その手は菓子である
ーeclipseによるstruts利用ー

岐阜経済大学　経営学部 経営情報学科　井戸　伸彦

来歴：　0．0版　２００４年１０月２９日
１．0版　２００５年2月１７日（strtus1.2対応、JavaBeans説明を削除）
２．0版　２００８年2月1日（eclipse3.3, strtus1.38対応）

スライドの構成

はじめに

■本スライドは、eclipse3.3、および、Tomcat6を用いたJavaサーブレット作成にて、strutsを利用する手順について記します。

■次の3つのスライドを学習済みであることを前提としています。

- 「月に吠える　ーeclipseを用いたJavaアプリケーションの作成ー」
  - eclipseの起動方法等は、このスライドを参照してください。
- 「ただ一疋の青い猫のかげ　ーeclipseを用いたJavaサーブレットの作成ー」
  - Eclipse3.3にて、JSP/サーブレットを作成します。
- 「されど我らが日々　ーJavaサーブレット入門ー」
  - 簡単なサーブレットを作成するときの方法を説明しています。

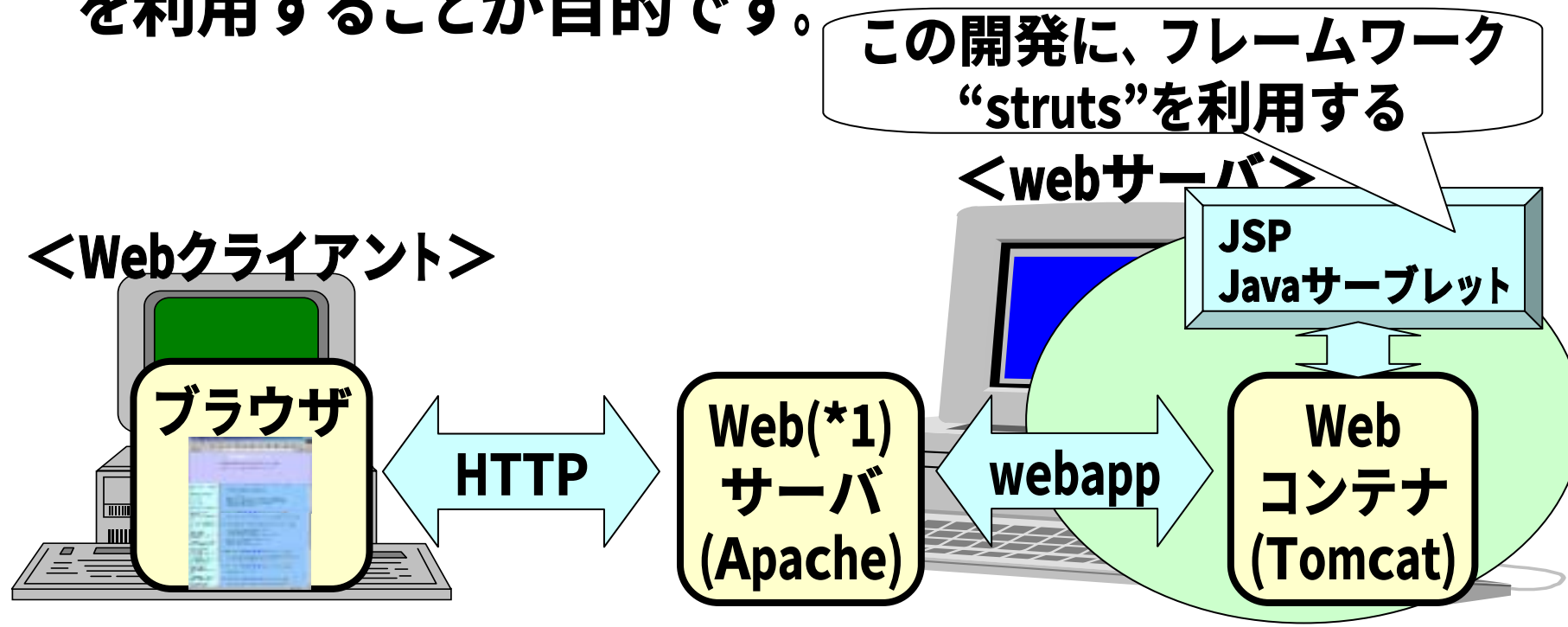
■岐阜経済大学では、全WindowsPCに、eclipse3.3、Tomcat6はインストールされています。本スライドでは、これらでの環境での操作を記します。

■井戸ゼミではstrutsの参考書として、次の本を使っています。本文中の参考書とは、次の本を指しています。

- 「10日でおぼえるJakarta入門教室」、山田祥寛、翔泳社

(1．1) 何がやりたいのか？

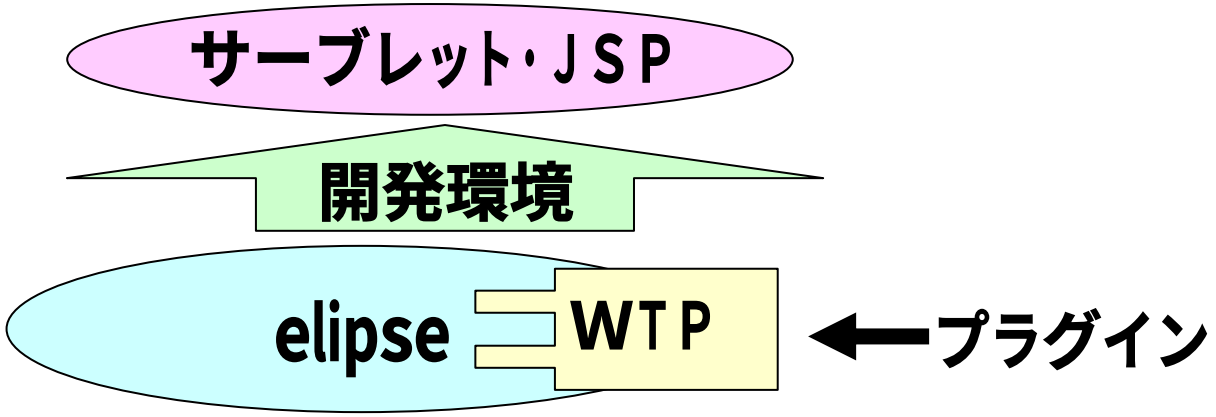
■下図のようなシステム構成で動作するサーブレットとＪＳＰをeclipse上で作成する際、フレームワーク“struts”を利用することが目的です。



- (＊1) 大学のＰＣに設定した環境ではApacheは実装せず、Tomcatの持つＷｅｂサーバ機能を利用しています。

(1．2) プラグイン

■スライド「eclipseを用いたＪａｖａサーブレットの作成」に記したとおり、Javaサーブレットをeclipse上で作成する場合、プラグインＷＴＰを利用することが一般的です。



■eclipse上でstrutsを利用するためのプラグインには、”StrutsIDE”などがありますが、本スライドではそのようなプラグインは使用しません。

(1．3) 準備

■次のサイトからファイルをダウンロードして解凍しておきます。

- http://struts.apache.org/download.cgi

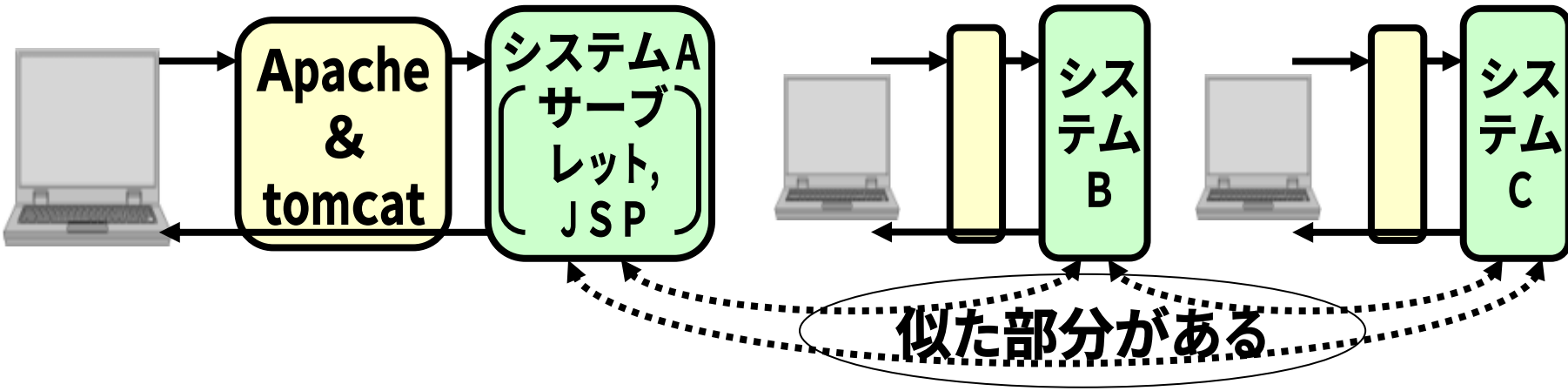
ファイル名は次のとおりです。

- struts-1.3.8-all.zip

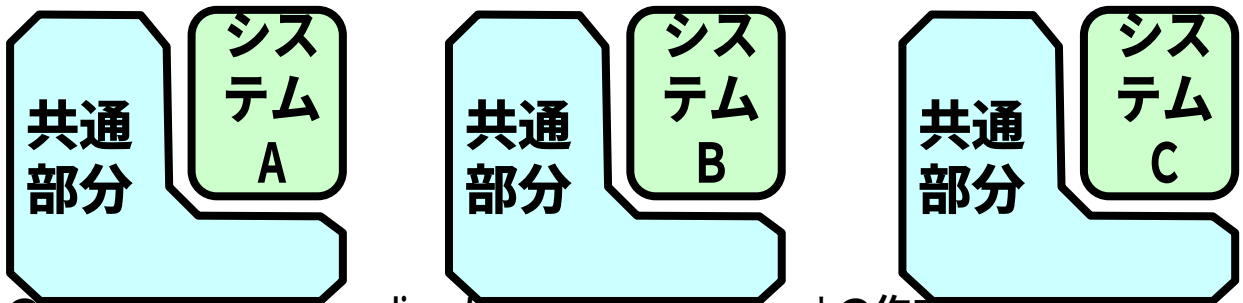
■岐阜経済大学内のＰＣで準備が済んでいます。次のフォルダに解凍したファイルがあります。

(2．1) フレームワーク

■tomcat等のWebコンテナ上で動作する、Javaサーブレット・JSPによるシステムをいろいろ開発すると、似たような部分があるシステムが出来ることになります。

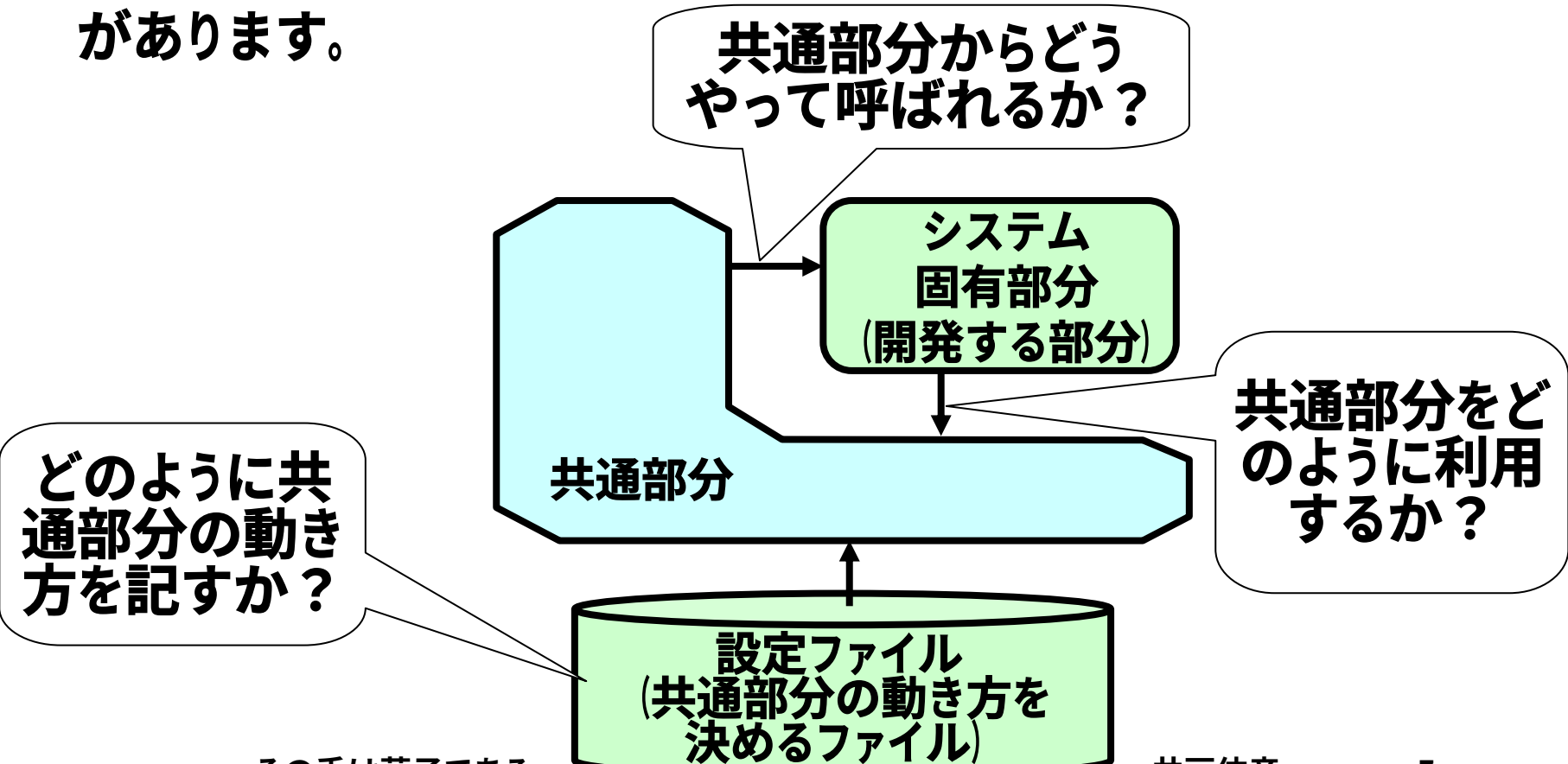


■どのシステムでも“同じ作り方”(これについては後述)をするとすれば、共通部分は1回作ればよい訳です。



(2．2) フレームワークの利用

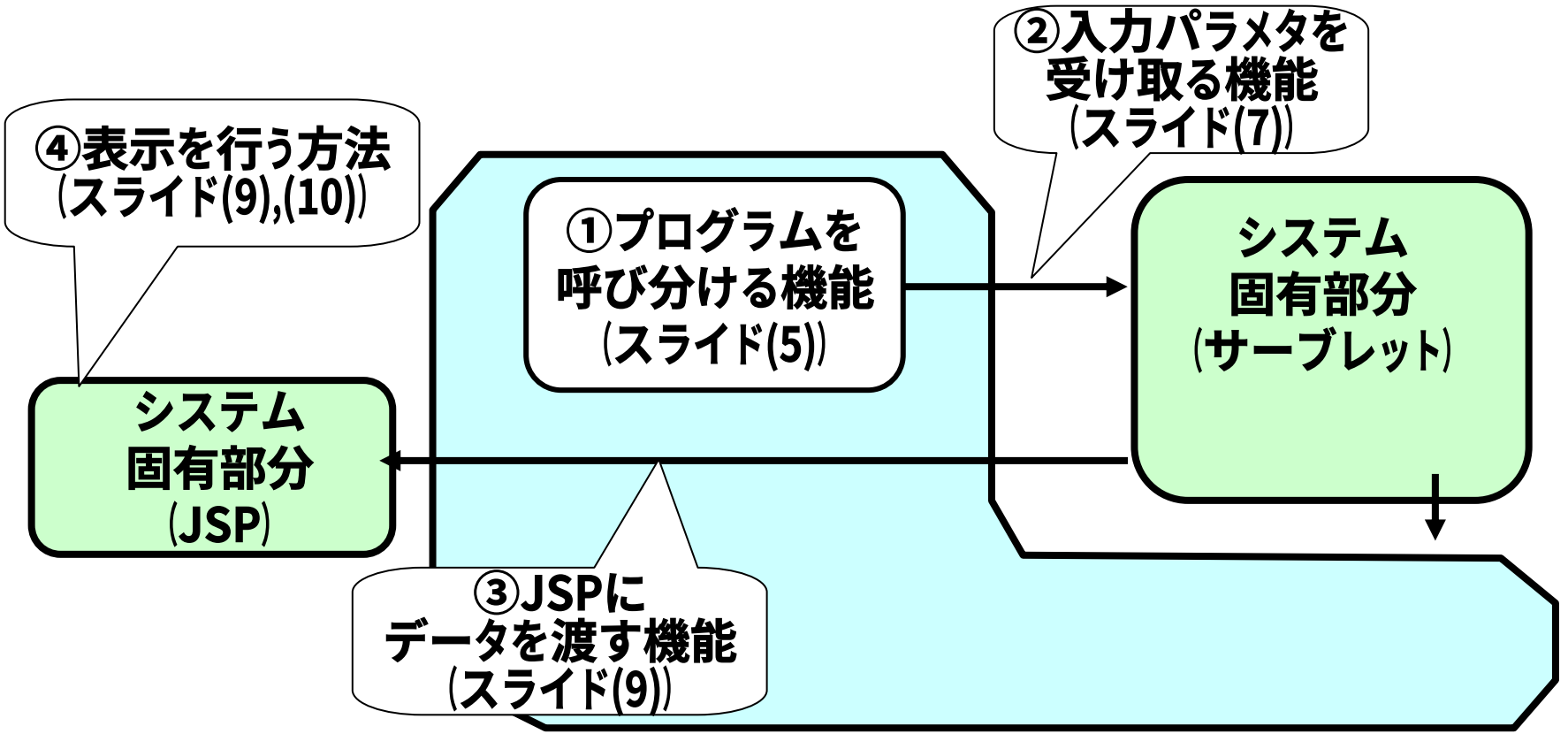
- “同じ作り方”を前提に、共通部分を供給するのが、strutsのようなフレームワークです。フレームワーク(共通部分)を利用すれば、作成するのはシステムに固有な部分だけです。
- システム固有部分を作成する際には、次の3点を理解する必要があります。



（2．3）strutsの機能

■strutsはフレームワークとして、さまざまな機能を提供しています。

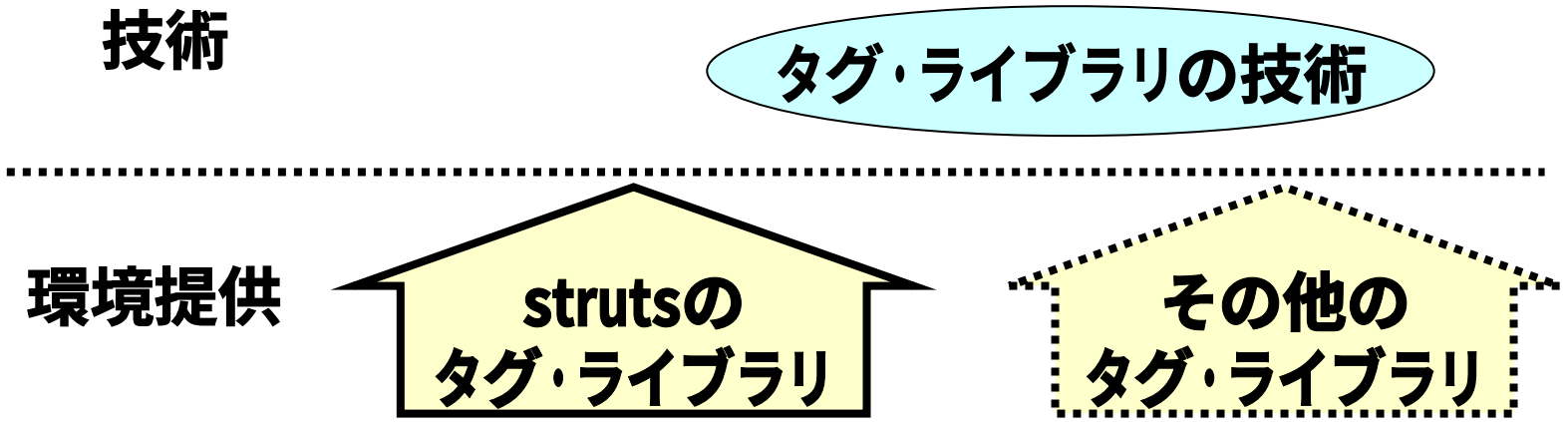
■本スライドで扱うのは、次の機能のみです。



(2．4) タグ･ライブラリ

■タグ･ライブラリについても記しています。

■ＪＳＰで利用するタグ･ライブラリは、strutsだけの技術ではありません。このスライドで説明するタグ･ライブラリは、strutsが提供するタグ･ライブラリです。



(3) プロジェクトの作成

■次の資料にしたがって、動的Webプロジェクトを作成します。

- 別資料「ただ一疋の青い猫のかげ　－eclipseによるJavaサーブレット作成－」(2．1)～(2．6)

■今回作成したのは、次のような名前のプロジェクトです。

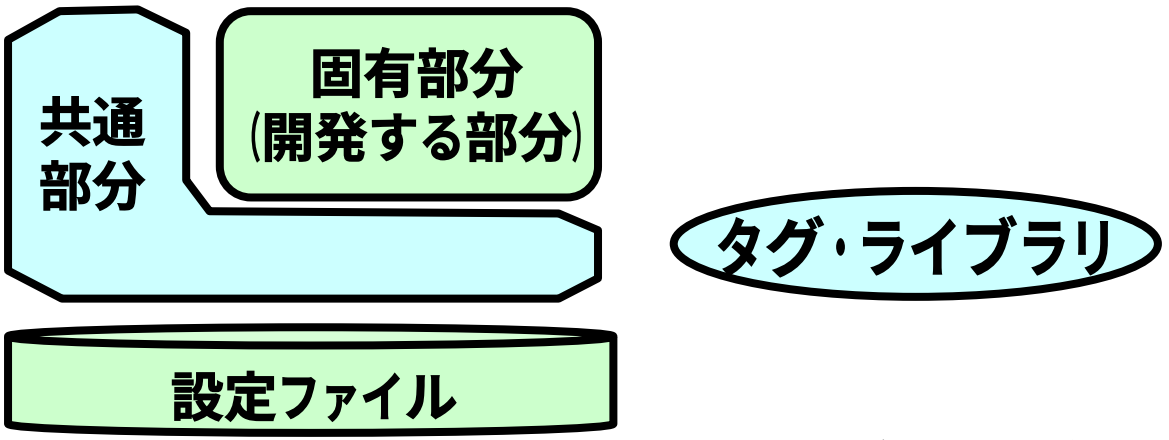
- プロジェクト　：　“strutsTest”

(3．1) 必要なファイルの配置

■strutsを利用してシステムを開発するとき、共通部分にあたるファイルをコピー等により配置する必要があります。また、設定ファイルや固有部分なども、同様に配置する必要があります。

■これらのファイルを一から作成していっても良いのですが、strutsにはアプリケーションとして動くようになっている一群のファイルが雛形として用意されています。

■eclipse上では、このような雛形をインポートすれば、必要なファイルの配備はすべて終わり、後は開発部分を作っていけばよい状態になります。



(3．2) 雛形のインポート

■strutsには、雛形のアプリケーションが幾つも用意されています。作成するアプリケーションによってそれらをインポートすれば、どのようにプログラムを作成して行けばよいかの道標になります。

■今回は、ほとんど空 (Blank) の雛形である、次のディレクトリ配下アプリケーションをインポートしていきます。

C:\eclipse\struts-1.3.8\apps\struts-blank-1.3.8

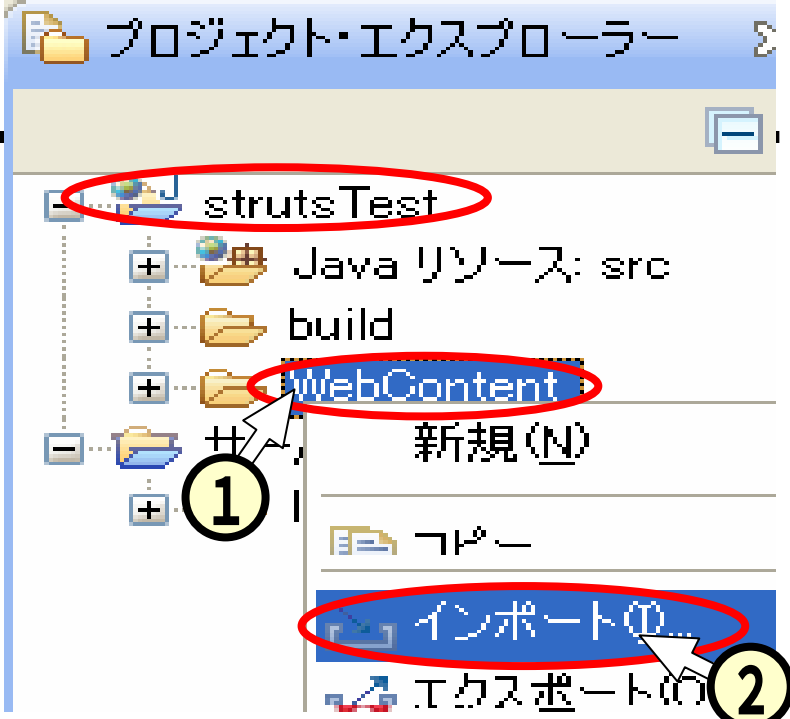
※ 上記のフォルダは、大学のパソコンについて井戸がコピーしておいたものです。一般的なeclipseの環境に上記のフォルダにある訳ではありません。自分のＰＣで行う場合には、該当するファイルを次のサイトからダウンロードして入手してください。

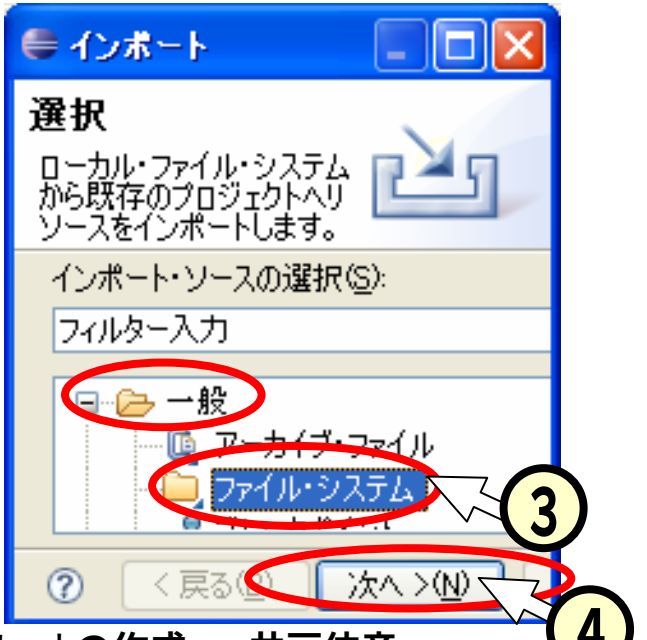
http://struts.apache.org/

(3.3.1) インポート (1 / 3)

■「プロジェクト・エクスプローラ」中、作成した[strutsTest]-[WebContent]を右クリック (①) して、ポップアップメニューから、[インポート]をクリック (②) します。

■「インポート：選択」のウインドウにて、[一般]-[ファイルシステム]をクリック (③) し、[次へ]をクリック (④) します。

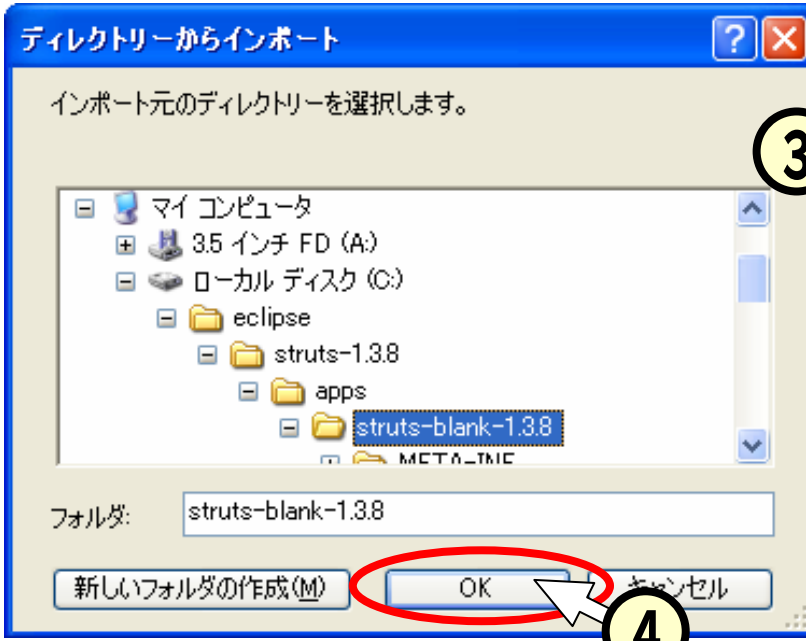




(3．3．2) インポート (2/3)

■「インポート：ファイルシステム」のウインドウにて、[ソース・ディレクトリ]の欄 (①) を埋めるために、次の操作をします。

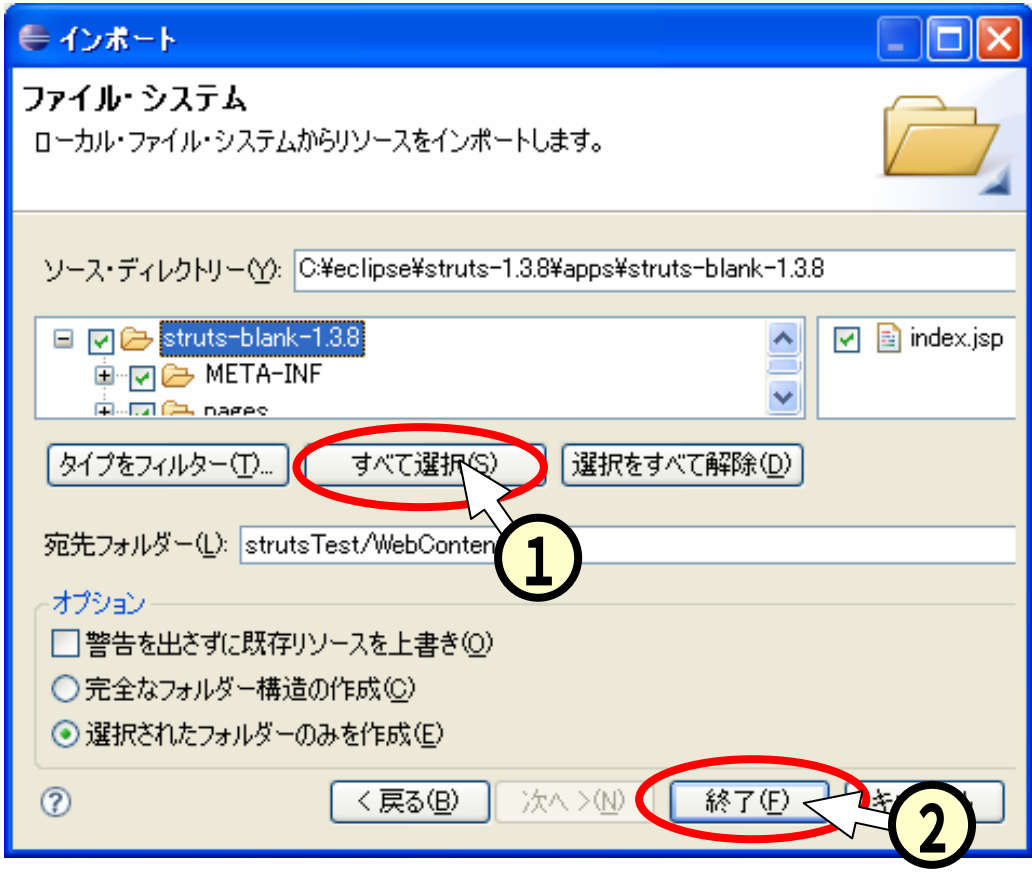
- 右の[ブラウズ]ボタンをクリック (②) する。
- 「ディレクトリからインポート」のウィンドウ (③) にて、次のフォルダを選択し、[OK]をクリック (④) する。
- C:\eclipse\struts-1.3.8\apps\struts-blank-1.3.8

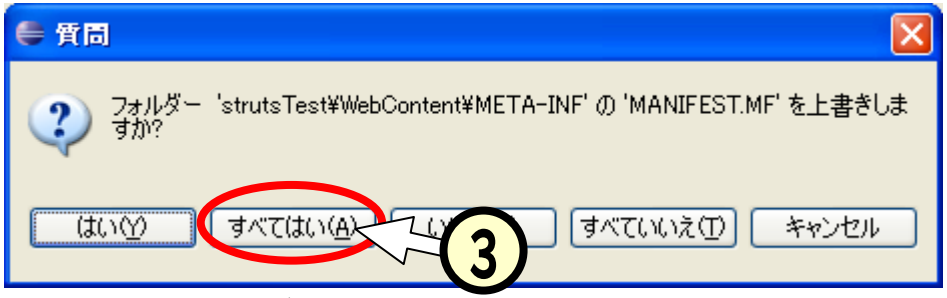


(3．3．3) インポート (3/3)

■「インポート：ファイルシステム」のウインドウにて、[すべてを選択]をクリック (①) し、[終了]をクリック (②) します。

■上書きの確認のウインドウが表示されますが、[すべて「はい」]をクリック (③) します。



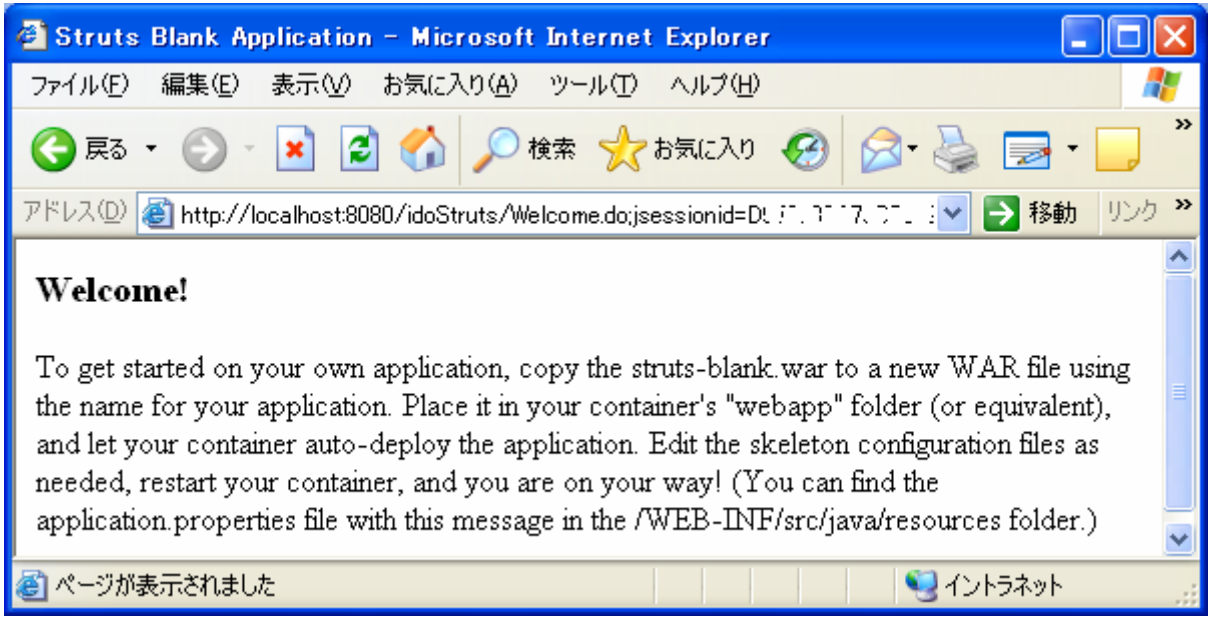


(3．4)動作確認

■正常にstruts-blankが動作するかを、次のＵＲＬにアクセスして確認します(Tomcatの立ち上げを忘れずに！)。

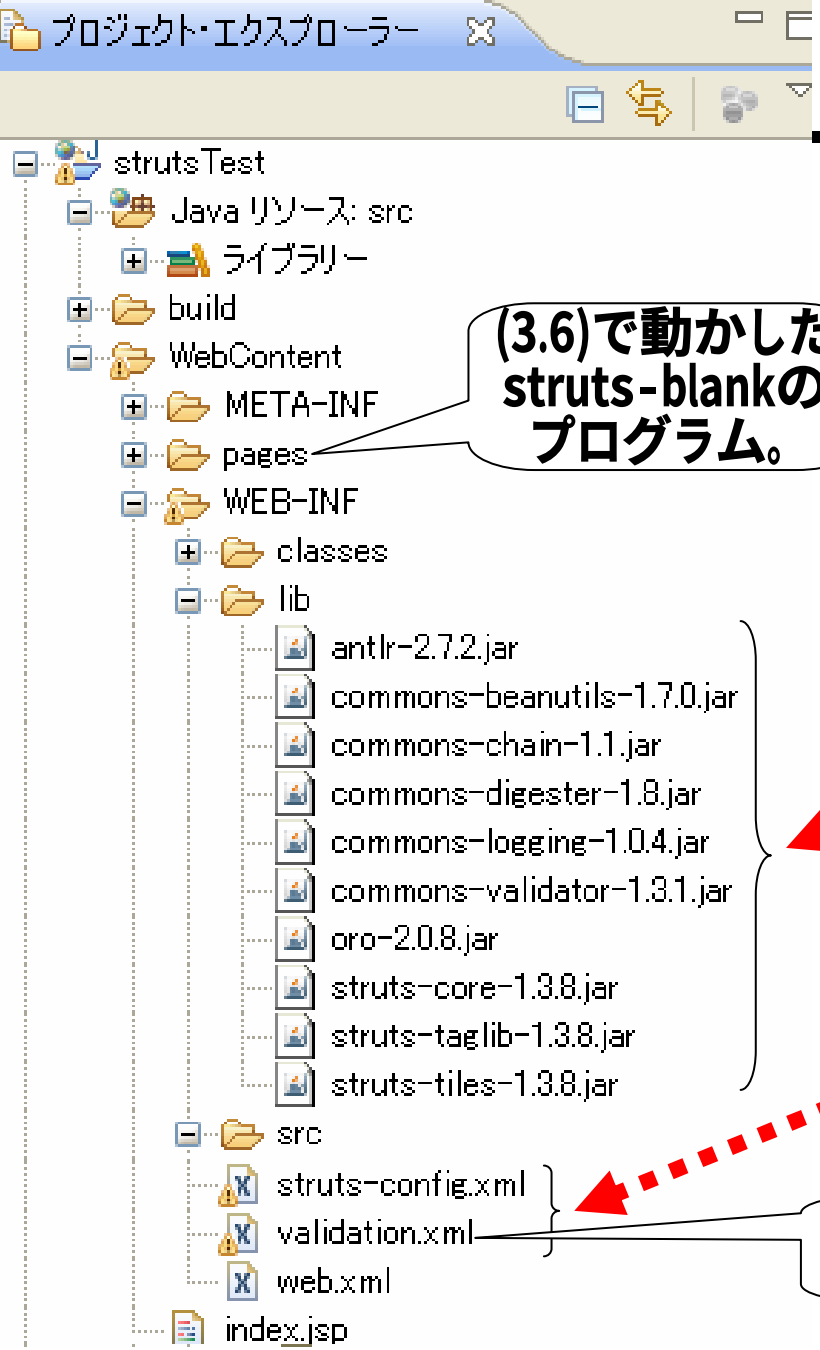
- http://localhost:8080/inquery/index.jsp

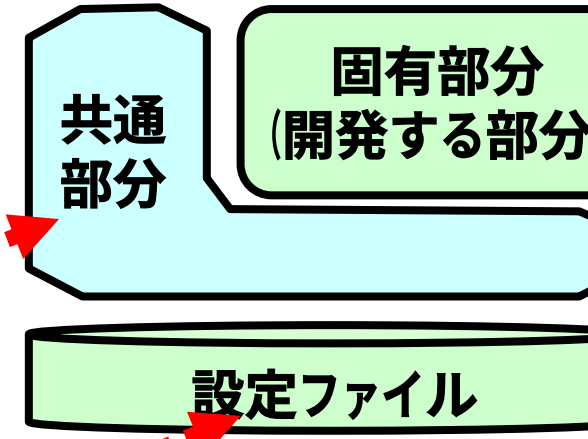
■リクエストは別のＪＳＰに送られ、次のようなメッセージが表示されます。



(3．5) ファイル構成

■インポートすることにより
配置された主なファイルは、
図のようになっています。

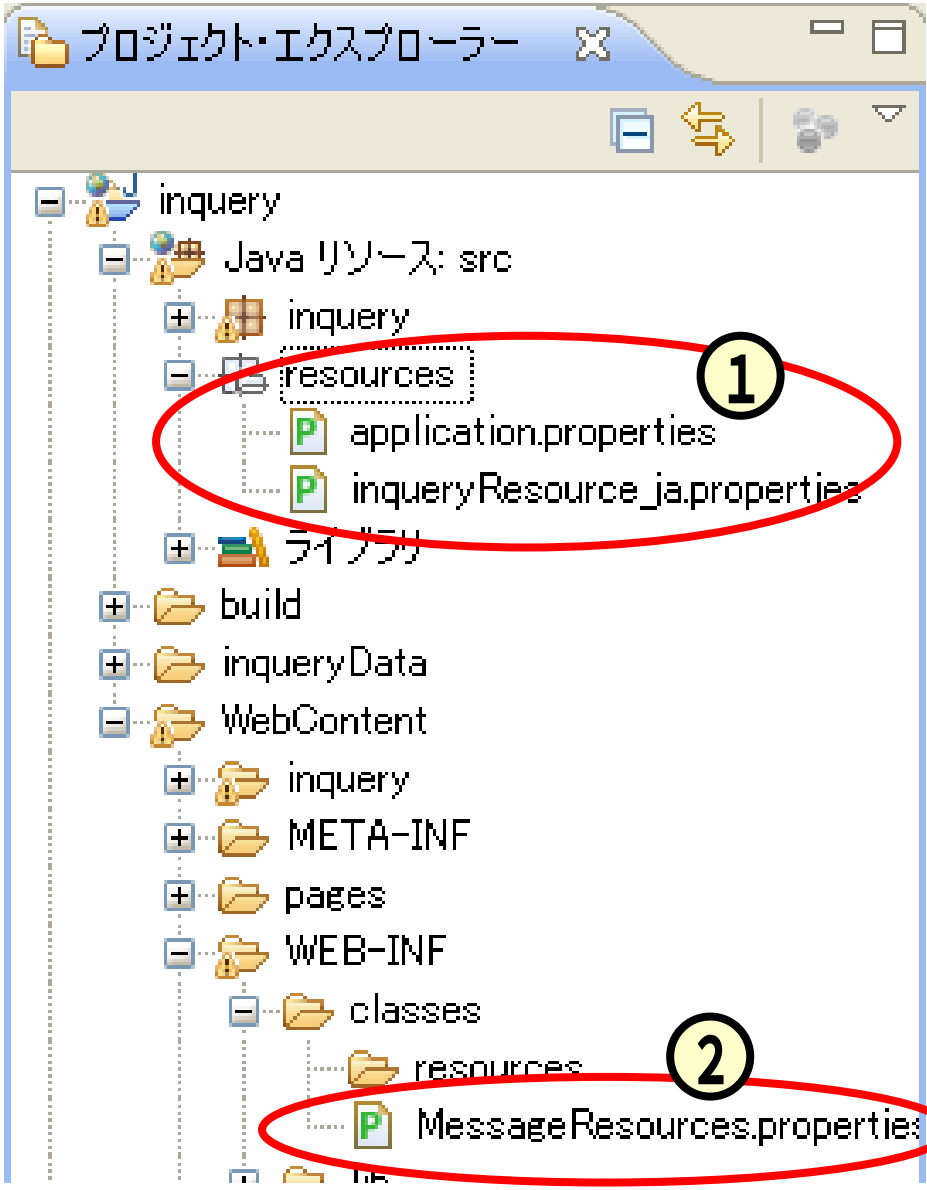




(3．6) リソースファイル

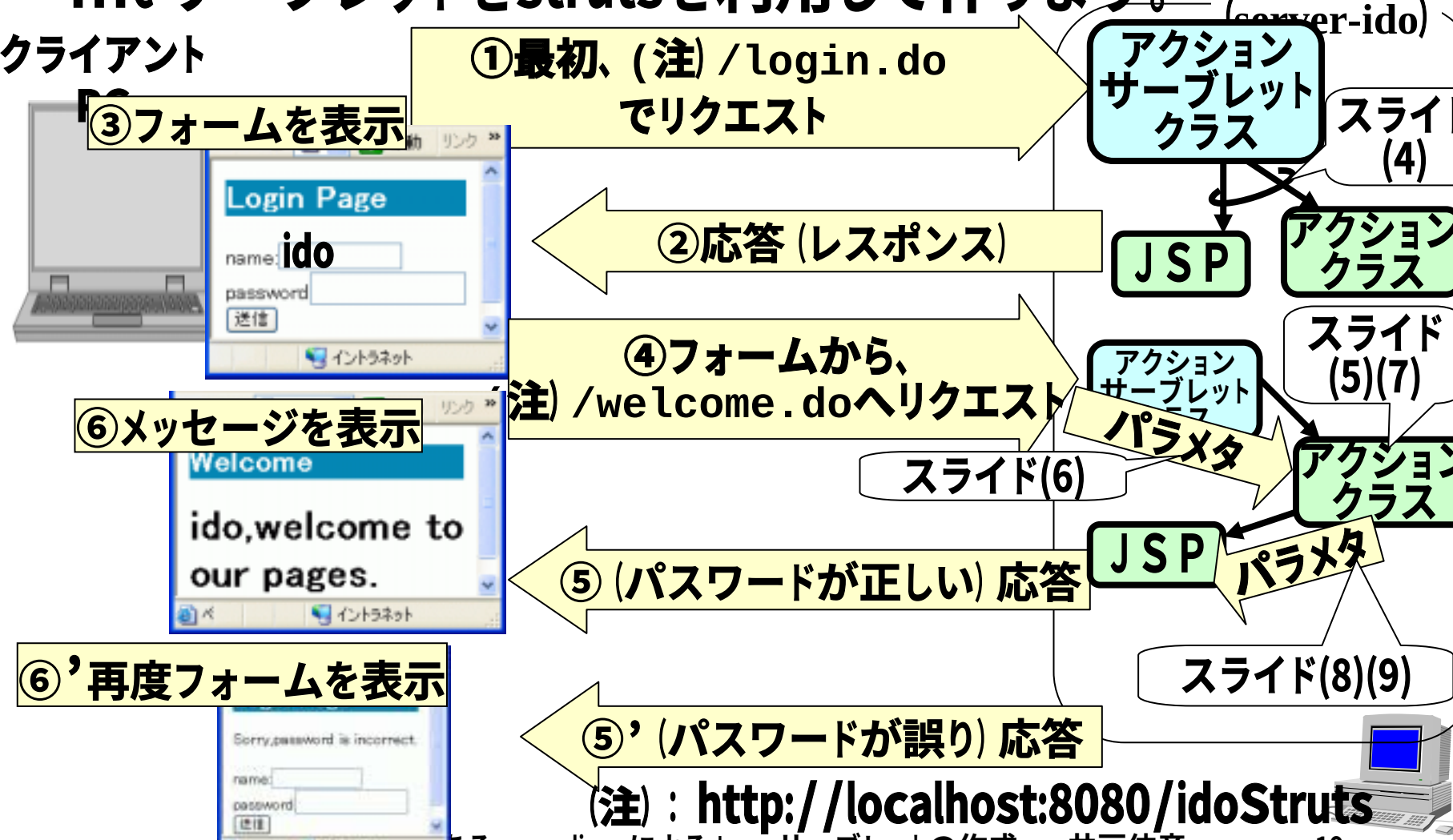
■リソースファイルの配置

- 右図のように、[Javaリソース:src]フォルダの中にパッケージ(右図では"resources"、①)をつくり、この中に配置します。
- Eclipseはパッケージなし(すなわち、デフォルトパッケージ)を許してくれないので、strutsのブランクのサンプルにあるリソースファイル(②、MessageResource.properties)は、このフォルダにおくことができません。



(4) Welcomeサーブレット

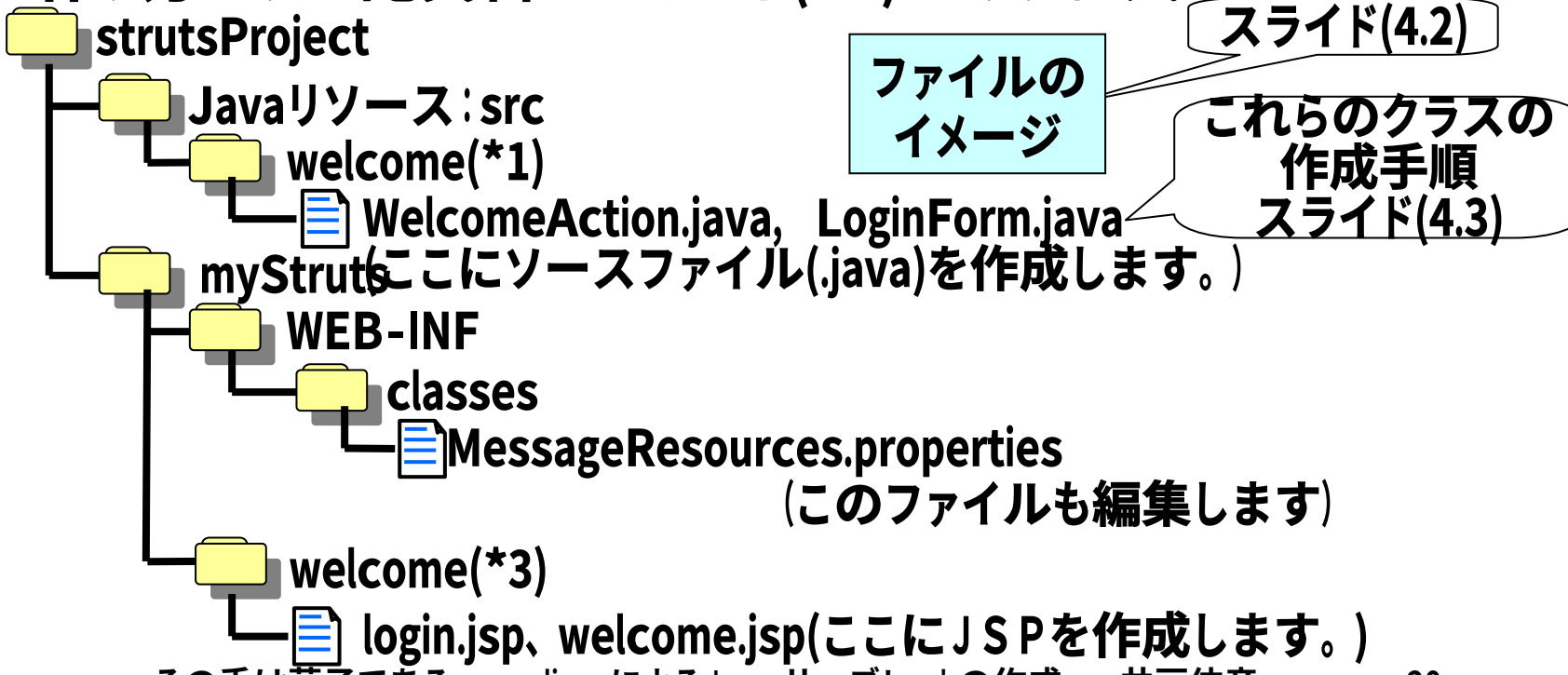
■スライド「Javaサーブレット入門」で作成した、Welcomeサーブレットをstrutsを利用して作ります。



(4．1) パッケージ、フォルダの作成

■パッケージ名は、“welcome”にします。
- eclipseでのパッケージの作り方は、次の資料にあります。
  - ◆されど我らが日々　－サーブレット入門－ スライド (6.4)
- パッケージを作成すると、下図のフォルダ(*1)(*2)が出来ます。

■ＪＳＰ用のフォルダ“welcome”(*3)を作っておきます。
- 作り方は、上記資料のスライド(6.7)にあります。



(4．2．1) login.jsp

```
<%@ page language="java" contentType="text/html; charset=windows-31j"
    pageEncoding="windows-31j"%>
<%@ taglib uri="http://struts.apache.org/tags-html" prefix="html" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html:html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-
31j">
<title>login page</title>
</head>
<body>
<h2 style="color:white;background-color:#0086b2;">
        Login Page</h2>
<html:errors />
<html:form action="/welcome">
name:<html:text property="username" size="15" /><br>
password<html:password property="passwd" size="15" /><br>
<html:submit property="submit" value="送信" /><br>
</html:form>

</body>
</html:html>
```

スライド(5.2)

スライド(6.3)

（4．2．2）welcome.jsp

```
<%@ page language="java" contentType="text/html; charset=
    pageEncoding="windows-31j"%>
<%@ taglib uri="http://struts.apache.org/tags-bean" prefix="bean" %>
<%@ taglib uri="http://struts.apache.org/tags-html" prefix="html" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html:html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=windows-
31j">
<title>Insert title here</title>
</head>
<body>
<h2 style="color:white;background-color:#0086b2;">
        Welcome</h2>
<h1><bean:write name="indLoginForm" property="username" />
,welcome to our pages.</h1>
</body>
</html:html>
```

スライド(8)

スライド(8)

（4．2．3）WelcomeAction.java

スライド(7)

作成方法
スライド(4.3)

```
package welcome;
import javax.servlet.http.*;
import org.apache.struts.action.Action;
import org.apache.struts.action.*;
public class WelcomeAction extends Action {
  private static final String sitePasswd = "aaaaaaa";
  public ActionForward execute(ActionMapping map,
    ActionForm form,HttpServletRequest request,
    HttpServletResponse response){
       LoginForm loginForm = (LoginForm)form;
       String inputPasswd = loginForm.getPasswd();
       if((inputPasswd==null)||(!inputPasswd.equals(sitePasswd))){
         ActionMessages errors = new ActionMessages();
         errors.add(ActionMessages.GLOBAL_MESSAGE,
                    new ActionMessage("errors.login"));
         saveErrors(request,errors);
         return map.findForward("fault");
       }
       if(loginForm.getUsername()==null){
         loginForm.setUsername("a Mr. or Ms. Unknown");
       }
       request.setAttribute("indLoginForm",loginForm);
       return map.findForward("success");
  }
}
```

スライド(9)

スライド(8)

スライド(5.8)

（4．2．4）LonginForm.java

スライド(6)

```
package welcome;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;
public class LoginForm extends ActionForm {
  private String username;
  private String passwd;
  public void setUsername(String username){
    this.username = username;     }
  public String getUsername(){
    return this.username;  }
  public void setPasswd(String passwd){
    this.passwd = passwd;  }
  public String getPasswd(){
    return this.passwd;    }
  public void reset(ActionMapping map,
                    HttpServletRequest request){

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

スライド(6.3)

編集方法
スライド(6.5)

(4．2．5) struts-config.xml (フォーム･ビーンズ)

■設定ファイル“struts-config.xml”には、様々な設定が記されています。

■次の部分は、フォーム･ビーンズについて記されています。

```
<!-- ================================== Form Bean Definitions -->

    <form-beans>
        <form-bean
            name="LoginForm"
            type="welcome.LoginForm"/>
```

スライド(6.2)

（4．2．6）struts-config.xml ーアクションマッピングー

■設定ファイル“struts-config.xml”中の次の部分には、アクションマッピングについて記されています。

```
<!-- ========================== Action Mapping Definitions -->
  <action-mappings>
    <action
       path="/login"
       forward="/welcome/login.jsp"/>
    <action
       path="/welcome"
       type="welcome.WelcomeAction"
       name="LoginForm" scope="request">
      <forward name="success" path="/welcome/welcome.jsp" />
      <forward name="fault" path="/welcome/login.jsp" />
    </action>
```

スライド(5.4)

スライド(5.7),(5.8)

（4．2．7）MessageResources.properties

```
# -- other --
errors.cancel=Operation cancelled.
errors.detail={0}
    :
errors.login=password is not correct.
    :
```

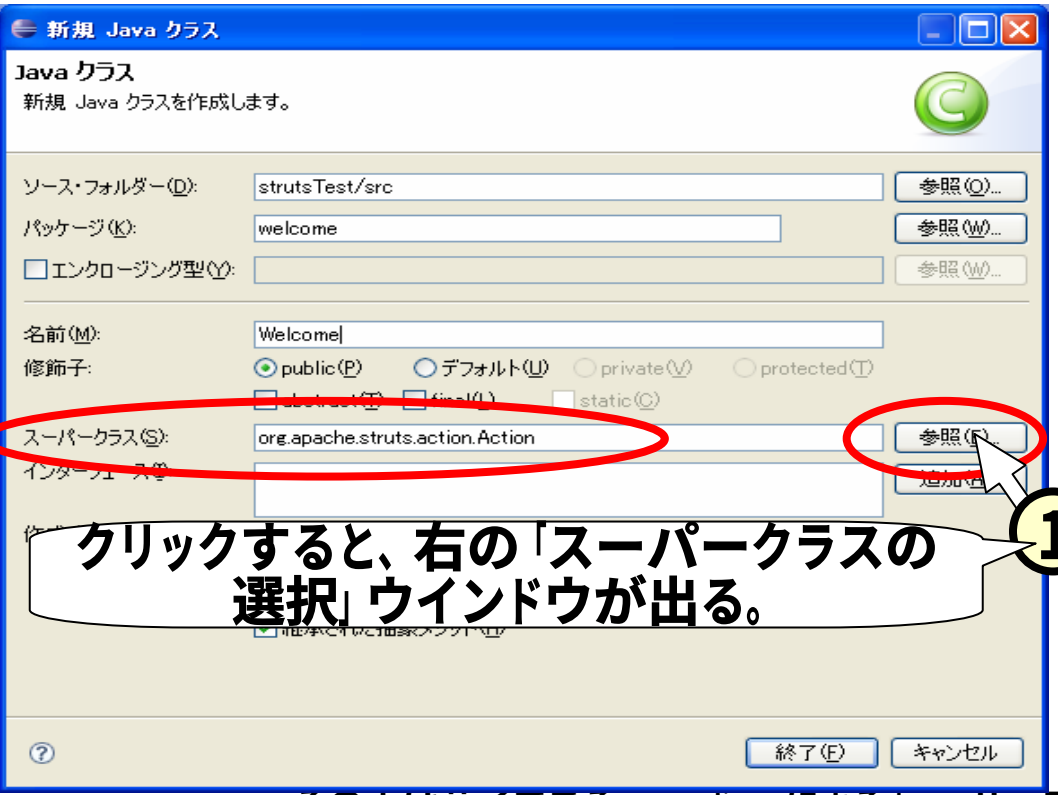
スライド(9.3)

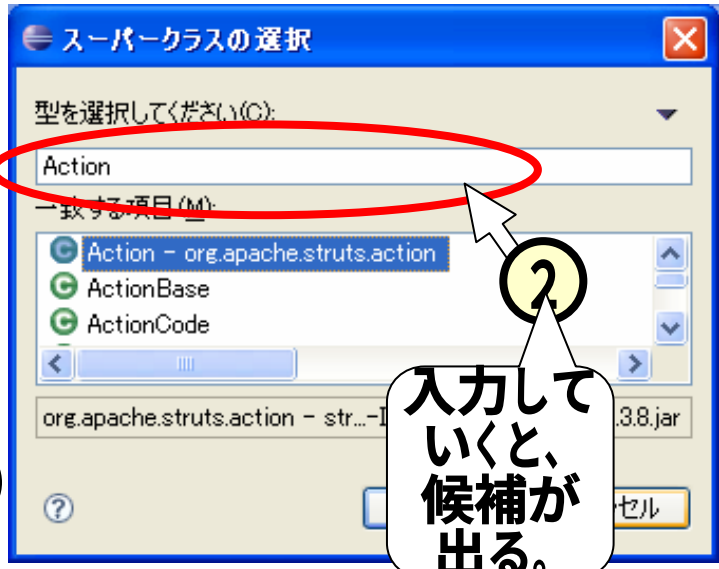
（4．3）アクション･クラスの作成

■WelcomeAction.javaの作成時に、スーパークラスとして次のクラスを用います。

- org.apache.struts.action.Action

■eclipseの場合、「新規Ｊａｖａクラス」のウインドウにて指定しておきます。

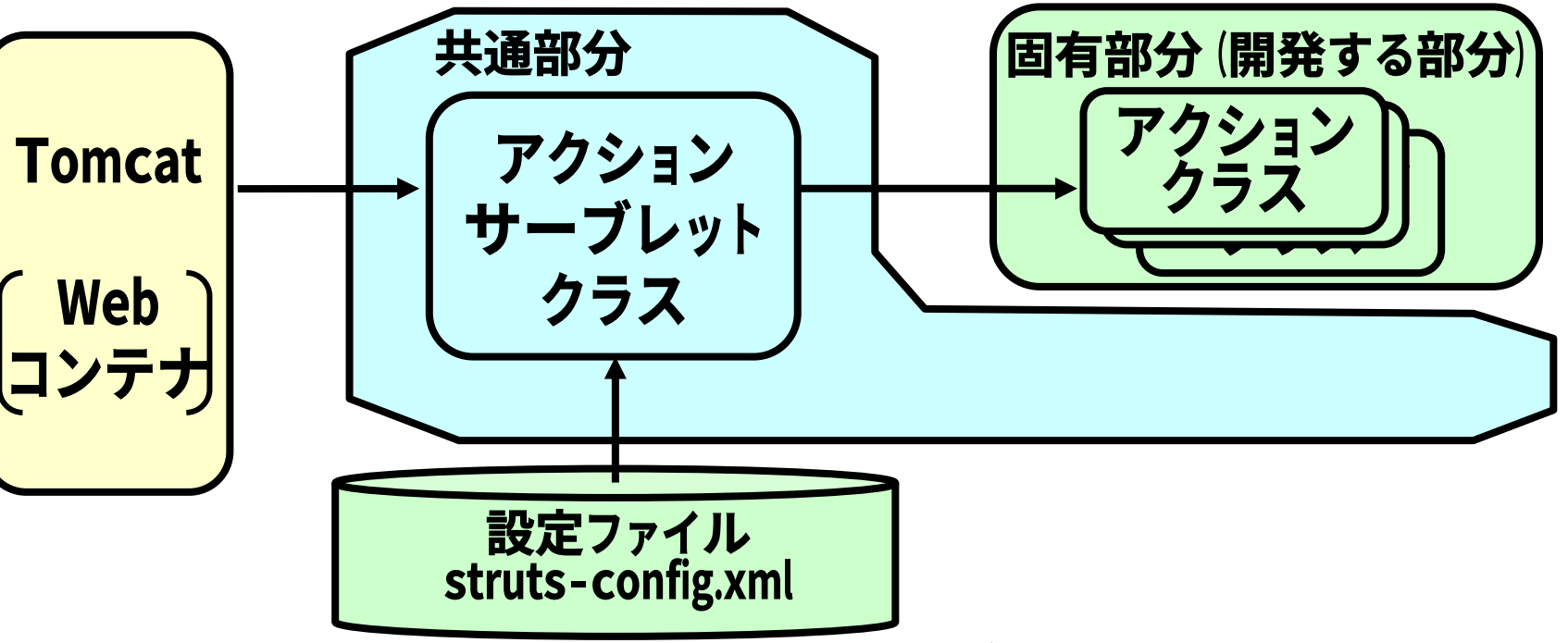




(5．1) アクション･サーブレット

■strutsでは、サーブレット本体 (HttpServletを拡張するクラス) は共通部分として提供されます。このサーブレットのことをアクションサーブレット(ActionServlet)と呼びます。

■アクションサーブレットがどのように動くかは、設定ファイルの“struts-config.xml”に記述します。



（5．2）web.xml

■ActionServletに特別なことはなく、我々が作成するサーブレットと同様に作成されています。

■ActionServletは、“xxxx.do”という形のＵＲＬで呼ばれますが、これはweb.xmlにそのように記しているからであり、特別なことではありません（web.xmlの内容は、struts-blankをインポートした際に設定されています）。

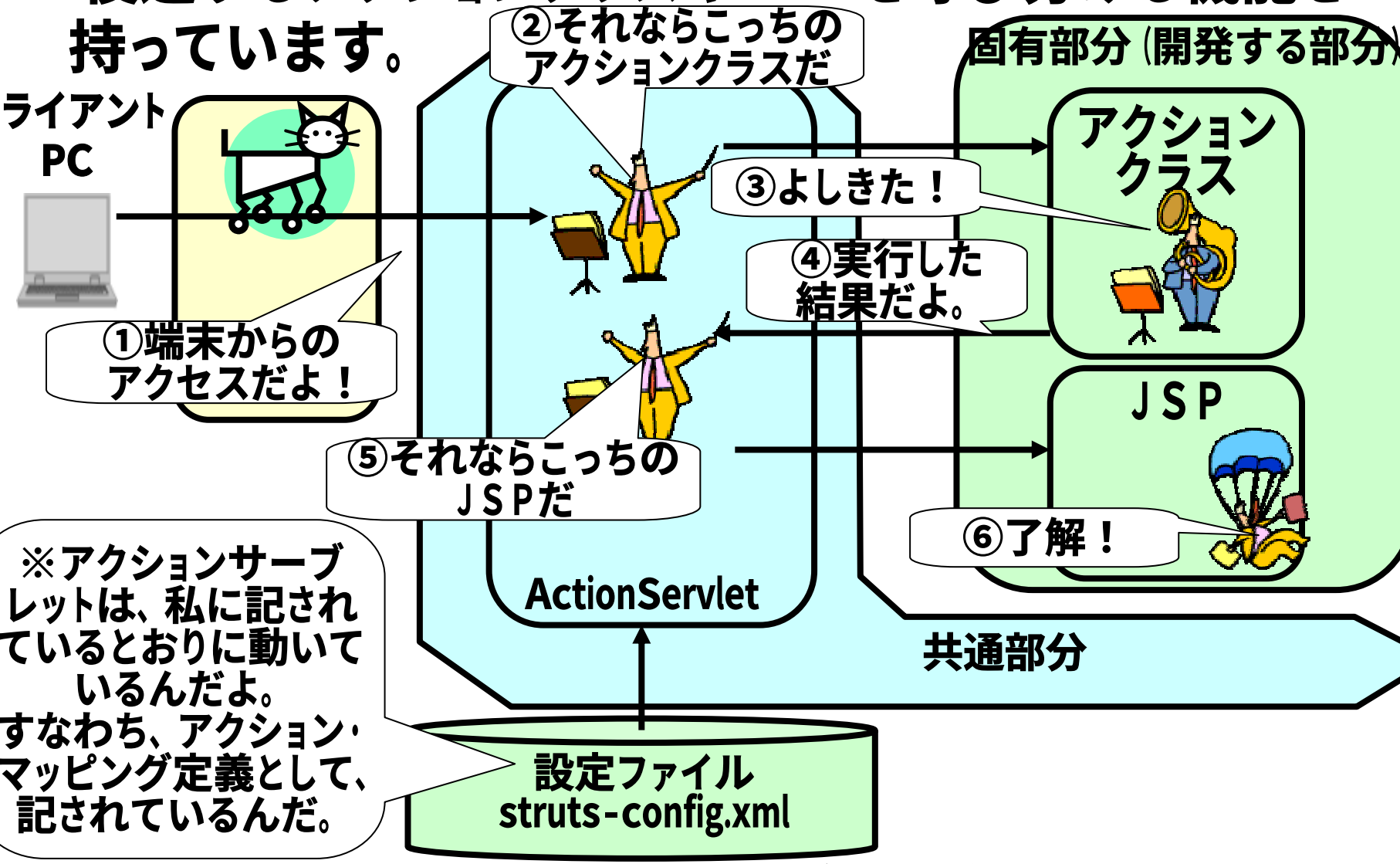
```
<!-- Standard Action Servlet Configuration (with debugging) -->
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>
      org.apache.struts.action.ActionServlet
    </servlet-class>
  </servlet>
  <!-- Standard Action Servlet Mapping -->
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>
```

共通部分としてstrutsが供給するアクションサーブレットクラス

“xxx.do”という形にマッチするＵＲＬ

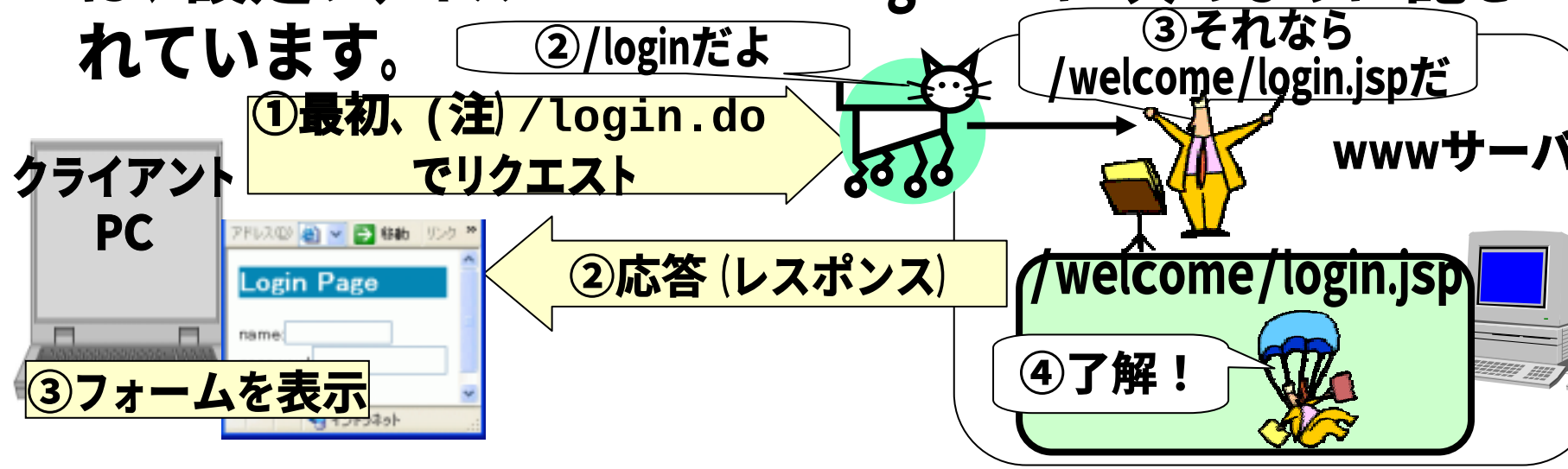
(5．3) 呼び分ける機能(アクション･マッピング)

■後述するアクションクラスやＪＳＰを呼び分ける機能を持っています。



(5．4) アクションマッピング定義(struts-config.xml)

■最初にhttp://localhost:8080/strutsTest/login.doにアクセスがあった時、login.jspを起動して表示を行うことは、設定ファイル“struts-config.xml”に次のように記されています。




```
<!-- ============================ Action Mapping Definitions -->
  <action-mappings>
    <action
       path="/login"
       forward="/welcome/login.jsp"/>
```

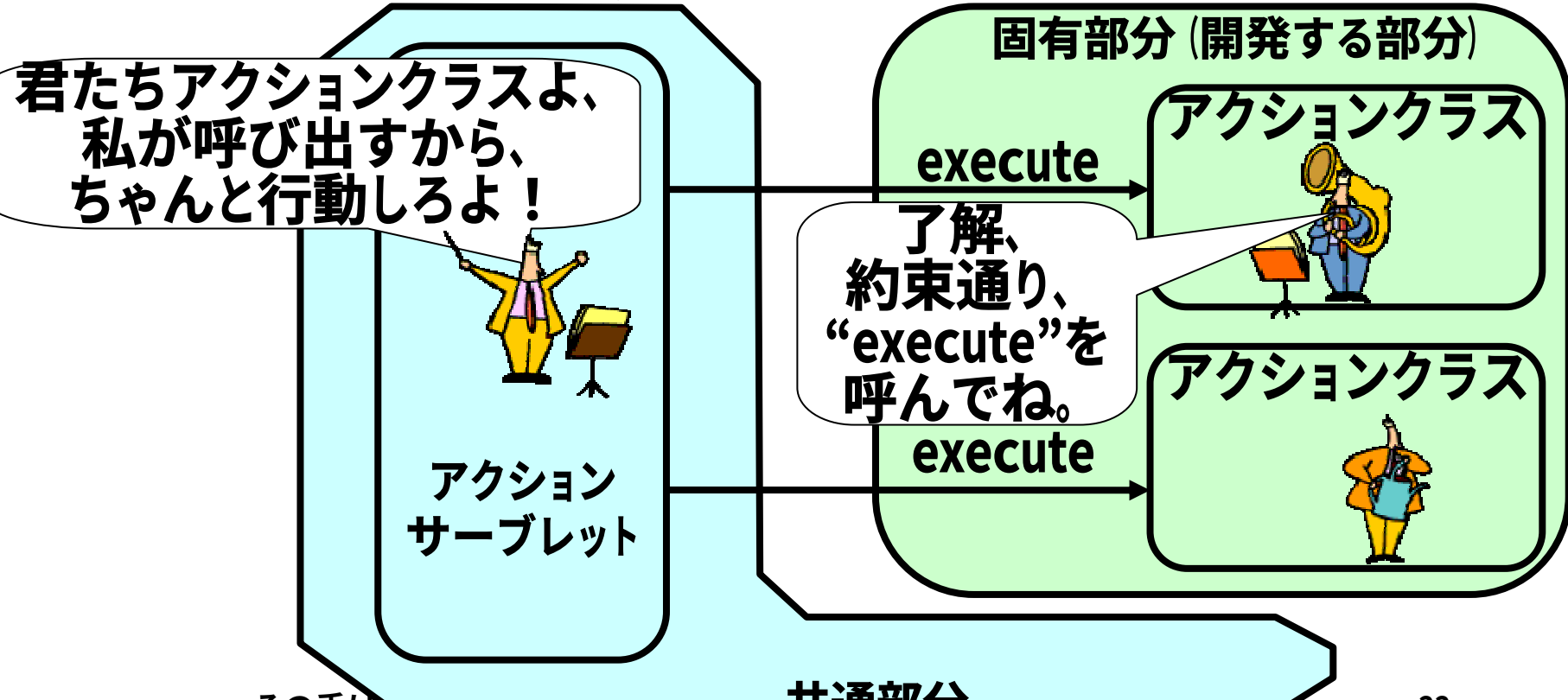
②/loginだよ

③それならwelcome/login.jspだ

(5．5) アクション･クラス

■アクション･クラスは、アクションサーブレットの呼び出しに応えて処理を実行するクラスで、次のクラスを継承します。　org.apache.struts.action.Action

■アクションサーブレットからの呼ばれ方（＝メソッド）は、メソッド“execute”の形式に決められています。



(5．6) メソッド“execute”

■アクション･クラスのメソッド“execute”は、次のような形をしています。

```
public ActionForward execute(
                ActionMapping map,
                ActionForm form,
                HttpServletRequest request,
                HttpServletResponse response ){
                    :
    return map.findForward("success");
}
```

入力パラメタ

リターン値

■上記の入力パラメタとリターン値とを使って実現される機能のうち、次の二つについて説明します。

- 実行結果により、次の処理を決める機能。
- クライアント端末からリクエスト中のパラメタを、アクションクラスに渡す機能（スライド(7)以下）。

（5．7）WelcomeAction実行結果による呼び分け

■アクションクラス“WelcomeAction.java”では、パスワードの判定結果により、異なるＪＳＰを表示します。

(5．8)“struts-config.xml”と“execute”

■前スライドの処理は、次のように記述されます。

<struts-config.xml：アクション・マッピング定義の部分>

```
<!-- ========== Action Mapping Definitions -->
  <action
     path="/welcome"
     type="welcome.WelcomeAction“>
     <forward name="success" path="/welcome/welcome.jsp" />
     <forward name="fault" path="/welcome/login.jsp" />
  </action>
```

②/welcomeだよ

③それならアクションクラスのwelcome.WelcomeActionだ

⑥じゃあ、welcome.jsp行け！

⑥’じゃあ、login.jsp行け！

<welcome/WelcomeAction.java>

```
package welcome;
                :
public class WelcomeAction extends Action {
                :
   public ActionForward execute(.......){
                :
          return map.findForward("fault");
                :
         return map.findForward("success");
    }
}
```

④了解！

⑤’失敗！
(“fault”)

⑤成功！
(“success”)

(5．9) アクション・マッピング定義中のパス

■設定ファイル“struts-config.xml”中のアクションマッピング定義中では、次の2つのパスが用いられます。

- ①action要素中の属性のpath
- ②forward要素中の属性のpath

```
<action
  path="/welcome"                    ①
  type="welcome.WelcomeAction“>
  <forward name="success" path="/welcome/welcome.jsp" />
  <forward name=“fault”   path=“/welcome/login.jsp”   />
</action>                                              ②
```

■①のパスには、端末からのリクエストＵＲＬに結びついた、“/”から始まる一意の名前で記す必要があります。

■②には、次のいずれかが入ります。

- ＪＳＰへのパス (/inqueryから見たパスを、“/”をつけて記述)
- tilesで定義されたパス (これについては、別スライドで記します)。
- ①のパス(“.do”を付ける)。すなわち、上記の例では、次のように記述しても同じになります。

```
<forward name=“fault”   path=“/login”   />
```

(5．10) リクエストから直接ＪＳＰへ振る場合

■スライド(5.4)で示したアクション･マッピングは、次の点でやや特殊でした。

- 端末からのリクエストに対して、アクション･クラスではＪＳＰにて表示を行う以外の処理は特にない。

■このような場合にいちいちアクション･クラスを作るのは面倒なので、実際にはあらかじめ用意された次のクラスを用いています。

- org.apache.struts.actions.ForwardAction (①)

```
<!-- ========================== Action Mapping Definitions -->
  <action-mappings>
    <action
       path="/login"
       forward="/welcome/login.jsp"/>
```

②/loginだよ

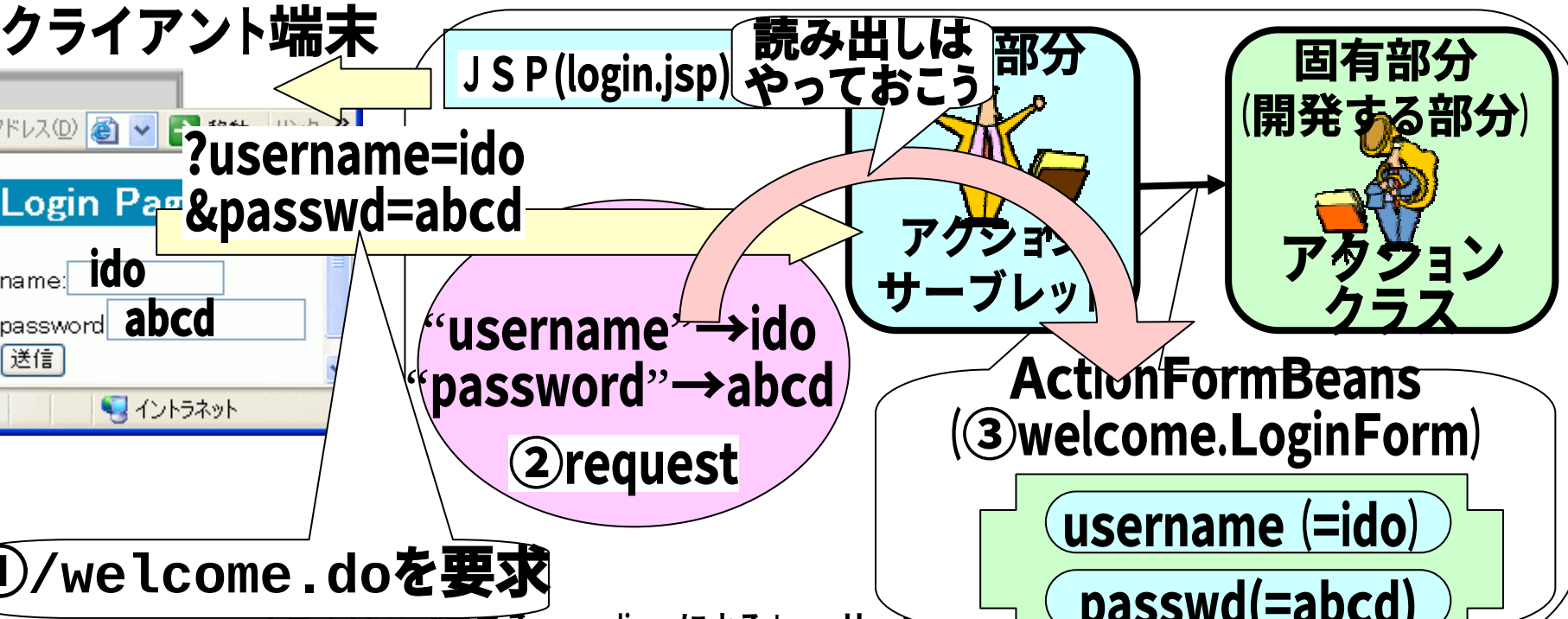
①

③それならwelcome/login.jspだ

⑹.1) アクション･フォーム･ビーンズ(ActionFormBeans)

■strutsでは、Java Beansを拡張した、ActionFormBeansを使用することが出来ます。

■Actonクラスでは、ＪＳＰ上のフォームへの入力値を、ActionFormBeansを介してサーブレット側で読むことが出来ます。すなわち、フォームの入力値の読み取りは、Ａｃｔｉｏｎサーブレットが自動的に実行してくれます。



（6．2）struts-config.xmlでの指定

■welcome.LoginFrom(前スライド③)を、フォーム･ビーンとして使い、その名前を④“LoginForm”とすることを指定します。

```
<!-- ================================ Form Bean Definitions -->
    <form-beans>
        <form-bean
            name="LoginForm"
            type="welcome.LoginForm"/>
```

④

前スライド③

■次にアクション･マッピング定義にて、ＵＲＬの/welcome.do（前スライド①）が要求された場合、④LoginFormの名前のフォーム･ビーンを用いることを指定します。

```
<!-- ============================ Action Mapping Definitions -->
  <action-mappings>
    <action
       path="/welcome"
       type="welcome.WelcomeAction"
       name="LoginForm" scope="request">
            :
```

前スライド①

前スライド②

④

(6．3) JSPとの符合

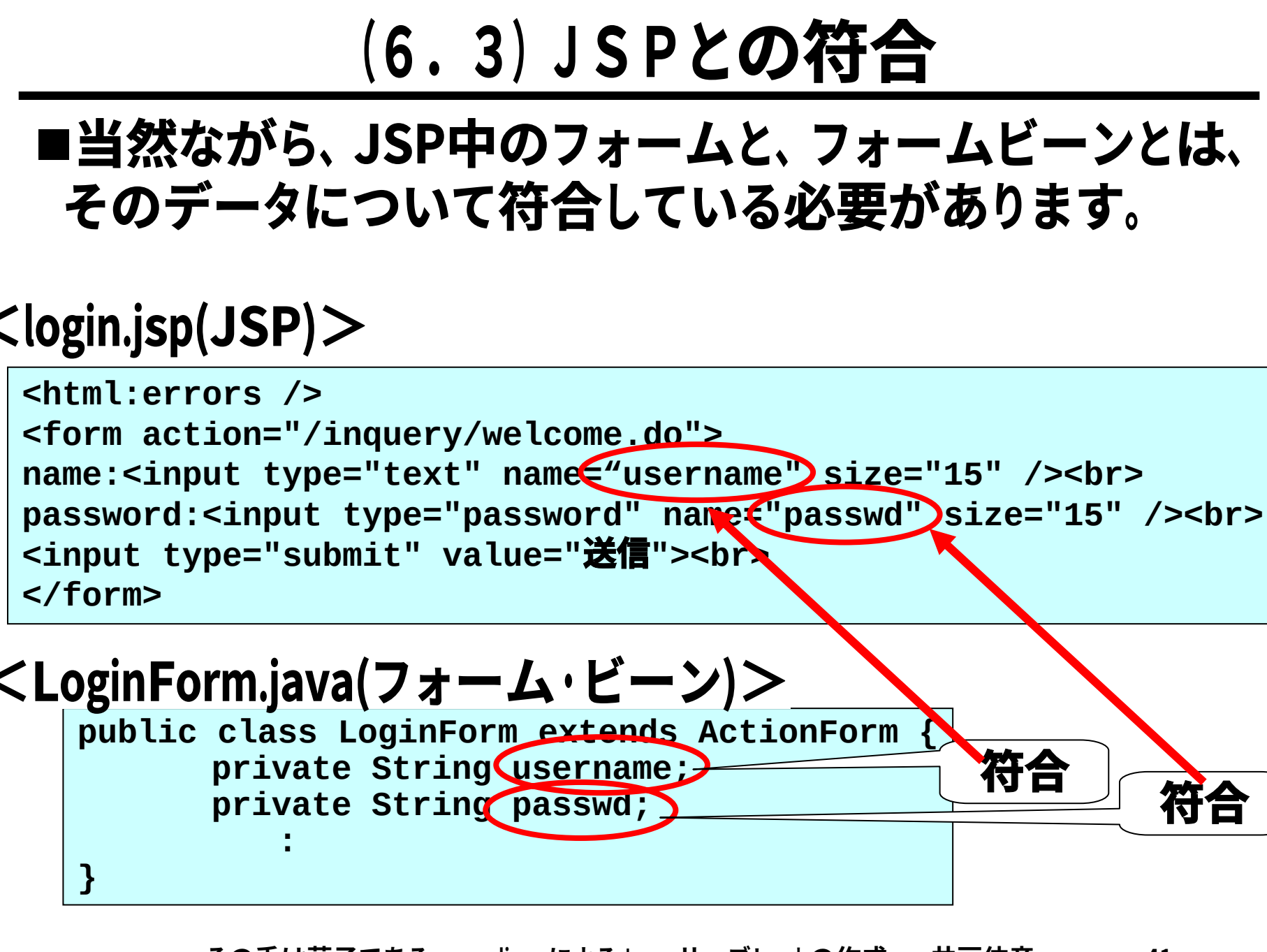
■当然ながら、JSP中のフォームと、フォームビーンとは、そのデータについて符合している必要があります。

＜login.jsp(JSP)＞

```
<html:errors />
<form action="/inquery/welcome.do">
name:<input type="text" name="username" size="15" /><br>
password:<input type="password" name="passwd" size="15" /><br>
<input type="submit" value="送信"><br>
</form>
```

＜LoginForm.java(フォーム･ビーン)＞

```
public class LoginForm extends ActionForm {
        private String username;
        private String passwd;
            :
}
```



(6．4) アクションクラスでの利用

■アクションクラスの引数で、“ActionForm form,”(①) と渡されているのが、アクション･フォーム･ビーンです。このプログラムでは、具体的には“LoginForm”クラスですので、これにキャスト (②) します。後は普通にクラスとして参照 (③) ／設定 (④) が出来ます。

```
public ActionForward execute(ActionMapping map,
          ActionForm form,
          HttpServletRequest request,
          HttpServletResponse response){
 LoginForm loginForm = (LoginForm)form;

  if(!loginForm.getPasswd().equals(sitePasswd))
                  :
 loginForm.setUsername("a Mr. or Ms. Unknown")
                  :
```

①

②

③:スライド(7.1)の場合であれば、”abcd”が読み出せる

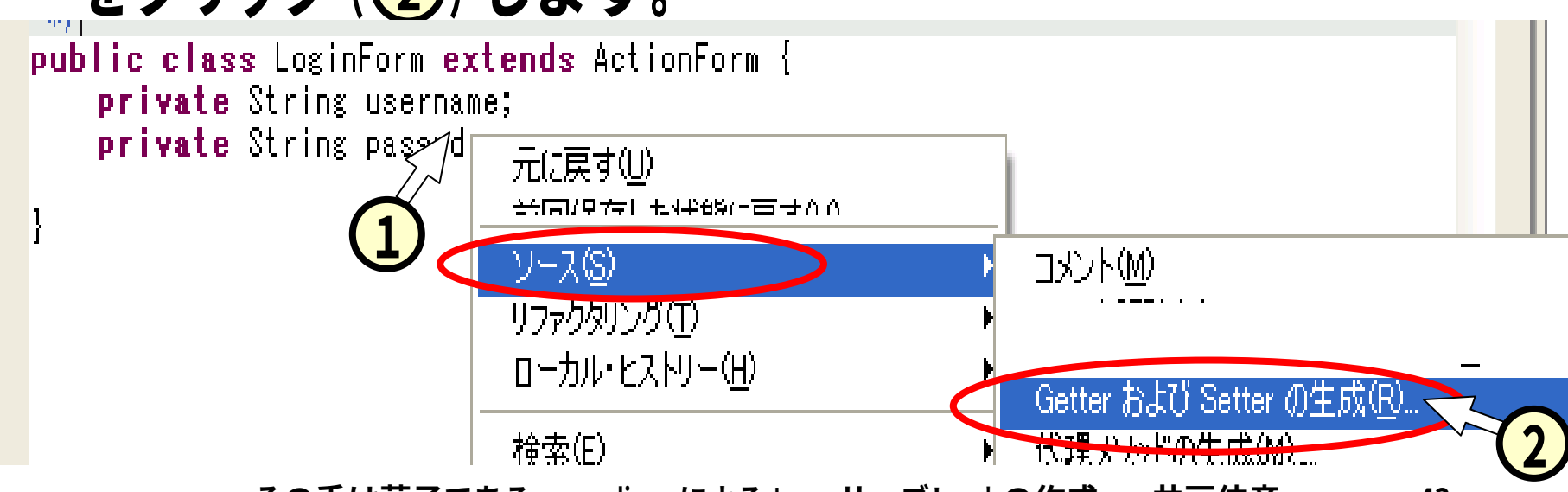
④

(6.5.1) eclipseでのアクセサ・メソッド編集ー1ー

■アクセサメソッドの編集は、eclipse上で簡単に行うことが出来ます。

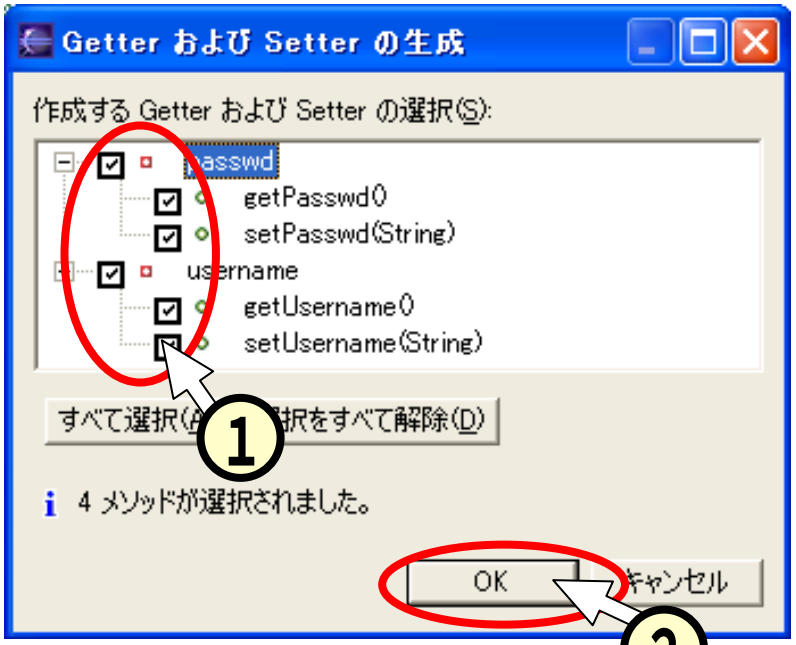
■変数（usernameとpasswd）の宣言のところまでは、入力したとします。

■エディター上のソースを右クリック（①）し、ポップアップメニューから、[ソース]-[GetterおよびSetterの生成]をクリック（②）します。



(6.5.2) eclipseでのアクセサ・メソッド編集ー2ー

- 「GetterおよびSetterの生成」のウインドウにて、作成するアクセサにチェック(①)します。
- [OK]をクリック(②)します。
- 変数“username”のアクセサ・メソッドが自動的に編集されます(③)。




```
 */
public class LoginForm extends ActionForm {
    private String username;
    private String passwd;

    /**
     * @return
     */
    public String getPasswd() {
        return passwd;
    }

    /**
     * @return
     */
    public String getUsername() {
        return username;
    }

    /**
     * @param string
     */
    public void setPasswd(String string) {
        passwd = string;
    }

    /**
     * @param string
     */
    public void setUsername(String string) {
        username = string;
    }

}
```

③

(7) WelcomeActionクラスの概要

■executeメソッドの概要を、ＰＡＤで示します。

■この処理は、スライド「Javaサーブレット入門」で作成したWelcomeサーブレットのdoGetメソッドの処理と同等です。

始まり

アクションフォームから、LoginFormクラスのインスタンス(loginForm)へキャストする。

パスワードがsitePasswdと等しくないか？

Yes

エラー情報を格納

処理失敗(fault)として、終了

名前が設定されていないか？

Yes

名前に、"a Mr. Unknown"を設定

request中に、"indA"をインデックスとして、loginFormを設定

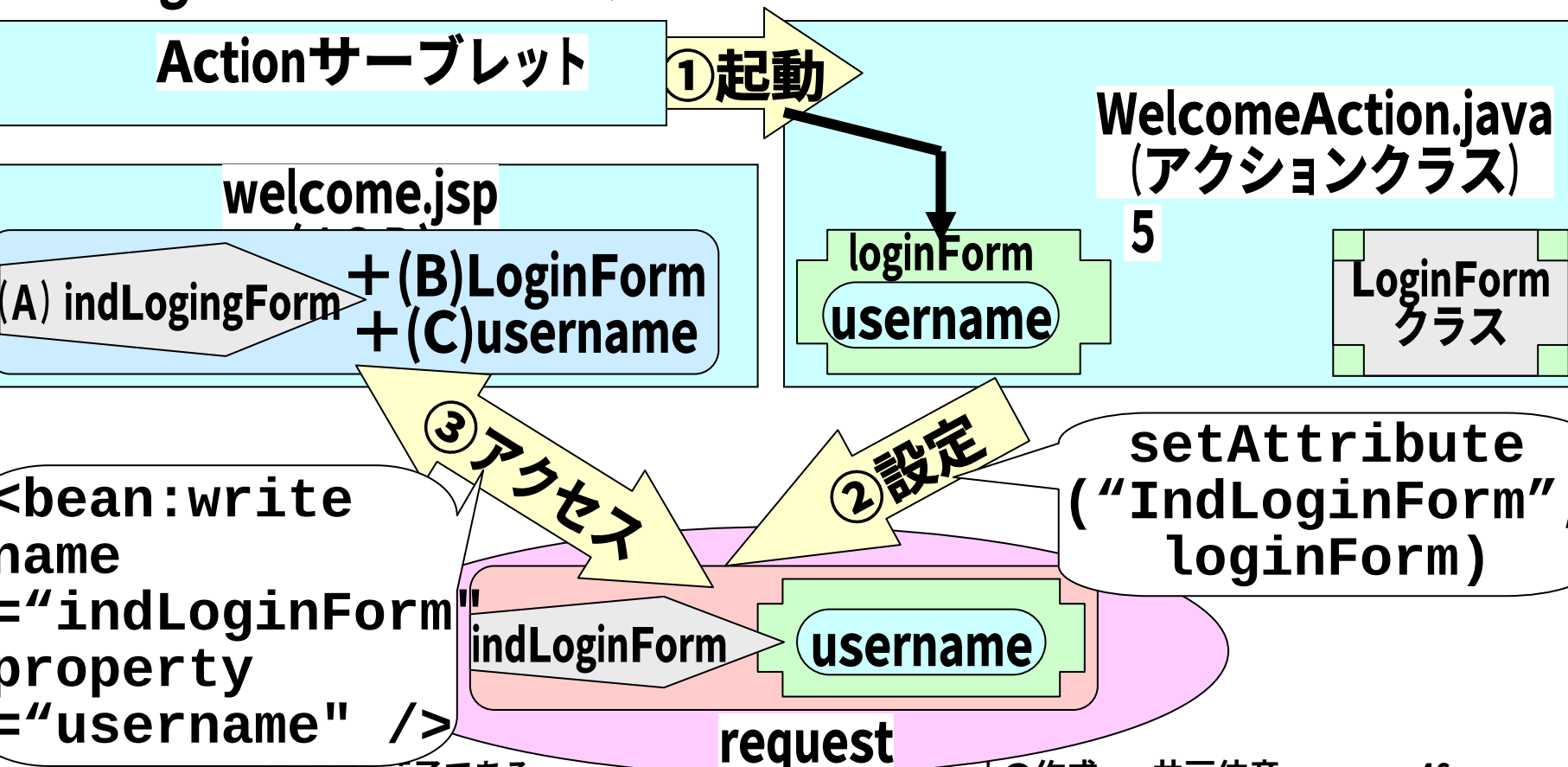
処理成功として終了

終わり

(8) アクション・クラスからＪＳＰへ

■アクションクラス (Welcome.java) から、JSP (welcome.jsp) へは、アクション・フォームクラスである、LoginFormをrequestに設定して渡しています。

■LoginFormは、前に見たJava Beansそのものです。

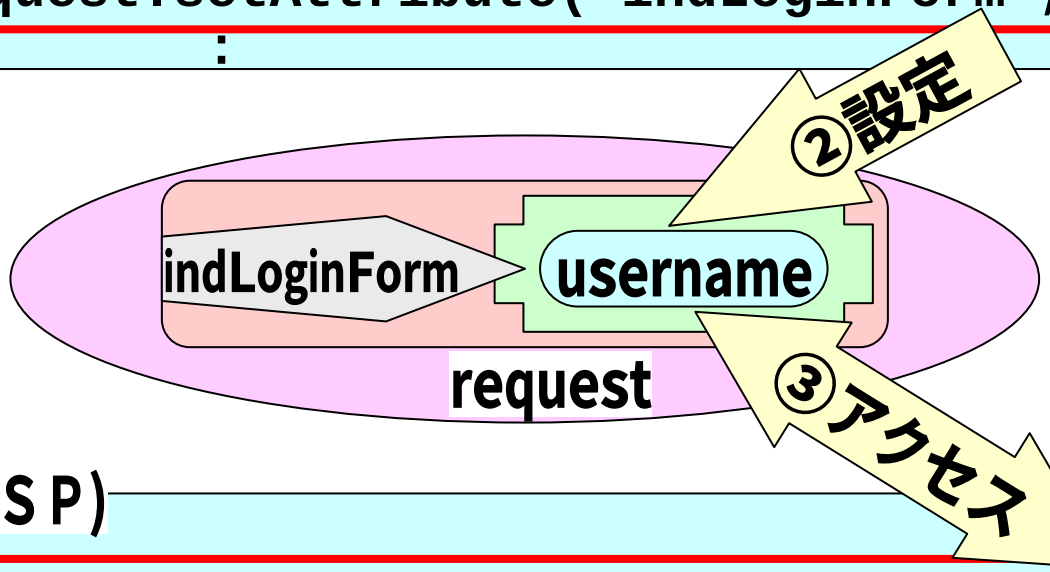


(8．1) ソースコード

■前スライドに対応するソースコードは、次のとおりです。

WelcomeAction.java (アクションクラス)

```
public class WelcomeAction extends Action {
  public ActionForward execute(ActionMapping map,
                :
    request.setAttribute(“indLoginForm",loginForm);
                :
```



welcome.jsp(ＪＳＰ)

```
            :
<h1><bean:write name=“indLoginForm" property=“username" />
welcome to our pages.</h1>
            :
```

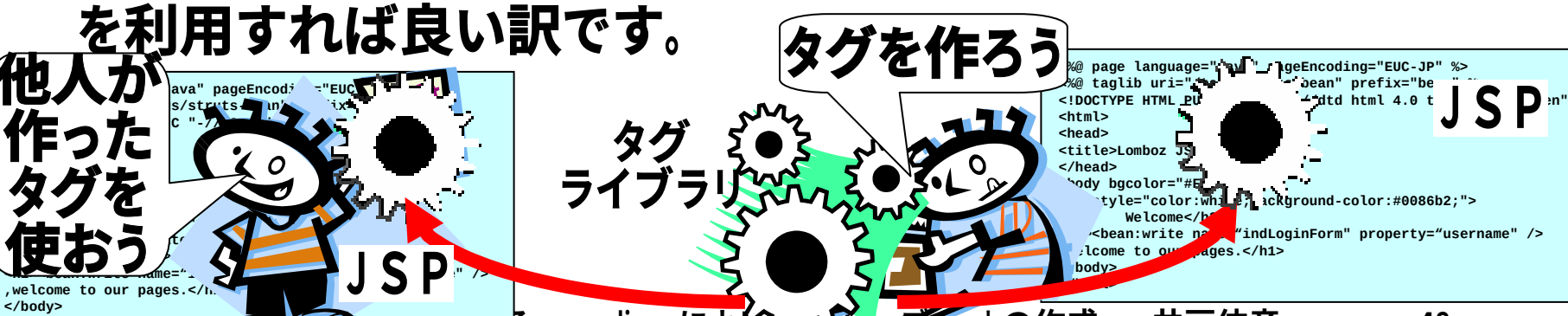
(8．2) タグ・ライブラリ

■前スライドのＪＳＰでのJava Beansへのアクセスは、スライド(6)で示したものとは異なるタグ (①) を使っています。この”<bean:”で始まるタグを使うことは、welcome.jspの２行めのtaglibディレクティヴ (②) に示されています。

welcome.jsp(ＪＳＰ

```
<%@ page language=“java” contentType=“text/html; ...(略)..."%>        ②
<%@ taglib uri="http://struts.apache.org/tags-bean" prefix="bean" %>
          :
<h1><bean:write name=“indLoginForm" property=“username" />        ①
,welcome to our pages.</h1>
```

■実は、ＪＳＰでは自由にタグを作ることが出来ます (これをカスタム・タグといいます)。しかしながら、いちいちカスタムタグを作らなくても、他人が作って提供されているタグ (＝タグ・ライブラリ) を利用すれば良い訳です。



(8．3) strutsのタグライブラリ

■welcome.jspで使用したBeanタグライブラリを含め、strutsには次のようなタグライブラリがあります。

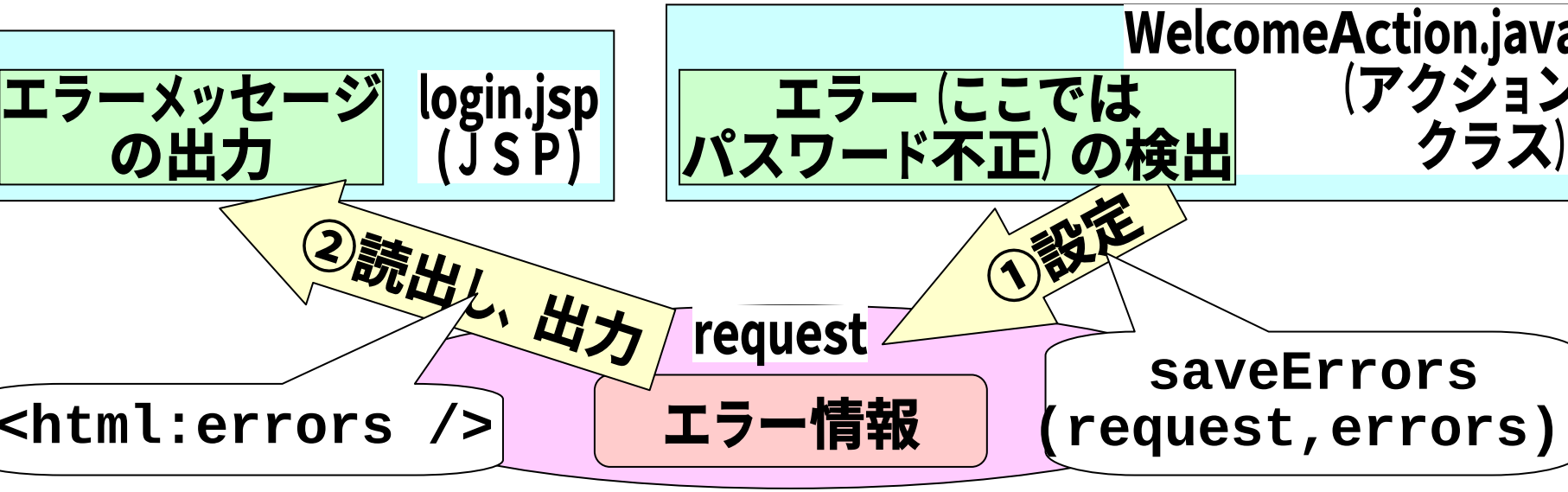
- HTMLタグライブラリ
  - ◆HTMLフォームやエラー表示を行います。
- Logicタグライブラリ
  - ◆繰り返しや条件分岐など、プログラムのような制御手順を提供します。
- Beanタグライブラリ
  - ◆JavaBeansにアクセスするための手段を提供します。
- Nestedタグライブラリ、Tilesタグライブラリ

■タグライブラリの詳細については、参考書を参照してください。

■Welcomeサーブレットでは、(9．1)に記したBeanタグライブラリの他、(10)にて説明するHTMLタグライブラリも使っています。

(9) エラーメッセージの出力方法

- アクションクラス“WelcomeAction.java”にてパスワードが正しくないことを検出した時、Welcomeサーブレットではログイン画面 (“login.jsp”) にてエラーメッセージを出力します。
- エラーメッセージの出力では、アクションクラスから表示用JSPへのデータの受け渡しが必要になります。
- これはよく現れる処理であることから、下記のような簡易に実現する仕組みがStrutsには備わっています。

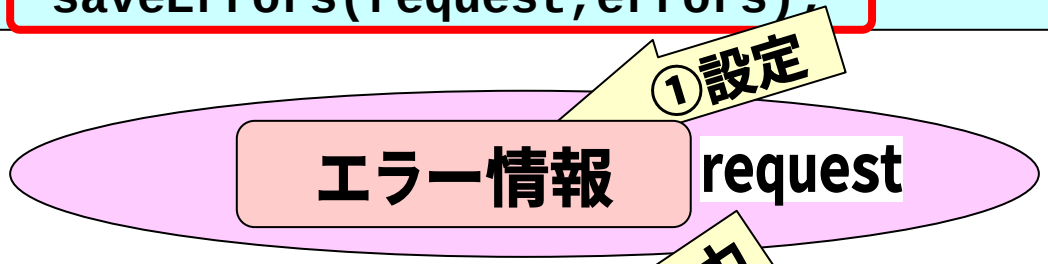


(9．1) ソースコード

■前スライドに対応するソースコードは、次のとおりです。

WelcomeAction.java（アクションクラス）

```
                :
public class WelcomeAction extends Action {
  public ActionForward execute(ActionMapping map,
                :
    ActionMessages errors = new ActionMessages();
      errors.add(ActionMessages.GLOBAL_Messages,
              new ActionMessage("errors.login"));
      saveErrors(request,errors);
```



welcome.jsp(ＪＳＰ)

```
                :
<h2 style="color:wh      background-color:#0086b2;">
        Login Pag      h2>
<html:errors />
                :
```

(9．2) アクション･エラー (ActionError)

■エラーに関するは,“ActionMessage”クラスに蓄えます。

■“ActionMessages”クラスは,“ActionMessage”クラスを複数格納するクラスです。ここ (WelcomeAction.java) では,１つの“ActionMessage”クラスのみを格納しています。

WelcomeAction.java (アクションクラス)

```
ActionMessages errors = new ActionMessages();
      errors.add(ActionMessages.GLOBAL_Message,
            new ActionMessage("errors.login"));
```

(9．3) メッセージ･キー

■“ActionMessage”クラスには、メッセージ･キーにより、エラーに関するメッセージが蓄えられます。

WelcomeAction.java (アクションクラス)

```
new ActionMessage("errors.login")
```

メッセージ･キー

■メッセージ･キーがどのようなメッセージを指すのかは、ファイル“application.properties”に次のとおり記されています。

MessageResources.properties

```
errors.login=password is not correct.
```

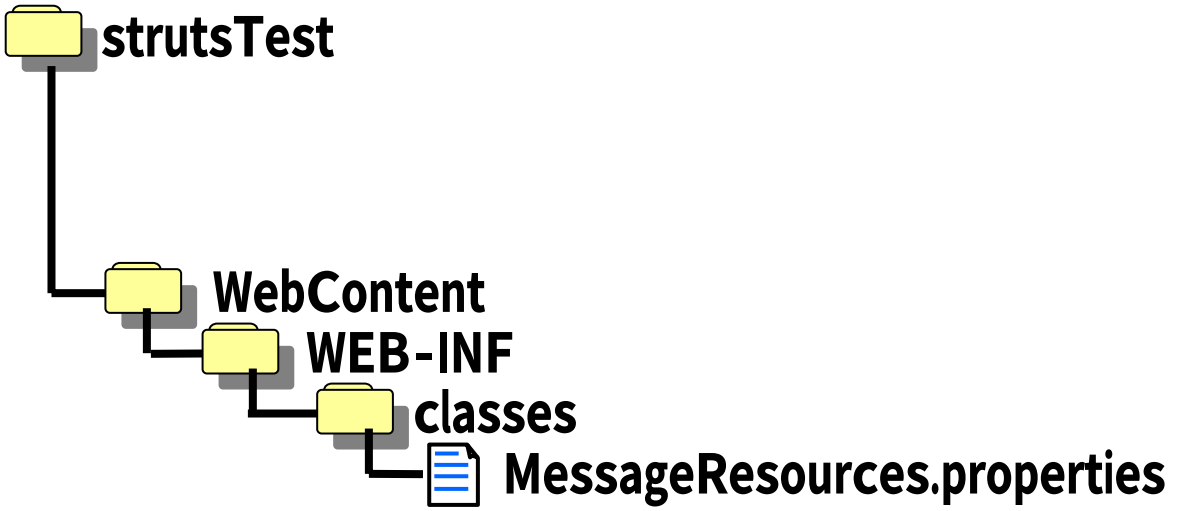
■「～.properties」の形のファイルは、リソースファイルと呼ばれ、**キー名＝値** の形で文字列情報を管理します。

(9．4) リソース･ファイル

■この“MessageResources.properties”というファイルを用いることは、ファイル“struts-config.xml”内に次のように記されています (.propertiesの部分は省きます) 。

Struts-config.xml

```
<message-resources parameter=“MessageResources”/>
```



(9．5) リソース･ファイルの利用

■リソース･ファイルは、エラー･メッセージのみならず、すべてのメッセージについて利用することができます。すなわち、利用者に示すメッセージや表示は、すべてプログラム (.java、.jsp) の外であるリソース･ファイルに追い出すことができます。

■また、リソースファイルを利用することにより、日本語･英語の表示を切り替えるようなことが、簡単に行えるようになります。

■詳しくは教科書を参照してください。

(10．1) MVCモデル

■フレーム･ワークの説明 (スライド(2.1)) のところで、「同じような作り方をすることにより、同じ部分を共通化できる」という旨のことを書きました。

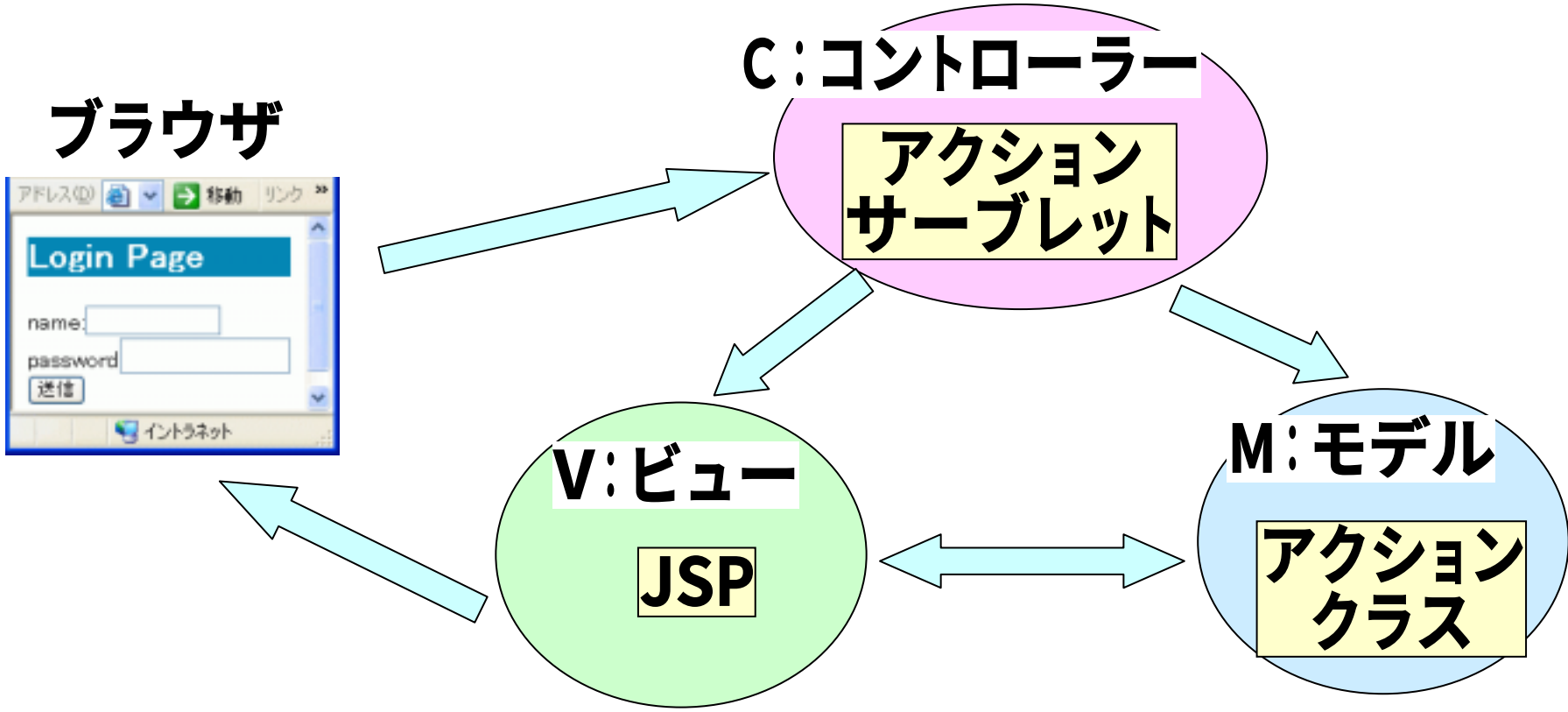
■同じような作り方をするために、 Strutsが前提としている考え方を、MVCモデルと言います。

■MVCモデルでは、システムは次の3つの部分に分けて考えます。

- M：Model (モデル)
  - ◆システムで行う仕事そのものの内容を担うプログラム。
- V：View (ビュー)
  - ◆表示･入出力を担当するプログラム。
- C：Controller (コントローラー)
  - ◆モデルとビューとを制御するプログラム。

(10．2) strutsでのMVC

■StrutsでのMVCは、アクション･クラス、JSP、アクション･サーブレットにより実装されます。



■このような枠組みを、MVC2と呼ぶことがあります。