
ECサイト初級

Struts設定ファイルの作成

2 時間目

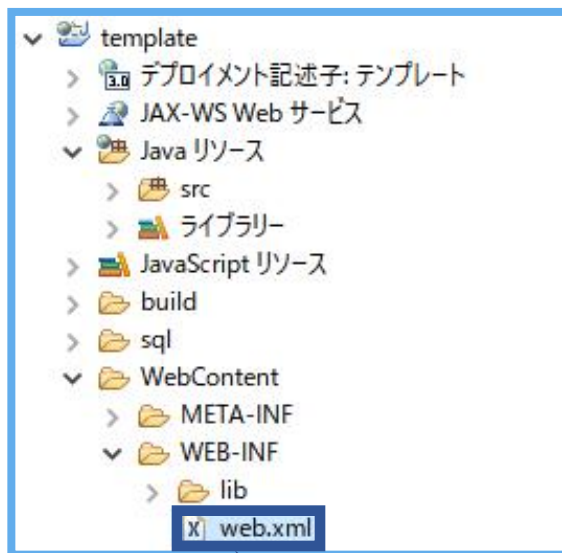
設定ファイルの作成

Strutsフレームワークはプロジェクトの仕様に従って設定行う必要がある為、各プロジェクト毎に設定ファイルを用意します。

ログイン認証プロジェクトと同様にweb.xmlとstruts.xmlが必要になります。

web.xmlの作成

1 Web.xml



① 「プロジェクト」「WebContent」「WEB-INF」の中にある「web.xml」ファイルを開きます。

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xmlns="http://xmlns.jcp.org/xml/ns/javaee"
4   xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app 3.1.xsd"
5   id="WebApp_ID" version="3.1">
6   <display-name>test</display-name>
7   <welcome-file-list>
8     <welcome-file>index.html</welcome-file>
9     <welcome-file>index.htm</welcome-file>
10    <welcome-file>index.jsp</welcome-file>
11    <welcome-file>default.html</welcome-file>
12    <welcome-file>default.htm</welcome-file>
13    <welcome-file>default.jsp</welcome-file>
14  </welcome-file-list>
15 </web-app> EOF
```

② このコードが初期状態で入っています。

③ 上記の内容を次のページの内容に変更します。

web.xmlの作成

④ 以下の内容に書き換えます。

web.xml(xmlファイル)

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
```

```
<display-name>template</display-name>
<welcome-file-list>
  <welcome-file>home.jsp</welcome-file>
</welcome-file-list>
```

プロジェクト名を設定します。
「template」と書いてください。

最初に表示するJSPファイルを設定します。
初回アクセスで表示させたいJSPファイル名を
書いてください。

```
<filter>
  <filter-name>struts2</filter-name>
  <filter-class>org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter</filter-class>
</filter>
```

Struts2を使うので、以下の設定を書いて
ください。

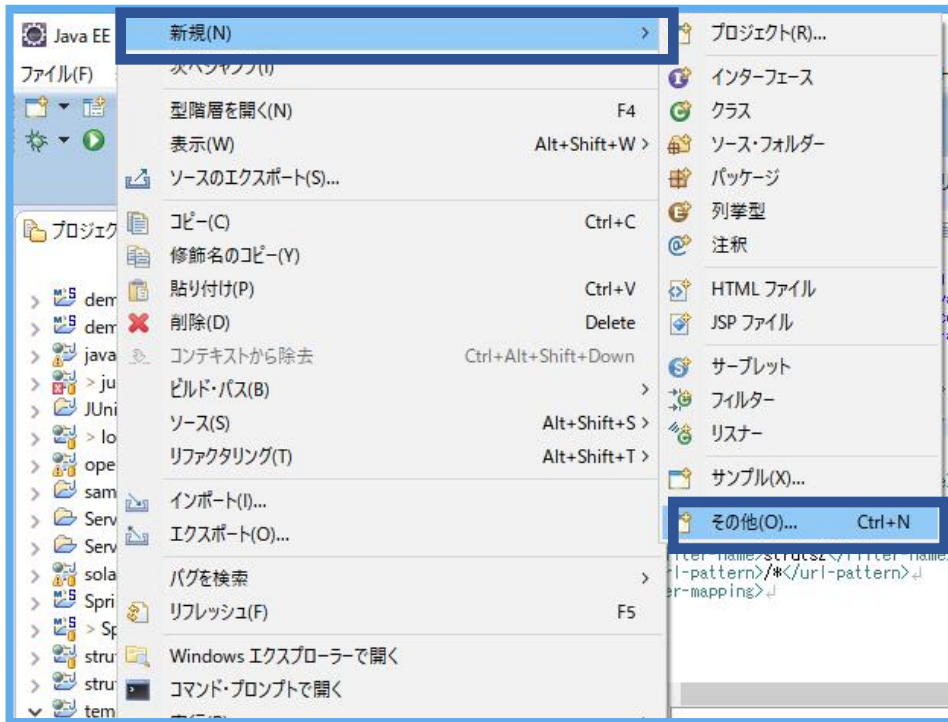
次へ続きます。

前の続きです。

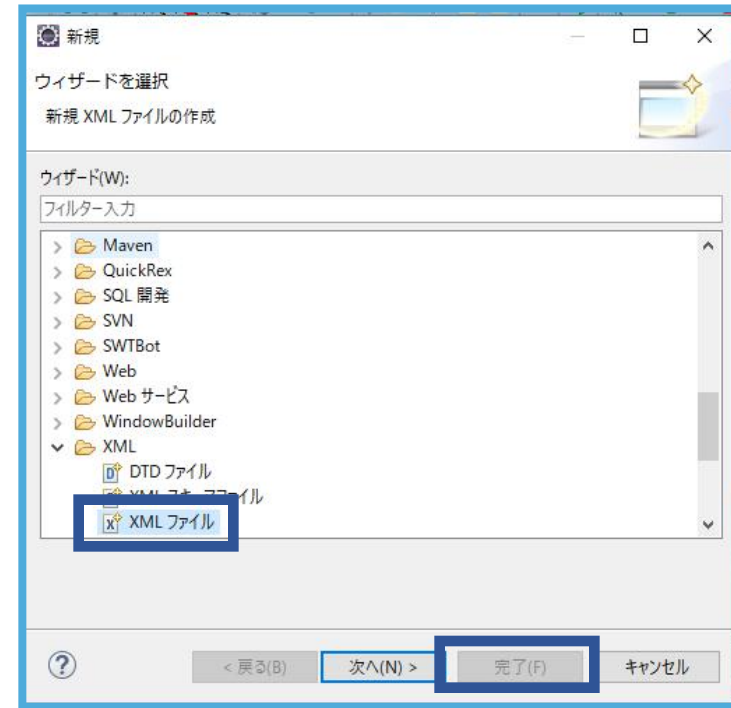
```
<filter-mapping>  
  <filter-name>struts2</filter-name>  
  <url-pattern>/*</url-pattern>  
</filter-mapping>  
  
</web-app>
```

struts.xmlの作成

2 Struts.xml

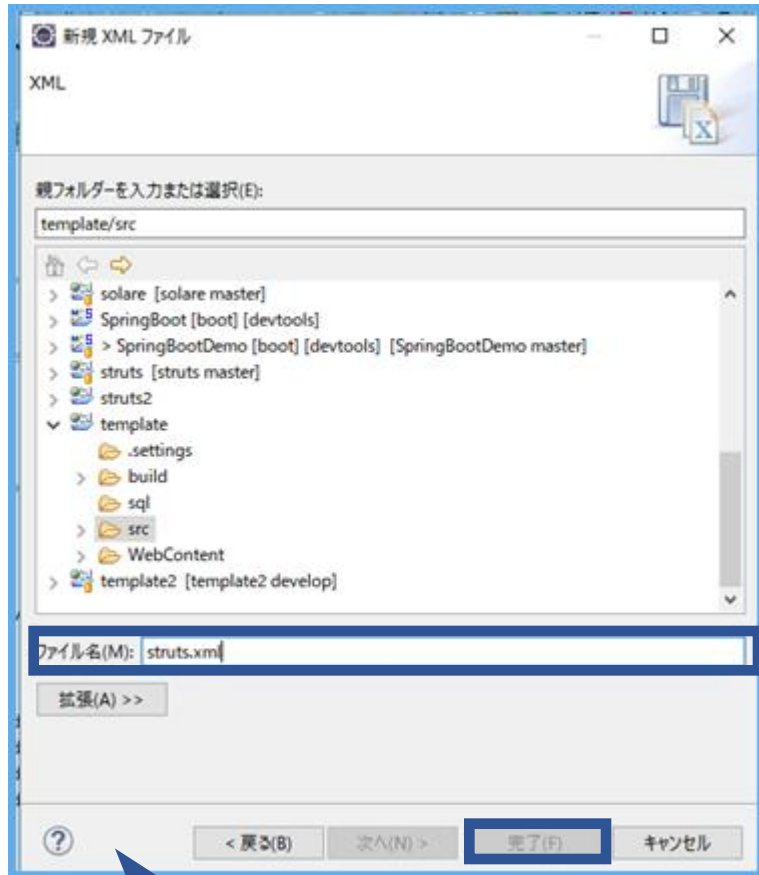


① 「Javaリソース」「src」を選択し、「プロジェクトを右クリック」「新規」「その他」を選択します。

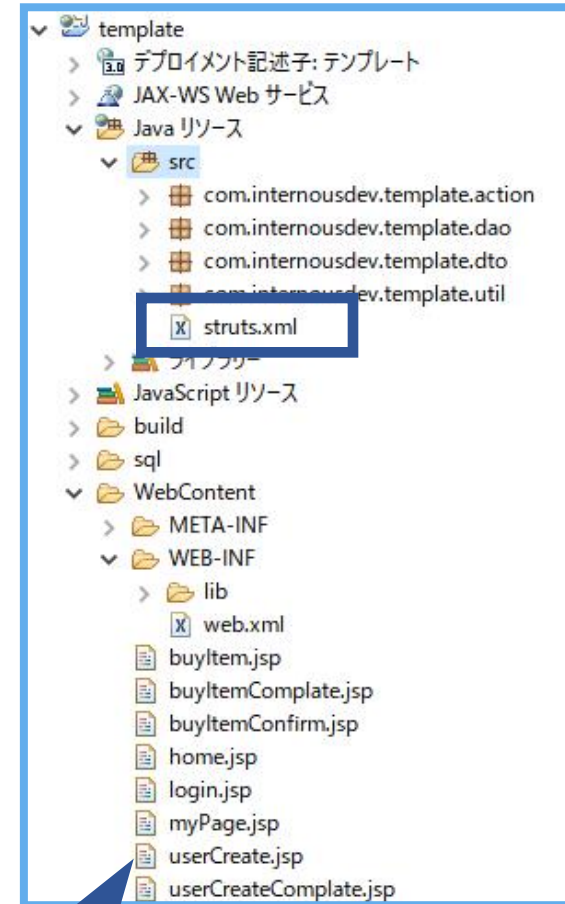


② 「ウィザード(W):」欄の「XML」「XMLファイル」を選択し、完了ボタンをクリックします。

struts.xmlの作成



③ ファイル名(M):の欄に「struts.xml」を入力して、完了をクリックします。



④ 「Javaリソース」「src」の直下に「struts.xml」が作成されていれば成功です。

struts.xmlの作成

⑤ 以下の内容を写経します。

struts.xml(xmlファイル)

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

<!DOCTYPE struts PUBLIC
"-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
"http://struts.apache.org/dtds/struts-2.3.dtd">
<!-- Strutsの詳細設定 -->

<struts>
    <constant name="struts.devMode" value="true" />
    <!-- 対象のpackageを設定 -->
    <package name="com.internousdev.template.action" extends="struts-default">

        <!-- HomeAction -->
        <action name="HomeAction" class="com.internousdev.template.action.HomeAction"
method="execute">
            <result name="success">login.jsp</result>
        </action>
    </package>
</struts>
```

呼び出されるJavaクラスファイルと実行するメソッドを登録します。

次へ続きます。

前の続きです。

```
<!-- LoginAction -->
<action name="LoginAction" class="com.internousdev.template.action.LoginAction"
method="execute">
    <result name="success">buyItem.jsp</result>
    <result name="error">home.jsp</result>
</action>

<!-- UserCreateAction -->
<action name="UserCreateAction"
class="com.internousdev.template.action.UserCreateAction" method="execute">
    <result name="success">userCreate.jsp</result>
    <result name="back">login.jsp</result>
</action>

<!-- UserCreateConfirmAction -->
<action name="UserCreateConfirmAction"
class="com.internousdev.template.action.UserCreateConfirmAction" method="execute">
    <result name="success">userCreateConfirm.jsp</result>
    <result name="error">userCreate.jsp</result>
</action>
```

次へ続きます。

前の続きです。

```
<!-- UserCreateCompleteAction -->
<action name="UserCreateCompleteAction"
class="com.internousdev.template.action.UserCreateCompleteAction" method="execute">
    <result name="success">userCreateComplete.jsp</result>
</action>

<!-- BuyItemAction -->
<action name="BuyItemAction"
class="com.internousdev.template.action.BuyItemAction" method="execute">
    <result name="success">buyItemConfirm.jsp</result>
    <result name="error">buyItem.jsp</result>
</action>

<!-- BuyItemConfirmAction -->
<action name="BuyItemConfirmAction"
class="com.internousdev.template.action.BuyItemConfirmAction" method="execute">
    <result name="success">buyItemComplete.jsp</result>
    <result name="back">buyItem.jsp</result>
</action>
```

次へ続きます。

前の続きです。

```
<!-- BuyItemCompleteAction -->
<action name="BuyItemCompleteAction"
class="com.internousdev.template.action.BuyItemCompleteAction" method="execute">
    <result name="success">home.jsp</result>
</action>

<!-- MyPageAction -->
<action name="MyPageAction"
class="com.internousdev.template.action.MyPageAction" method="execute">
    <result name="success">myPage.jsp</result>
</action>

<!-- LogoutAction -->
<action name="LogoutAction"
class="com.internousdev.template.action.LogoutAction" method="execute">
    <result name="success">home.jsp</result>
</action>

</package>
</struts>
```

struts.xmlの作成

```
<action name="LoginAction" class="com.internousdev.template.action.LoginAction"  
method="execute">
```

「LoginAction」クラスの中で「execute」メソッドが最初に呼び出されます。

「com.internousdev.template.action」パッケージの「LoginAction」に設定します。

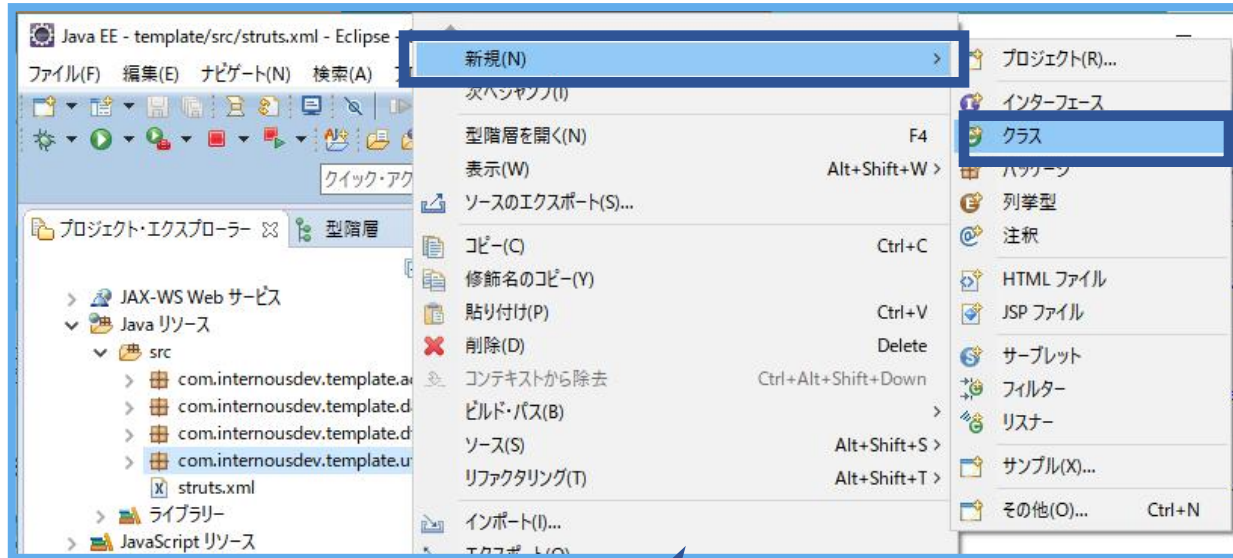
```
<result name="success">buyItem.jsp</result>  
<result name="error">home.jsp</result>
```

「execute」メソッドの処理結果に応じてフロントに送る「JSPファイル」を設定しています。

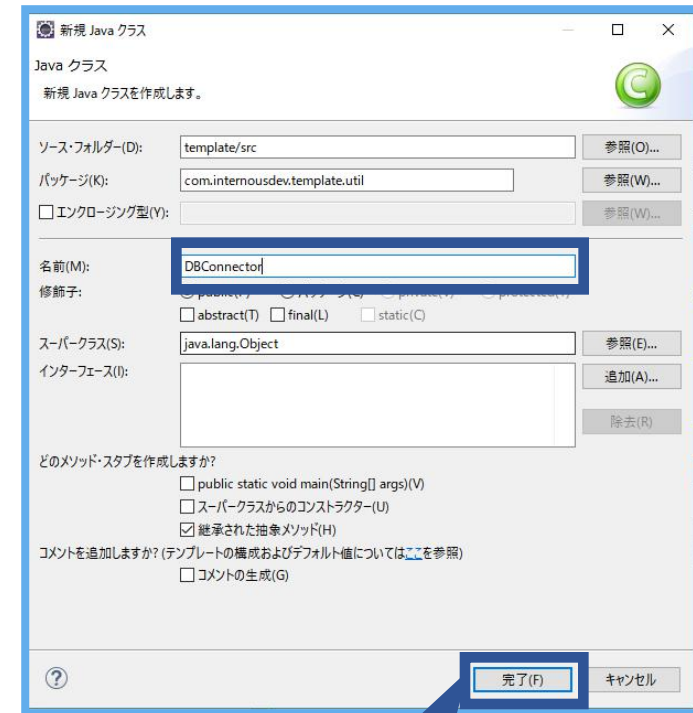
```
</action>
```

DBConnectorの作成

3 DBConnector



① 「src」「com.internousdev.template.util」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「DBConnector」と入力し、完了ボタンをクリックします。

DBConnectorの作成

DBConnector.java
(javaファイル)

```
package com.internousdev.util;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;
```

```
public class DBConnector {  
    private static String driverName = "com.mysql.jdbc.Driver";  
    private static String url = "jdbc:mysql://localhost/ecsita";
```

```
    private static String user = "root";  
    private static String password = "mysql";
```

```
    public Connection getConnection() {  
        Connection con = null;
```

```
        try {  
            Class.forName(driverName);  
            con = (Connection) DriverManager.getConnection(url,user,password);
```

③ 以下の内容を写経します。

MySQL接続に必要な情報を設定します。

接続情報から自分のパソコンにインストールされているMySQLへ接続する準備が整います。

次へ続きます。

DBConnectorの作成

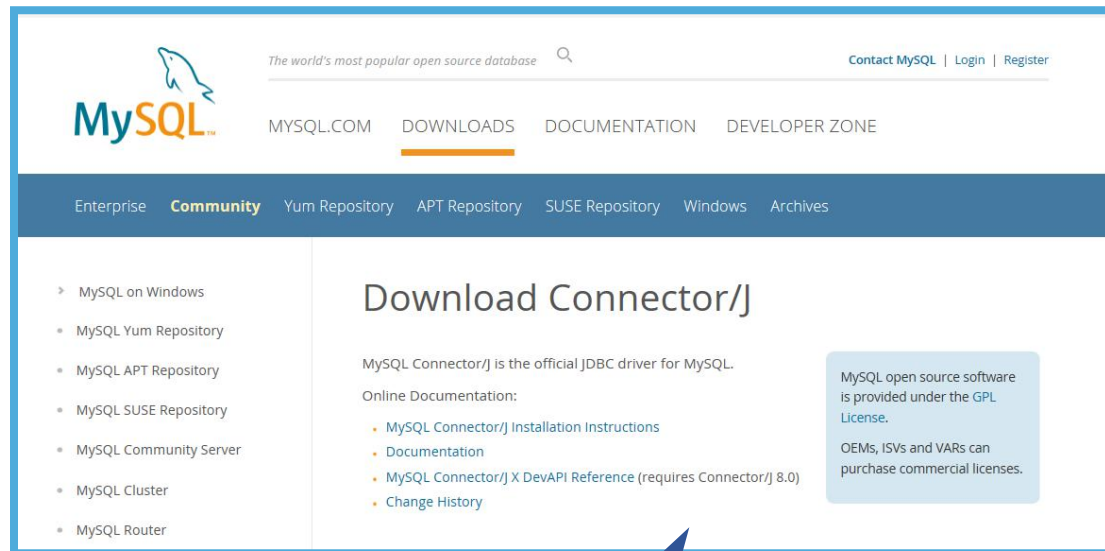
前の続きです。

```
    } catch(ClassNotFoundException e) {  
        e.printStackTrace();  
    } catch(SQLException e) {  
        e.printStackTrace();  
    }  
    return con;  
}
```

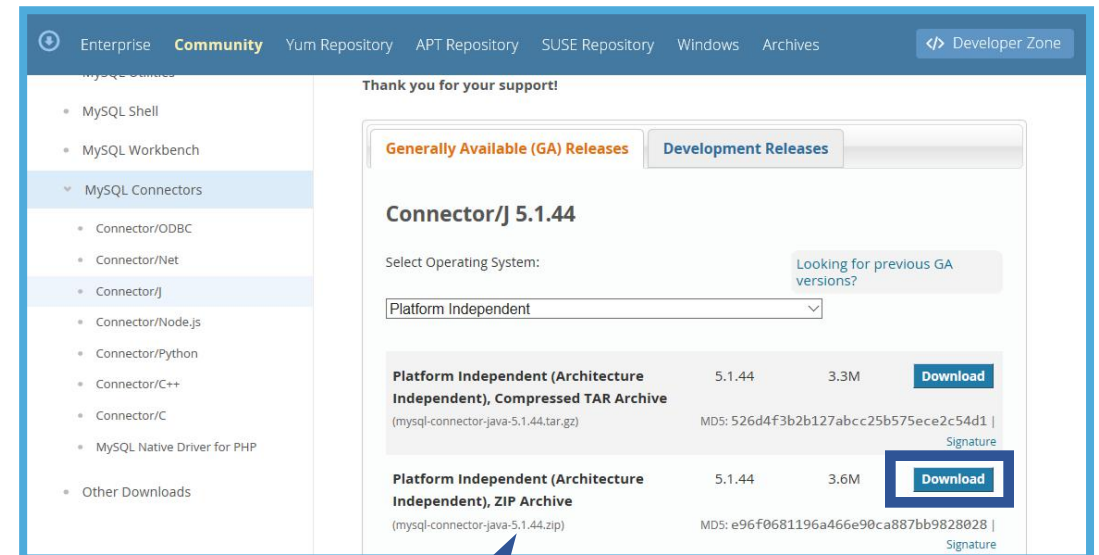
Mysqlに接続できたか情報を渡します。

MySQL JDBC ドライバーの配備

4 MySQL JDBC ドライバー

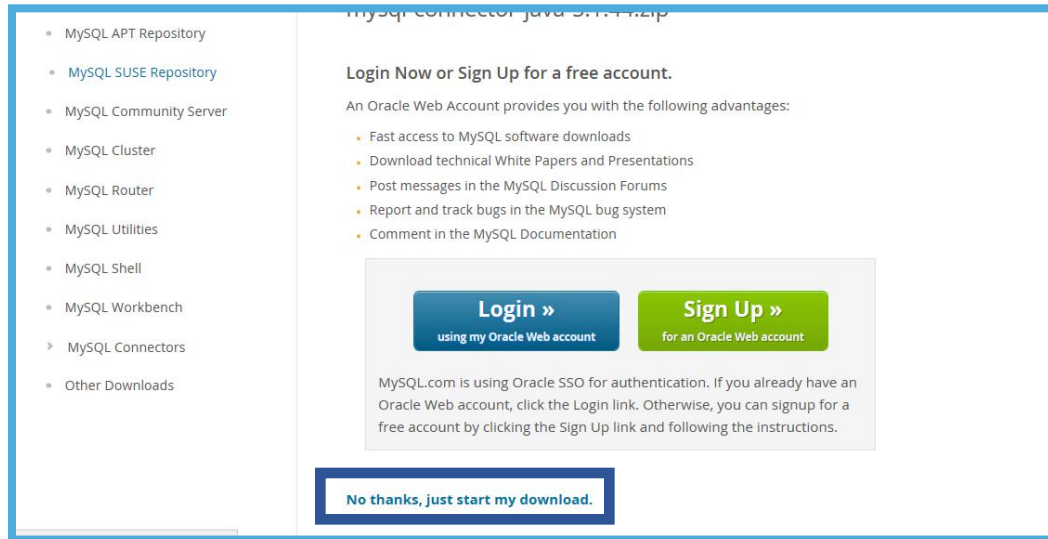


① 「<https://dev.mysql.com/downloads/connector/j/>」にアクセスします。



② 下へスクロールして、「zip」形式のDownloadボタンをクリックします。

MySQL JDBC ドライバーの配備

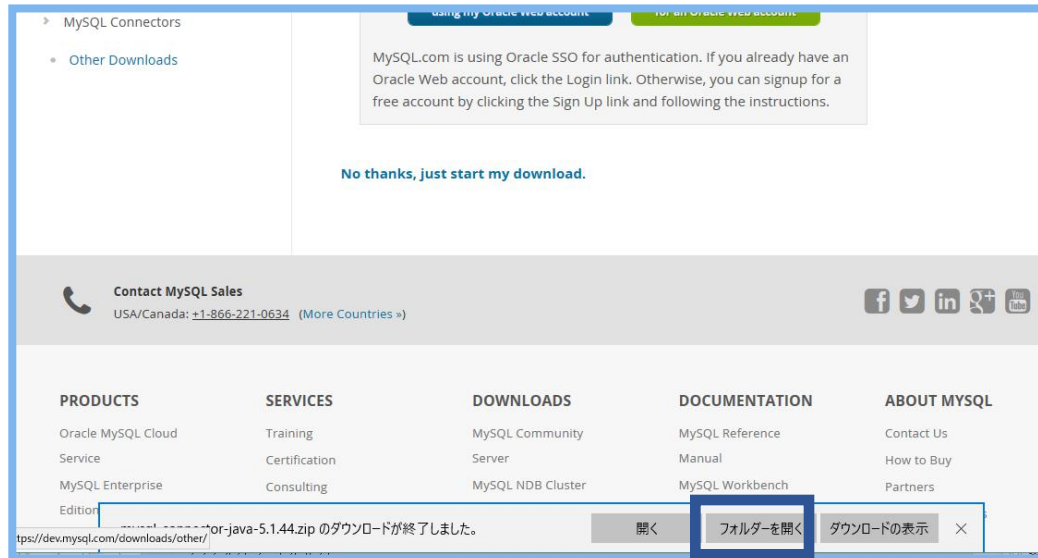


③ 遷移した先の画面でも、下へスクロールし、「No thanks, just start my download.」リンクをクリックします。

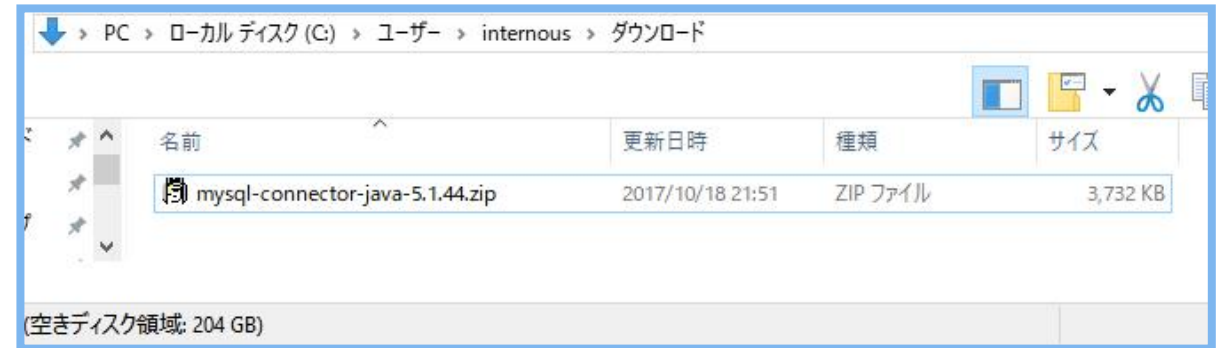


④ 「保存」ボタンをクリックします。

MySQL JDBC ドライバーの配備

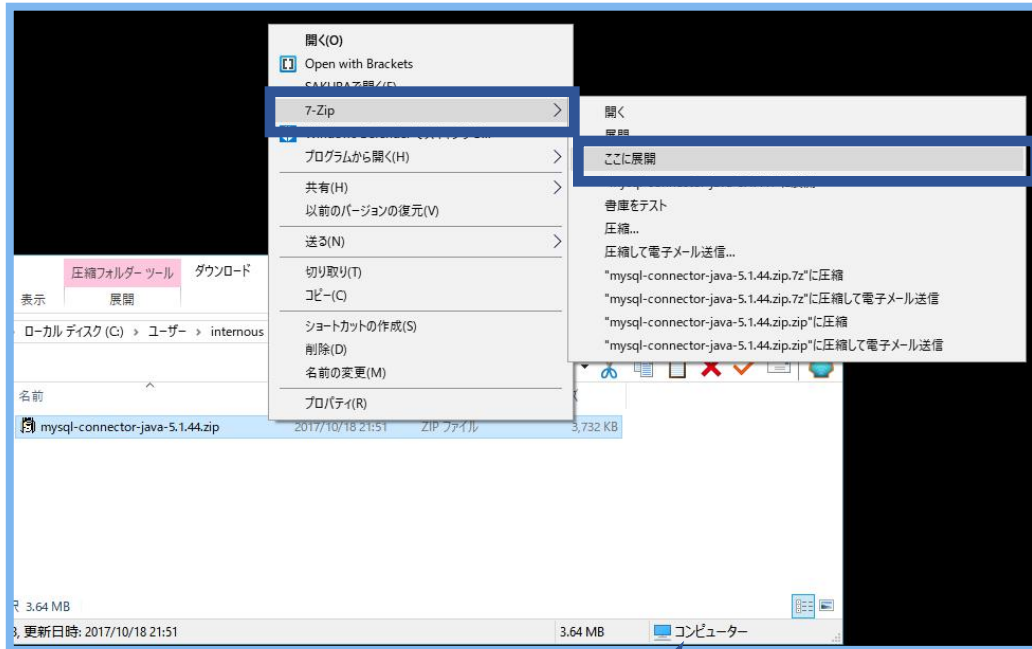


⑤ ダウンロードが完了後「フォルダを開く」ボタンをクリックします。

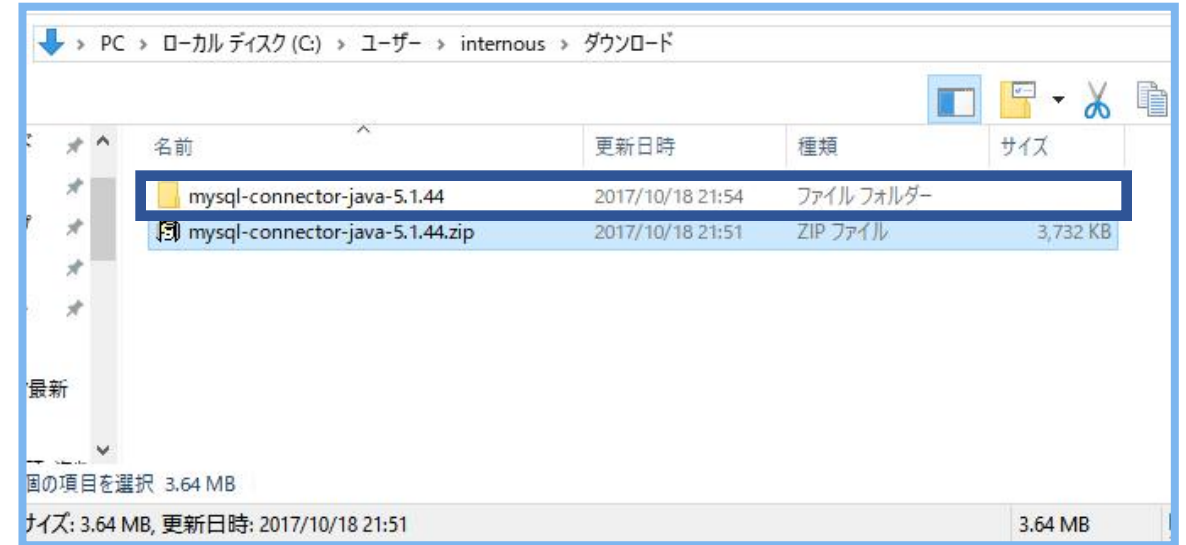


⑥ ダウンロードファイルが格納されている「ディレクトリ」が開きます。対象のディレクトリの中に「mysql-connector-java-x.x.xx.zip」を探します。

MySQL JDBC ドライバーの配備

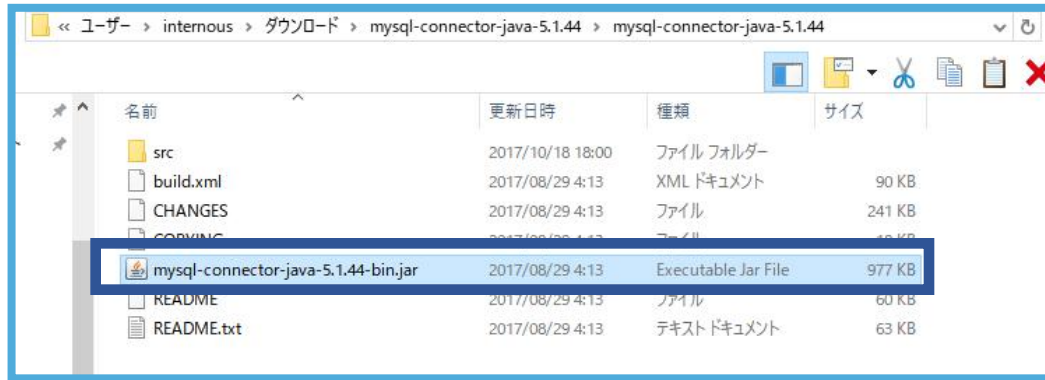


⑦ 「mysql-connector-java-x.x.xx.zip」を
右クリックし、「解凍」または「展開」します。

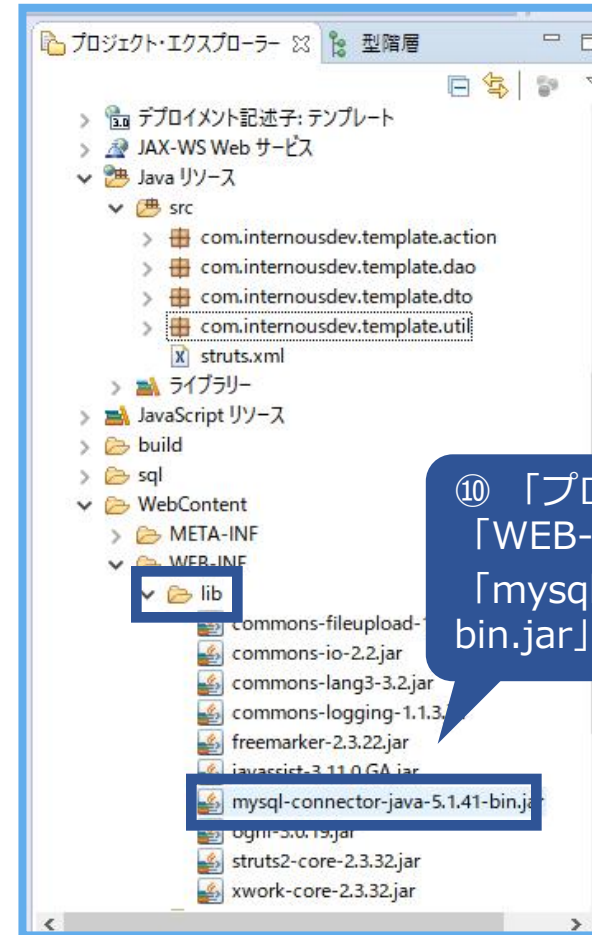


⑧ 解凍が終わると「mysql-connector-
java-x.x.xx」という名前のディレクトリ
が増えます。
「mysql-connector-java-x.x.xx」をダ
ブルクリックします。

MySQL JDBC ドライバーの配備

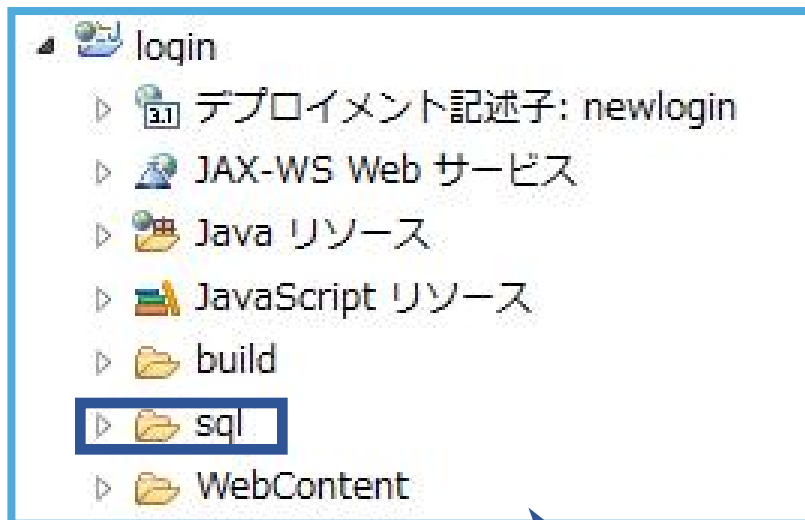


⑨ 「mysql-connector-java-x.x.xx」ディレクトリの中に「mysql-connector-java-x.x.xx-bin.jar」があることを確認します。

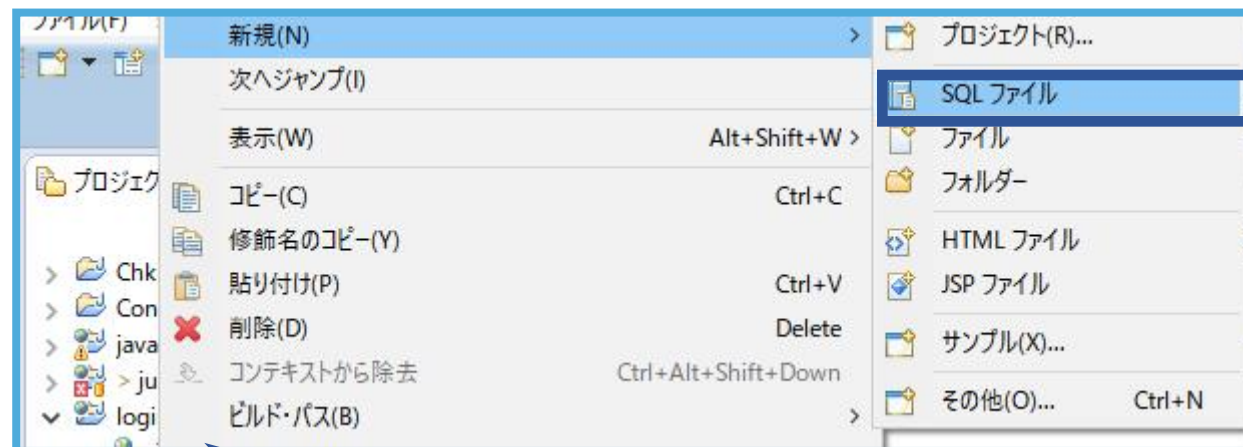


⑩ 「プロジェクト」「WebContent」「WEB-INF」「lib」の中に「mysql-connector-java-x.x.xx-bin.jar」をコピーします。

SQLファイルの配備

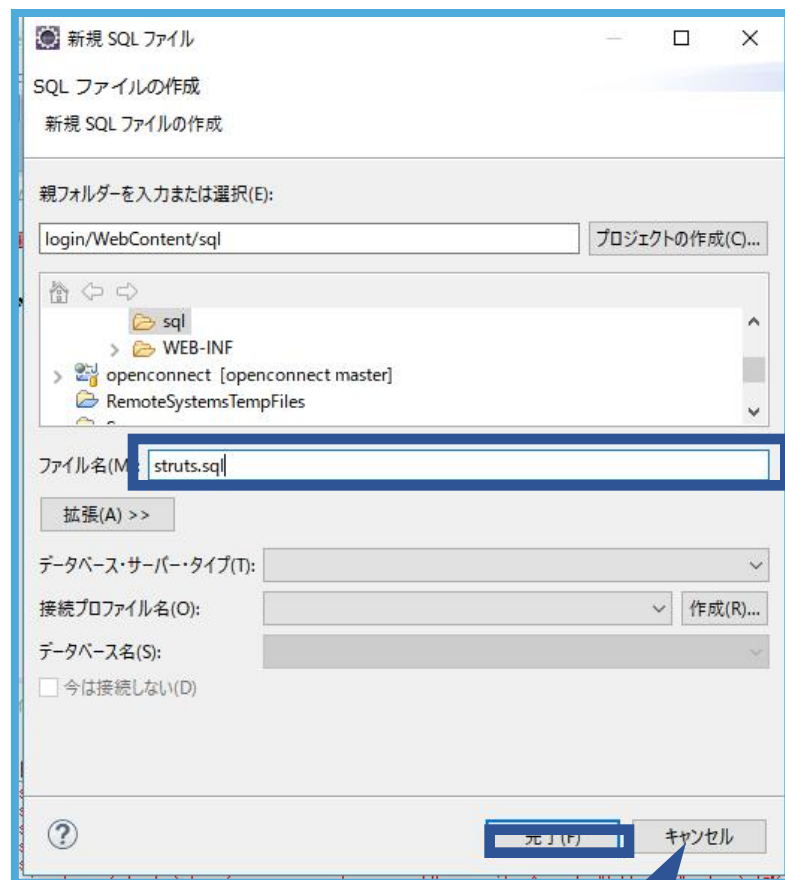


① 「プロジェクト」の直下に「sql」ディレクトリを作成します。

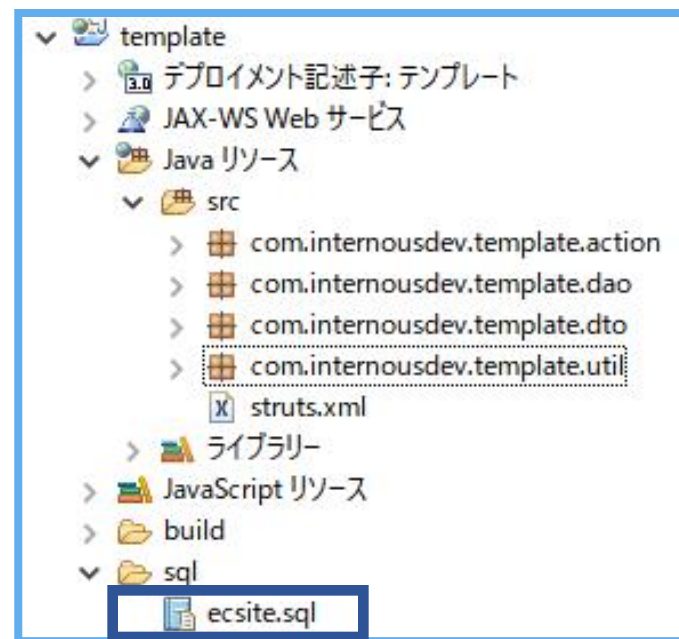


② 「sql」の上で「右クリック」「新規」「SQLファイル」を選択します。

SQLファイルの配備



③ 「ファイル名(M):」の欄に「ecsite.sql」と入力し、完了をクリックします。



④ SQLファイルが作成できていれば成功です。

SQLファイルの写経

③ 以下の内容を写経します。

ecsite.sql(sqlファイル)

```
set names utf8;  
set foreign_key_checks = 0;  
drop database if exists ebsite;
```

「ecsite」がデータベース名になります。

```
create database if not exists ebsite;  
use ebsite;
```

「login_user_transaction」がテーブル名になります。

```
drop table if exists login_user_transaction;
```

```
create table login_user_transaction(  
  id int not null primary key auto_increment,  
  login_id varchar(16) unique,  
  login_pass varchar(16),  
  user_name varchar(50),  
  insert_date datetime,  
  updated_date datetime  
);
```

「login_user_transaction」テーブルに情報を保存します。

次へ続きます。

前の続きです。

```
drop table if exists item_info_transaction;
```

```
create table item_info_transaction(  
  id int not null primary key auto_increment,  
  item_name varchar(30),  
  item_price int,  
  item_stock int,  
  insert_date datetime,  
  update_date datetime  
);
```

```
drop table if exists user_buy_item_transaction;
```

```
create table user_buy_item_transaction(  
  id int not null primary key auto_increment,  
  item_transaction_id int,  
  total_price int,  
  total_count int,  
  user_master_id varchar(16),  
  pay varchar(30),
```

次へ続きます。

前の続きです。

```
insert_date datetime,  
delete_date datetime  
);
```

作成したテーブルに情報を格納します。

```
INSERT INTO item_info_transaction(item_name, item_price, item_stock) VALUES("ノートBook", 100,  
50);
```

```
INSERT INTO login_user_transaction(login_id, login_pass, user_name) VALUES("internous",  
"internous01", "test");
```

SQLファイルの写経

drop database if exists `ecsite`;

もし「ecsite」というデータベースがあったら、削除します。

create database `ecsite`;

「ecsite」データベースを作成します。

use `ecsite`;

作成した「ecsite」データベースを利用します。