и указать в настройках Eclipse путь к распакованному архиву. Сделаем все по порядку.

Качаем архив с сайта:



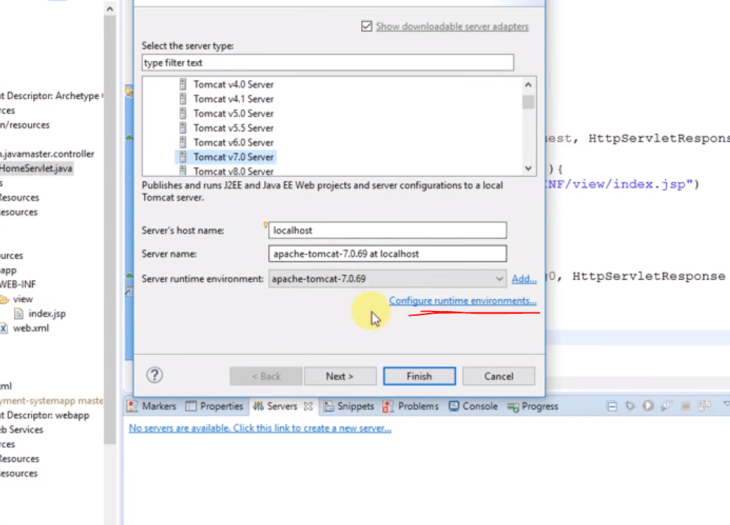
Распаковываем его в желаемую папку на диске.

Переходим в Eclipse и во вкладке сервера нажимаем создать новый:



Если такой вкладки у Вас нет, ее нужно добавить: Window -> View -> Servers

Теперь нужно выбрать нужный нам вариант из предложенных серверов, указать путь к нашему и готово.



Переходим к нашему проекту. В разделе Project explorer нужно нажать правой клавишей мыши и выбрать: run as -> run on server. Далее в окошке выбрать добавленный сервер подождать, пока проект запуститься и наслаждаться своим сайтом. Если у Вас нет опции  run on server, ее нужно добавить: run as -> run configuration -> run on server.



Вот вроде и все. Небольшая заметка превратилась в полноценную статью. И это только для запуска хело ворд приложения. Далее будет интереснее и проще)). Я записал видео, где можно посмотреть весь процесс создания этого приложения:

Код на гитхаб можно найти в описании к видео.