

# テスト用アプリを作成して接続確認

## 1 ・ testApp アプリの構成

```
□ /opt/tomcat
└─□ webapps
    └─■ testApp
        └─■ WEB-INF
            ├───■ java
            │   └─◆ TestServ.java
            ├───■ classes
            │   └─◆ TestServ.class
            └─◆ web.xml
```

/opt/tomcat 配下に、  
左図の構成で testApp を配置する

□ ■ ディレクトリ  
◇ ◆ ファイル  
■ ◆ 部を新規で作成

## 2 ・ コンテキストルート(アプリ最上位ディレクトリ)からディレクトリを作成していく

`cd /opt/tomcat/webapps` で webapps ディレクトリへ移動

`mkdir testApp; cd testApp` でコンテキストルートを作成し、そのディレクトリへ移動

```
[root@localhost ~]# cd /opt/tomcat/webapps
[root@localhost webapps]# mkdir testApp; cd testApp
[root@localhost testApp]#
```

`mkdir WEB-INF; cd WEB-INF` でディレクトリを作成し、そのディレクトリへ移動

```
[root@localhost testApp]# mkdir WEB-INF; cd WEB-INF
[root@localhost WEB-INF]#
```

`mkdir java classes` で2つのディレクトリを作成

`ls` で確認

```
[root@localhost WEB-INF]# mkdir java classes
[root@localhost WEB-INF]# ls
classes  java
```

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class TestServ extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public void doGet(HttpServletRequest req, HttpServletResponse res)
        throws IOException, ServletException {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Test</title>");
        out.println("</head>");
        out.println("<body>");
        try {
            out.println(conDB());
        } catch (Exception e) {
            out.println("DB 接続失敗 : " + e);
        }
        out.println("</body>");
        out.println("</html>");
    }

    private String conDB() throws Exception {
        StringBuilder sb = new StringBuilder();
        ResultSet rs = null;
        Statement stmt = null;
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        Connection con = DriverManager.getConnection(
            "jdbc:mysql://localhost/test_db", "root", "AT8_MySQL");
        stmt = con.createStatement();
        rs = stmt.executeQuery("select * from test_tb");
        while (rs.next()) {
            sb.append(rs.getString("name"));
        }
        rs.close();
        stmt.close();
        return sb.toString();
    }
}
```

#### 4・Servlet をコンパイルして classes ディレクトリへ配置

`find / -name servlet-api.jar` で `servlet-api` の所在を確認

```
[root@localhost WEB-INF]# find / -name servlet-api.jar
/opt/tomcat/lib/servlet-api.jar
```

```
javac -classpath ¥                                servlet-api のパスを指定してコンパイル
/opt/tomcat/lib/servlet-api.jar ¥
java/TestServ.java; ls java                      「ls」でjavaディレクトリ内を確認
```

```
[root@localhost WEB-INF]# javac -classpath ¥
> /opt/tomcat/lib/servlet-api.jar ¥
> java/TestServ.java; ls java
TestServ.class  TestServ.java
```

```
mv java/TestServ.class classes; ls java classes
```

で、class ファイルの移動と確認

```
[root@localhost WEB-INF]# ¥
> mv java/TestServ.class classes; ls java classes
classes:
TestServ.class

java:
TestServ.java
```

## 5・配備記述子（web.xml）の作成

## 1. Tomcat の web.xml を確認

vi /opt/tomcat/conf/web.xml      で、`<web-app ~>`タグのコードを確認

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

---

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
  version="3.1">
```

```
<!-- ===== Introduction ===== -->
```

## 2. アプリの web.xml を作成

vi web.xml で以下の内容を記載

```
<?xml version="1.0" encoding="UTF-8" ?>
```

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
  version="3.1">
```

```
  <servlet>
    <servlet-name> TestServ </servlet-name>
    <servlet-class> TestServ </servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name> TestServ </servlet-name>
    <url-pattern> /test_sv </url-pattern>
  </servlet-mapping>
</web-app>
```

<web-app ~>タグのコードは、/opt/tomcat/conf/web.xml で確認したものと同じ

「ls」で直下に作成されたことを確認

```
[root@localhost WEB-INF]# vi web.xml
[root@localhost WEB-INF]# ls
classes  java  web.xml
```

## 6 ・ Apache の設定

vi /etc/httpd/conf/httpd.conf で、末尾に以下のコードを追加

```
ProxyPass /test/ ajp://localhost:8009/testApp/
```

```
ProxyPassReverse /test/ ajp://localhost:8009/testApp/
```

```
ProxyPass /test/ ajp://localhost:8009/testApp/
ProxyPassReverse /test/ ajp://localhost:8009/testApp/
-- INSERT --
```

## 7 ・ 再起動させて読み込ませる

systemctl restart tomcat httpd で、Tomcat と Apache を同時に再起動

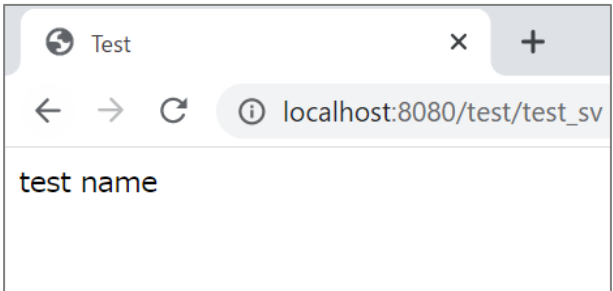
```
[root@localhost WEB-INF]# systemctl restart tomcat httpd
```

## 8・MySQL にテスト用データベースを作成

MySQL にログインし、データベースを作成

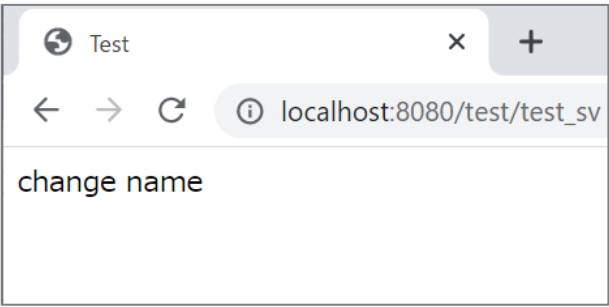
mysql> CREATE DATABASE test_db; Query OK, 1 row affected (0.01 sec)	CREATE DATABASE test_db;
mysql> USE test_db; Database changed	USE test_db;
mysql> CREATE TABLE test_tb( -> mid CHAR(5) PRIMARY KEY, -> name VARCHAR(20)); Query OK, 0 rows affected (0.03 sec)	CREATE TABLE test_tb( mid CHAR(5) PRIMARY KEY, name VARCHAR(20));
mysql> INSERT INTO test_tb -> VALUES('00001','test name'); Query OK, 1 row affected (0.01 sec)	INSERT INTO test_tb VALUES('00001','test name');
mysql> SELECT * FROM test_tb; +-----+-----+   mid   name   +-----+-----+   00001   test name   +-----+-----+ 1 row in set (0.00 sec)	SELECT * FROM test_tb;

## 9・ブラウザからアクセスする

 <p>The screenshot shows a web browser window with the address bar displaying 'localhost:8080/test/test_sv'. The page content shows 'test name'.</p>	<p>「<a href="http://localhost:8080/test/test_sv">http://localhost:8080/test/test_sv</a>」 でアクセスし、 「test name」と表示されれば OK</p>
---	--

## 10・データベースの編集が反映されるかを確認

mysql> UPDATE test_tb SET -> name='change name' WHERE mid=00001; Query OK, 1 row affected (0.00 sec) Rows matched: 1 Changed: 1 Warnings: 0	UPDATE test_tb SET name='change name' WHERE mid=00001;
mysql> SELECT * FROM test_tb; +-----+-----+   mid   name   +-----+-----+   00001   change name   +-----+-----+ 1 row in set (0.01 sec)	SELECT * FROM test_tb;

 <p>The screenshot shows a web browser window with a single tab titled 'Test'. The address bar displays 'localhost:8080/test/test_sv'. Below the address bar, the page content consists of the text 'change name'.</p>	<p>web ページを「F5」で更新し、編集内容通りに表示されれば OK</p>
---	--

Creation Date 2020 12