

テスト用アプリの作成

MySQL との接続テストアプリの用意

1 ・ test アプリの構成

```
□/opt/tomcat
├─□work
│   └─□Catalina
│       └─□localhost
│           └─◆test.xml
└─□webapps
    └─■test
        └─■WEB-INF
            ├─■lib
            │   └─◇mysql-connector-java-8.0.22.jar
            ├─■java
            │   └─◆Test.java
            ├─■classes
            │   └─◆Test.class
            └─◆web.xml
```

/opt/tomcat 配下に、
左図の構成で test アプリを配置する

■◆部を新規で作成

- ・ jar ファイルを lib 配下に移動
- ・ Test.java を作成して
classes 配下にコンパイル
- ・ 2 つの xml ファイルを作成/配置

2 ・ test.xml を作成（コンテキスト(アプリの単位)の指定）

「find / | grep Catalina/localhost」でディレクトリの所在を確認

```
[root@localhost ~]# find / | grep Catalina/localhost
/opt/tomcat/work/Catalina/localhost
```

バージョンなどによっては cnof ディレクトリに配置されていたりする

「vi /opt/tomcat/work/Catalina/localhost/test.xml」を実行し、下記のコードを記載する

```
<Context path="/test"
  docBase="/opt/tomcat/webapps/test"
  reloadable="false">
</Context>
```

```
<Context path="/test"
  docBase="/opt/tomcat/webapps/test"
  reloadable="false">
</Context>
-- INSERT --
```

```
[root@localhost ~]# vi /opt/tomcat/work/Catalina/localhost/test.xml
[root@localhost ~]# ls /opt/tomcat/work/Catalina/localhost
DocoTsubu ROOT docs examples host-manager manager test.xml
```

「ls」でファイルが作成されたことを確認

3・test コンテキスト(アプリ)の作成

「cd /opt/tomcat/webapps」で webapps ディレクトリへ移動し、
「mkdir test; cd test」で test ディレクトリを作成し、「cd」で移動

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

「mkdir WEB-INF; cd WEB-INF」で WEB-INF ディレクトリを作成し、「cd」で移動

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

「mkdir lib java classes」で3つのディレクトリを作成

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

4・lib ディレクトリに入手しておいた JDBC の jar ファイルを配置

「cp ¥
/opt/tomcat/temp/mysql-connector-java-8.0.22/¥
mysql-connector-java-8.0.22.jar lib; ls lib」

コピー元を絶対パスで指定
[¥]長いので改行
「ls」で確認

```
[root@localhost WEB-INF]# cp ¥
> /opt/tomcat/temp/mysql-connector-java-8.0.22/¥
> mysql-connector-java-8.0.22.jar lib; ls lib
mysql-connector-java-8.0.22.jar
```

5・Servlet(Test.java)の作成 「vi java/Test.java」で以下のコードを記載する

```
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 Test 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", "root", "AT8_MySQL");
        stmt = con.createStatement();
        rs = stmt.executeQuery("select * from sample");
        while (rs.next()) {
            sb.append(rs.getString("name"));
        }
        rs.close();
        stmt.close();
        return sb.toString();
    }
}

```

「ls java」でファイルが作成されたことを確認

```

[root@localhost WEB-INF]# ls java
Test.java

```

6・Servlet のコンパイル

「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 ¥
/opt/tomcat/lib/servlet-api.jar ¥
java/Test.java; ls java」

servlet-api のパスを指定してコンパイル

「ls」で class ファイルを確認

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

「mv java/Test.class classes」で移動、「ls java classes」で確認

```
[root@localhost WEB-INF]# mv java/Test.class classes
[root@localhost WEB-INF]# ls java classes
classes:
Test.class

java:
Test.java
```

7・配備記述子 (web.xml) の作成

「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>Test</servlet-name>
    <servlet-class>Test</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Test</servlet-name>
    <url-pattern>/test</url-pattern>
  </servlet-mapping>
</web-app>
```

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

```
<?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>Test</servlet-name>  
    <servlet-class>Test</servlet-class>  
  </servlet>  
  <servlet-mapping>  
    <servlet-name>Test</servlet-name>  
    <url-pattern>/test</url-pattern>  
  </servlet-mapping>  
</web-app>
```

<web-app>タグの内容は、「/opt/tomcat/conf/web.xml」に記載されている内容と同じ

8・再起動させて読み込ませる

```
「systemctl restart tomcat」
```

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

Creation Date 2020 12