
ログイン認証

Action DAO DTOの作成

3 時間目

- 1) MVCモデルについて
- 2) クラスファイルの作成
 - 1 : DTOクラスの作成
 - 2 : DAOクラスの作成
 - 3 : Actionクラスの作成

MVCモデルとは

元々Javaでwebアプリケーションを作る際には、JSPとServletのみを使用して開発を行ってきました。（テキスト教材の前章参照）

しかし、webアプリケーションの規模が大きくなり複雑になると、JSPとServletだけでは全体の制御がしにくくなってきました。そこでアプリケーションの各機能を分けて独立させる事で、より開発やメンテナンスをしやすくしたのがMVCモデルというものです。Model、View、Controllerの3つの機能を分けてプログラムする手法です。

Model : 処理

View : 画面

Controller : 司令塔

MVCモデルとは

その後、そのMVCモデルの概念を骨組み（テンプレート）として、より簡単にWebアプリケーションを開発できる「フレームワーク」（土台、枠組み）が使われるようになりました。

この研修ではstruts2.3というフレームワークを使います。

■ struts2.3フレームワーク内でのファイルの役割

Model : 処理（Action）

View : 画面（JSP）

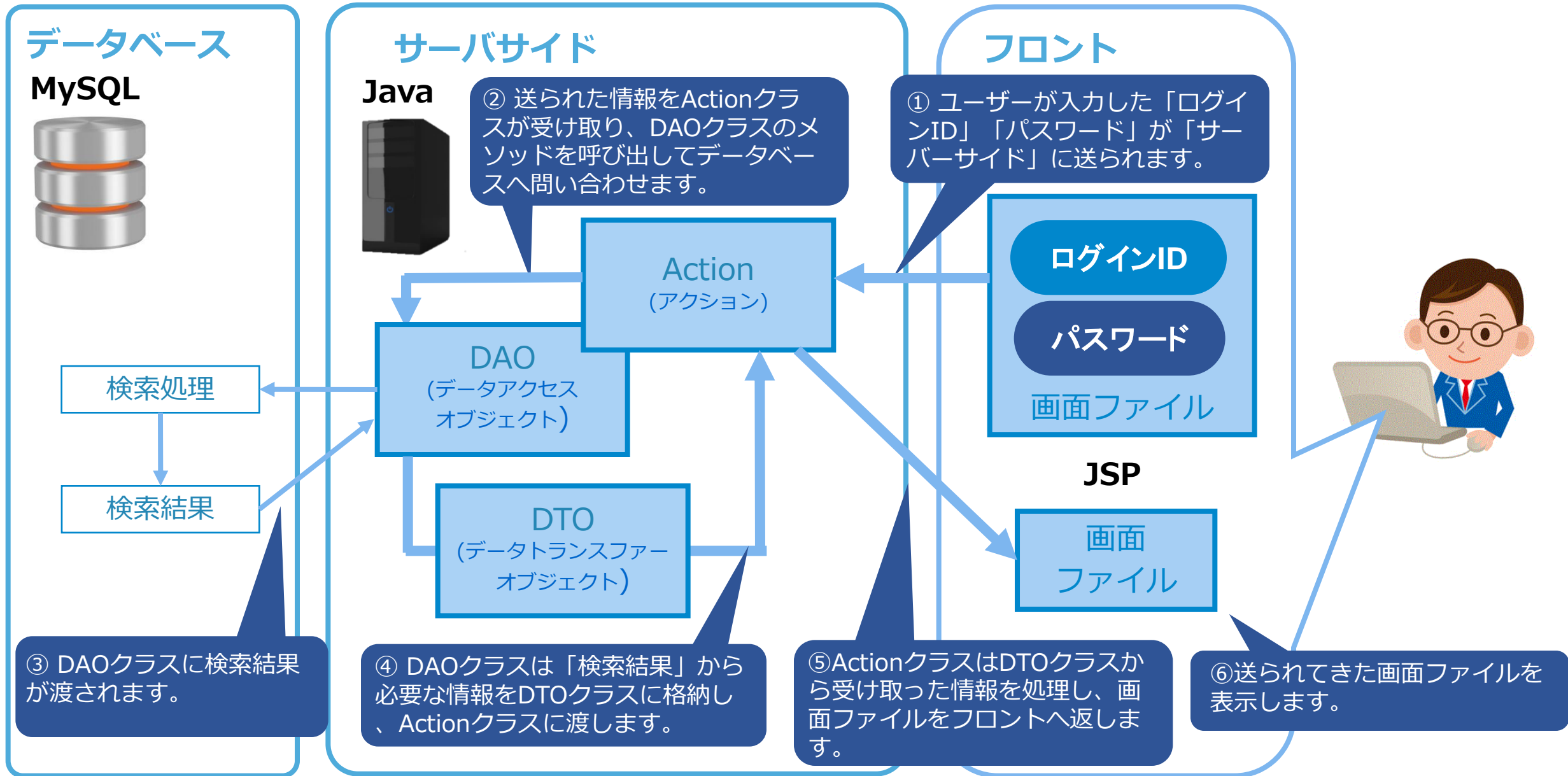
Controller : 司令塔（struts2のクラス群）



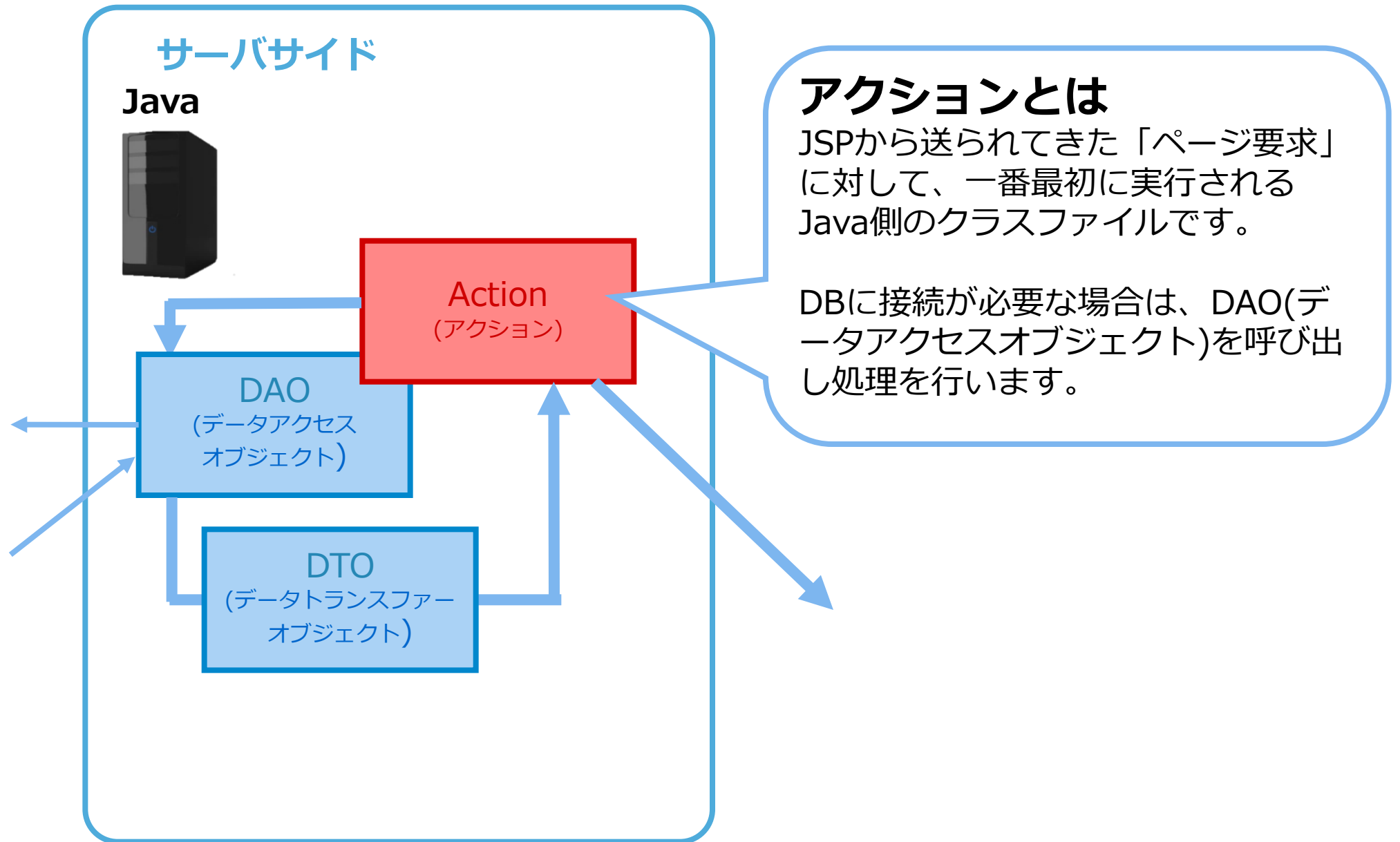
自分で作成

Struts2が提供

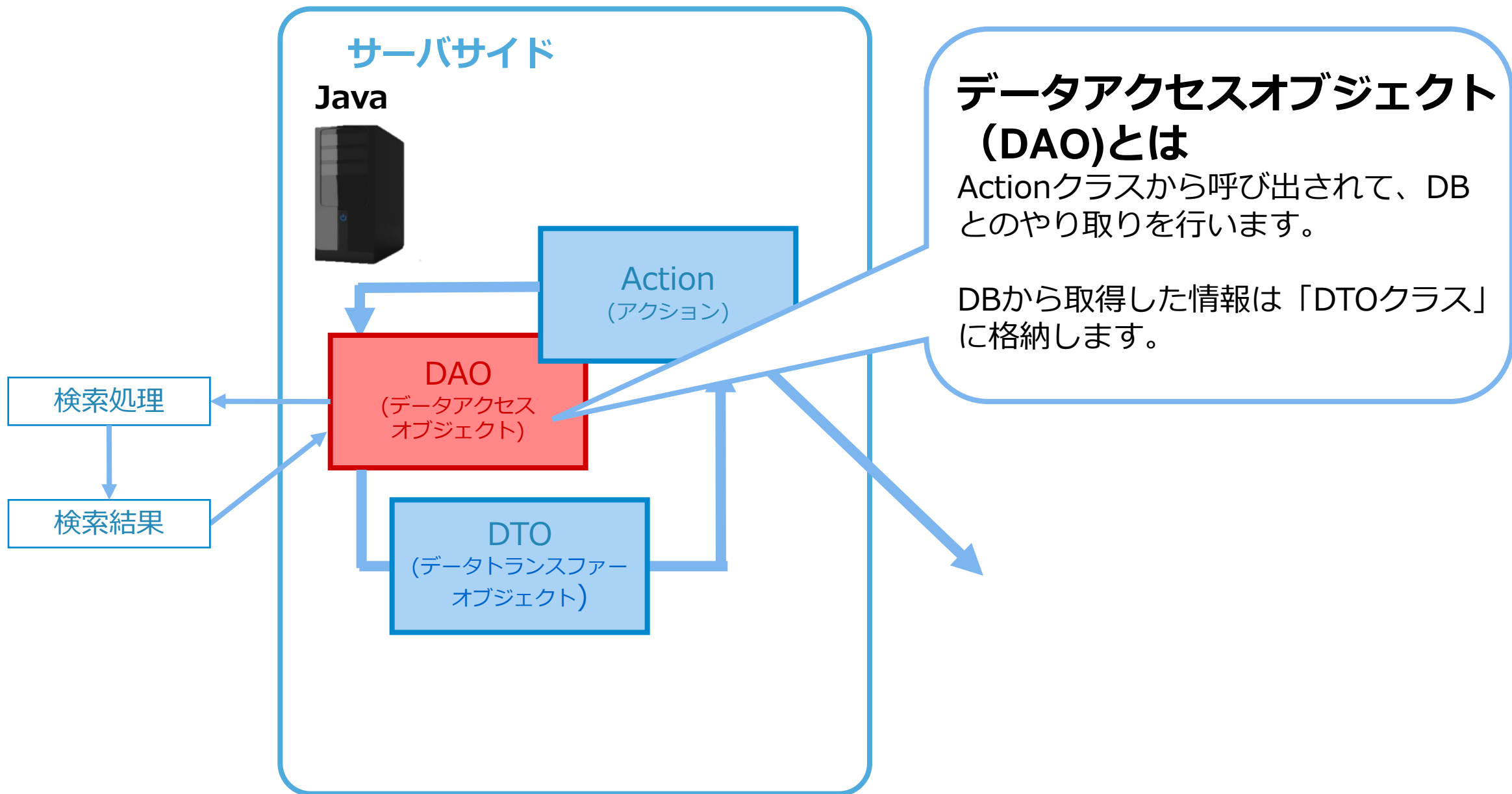
MVCモデルについて



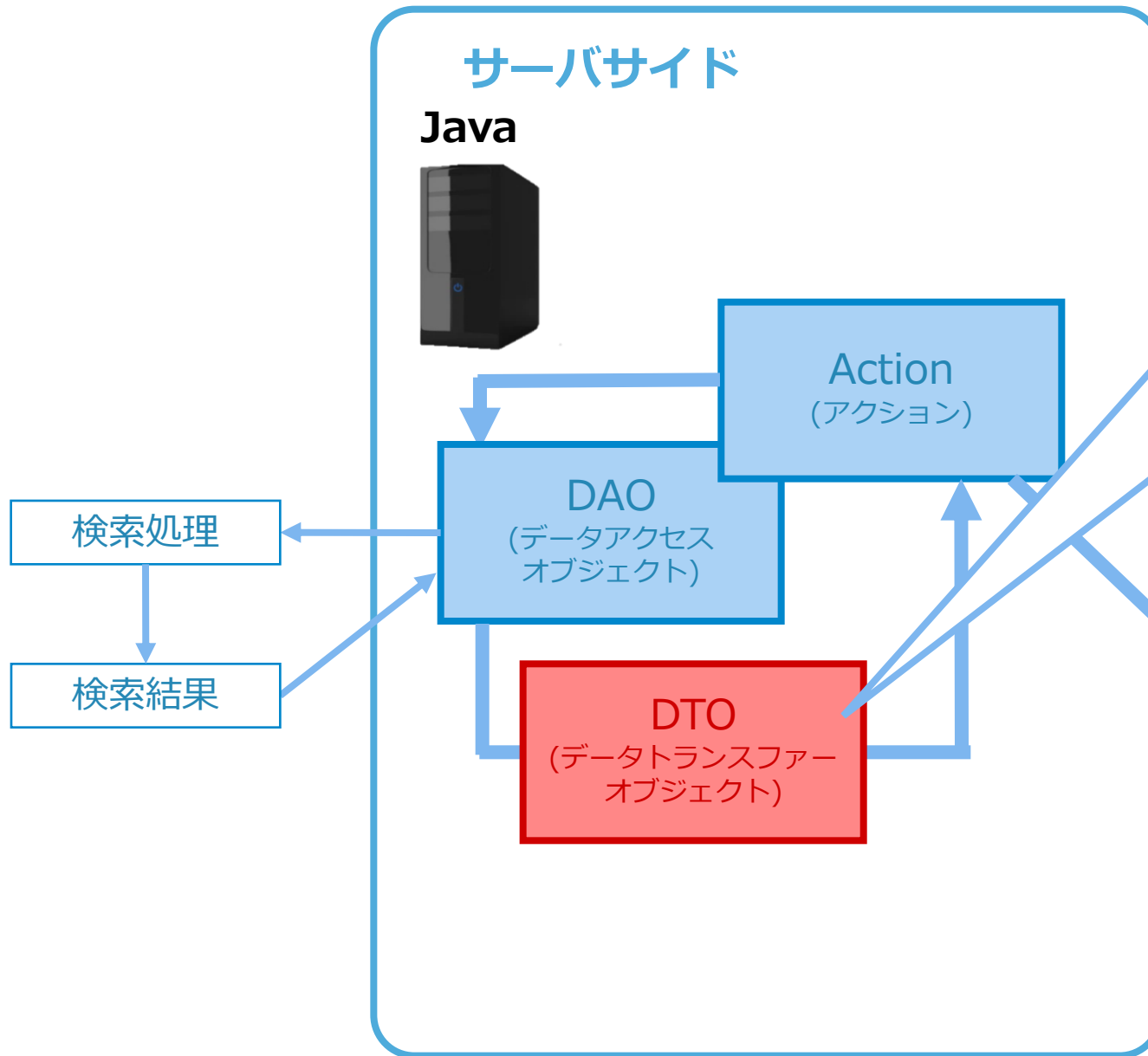
MVCモデルについて



MVCモデルについて



MVCモデルについて



データ転スファ-オブジェクト (DTO) とは
DBから取得した情報を格納する為のクラスです。

DAOはDTOに格納した情報をActionクラスへ渡します。

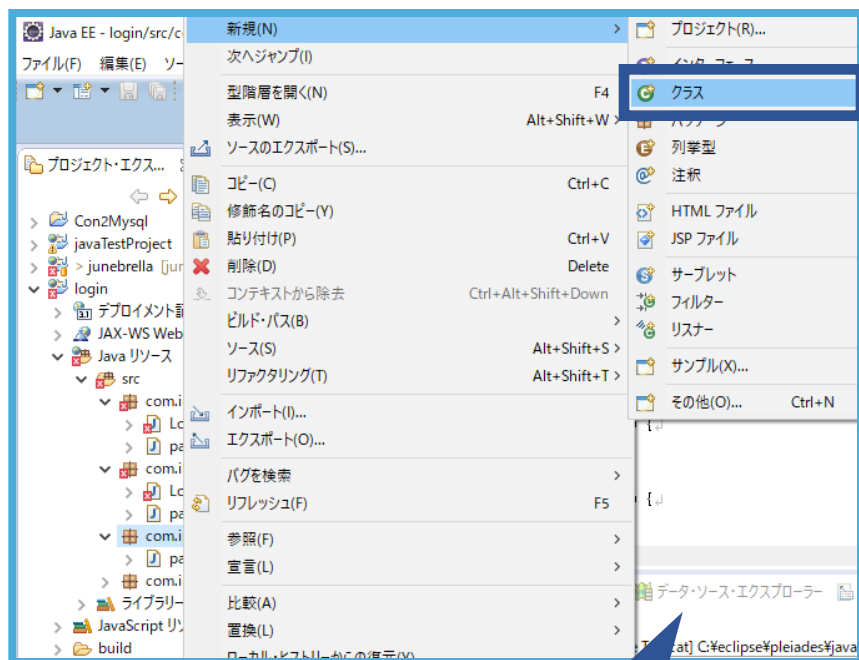
DTOに格納された情報をActionクラスが処理し、「表示画面」へ渡します。

DTOクラスの作成

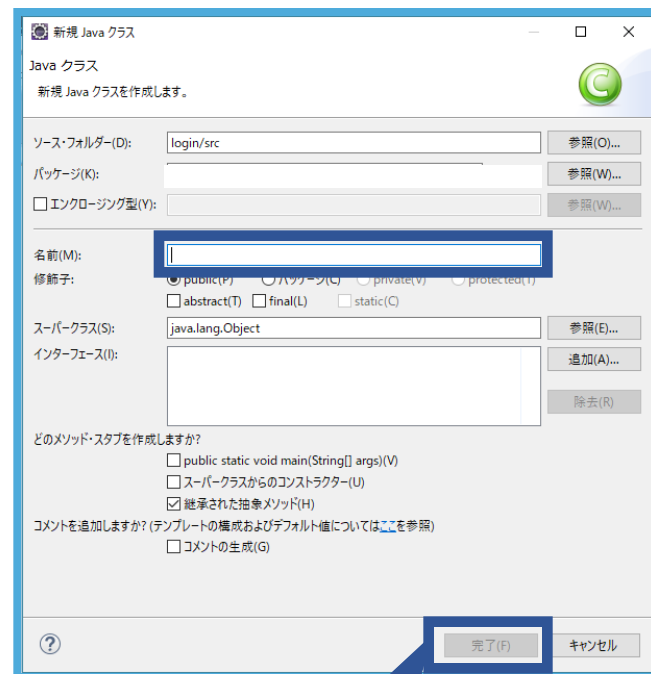
解説

DTOクラスは、DAOがDBから取得した値をActionへ戻す時、値を格納するのに利用されます。
DTOクラスには、必要なテーブルのデータの列に対応したフィールド変数とgetter/setterのみを定義します。

1 DTOクラス



① 「com.internousdev.login.dto」を右クリックし、「新規」「クラス」を選択します。



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

DTOクラスの作成

LoginDTO(javaファイル)

```
package com.internousdev.login.dto;  
public class LoginDTO {
```

```
    private int id;  
    private String name;  
    private String password;
```

```
    public int getId() {  
        return id;  
    }
```

```
    public void setId(int id) {  
        this.id = id;  
    }
```

```
    public String getName() {  
        return name;  
    }
```

テーブルから取得するデータに対応したフィールド変数を宣言します。

フィールド変数に対応したgetter と setterを定義します。

Actionクラスから呼び出され、nameフィールドの値をActionに渡します。

次へ続きます。

DTOクラスの作成

前の続きです。

```
public void setName(String name) {  
    this.name = name;  
}
```

DAOクラスから呼び出され、引数として受け取ったテーブルの値を自身のnameフィールドに格納します。

```
public String getPassword() {  
    return password;  
}
```

Actionクラスから呼び出され、passwordフィールドの値をActionに渡します。

```
public void setPassword(String password) {  
    this.password = password;  
}
```

DAOクラスから呼び出され、引数として受け取ったテーブルの値を自身のpasswordフィールドに格納します。

```
}
```

DTOクラスのまとめ

①DAOクラスでselectされた値を格納するためのクラス

②フィールドで格納する値を宣言

③②の各フィールドのgetterとsetterを定義

getter : Actionクラスから呼び出されActionへ値を渡す

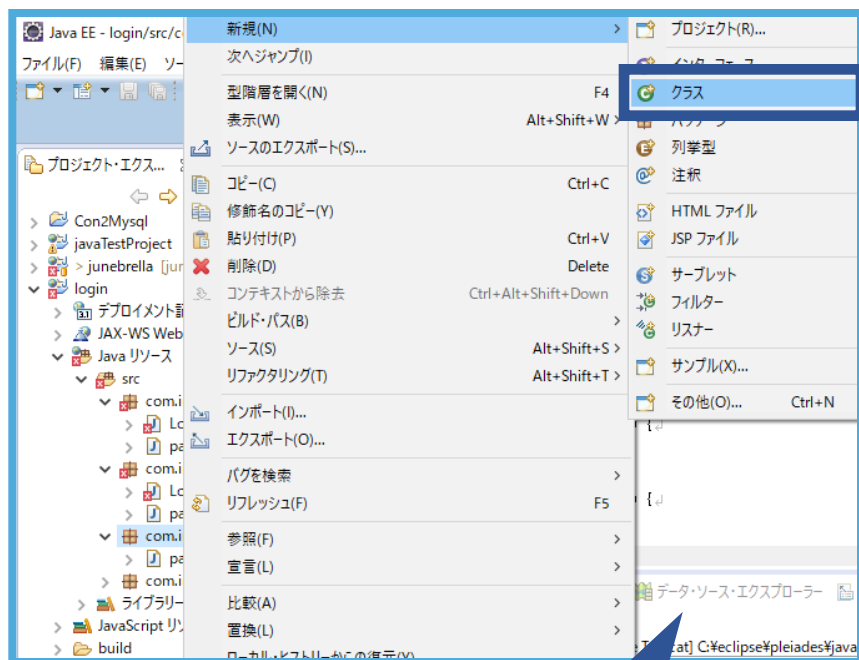
setter : DAOクラスから呼び出され、テーブルの値を自身（DTO）のフィールドに格納

DAOクラスの作成

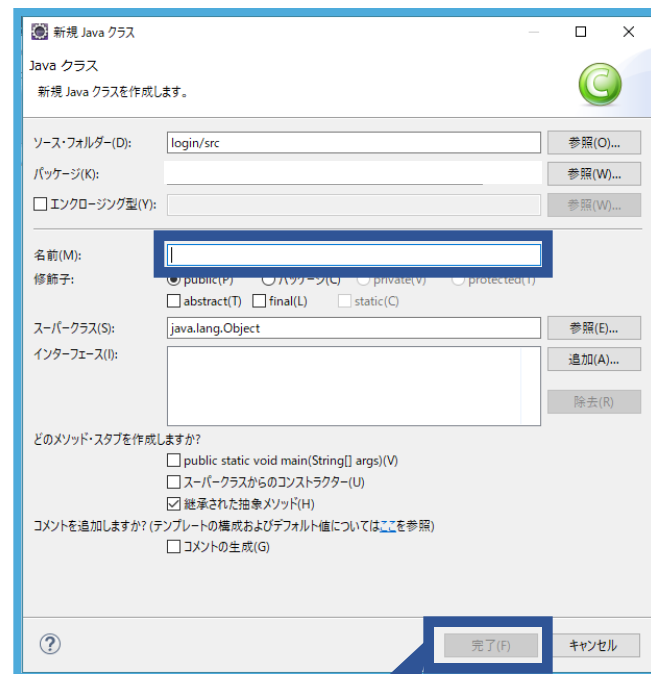
解説

DAOクラスでは、Actionから送られてきた情報を使ってDBへ問い合わせを行います。
問い合わせで取得した値をDTOクラスに格納します。

2 DAOクラス



① 「com.internousdev.login.dao」を右クリックし、「新規」「クラス」を選択します。



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

DAOクラスの作成

LoginDAO(javaファイル)

```
package com.internousdev.login.dao;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import com.internousdev.login.dto.LoginDTO;
import com.internousdev.login.util.DBConnector;
```

```
public class LoginDAO {
```

```
    public LoginDTO select(String name,String password) throws SQLException{
```

```
        LoginDTO dto=new LoginDTO();
        DBConnector db = new DBConnector();
        Connection con = db.getConnection();
```

```
        String sql="select * from user where user_name=? and password=?";
    try {
```

```
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1, name);
        ps.setString(2, password);
```

LoginDTO型を最後に呼び出し元に渡すので、LoginDTO型を戻り値にしたメソッドを作ります。Actionクラスの値を引数として受け取ります。

定義したSQL文の1番目の?にActionから送られたname、2番目の?にActionから送られたpasswordがそれぞれ入ります。

次へ続きます。

DAOクラスの作成

前の続きです。

```
ResultSet rs=ps.executeQuery();
```

select文のSQL文を実行するメソッドで、
戻り値はResultSet になります。

```
if(rs.next()) {
```

```
    dto.setName(rs.getString("user_name"));  
    dto.setPassword(rs.getString("password"));
```

```
}
```

tryとcatchで一括り
tryの中でエラーが起きた場合に
catchで処理する。

```
} catch (SQLException e) {  
    e.printStackTrace();
```

```
} finally {  
    con.close();  
}
```

処理中にSQL関連のエラーが発生した際
に実行する処理です。

```
return dto;
```

DB接続を終了する際は
必ず書くメソッドです。

```
}
```

dtoに入った値を、
呼び出し元であるActionクラスに渡す。

```
}
```

select文でDBから取得した情報を
String型に変換してDTOクラスに格納します。
LoginDTOクラスのsetName、setPassword
(setter) を利用します。

DAOクラスのまとめ

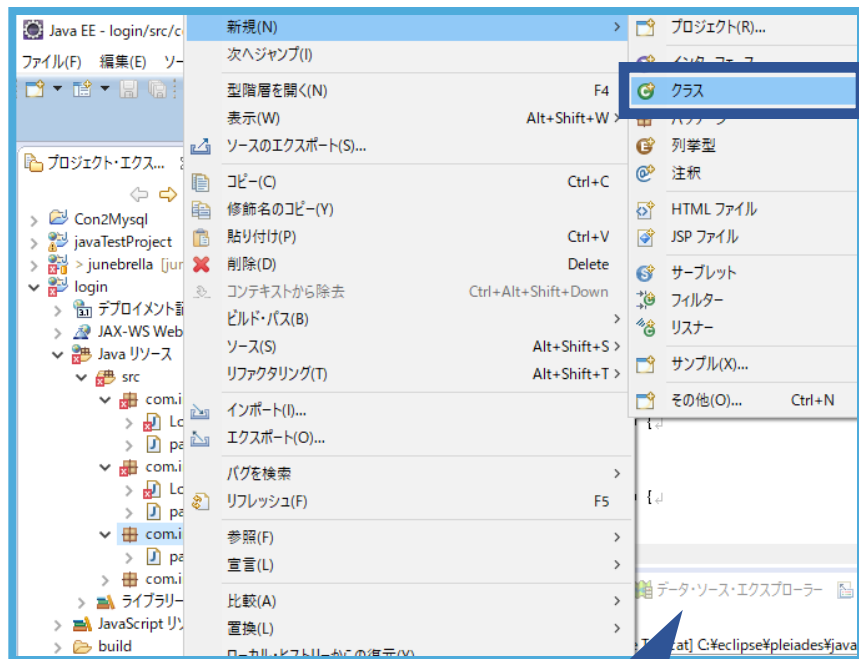
- ①クラス、メソッドの定義
- ②DBConnectorのインスタンス化
- ③getConnectionの呼び出し（DBと接続する）
- ④sql文を書く：値は？を入れておく（どんな値でも使いまわしできるようにするため）
- ⑤PreparedStatement（DBまで運んでくれる箱のイメージ）に代入
- ⑥sql文の?に入れる値をsetする
- ⑦executeQuery()/executeUpdate()で実行（select文の場合はexecuteQuery()を使う）
- ⑧結果の処理（select文で取得した値をDTOに格納）
- ⑨con.close()で接続を切る

Actionクラスの作成

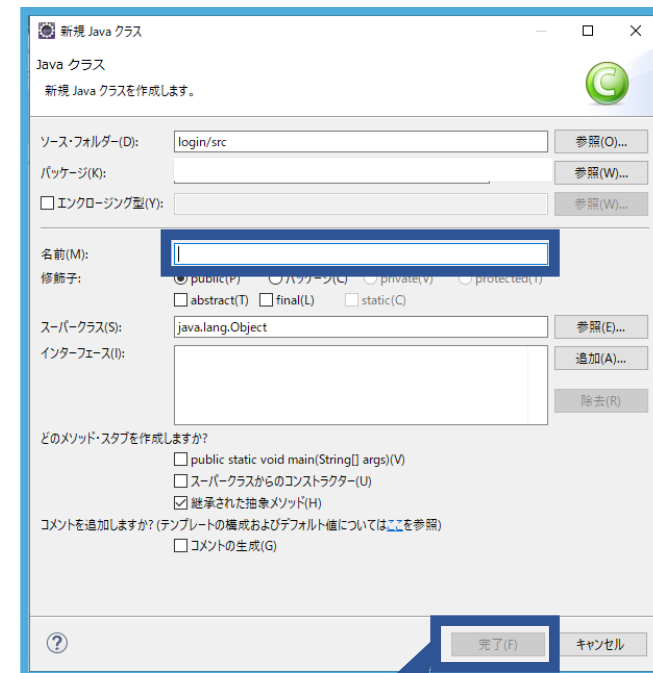
解説

Actionクラスでは、画面から送られてきたリクエストを取得します。
内部処理に応じてDAOやDTOクラスを呼び出し、最終的に次のJSPへ値を返します。

3 Actionクラス



① 「com.internousdev.login.action」を右クリックし、「新規」「クラス」を選択します。



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

Actionクラスの作成

LoginAction(javaファイル)

```
package com.internousdev.login.action;  
  
import java.sql.SQLException;  
import com.internousdev.login.dao.LoginDAO;  
import com.internousdev.login.dto.LoginDTO;  
import com.opensymphony.xwork2.ActionSupport;
```

```
public class LoginAction extends ActionSupport {
```

```
    private String name;  
    private String password;
```

struts2が持つActionSupportというクラスを継承します。
(Actionクラスは基本的にこのクラスを継承します)

フィールド変数
JSPから受け取る値、ここではnameとpassword を定義します。

※必ずJSPでの定義と同じ名前にします！

次へ続きます。

Actionクラスの作成

前の続きです。

```
public String execute() throws SQLException {
```

```
String ret = ERROR;
```

メソッド名は「execute」にします。

メソッドの戻り値「ret」を定義し、初期値としてERRORを代入します

```
LoginDAO dao = new LoginDAO();  
LoginDTO dto = new LoginDTO();
```

```
dto = dao.select(name,password);
```

JSPから送られてきたnameとpasswordを引数として、LoginDAOクラスのselectメソッドを呼び出します。その後、DAOで取得した結果をLoginDTOに代入します。

```
if(name.equals(dto.getName())) {  
    if(password.equals(dto.getPassword())){  
        ret = SUCCESS ;
```

ユーザーが入力した「ログインID」(name)と「パスワード」(password)が、DTOからもってきた値(dto.getName())と(dto.getPassword())にそれぞれ一致するか確認をします。

```
    }  
}  
return ret;  
}
```

if文の条件を満たした場合、戻り値「ret」の内容をSUCCESSに書き換えます。

次へ続きます。

Actionクラスの作成

前の続きです。

```
public String getName() {  
    return name;  
}  
public void setName(String name) {  
    this.name = name;  
}  
public String getPassword() {  
    return password;  
}  
public void setPassword(String password) {  
    this.password = password;  
}  
}
```

setName、setPassword(setter) を定義することで、JSPでユーザーが入力したnameとpasswordの値がそれぞれのフィールド変数に格納されます。

※このログイン認証機能の課題では、getNameとgetPasswordは使われませんが、次画面に値を引き渡すサイトの場合、getterが必要です。後々の不具合を防ぐため、現段階ではgetterとsetterは両方書くようにしてください。

- ① setterを定義することで、JSPでユーザーが入力した値がフィールドに格納される
- ② execute()メソッドを定義
- ③ 条件分岐でSUCCESSかERRORかを定める
(ここでは、ユーザーがJSPで入力した値とDTOに格納してある値を比較している)
- ④ execute()メソッドの結果 SUCCESS、ERRORを返す
(それにより、あらかじめstruts.xmlに遷移先として定義したそれぞれのJSPに振り分けられる)