

Tomcat servlet container

Regular meeting for get knowledge and tech in
EDUC

- контейнер сервлетов" - что и зачем
- Apache Tomcat - обзор и другие реализации
- Где взять
- Как запустить
- Как проверить
- Создание Приложения
- Деплой WAR
- Проверка

Контейнер сервлетов — программа, представляющая собой сервер, который занимается системной поддержкой

сервлетов и обеспечивает их жизненный цикл в соответствии с правилами.

Servlet – в первую очередь это простой Java интерфейс, реализация которого расширяет функциональные возможности сервера.

контейнер сервлетов управляет четырьмя фазами жизненного цикла сервлета:

ЖИЗНЕННЫЙ ЦИКЛ

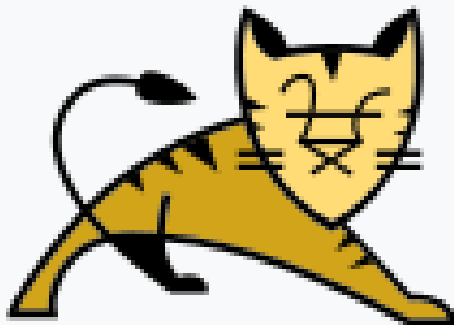
Загрузка класса сервлета — когда контейнер получает запрос для сервлета, то происходит загрузка класса сервлета в память и вызов конструктора без параметров.

Инициализация класса сервлета — это и есть момент когда код преобразуется из обычного класса в сервлет. путем вызова `init()` метода.

Обработка запросов — после инициализации сервлет готов к обработке запросов.

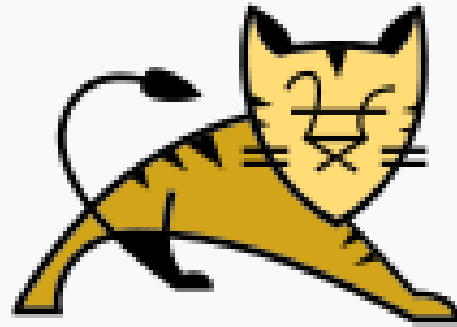
Удаление — когда контейнер останавливается или останавливается приложение, то контейнер сервлетов уничтожает классы сервлетов путем вызова `destroy()` метода.

Apache Tomcat



- Tomcat (в старых версиях — Catalina) — контейнер сервлетов с открытым исходным кодом, разрабатываемый Apache Software Foundation.
- Реализует спецификацию сервлетов, спецификацию JavaServer Pages (JSP) и JavaServer Faces (JSF). Написан на языке Java.
- Известные реализации: Apache Tomcat, Jetty, JBoss, GlassFish, IBM WebSphere, Oracle Weblogic.

Apache Tomcat

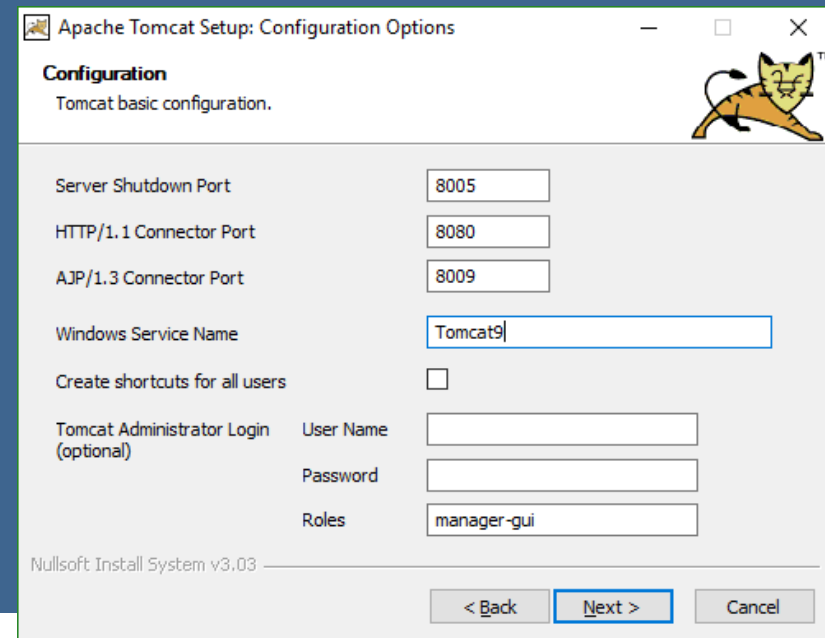
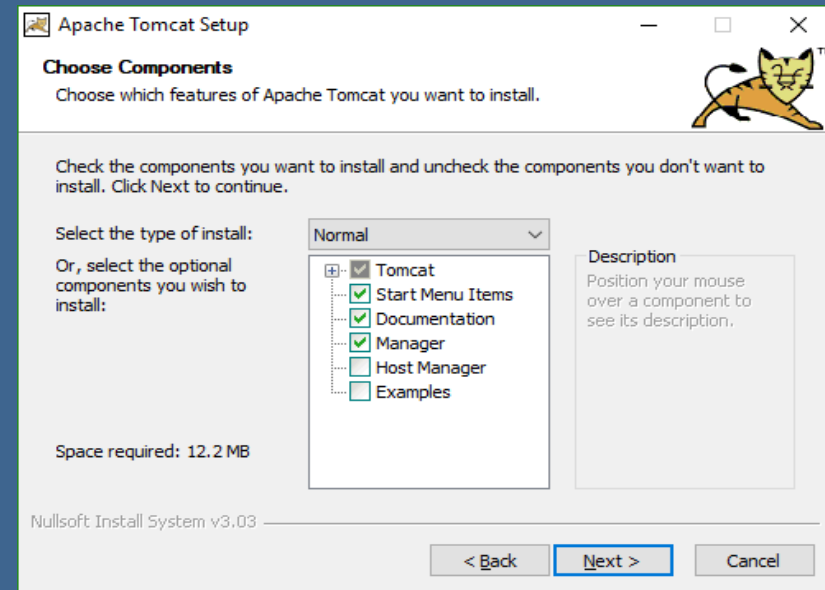
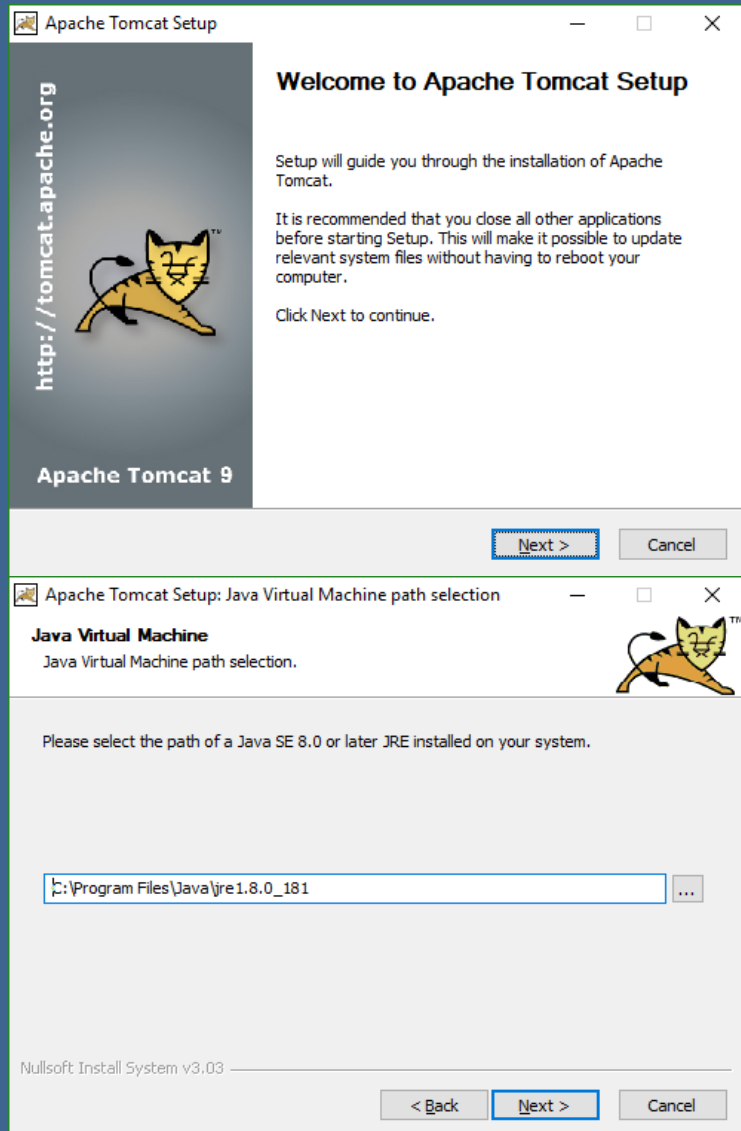


- 1 Переходим на сайт <https://tomcat.apache.org/>
- 2 Скачиваем установочный файл.

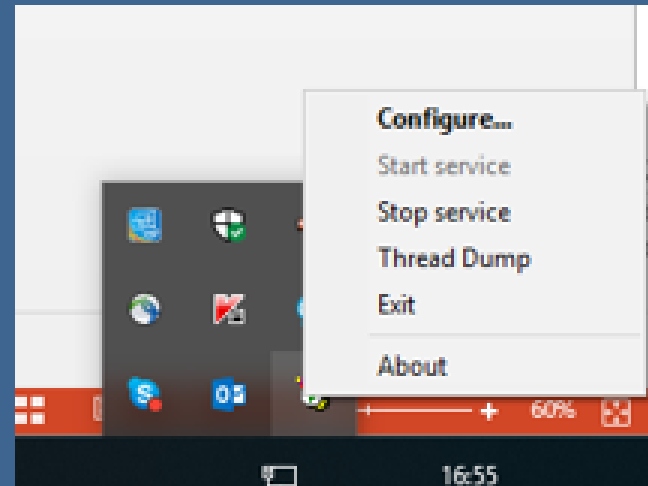
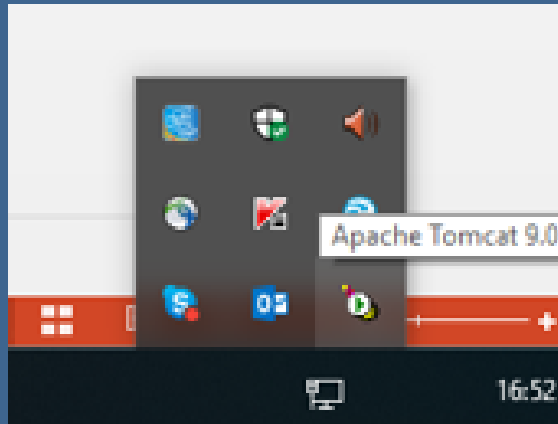


apache-tomcat-9.0.12.exe

• 3 Устанавливаем Tomcat.

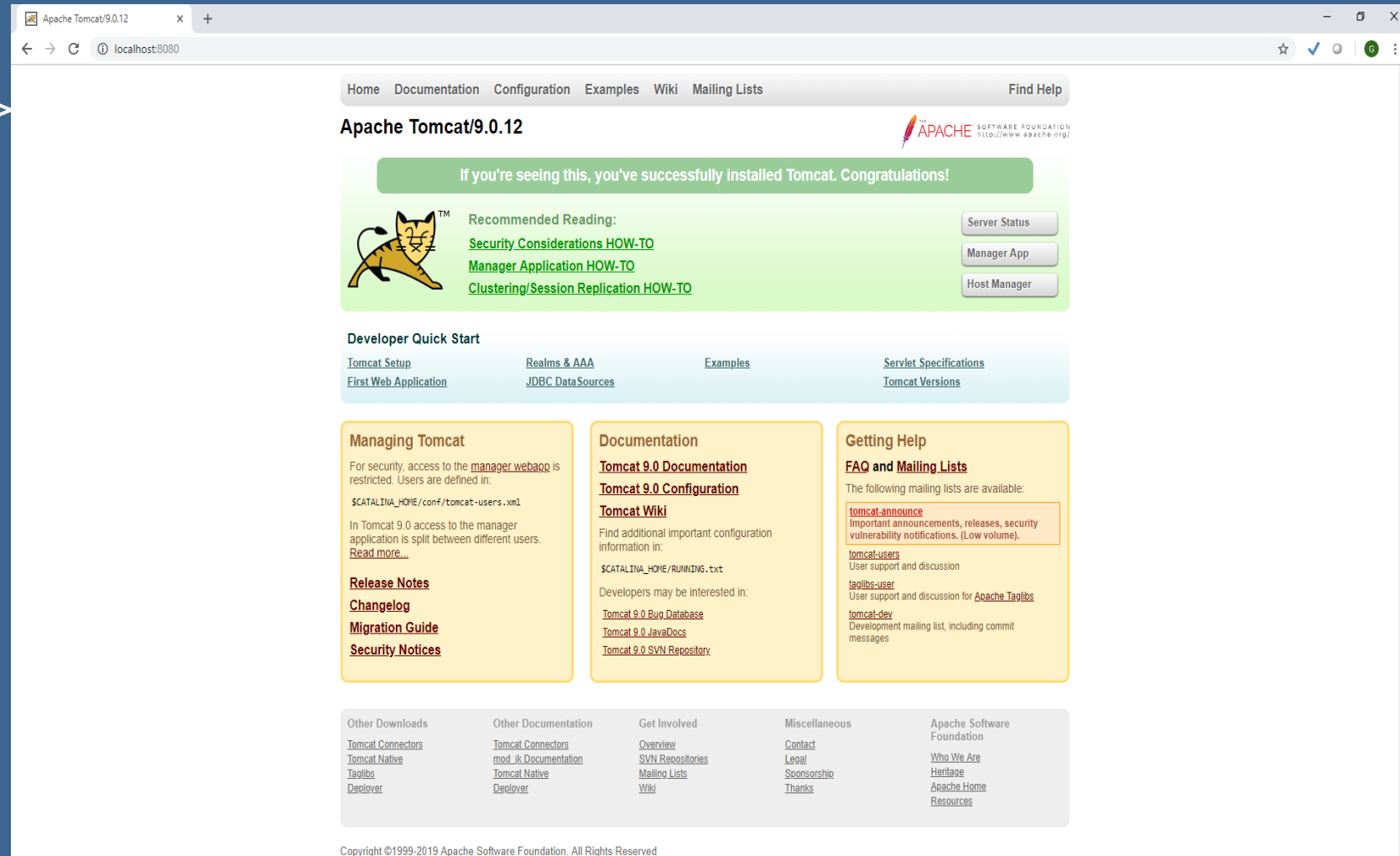


- 4 После этого в трее должен появиться значок запущенного сервиса.

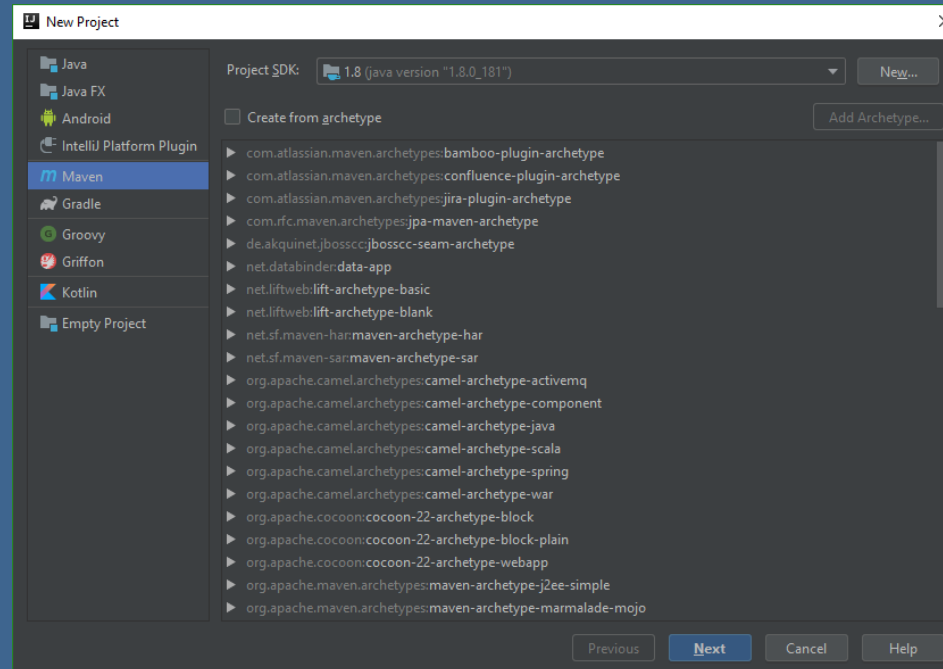


- Также через этот значок можно останавливать / запускать

- 5 Перейдите по адресу
- <http://localhost:8080/>.
- Если вы видите это =>
- то всё хорошо 😊

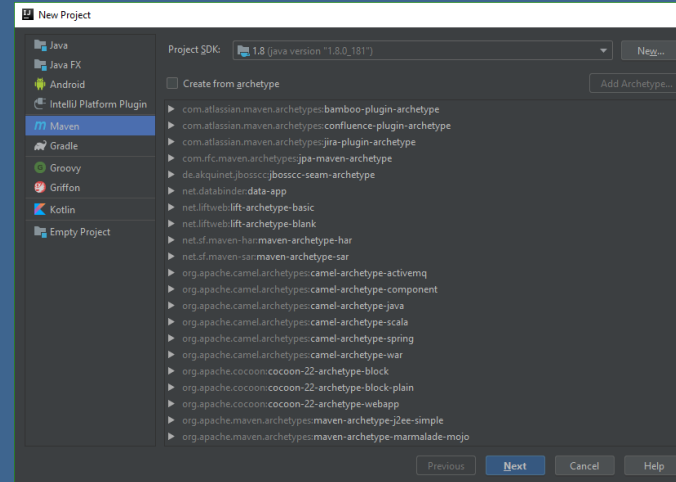
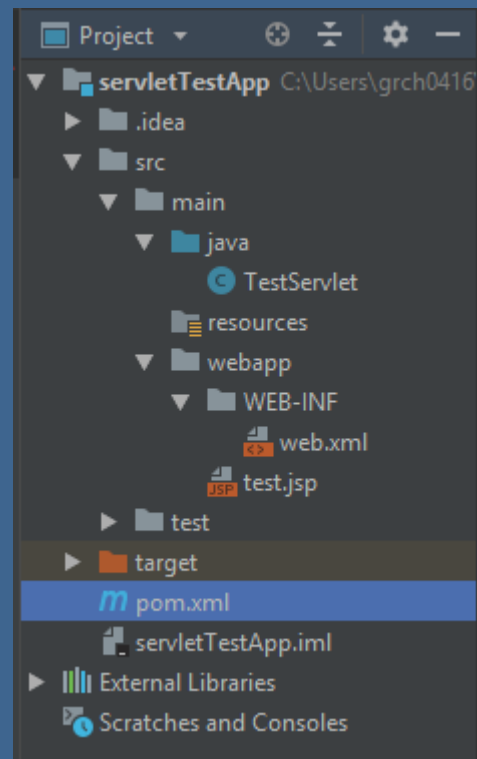


- Пример простого проекта
- Создать Maven Project



- Пример простого проекта

Создать Maven Project



Структура проекта

POM.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>servletTest</groupId>
  <artifactId>st</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>war</packaging>

  <dependencies>

  <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
  <dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>javax.servlet-api</artifactId>
    <version>3.1.0</version>
    <scope>provided</scope>
  </dependency>

  </dependencies>

</project>
```

```

TestServlet.java ×
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

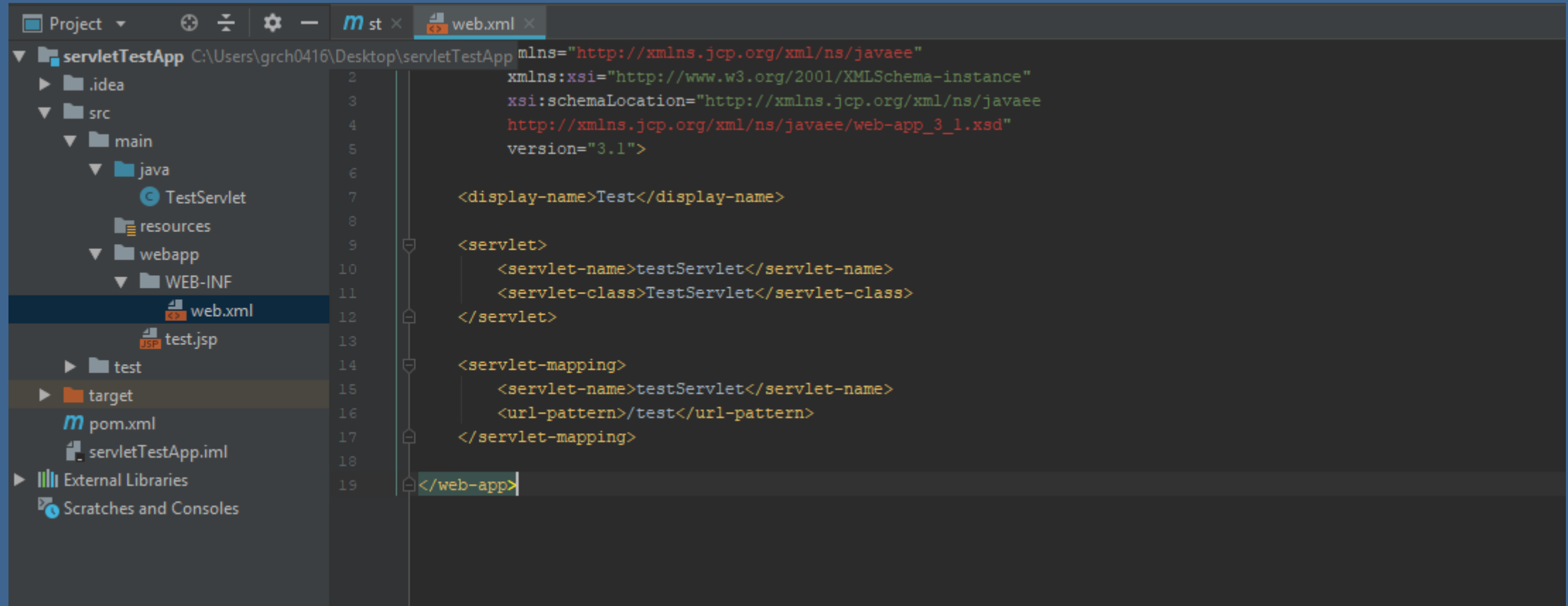
public class TestServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setContentType( "text/html" );
        PrintWriter out = response.getWriter();
        out.println( "<html><head>" );
        out.println( "<title>A Sample Servlet!</title>" );
        out.println( "</head>" );
        out.println( "<body>" );
        out.println( "<form method='get' action='/st/test'>" );
        out.println( "<input type='text' name='a' />" );
        out.println( "<input type='text' name='b' />" );
        out.println("<input type='submit' value='Show text' />");
        out.println("</form>");

        out.println( "<p>Param A " + request.getParameter( "a" )+ "</p>" );
        out.println( "<p>Param B " + request.getParameter( "b" )+ "</p>" );
        out.println( "</body></html>" );
        out.close();
    }
}

```

WEB.XML



```
1<?xml version="1.0" encoding="UTF-8"?>
2  xmlns="http://xmlns.jcp.org/xml/ns/javaee"
3  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
5    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
6  version="3.1">
7
8    <display-name>Test</display-name>
9
10   <servlet>
11     <servlet-name>testServlet</servlet-name>
12     <servlet-class>TestServlet</servlet-class>
13   </servlet>
14
15   <servlet-mapping>
16     <servlet-name>testServlet</servlet-name>
17     <url-pattern>/test</url-pattern>
18   </servlet-mapping>
19 </web-app>
```

Создание WAR - Web Archive или Web Application Resource

The screenshot shows an IDE interface for creating a WAR file. The main editor displays the `TestServlet.java` file with the following code:

```
1 import javax.servlet.ServletException;
2 import javax.servlet.http.HttpServlet;
3 import javax.servlet.http.HttpServletRequest;
4 import javax.servlet.http.HttpServletResponse;
5 import java.io.IOException;
6 import java.io.PrintWriter;
7
8 public class TestServlet extends HttpServlet {
9
10     @Override
11     protected void doGet(HttpServletRequest request, HttpServletResponse response)
12         throws ServletException, IOException {
13         response.setContentType("text/html");
14         PrintWriter out = response.getWriter();
15         out.println("<html><head>");
16         out.println("<title>A Sample Servlet!</title>");
17         out.println("</head>");
18         out.println("<body>");
19         out.println("<form method='get' action='/st/test'>");
20         out.println("<input type='text' name='a' />");
21         out.println("<input type='text' name='b' />");
22         out.println("<input type='submit' value='Show text' />");
23         out.println("</form>");
24
25         out.println("<p>Param A " + request.getParameter("a") + "</p>");
26         out.println("<p>Param B " + request.getParameter("b") + "</p>");
27         out.println("</body></html>");
28         out.close();
29     }
30 }
31
```

The left sidebar shows the project structure for `servletTestApp`, including `src/main/java`, `resources`, `webapp`, `WEB-INF`, `test`, and `target`. The `target` directory is highlighted. The right sidebar shows the Maven lifecycle with the `install` phase selected.

Создание WAR - Web Archive или Web Application Resource

The screenshot displays the IntelliJ IDEA IDE interface for a project named 'servletTestApp'. The main editor shows the 'TestServlet.java' file, which implements the 'HttpServlet' interface. The code includes imports for 'ServletException', 'HttpServlet', 'HttpServletRequest', 'HttpServletResponse', 'IOException', and 'PrintWriter'. The 'doGet' method is overridden to send an HTML response with a form and two parameters, 'A' and 'B'.

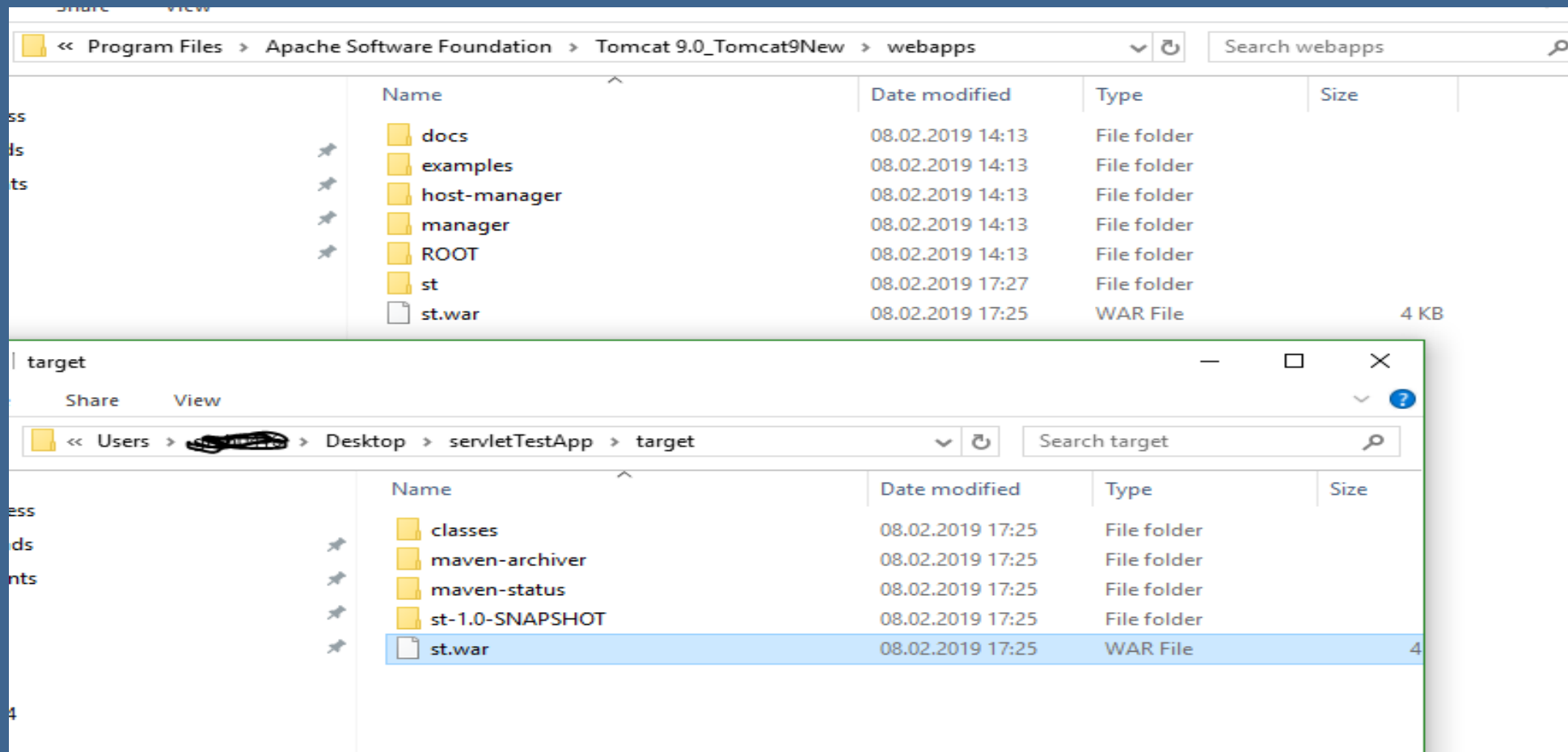
The left sidebar shows the project structure, including 'src/main/java', 'resources', 'webapp', 'WEB-INF', and 'target'. The 'target' directory is expanded, showing 'classes', 'maven-archiver', 'maven-status', 'st-1.0-SNAPSHOT', and 'st-1.0-SNAPSHOT.war'. The 'pom.xml' file is also visible.

The right sidebar shows the 'Maven Projects' panel, with the 'Lifecycle' tab selected. The 'install' goal is highlighted under the 'Lifecycle' section.

The bottom panel shows the 'Run' configuration for 'st [clean,install]'. The output log displays the following information:

```
[INFO] --- maven-install-plugin:2.4:install (default-install) @ st ---
[INFO] Installing C:\Users\grch0416\Desktop\servletTestApp\target\st-1.0-SNAPSHOT.war to C:\Users\grch0416\.m2\repository\servletTest\st-1.0-SNAPSHOT\st-1.0-SNAPSHOT.war
[INFO] Installing C:\Users\grch0416\Desktop\servletTestApp\pom.xml to C:\Users\grch0416\.m2\repository\servletTest\st-1.0-SNAPSHOT\st-1.0-SNAPSHOT.pom
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 3.603 s
[INFO] Finished at: 2019-02-08T17:25:22+04:00
[INFO] Final Memory: 13M/31M
[INFO]
Process finished with exit code 0
```


Деплой WAR Web Archive или Web Application Resource



- 1 скопировать WSR архив
- 2 перезапустить сервер



Tomcat Web Application Manager

Message: OK

Manager

[List Applications](#) [HTML Manager Help](#) [Manager Help](#) [Server Status](#)

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/test	None specified	Test	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy

Deploy directory or WAR file located on server

