Strutsによるアンケートシステム
ープログラムの説明ー

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

スライドの構成

はじめに
(1) システムの概要
(2) UMLで記した概要
(3) ファイル構成
(4) 作成済みファイルの利用
(5) tilesによる表示の共通化
(6) DynaActionForm
(7) データ受け渡しのクラス
(8) ファイル操作
(9) タグによる繰り返し
(10) タグによる2重の繰り返し

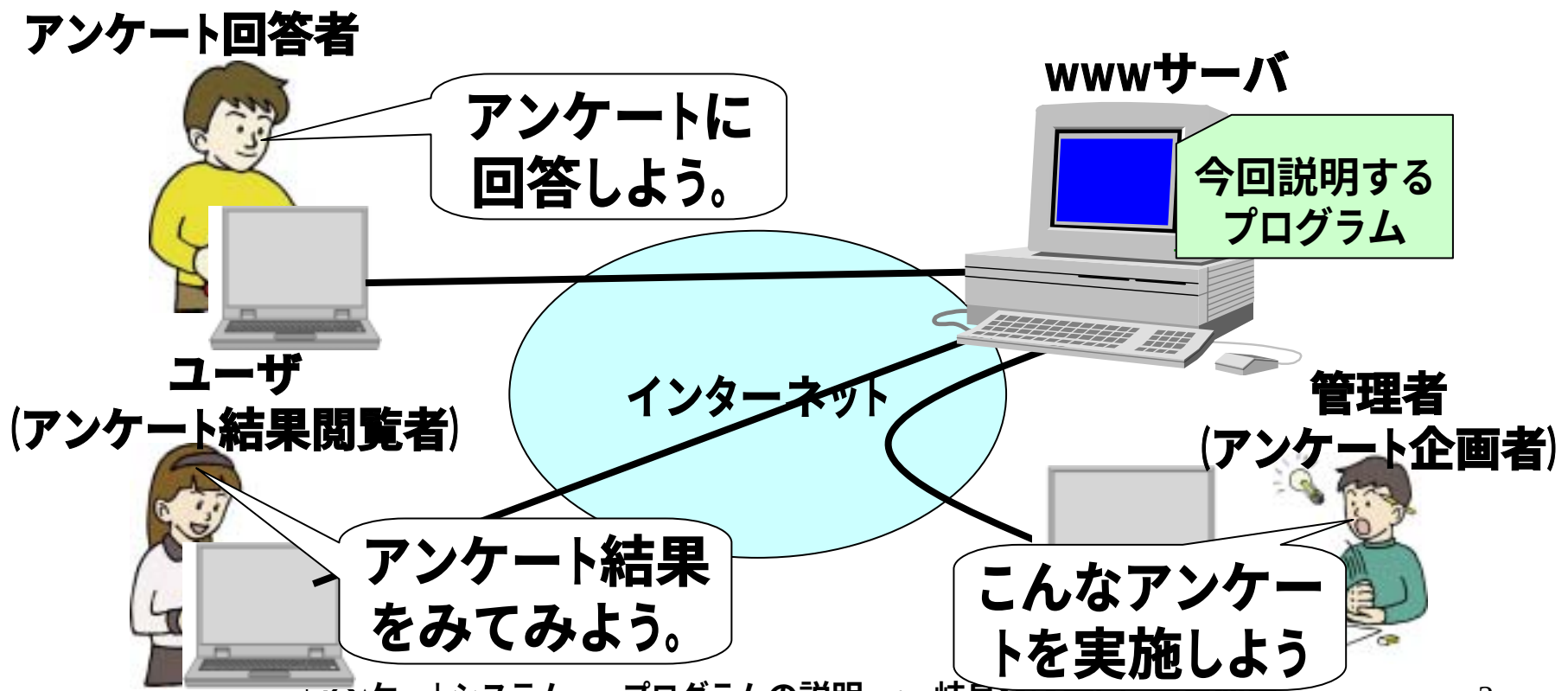
はじめに

- Web上で動作する「アンケート収集表示システム」のプログラムについて説明します。
- プログラムを理解するには、Java、サーブレット、strutsの知識が必要な部分もあります。それらの知識は次のスライドにて説明しています。
  - されど我らが日々　－サーブレット入門－
  - 月に吠える　－eclipseによるJavaアプリケーション作成－
  - ただ一疋の青い猫のかげ　－eclipseによるJavaサーブレット作成－
  - その手は菓子である　－eclipseによるstruts利用－
  - シャバドゥビ、ジャバ　－Java覚書－
- 説明は直感的な分かりやすさを重視し、厳密には不正確な言い方もしています。

(1) システムの概要

■Webでのアンケートシステムを作成します。

- Web上でアンケートを行い、その結果を閲覧できるようにします。
- アンケート管理者に対応するプログラムの作成は、練習問題としています。



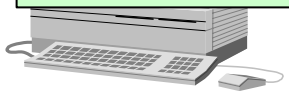
(1．1．1) アンケート回答者の操作

ケート回答者

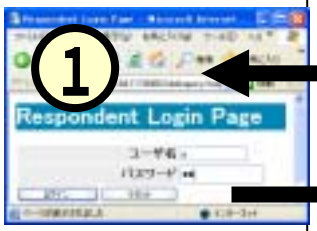
アンケートに
回答しよう。

wwwサーバ

今回説明する
プログラム

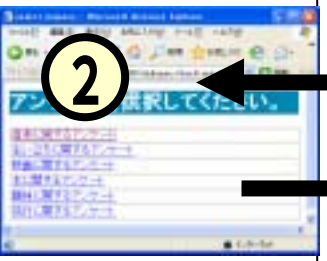


①http://(*)/idoInquery/respondentLogin.do
にてアクセス



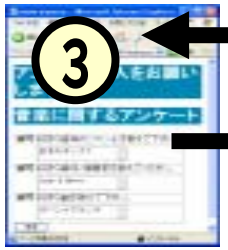
②ログイン画面を表示

③ユーザ名とパスワードを入力



④アンケート選択画面を表示

⑤アンケートを選択



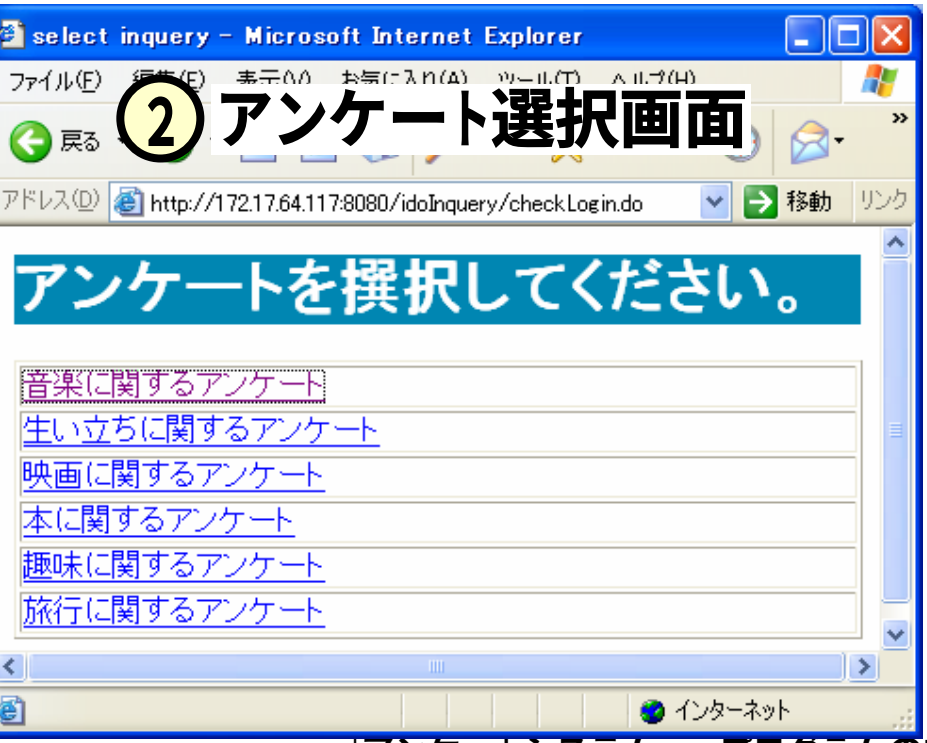
⑥アンケート回答作成画面を表示

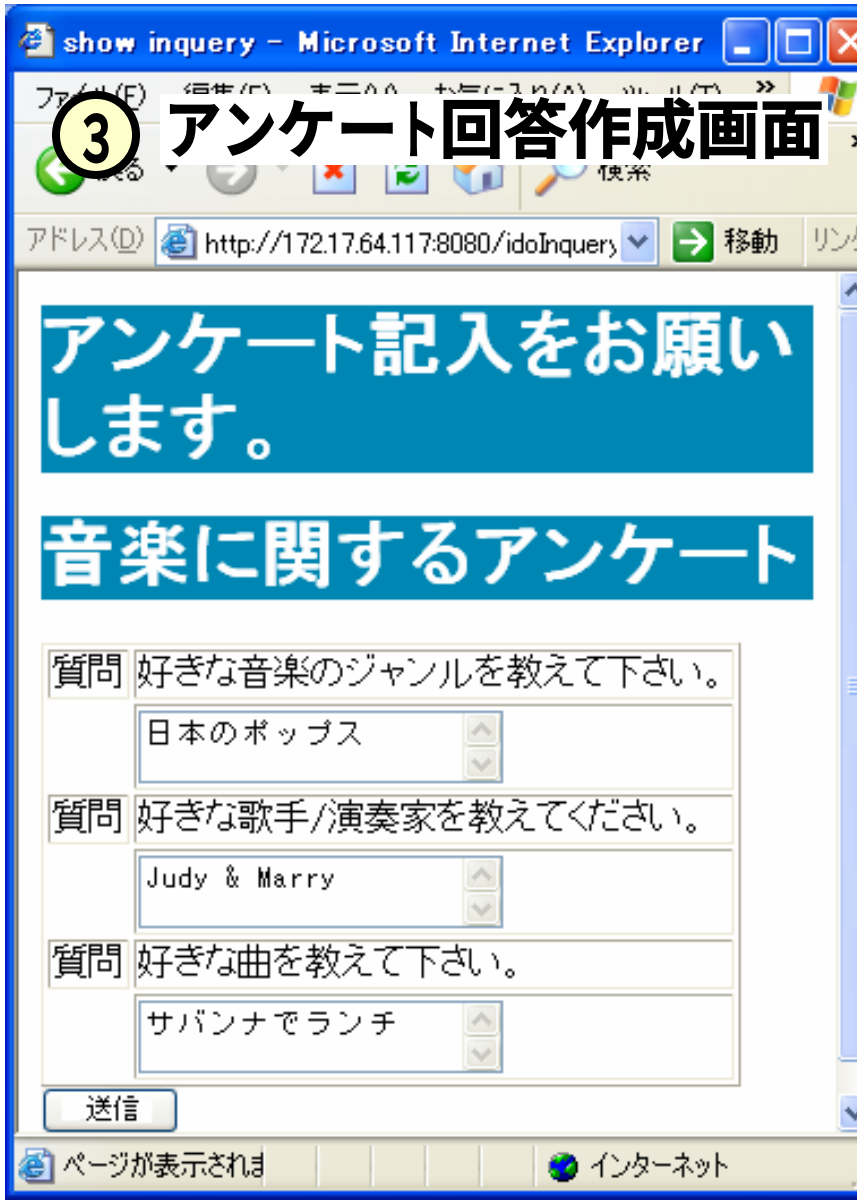
⑦アンケート回答を送信

(*)自身のPCで確認する場合、
“localhost:8080”となります。

(1．1．2) 画面イメージ

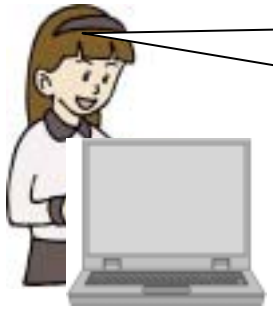

①ログイン画面
Respondent Login Page
ユーザ名
パスワード
ログイン
リセット


②アンケート選択画面
select inquery - Microsoft Internet Explorer
アンケートを撰択してください。
音楽に関するアンケート
生い立ちに関するアンケート
映画に関するアンケート
本に関するアンケート
趣味に関するアンケート
旅行に関するアンケート


③アンケート回答作成画面
show inquery - Microsoft Internet Explorer
アンケート記入をお願いします。
音楽に関するアンケート
質問 好きな音楽のジャンルを教えて下さい。
日本のポップス
質問 好きな歌手/演奏家を教えてください。
Judy & Marry
質問 好きな曲を教えて下さい。
サバンナでランチ
送信

(1．2．1) アンケート結果閲覧者の操作

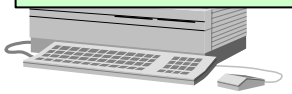
ケート回答者



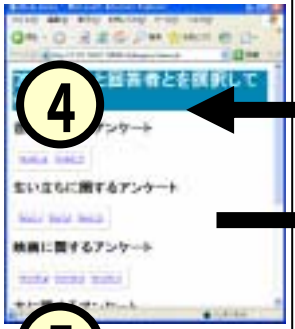
アンケート結果
をみてみよう。

wwwサーバ

今回説明する
プログラム

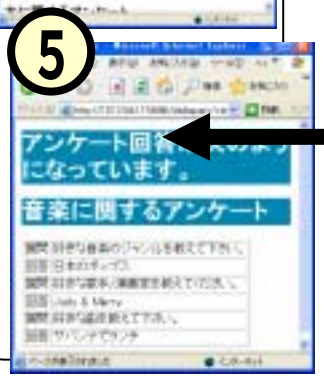


①http://(*)/idoInquery/menu.do



②アンケート回答選択画面を表示

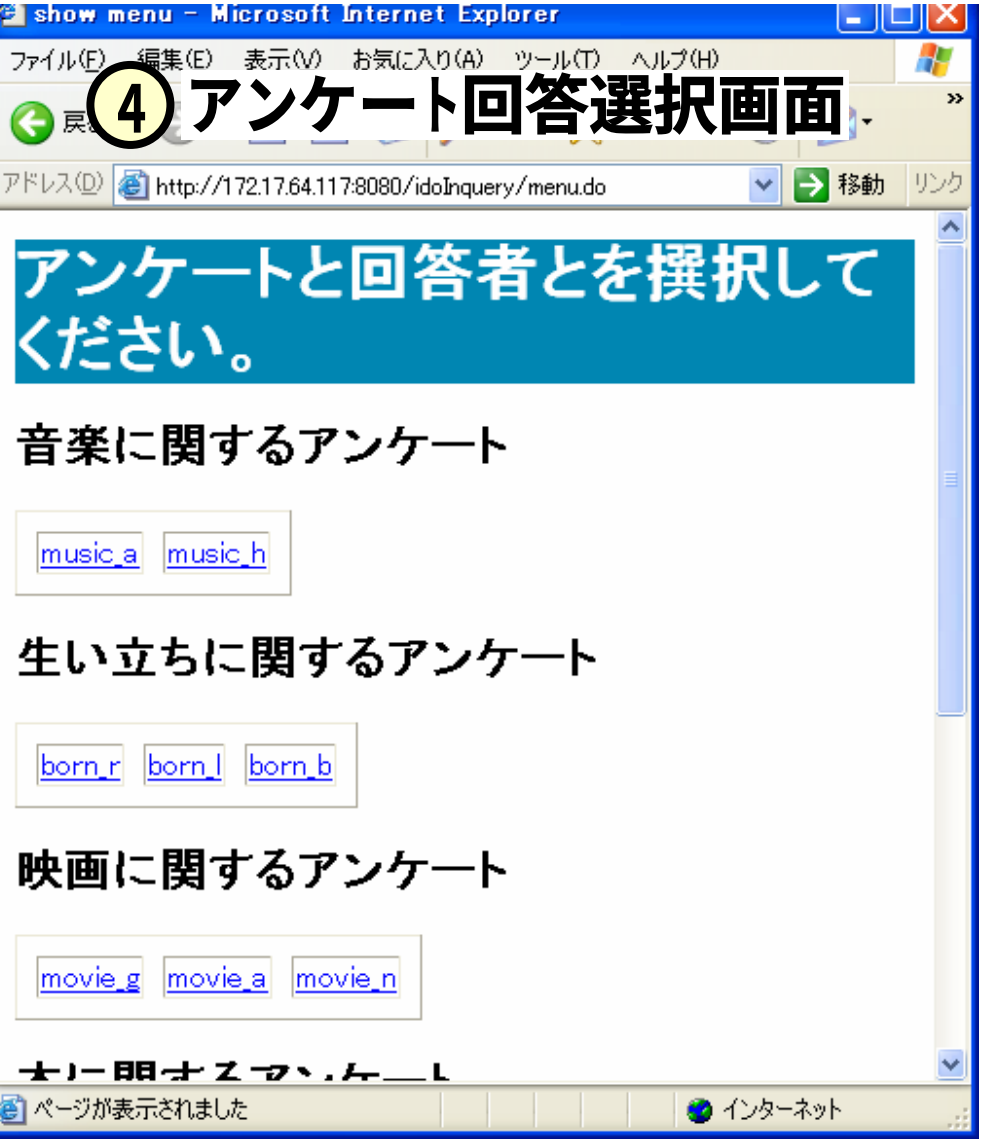
③アンケートを選択

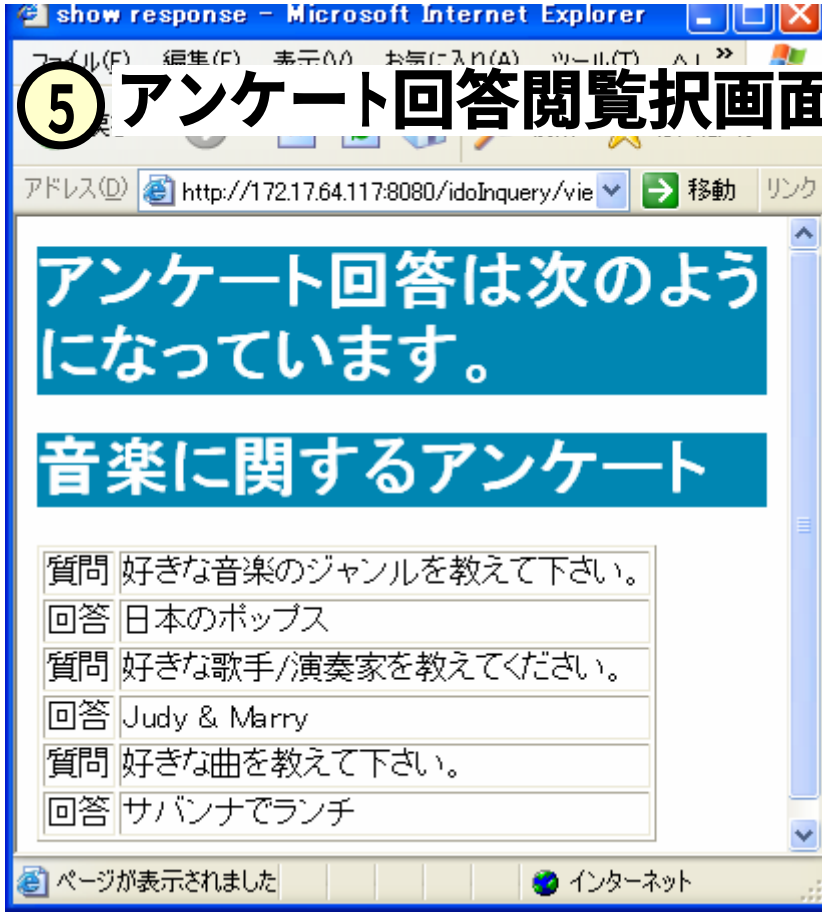


④アンケート回答閲覧画面を表示

(*)自身のPCで確認する場合、
“localhost:8080”となります。

(1．2．2) 画面イメージ


show menu - Microsoft Internet Explorer
ファイル(F) 編集(E) 表示(V) お気に入り(A) ツール(T) ヘルプ(H)
④アンケート回答選択画面
アドレス(D) http://172.17.64.117:8080/idoInquery/menu.do
移動
リンク
アンケートと回答者とを撰択してください。
音楽に関するアンケート
music_a
music_h
生い立ちに関するアンケート
born_r
born_l
born_b
映画に関するアンケート
movie_g
movie_a
movie_n
ページが表示されました
インターネット


show response - Microsoft Internet Explorer
⑤アンケート回答閲覧択画面
アドレス(D) http://172.17.64.117:8080/idoInquery/vie
移動
リンク
アンケート回答は次のようになっています。
音楽に関するアンケート
質問 好きな音楽のジャンルを教えて下さい。
回答 日本のポップス
質問 好きな歌手/演奏家を教えてください。
回答 Judy & Marry
質問 好きな曲を教えて下さい。
回答 サバンナでランチ
ページが表示されました
インターネット

(2) UMLで示した概要

■ここでは、UMLによる図を用いてアンケートシステムの概要を記します。

■UMLについては、次のスライド等を参考にしてください。

- 「ああオブジェクトにしあらましかばーオブジェクト指向」
  http://www.gifu-keizai.ac.jp/~ido/doc/system_design/object_oriented12.pdf

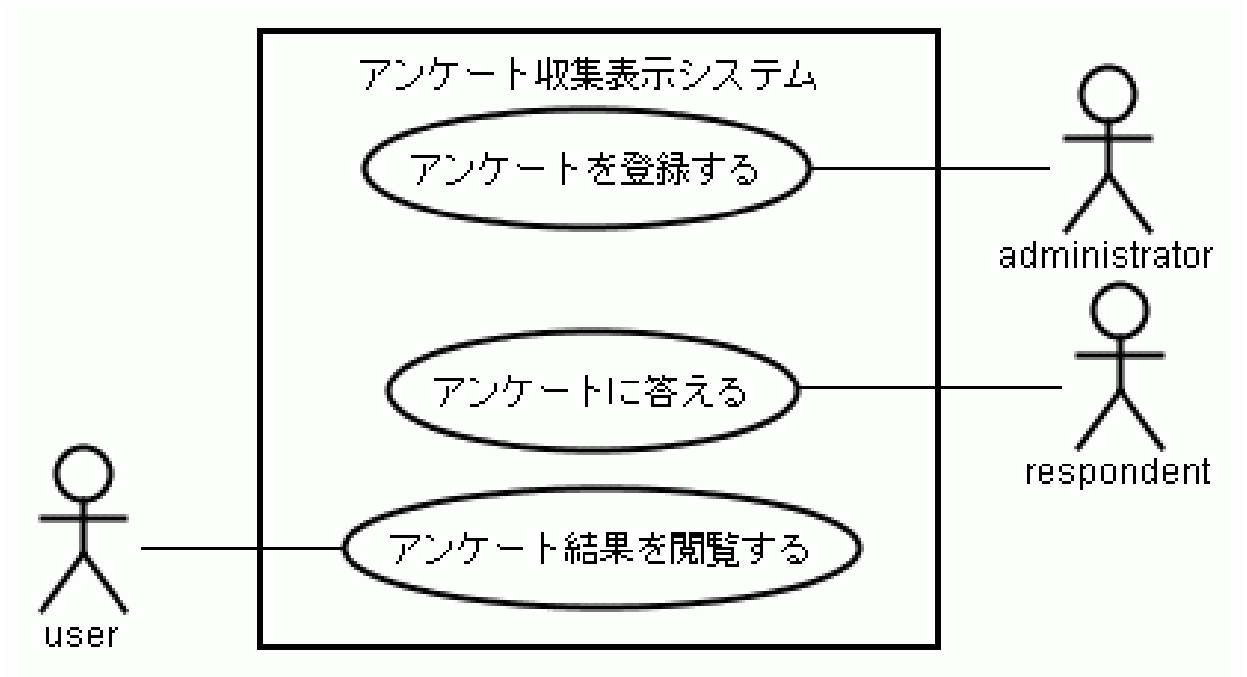
■UMLについては詳しく知らなくても、UMLによる図をなんとなく理解することは出来るのではないかと思います。本スライドではUMLそれ自体を説明することはないのですが、プログラムの理解の一助にはなると思います。

(2．1) ユースケース図

■アンケート収集表示システムは、次の3つの立場の人が利用します。

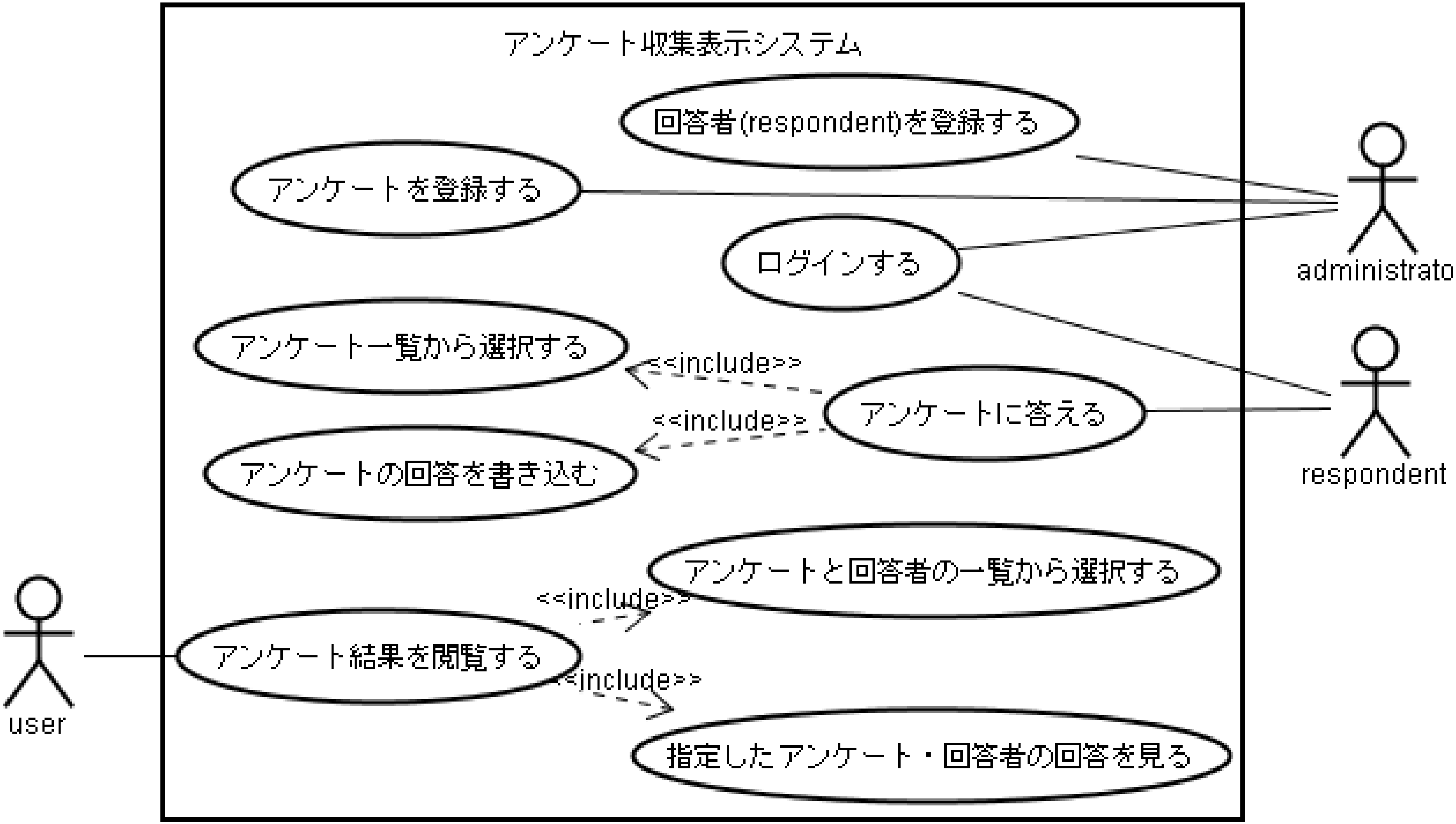
- 管理者(administrator)：アンケートを企画・実施する人
- アンケート回答者(respondant)：アンケートに答える人
- ユーザ (user) ：アンケート結果を見る人

■ユースケース図は、次のようになります。



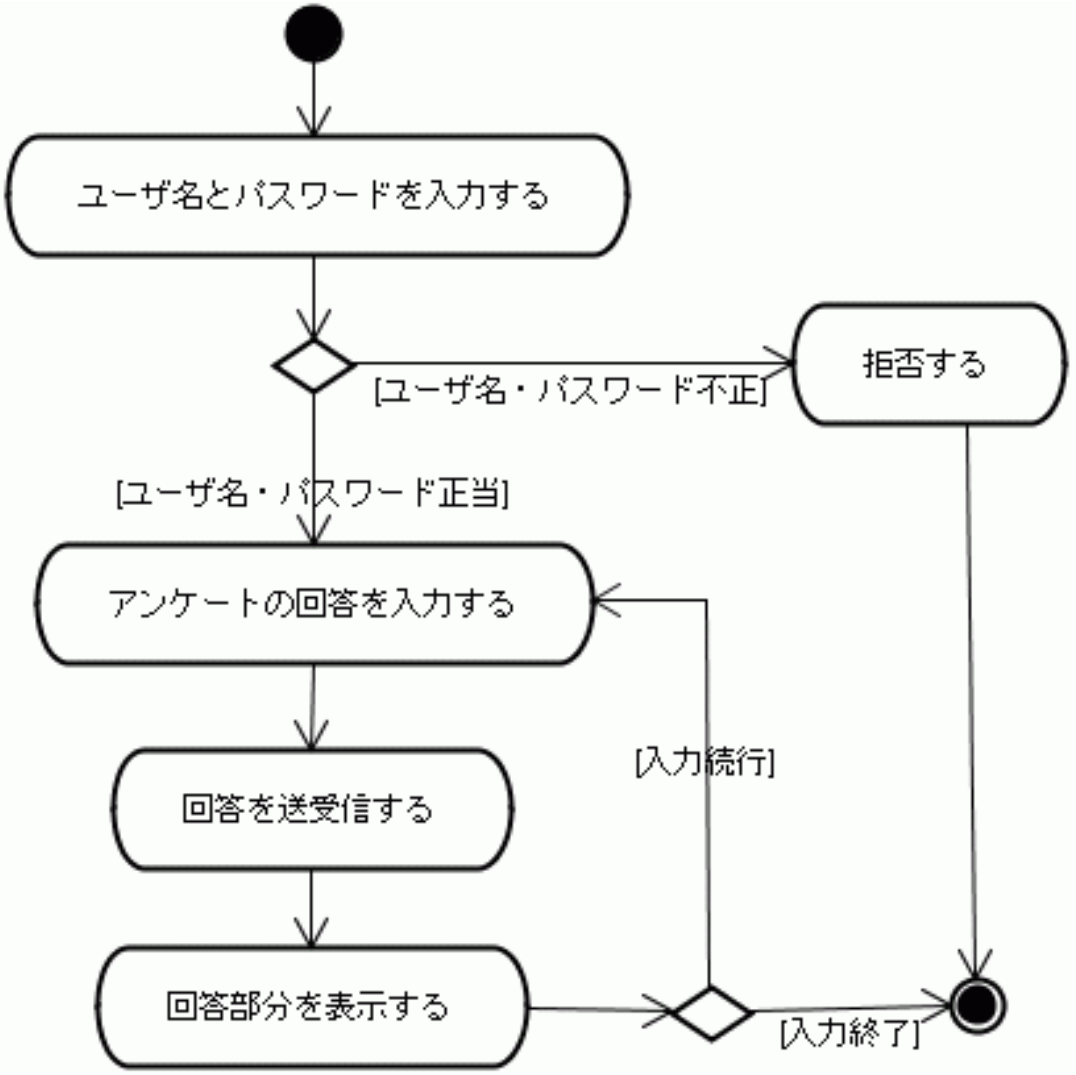
(2.2) 少し詳しくしたユースケース図

■もう少し詳細化したユースケース図を示します。



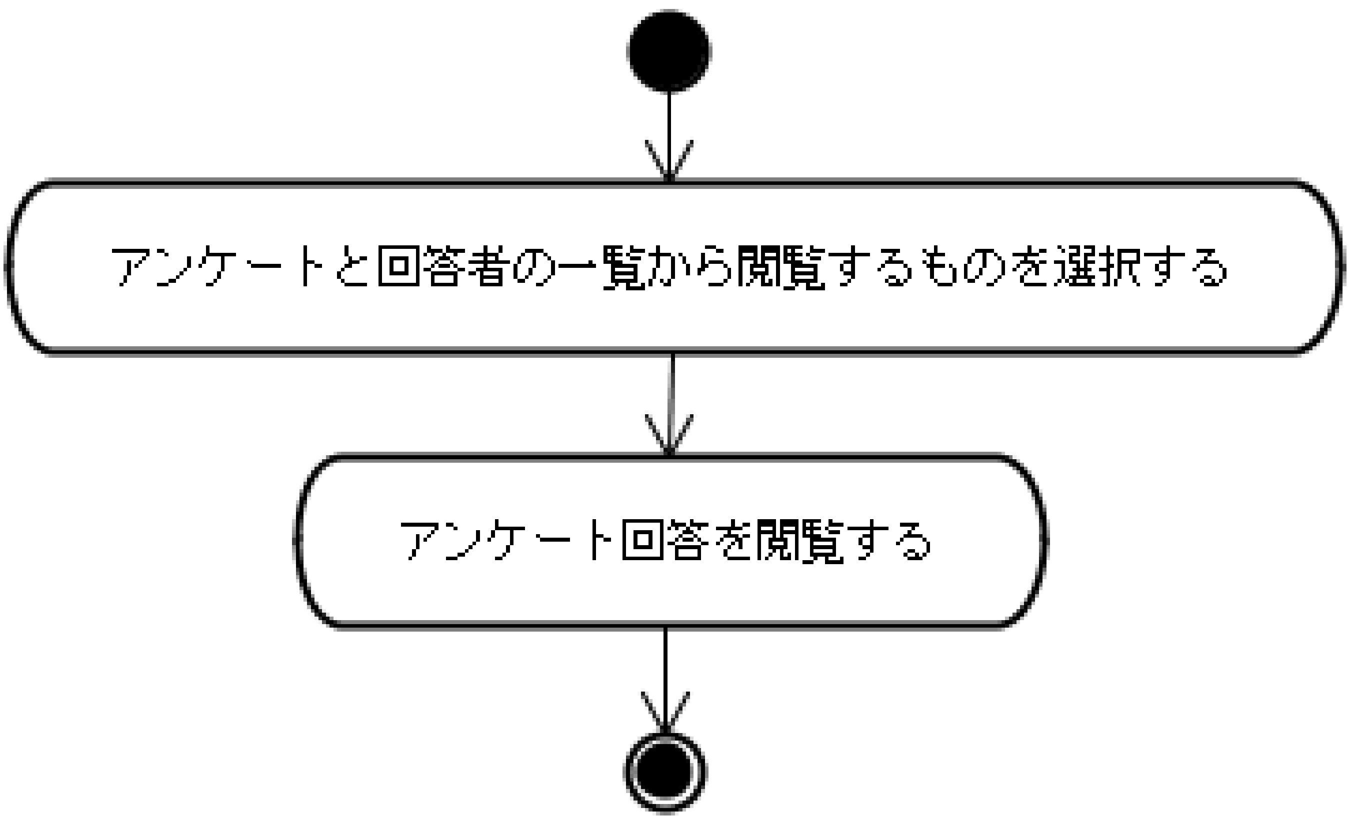
(2．3) アクティビティ図 (回答者)

■回答者のアクティビティ図を示します。

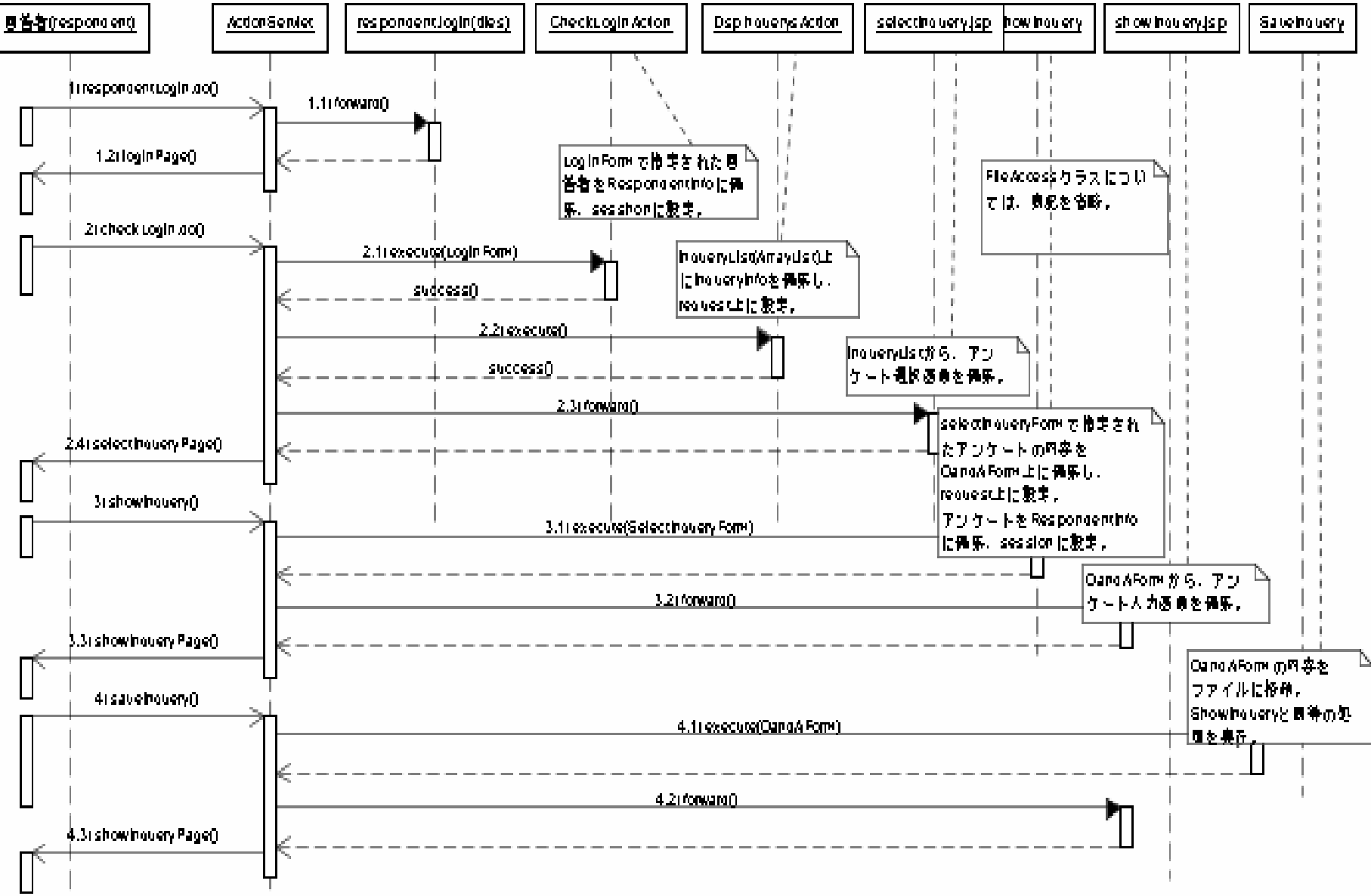


(2.4) アクティビティ図 (ユーザ)

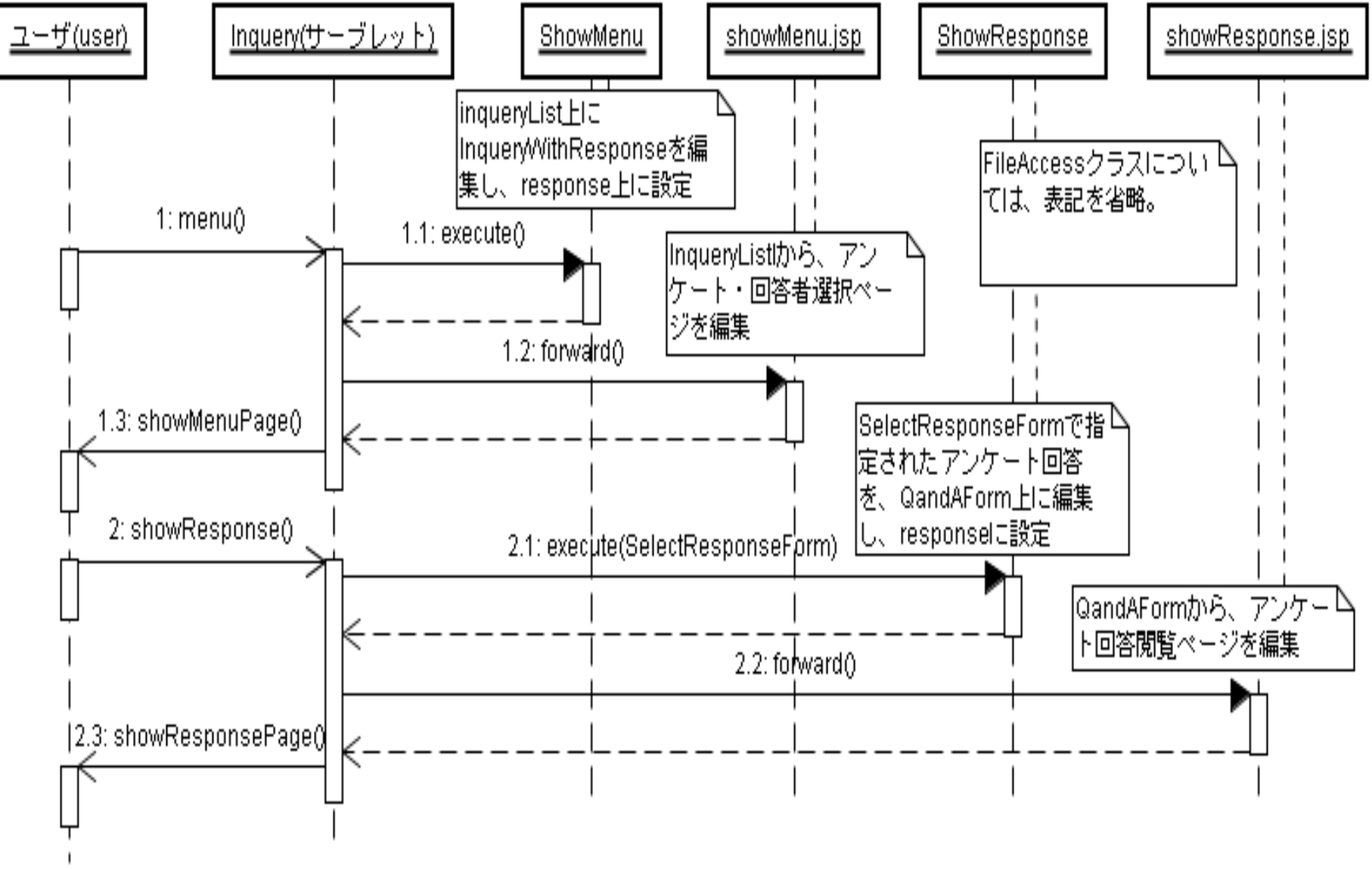
■ユーザのアクティビティ図を示します。



(2．6) シーケンス図 (回答者)

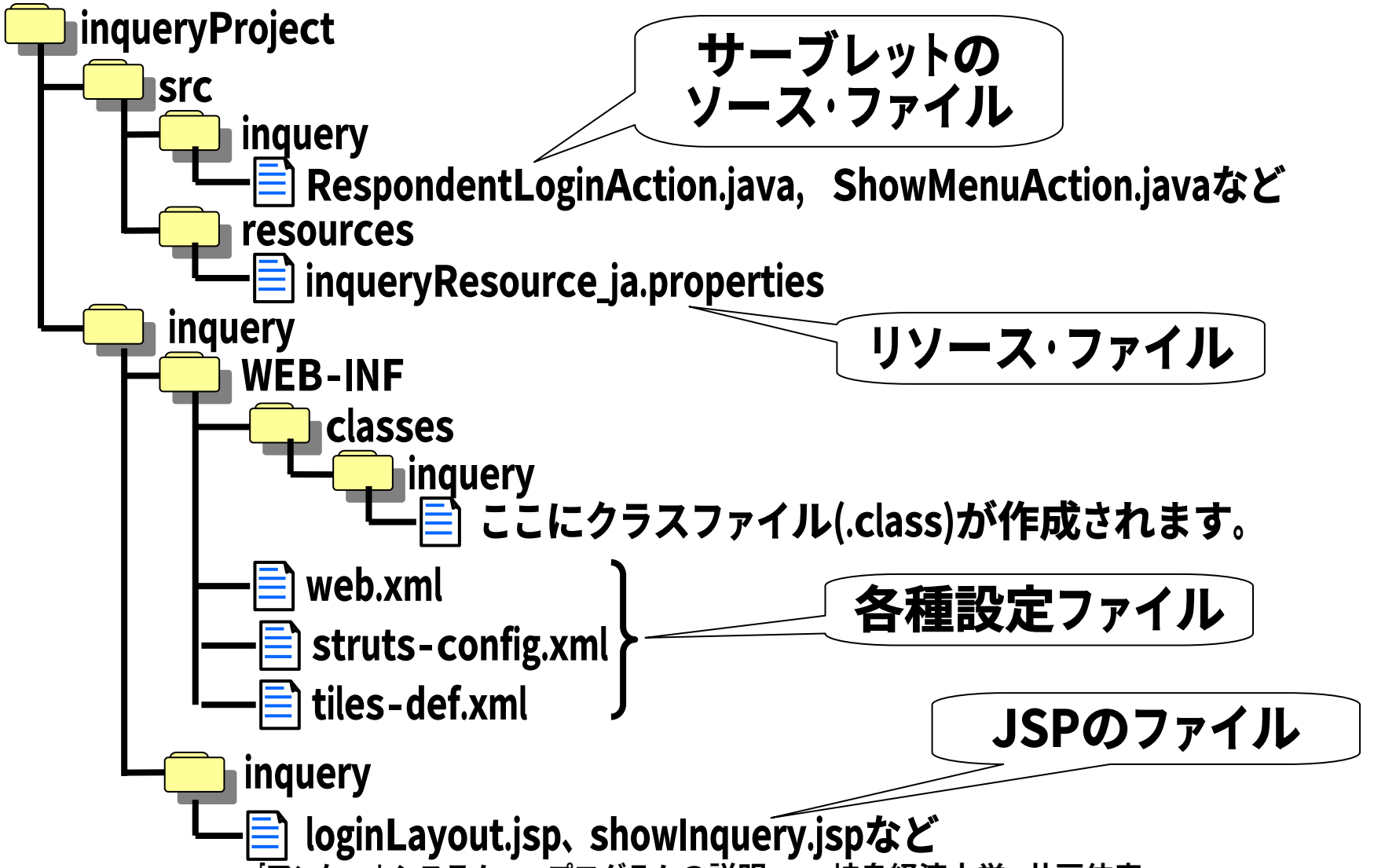


(2．7) シーケンス図 (ユーザ)



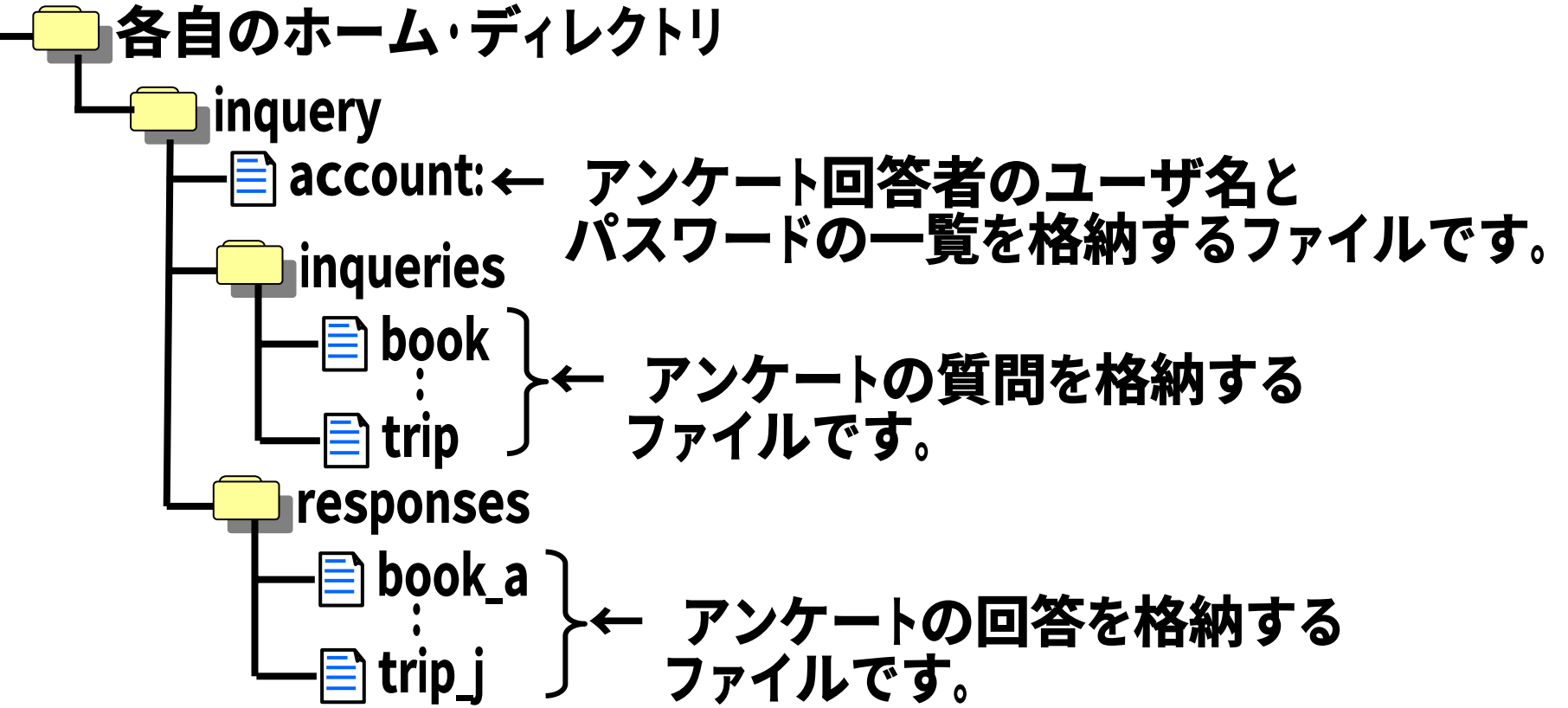
(3．1) サーブレット･JSPのファイル

■次のようなファイルを作成･編集します。



(3．2) アンケートの保存ファイル

■アンケートシステムでは、各自のホームディレクトリ灰化に次のようなファイルを作成し、これにアンケートの内容を保存します。



(3．2．1) ファイル“account”

■このファイルには、各行に回答者の情報が、次の形式で保持されています。

```
役割名  ユーザ名  パスワード
```

■役割名には、“respondent”(回答者)と、“administrator”(管理者)とを想定していますが、現状、管理者についてはサポートしていません。

■次のような値がデフォルトで含まれています。

```
respondent ido idoido
respondent a aa
respondent b bb
respondent c cc
respondent d dd
        :
```

(3．2．2) inqueries配下のファイル

■ディレクトリ“inqueries”の配下には、アンケートの質問を記したファイルが保持されています。

■たとえば、ファイル“music”は次のような内容になっています。

```
t:音楽に関するアンケート
q:好きな音楽のジャンルを教えて下さい。
q:好きな歌手/演奏家を教えてください。
q:好きな曲を教えて下さい。
```

■先頭に“t:”と記した行には、質問のタイトルが記されています。

■先頭に“q:”と記した行には、質問の内容そのものが記されています。

（3．2．3）responses配下のファイル

■ディレクトリ“responses”（回答）の配下には、アンケートの回答を記したファイルが保持されています。

■ファイル名は、次の形になっています。

- “アンケートのファイル名”＋“_”＋“回答者のユーザ名”
- 例　：　music_a（アンケートのファイル“music”、ユーザ名“a”）

■たとえば、ファイル“music_a”は、スライド(1.2.2)のような入力を行った場合、次のような内容になります。

```
r:日本のポップス
r:Judy & Marry
r:サバンナでランチ
```

■先頭に“r:”と記した行が、回答の内容です。

(4) 作成済みのファイルの利用

■ここでは、講師 (井戸) が作成したファイルを利用する方法を示します。

■次の2つの作業を順に行います。

- プロジェクトの作成 (スライド(4.1))
- プログラムファイルのインポート (スライド(4.2))
- アンケート・ファイルのコピー (スライド(4.3))
- Web.xmlの編集 (スライド(4.4))
- 起動の確認
  - ◆スライドに示したように起動できるか確認します。

■なお、次の作業は、インポートしたファイルの元となったプロジェクトにて既に実施されており、今回は特に実行する必要はありません (スライド「その手は菓子であるーeclipseによるstruts利用ー」では実施しています)。

- Strutsのパスの設定
- ソースディレクトリの移動

(4．1)プロジェクトの作成

■まず、Lombozプロジェクトを作成します。次の資料を参照してください。

- eclipseでのstrutsの使用準備(Linux)
  http://www.gifu-keizai.ac.jp/~ido/doc/linux_text/man_linux_struts.pdf
- eclipseでのstrutsの使用準備(Windows)
  http://www.gifu-keizai.ac.jp/~ido/doc/windows_text/man_win_eclipse.pdf
- 「ただ一疋の青い猫のかげ　－eclipseによるJavaサーブレット作成－」(2．1．1)～(2．3．4)
  http://www.gifu-keizai.ac.jp/~ido/doc/java/eclipse_servlet.pdf

■今回作成したのは、次のような名前のプロジェクト、および、モジュールです(もちろん、任意の名前でOKです)。

- プロジェクト　：　“inqueryProject”
- モジュール：　“inquery”
- ＵＲＬ：“/idoInquery”

自分の名前もしくは学籍番号に関する名前としてください。

(4．2)インポート

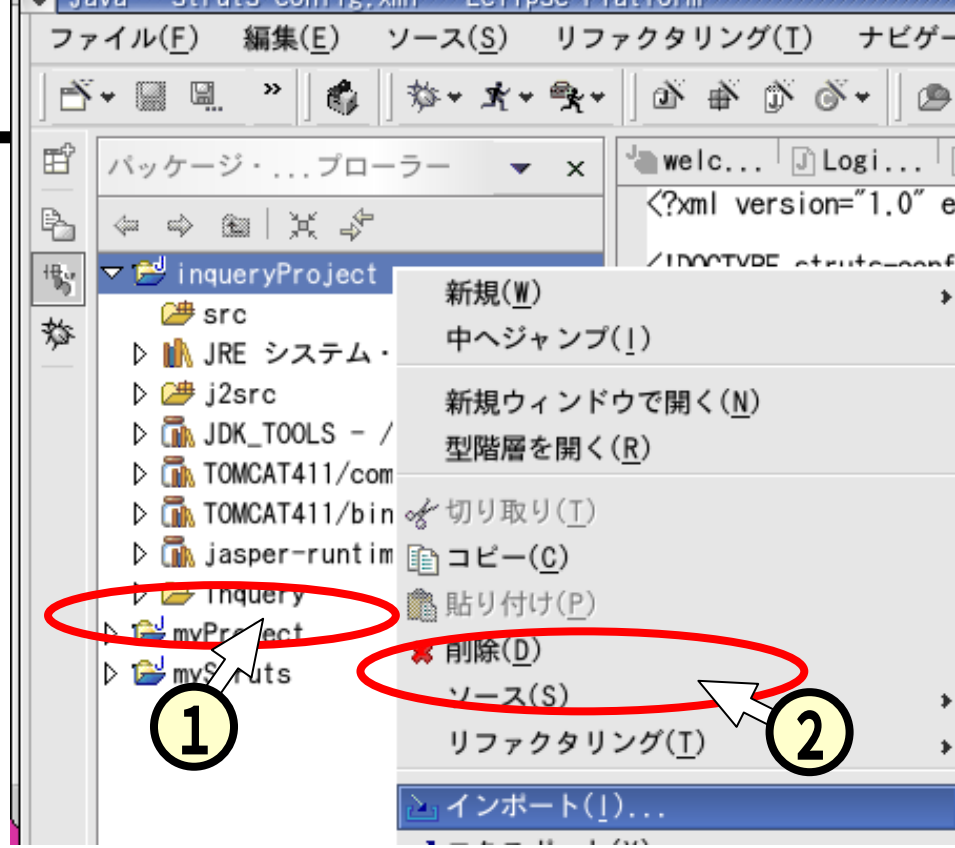
■次のディレクトリ配下のファイルをインポートします。
- /home/ido/javaSources/inqueryProject

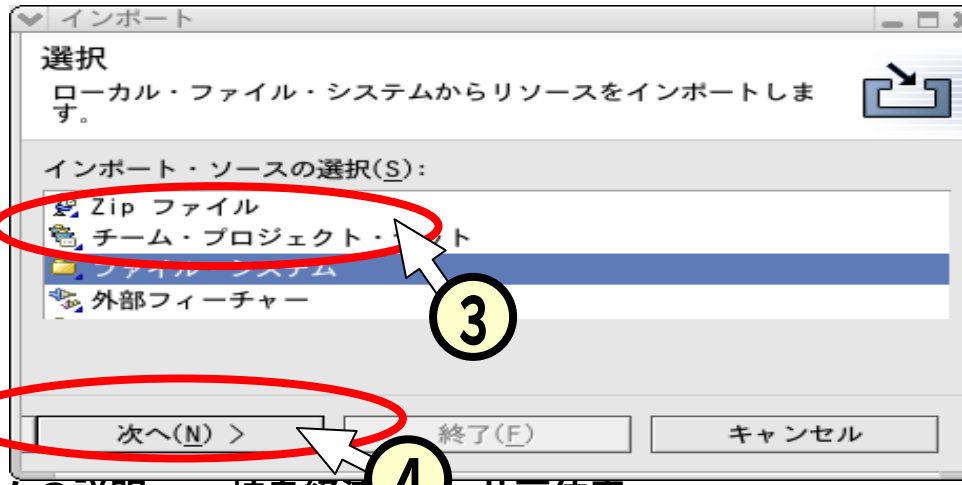
■インポート先は、作成したプロジェクト“inqueryProject”です。

■この手順は、次の資料と同等ですが、以下のスライドに再掲します。
- 「その手は菓子であるーeclipseによるstruts利用ー」(3．3)
  http://www.gifu-keizai.ac.jp/~ido/doc/java/eclipse_struts.pdf

(4.2.1) インポート1

- 「パッケージ・エクスプローラ」中、作成した[inqueryProject]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、[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。

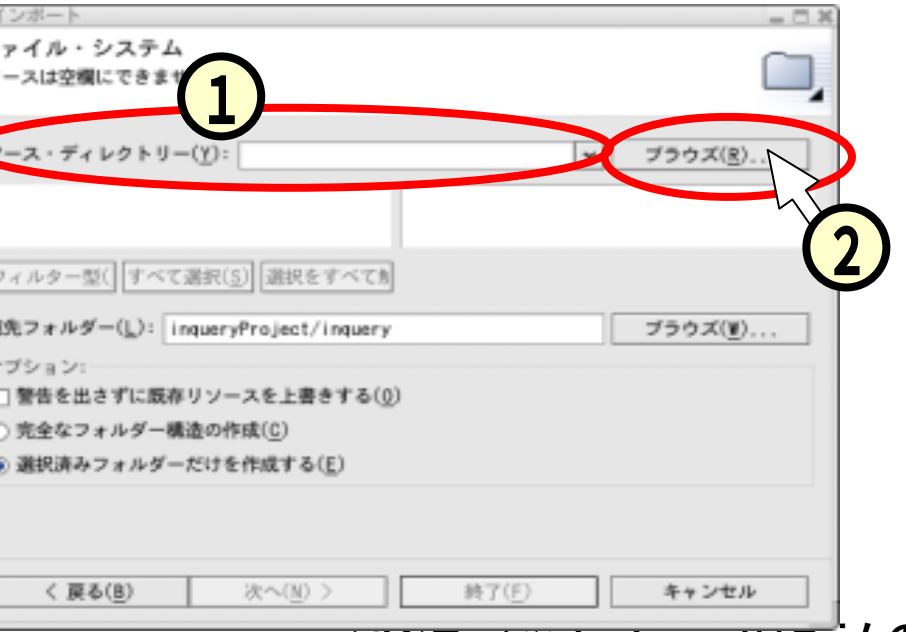


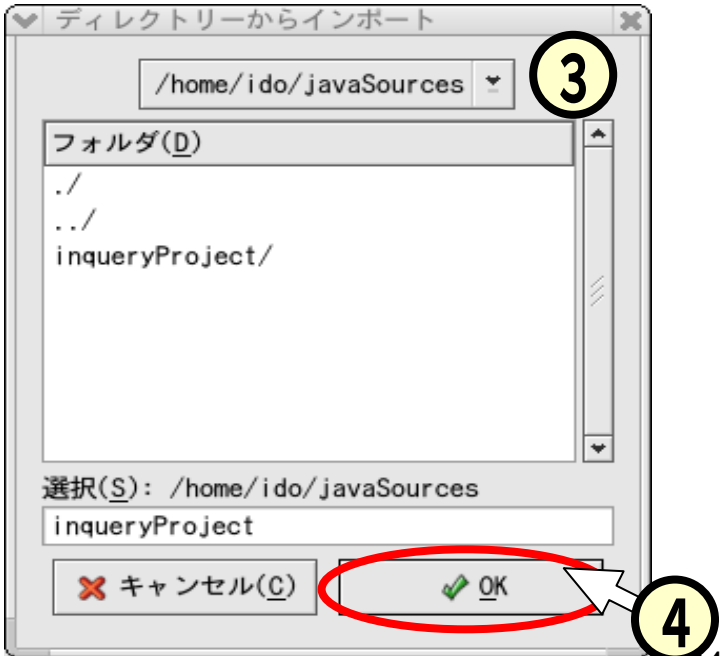


(4．2．2) インポート2

■「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。
- /home/ido/javaSources/inqueryProject

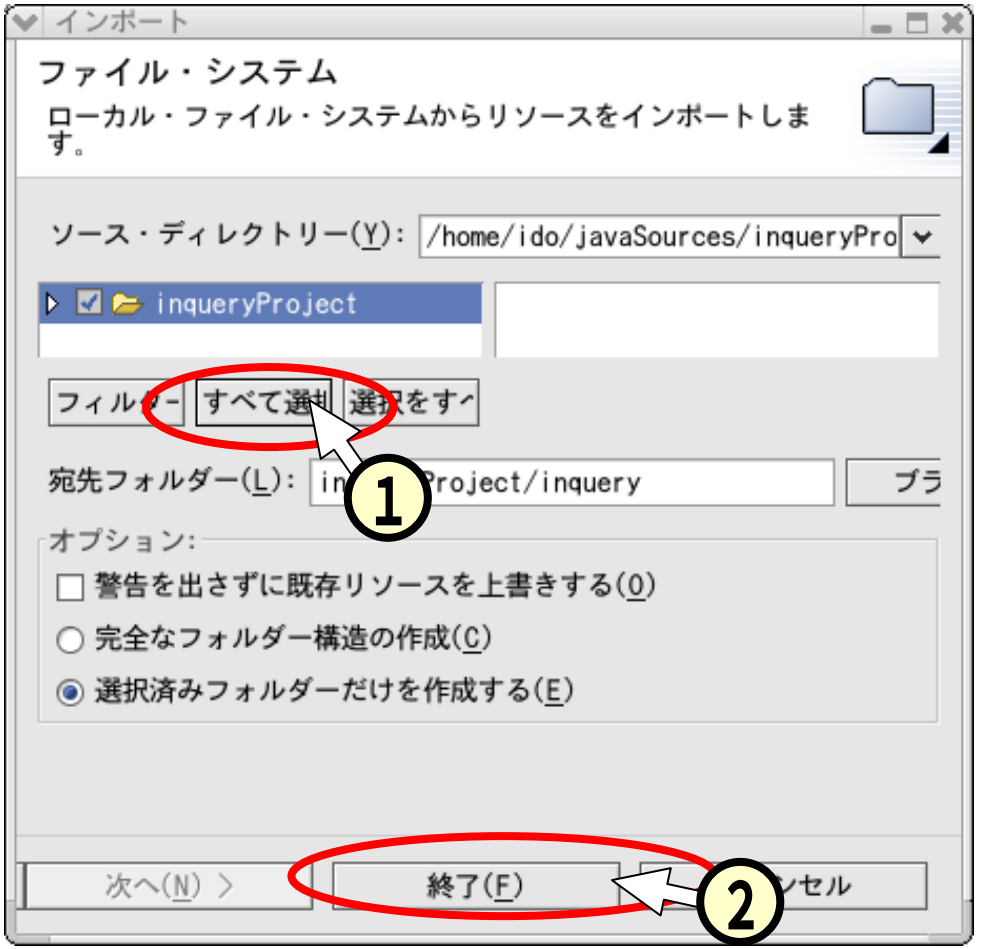


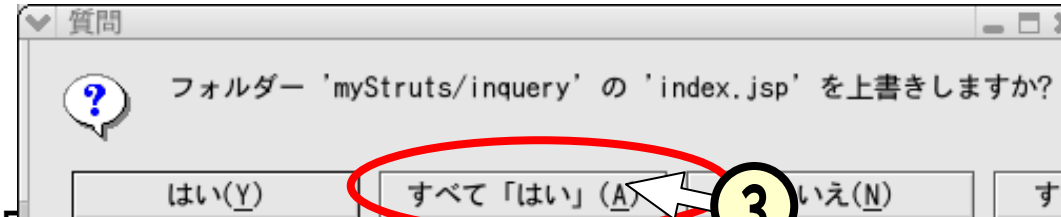


(4．2．3)インポート3

■「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た場合は、[すべて「はい」]をクリック(③)します。



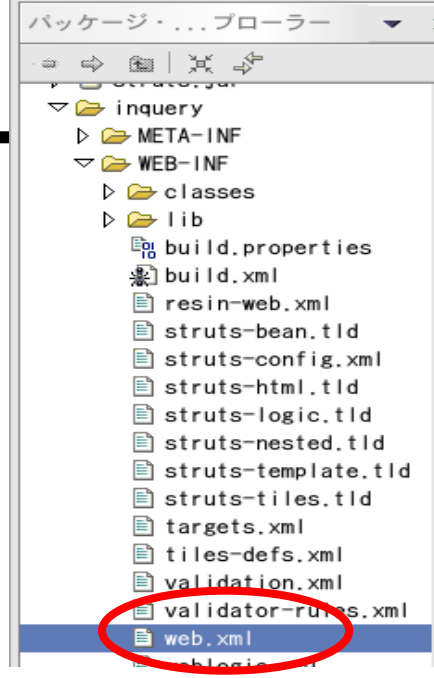


（4．4）web.xmlの編集(Linux)

■次のファイルを編集します。

- inqueryProject/inquery/WEB-INF/web.xml

■井戸のディレクトリを参照しているところを、自分のディレクトリへの参照に変更します。



＜旧＞

<Web.xml>

```
<init-param>
  <param-name>inquery-directory</param-name>
  <param-value>/home/ido/inquery/</param-value>
</init-param>
```

＜新＞

<Web.xml>

```
<init-param>
  <param-name>inquery-directory</param-name>
  <param-value>/home/c30xxxxx/inquery/</param-value>
</init-param>
```

自分のホーム・ディレクトリ

（4．5）web.xmlの編集（Windows）

■次のファイルを編集します。

- inqueryProject/inquery/WEB-INF/web.xml

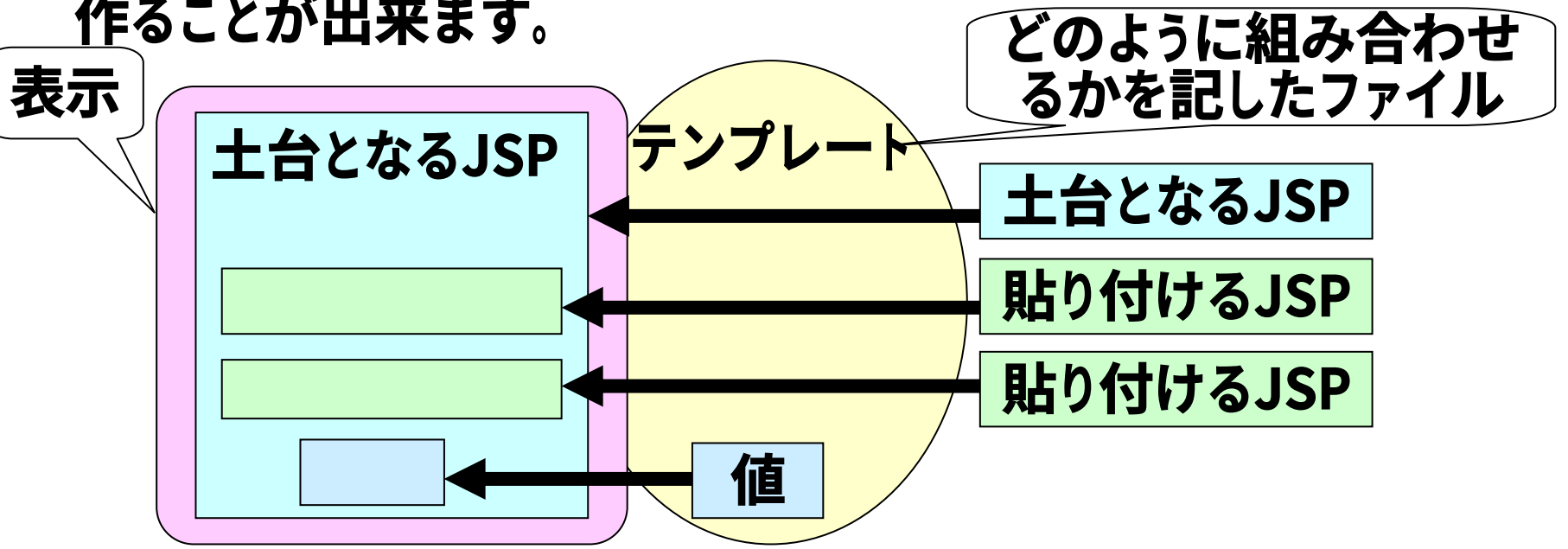
■各自の環境に合わせて、次の部分のフォルダを変更してください。

<Web.xml>

```
<init-param>
  <param-name>inquery-directory</param-name>
  <param-value>C:¥eclipsework¥inqueryProject¥inqueryData¥
               </param-value>
</init-param>
<init-param>
  <param-name>logApi-init-file</param-name>
  <param-value>
C:¥eclipsework¥inqueryProject¥inqueryData¥inqueryLog.prop
  </param-value>
</init-param>
```

(5) tilesによる表示の共通化

- Tilesとは、Webサイト内の各ページの表示に、形式として統一性を持たすことが出来る仕組みです。
- タイル(tile)を貼り付けるように、土台となるJSPのページにさまざまなJSPの表示や値（文字など）を貼り付けることで、表示を作ることが出来ます。

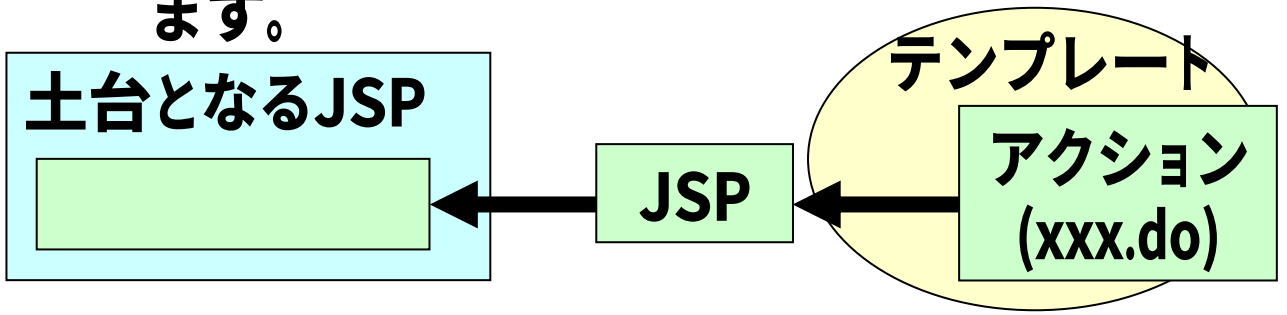


- テンプレートと呼ぶファイル中に、貼り付ける組み合わせを定義するだけで表示を定義出来ます。
- 同じ土台、同じJSPを組み合わせに用いることで、表示に統一性を持たすことが出来ます。

(5．1) tilesの何が便利なのか？

■JSPの表示を組み合わせで実現する方法には、タグ・ライブラリ (<jsp:include>) やBeansを用いる方法もあります。Tilesを使うと、次のような利点があります。

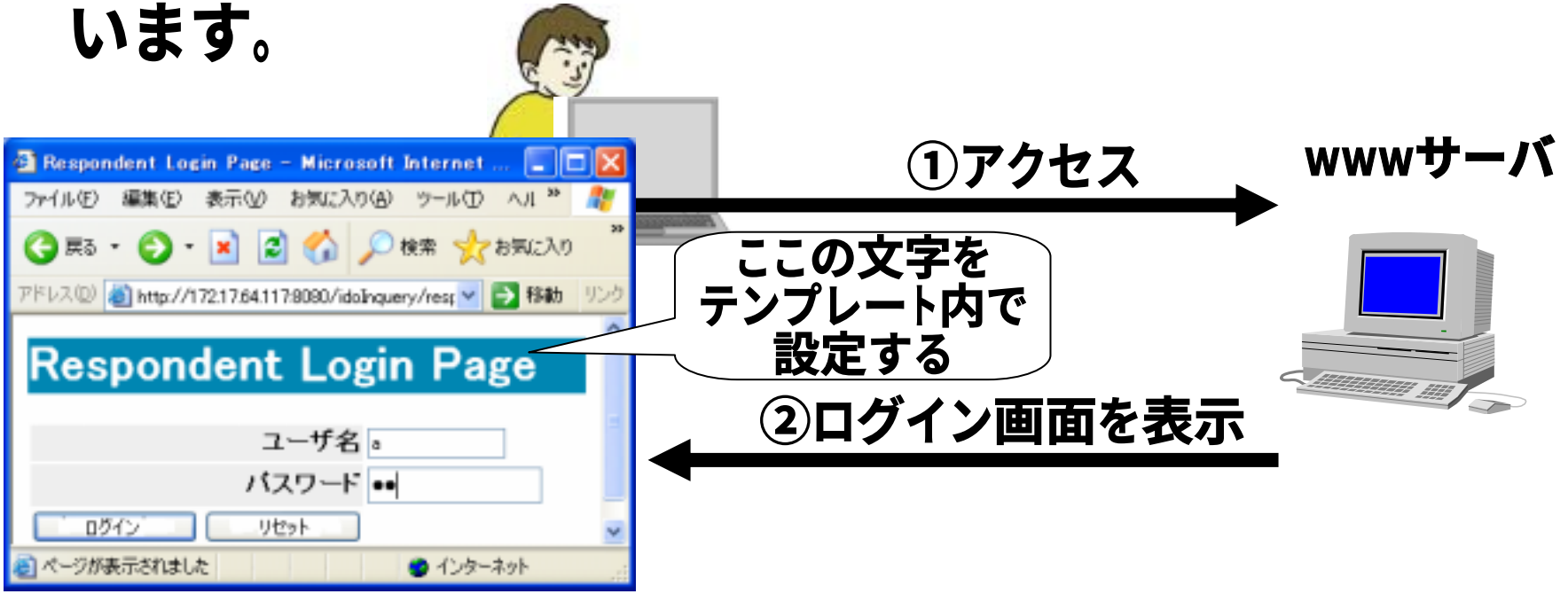
- プログラム／JSPから離れ、テンプレートの中で柔軟に表示の組み合わせを設定することが出来ます。
  - ◆例えば、単純なJSPだけでなく、アクション (xxx.do) を実行した結果のJSPを貼り付けることを指定することも出来きます。



- 定義した表示は、継承により効率よく利用できます。
- 表示に伴う処理 (tiles controller) を定義することにより、あるページの表示の可否の条件等を統一的に決めることが出来ます。

(5．2) アンケート･システムでの実装

■アンケートシステムでは、回答者のログイン画面の表示において、単純に値 (文字) を設定するために用いています。

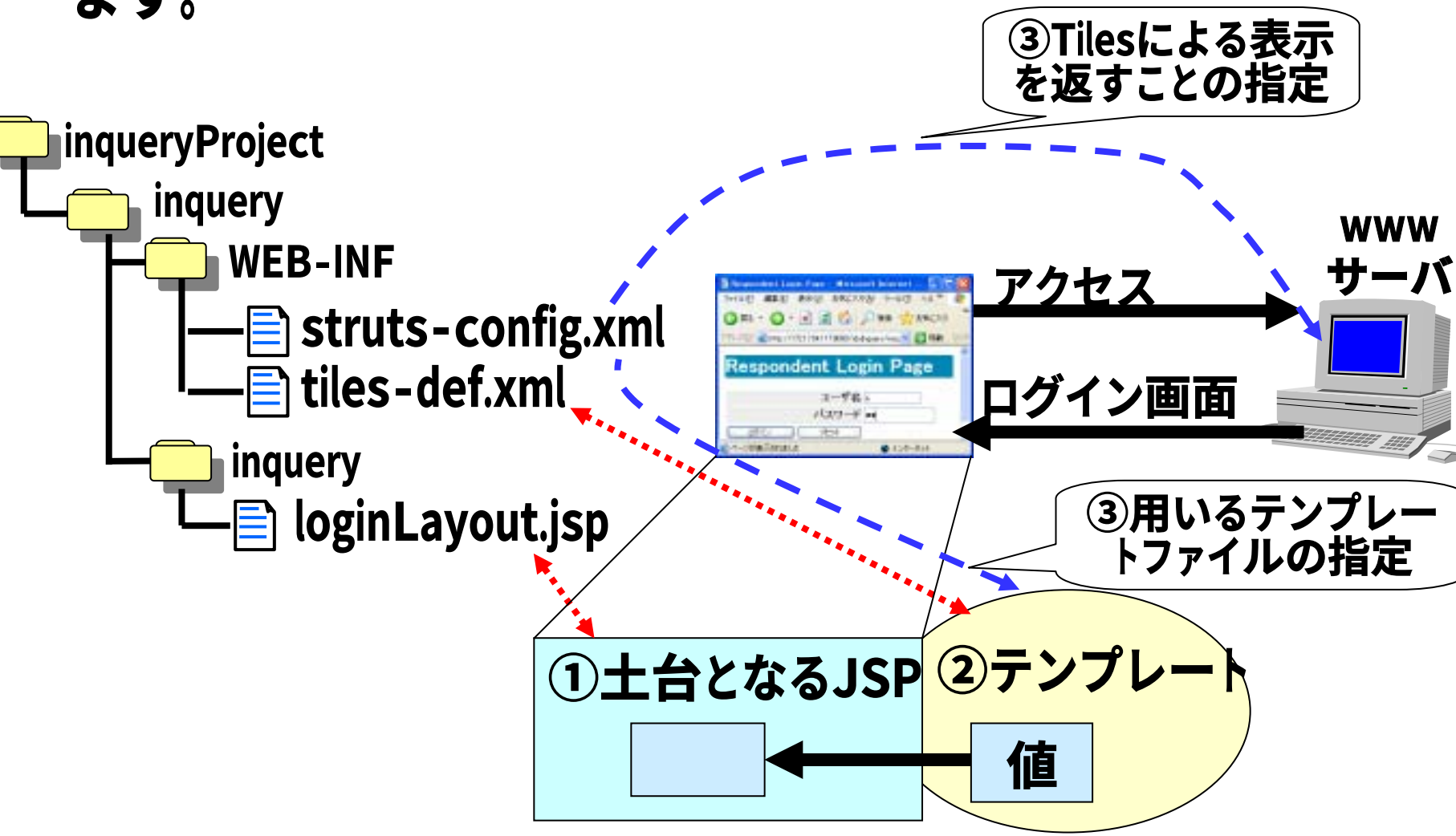


■管理者のログイン画面を作成する練習問題の際に、同じように利用してみてください。

■Tilesのさまざまな使い方については、教科書を参照してください。

(5．3) 使用するファイル

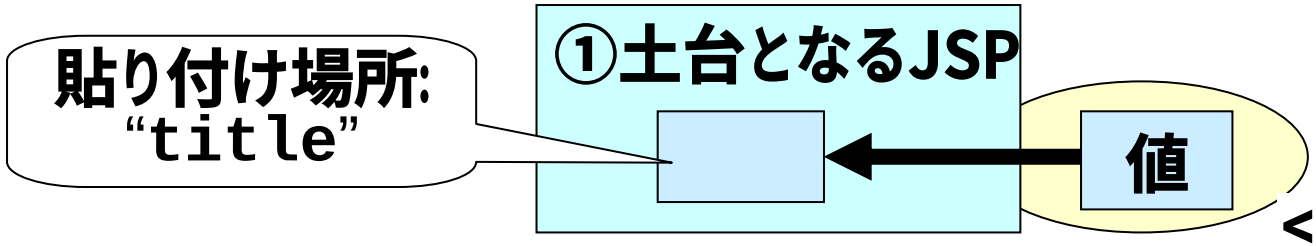
■図中、①～④の順に、次のスライドから説明していきます。



(5．4) 土台となるJSP、“loginLayouot.jsp”

■土台となるJSPは、次の点を除けば普通のJSPです。

- 貼り付けを行う場所が、タグ・ライブラリで指定されている。



<loginLayout.jsp>

```
<%@ page contentType="text/html;charset=iso-2022-jp" %>
(略)
<%@ taglib uri="/tags/struts-tiles" prefix="tiles" %>
(略)
<html:html locale="true">
<head>
<title><tiles:getAsString name="title" /></title>
</head>
<body bgcolor="white">
<h1 style="color:white;background-color:#0086b2;">
        <tiles:getAsString name="title" /></h1>
(略)
```

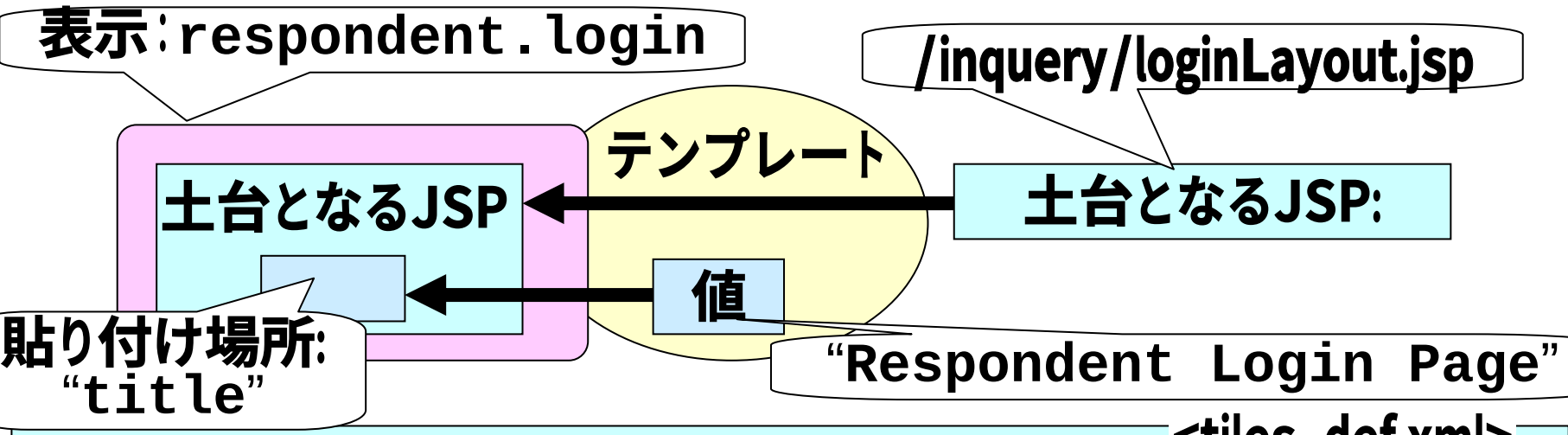
Tilesのタグライブラリを使用することを記したtaglibディレクティヴ

貼り付け場所“title”を示したタグ(文字として貼り付け：getAsString)

同上

(5．5) テンプレート、“tiles-def.xml”

■“respondent.login”という名前の表示 (tile) が定義されています。



<tiles-def.xml>

```
(略)
<tiles-definitions>
  <definition name="respondent.login"
              path="/inquery/loginLayout.jsp" >
    <put name="title"  value="Respondent Login Page" />
  </definition>
(略)
</tiles-definitions>
```

表示 (tile) の名前

土台となるJSP

貼り付け場所

値

(5．6) テンプレートの指定

■前スライドに記したファイル“tiles-def.xml”をテンプレートとして用いることを、“struts-config.xml”に指定します。

<struts-config.xml>

```
(略)
<!-- ================ Plug Ins Configuration -->
  <!-- ========== Tiles plugin ====================  -->
(略)
<plug-in classname="org.apache.struts.tiles.TilesPlugin">
    <set-property property="definitions-config"
                          value="/WEB-INF/tiles-defs.xml" />
    <set-property property="moduleAware" value="true" />
    <set-property property="definitions-parser-validate"
                  value="true" />
</plug-in>
(略)
```

使用するテンプレートの指定

(5．7) tilesによる表示を返すことの指定

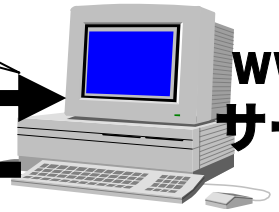
■アクション･マップには、“/idoInquery/respondentLogin.do”にてアクセスされた際、スライド(5.5)で定義した表示(tile)“respondent.login”が指定されています。

アクセス

http://(*)/idoInquery/respondentLogin.do

“respondent.login”による表示

表示



www
サーバ

<struts-config.xml>

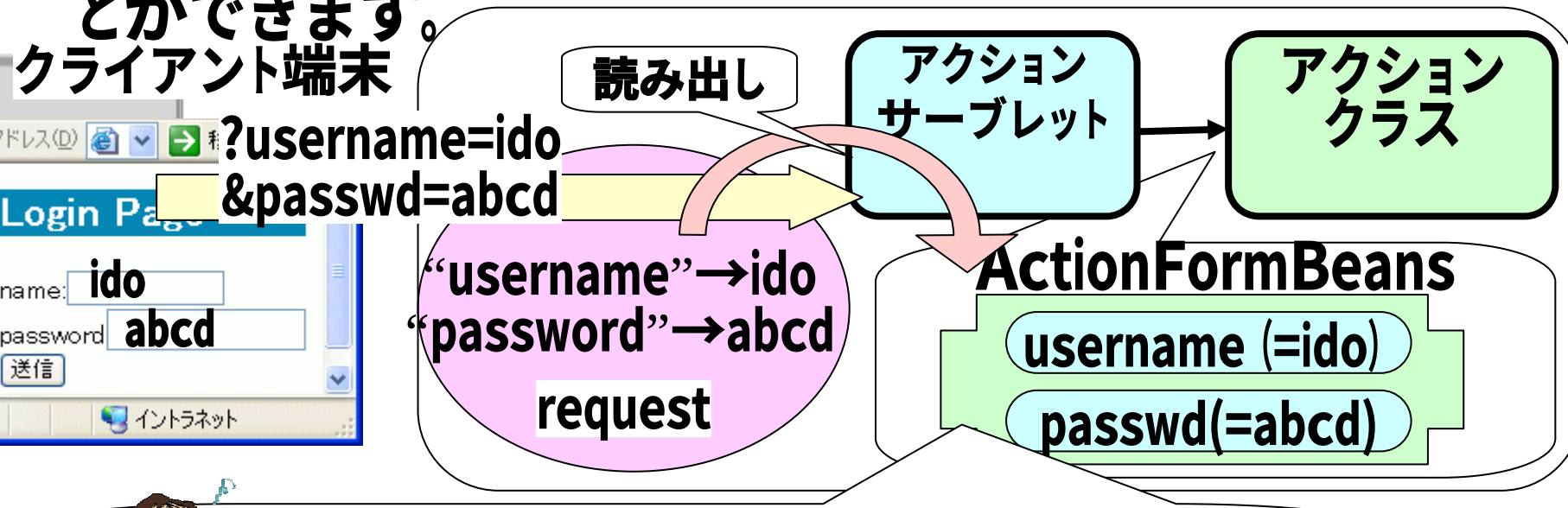
```
(略)
<!-- ===================== Action Mapping Definitions -->
   <action-mappings>
(略)
  <!-- Respondent -->
     <action
        path="/respondentLogin"
        type="org.apache.struts.tiles.actions.NoOpAction">
       <forward name="success" path="respondent.login" />
     </action>
(略)
```

アクセス

表示

(6．1) アクション･フォーム･ビーンズ

■Strutsでは、アクション･フォーム･ビーンズによりクライアント端末からの入力をアクションクラスに受け取ることができます。



いつも型どおりのアクセサのコーディング

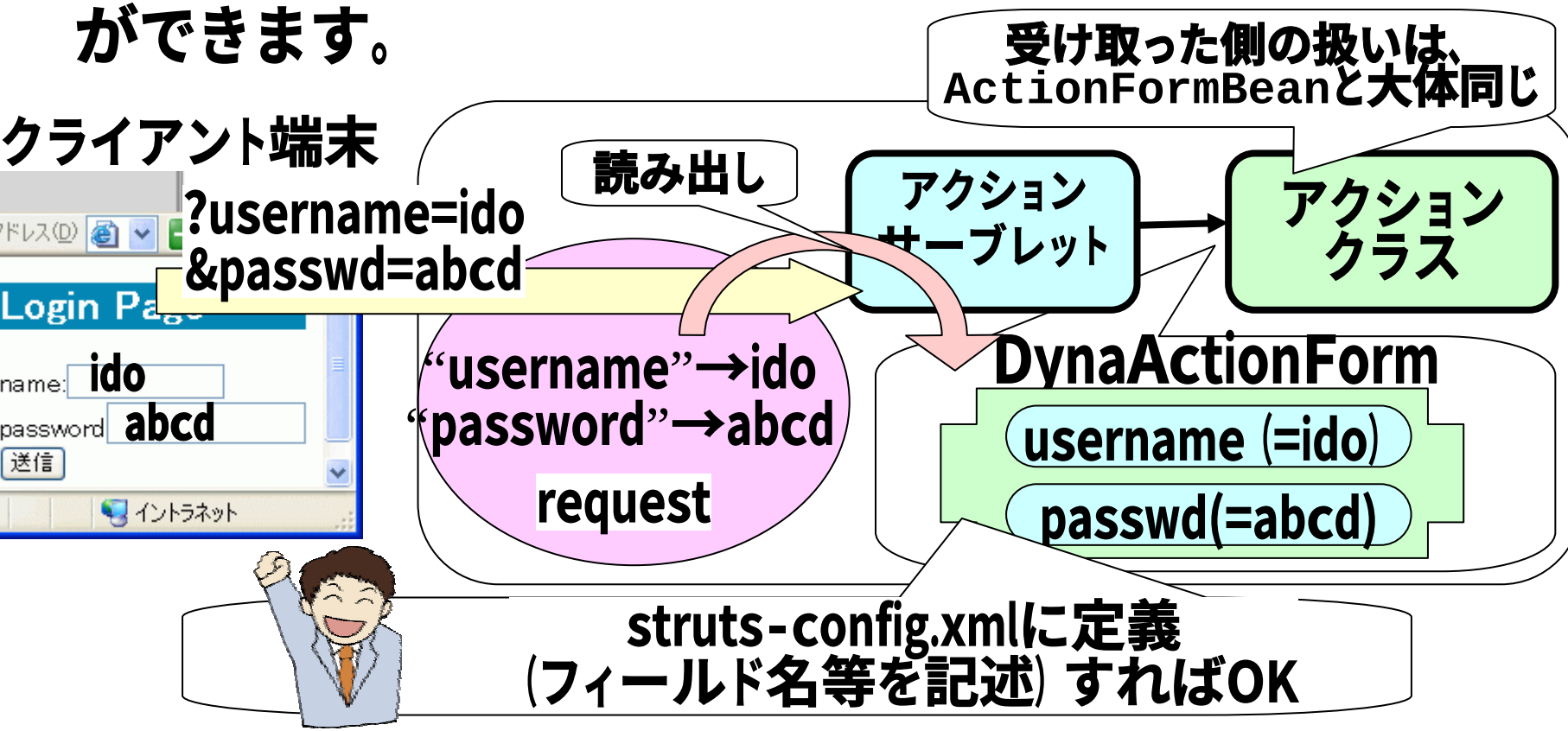
```
public class loginForm extends ActionForm {
            private String username;
            private String passwd;
            public String getUsername(){
                        return this.username;
          :
```

■端末からの入力のためだけに、いつも同じ形をしたアクション･フォーム･ビーンズのクラスを定義するのは、やや面倒です。

（6．2）DynaActionForm

■DynaActionForm（ダイナ・アクション・フォーム）を使えば、いちいちクラスを作る必要はありません。

■DynaActionFormを“struts-config.xml”内に定義しておけば、アクション・フォーム・ビーンと同じように使うことができます。



（6．3）DynaActionFormの定義

■DynaActionFormは、ActionFormBeanと似た感じで、コンフィグレーション･ファイル“strtus-config.xml”の中に定義します。

<struts-config.xml>

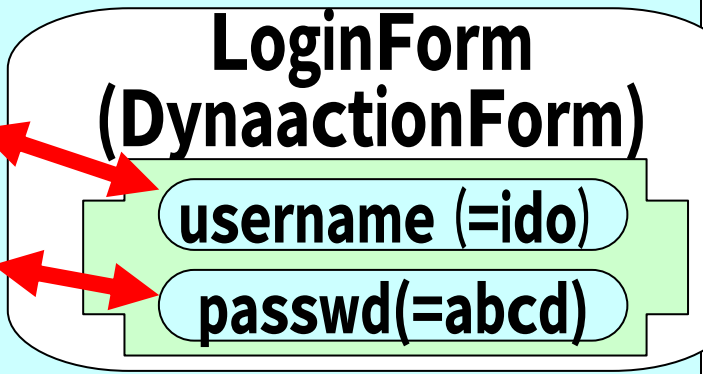
```
(略)
!-- ============================ Form Bean Definitions -->
 <form-beans>
   <form-bean
     name="LoginForm"
     type="org.apache.struts.action.DynaActionForm" >

     <form-property
       name="username"
       type="java.lang.String" />
     <form-property
       name="passwd"
       type="java.lang.String" />
   </form-bean>
```

DynaActionFormを使うことを指定

フィールドの定義

フィールド名

文字列型



(6．4) アクションクラスでの扱い

■クライアント端末からの入力をDynaActionFormで扱う際、アクションクラスでの処理はActionFormBeanと似たような感じになります。

<CheckLoginAction.java>

```
public ActionForward execute(ActionMapping map,
            ActionForm form,
            :                    ){
```

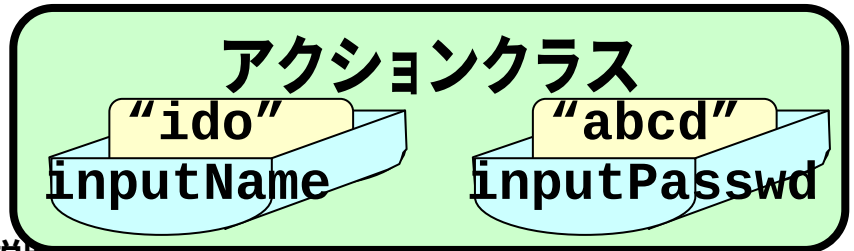
DynaActionFormを使う

```
DynaBean loginForm <CheckLoginAction.java>
 =(DynaBean)form;
String inputName
 =(String)loginForm.get("username");
String inputPasswd
 =(String)loginForm.get("passwd");
```

ActionFromBean(LoginForm)を使う

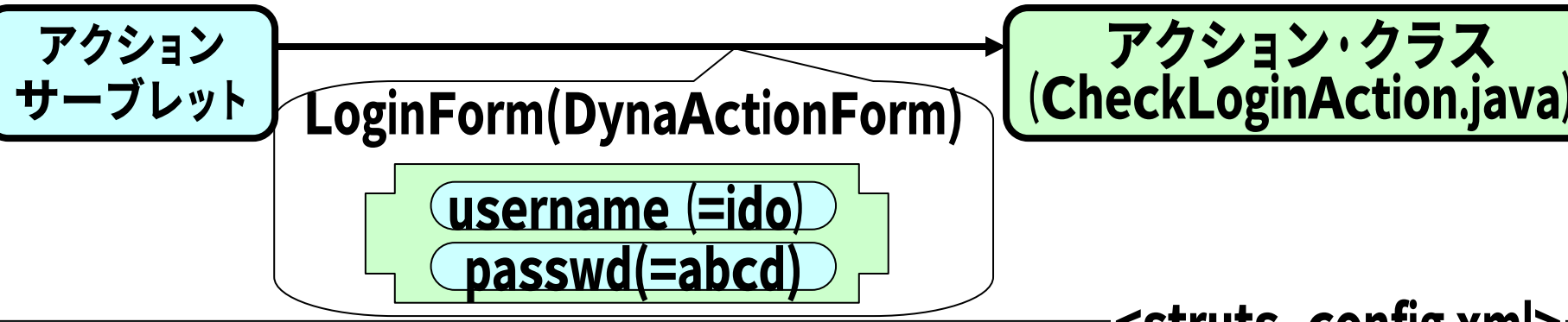
```
LoginForm loginForm
 =(LoginForm)form;
String inputName
 =loginForm.getUsername()
String inputName
 =loginForm.getPasswd();
```

?username=ido
&passwd=abcd



(6．5) アクション･マップでの指定

■コンフィグレーション･ファイル (“struts-config.xml”) でも、ActionFormBeanと同様に指定を行います。



<struts-config.xml>

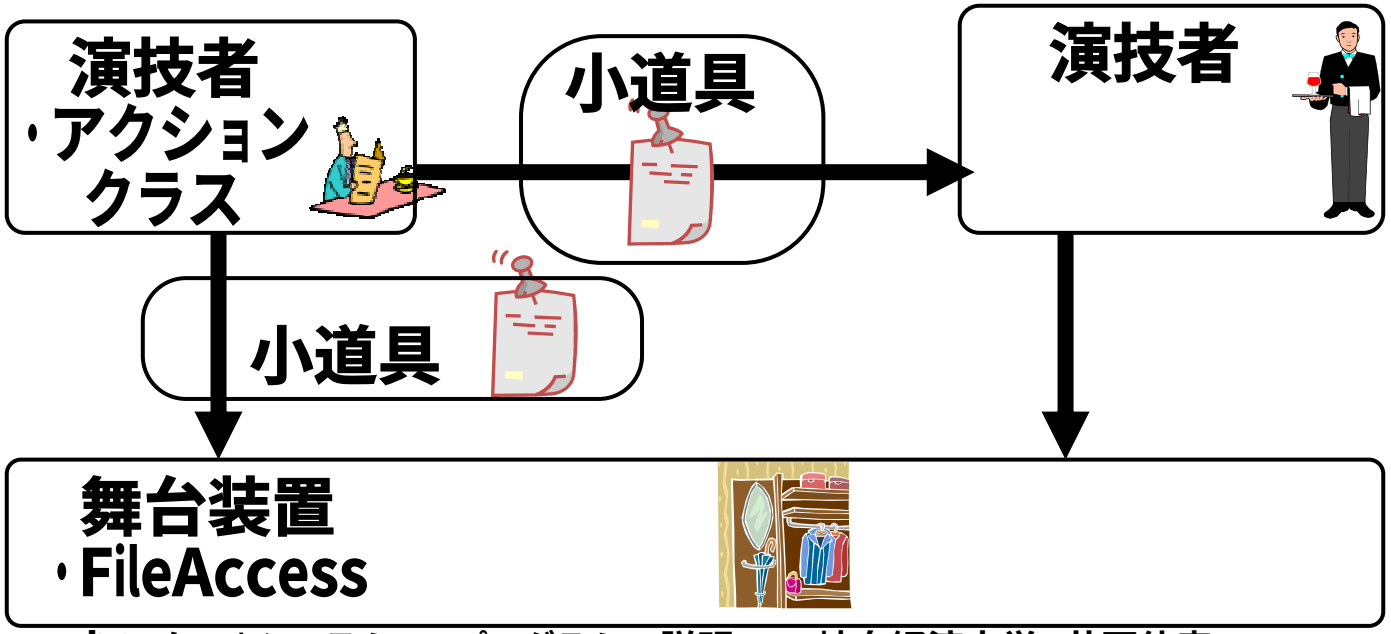
```
(略)
<!-- ===================== Action Mapping Definitions -->
   <action-mappings>
(略)
  <!-- Respondent -->
    <action
      path="/checkLogin"
      type="inquery.CheckLoginAction"
      name="LoginForm" scope="request">
(略)
```

(7) データ受け渡しのクラス

■オブジェクト指向プログラミングでは、プログラムの単位としてのクラスを、“もの”と考えます。

■アンケートシステムで考えるオブジェクト (＝もの) を舞台に例えると、小道具のようにオブジェクト間でやりとりされる“もの”に対応するクラスがいくつかあります。

■これらのクラスについて説明していきます。



(7．0) タグ・ライブラリ

■以下で説明するクラスのうち、次のものはJSPでのタグ・ライブラリによる表示の際に用いられます。

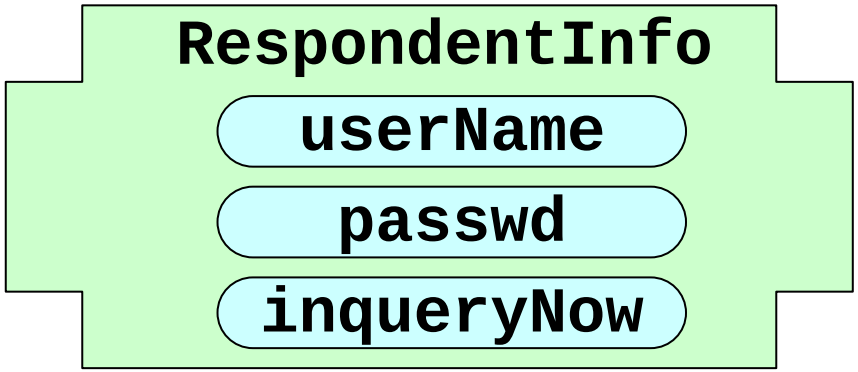
- (7．4．1) InqueryList
- (7．8) QandAForm

■タグ・ライブラリで用いるために、クラスとしての定義としては不自然な部分もあり、やや難解かも知れません。

■JSPでのタグ・ライブラリとの関係については、後で説明します。

（7．1）RespondentInfo(回答者情報)

■回答者の情報をまとめたものです。

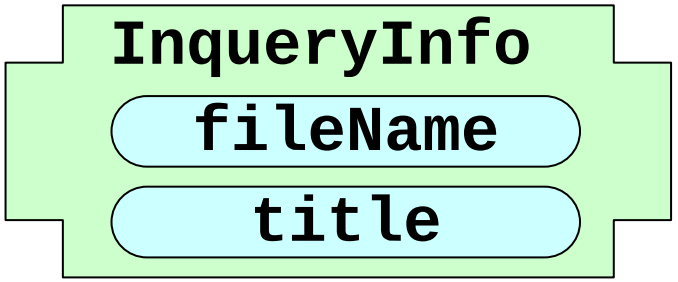


■それぞれのフィールドには、次のようなアクセサ（setterとgetter）があります。意向のクラスの説明では、アクセサについては記しません。

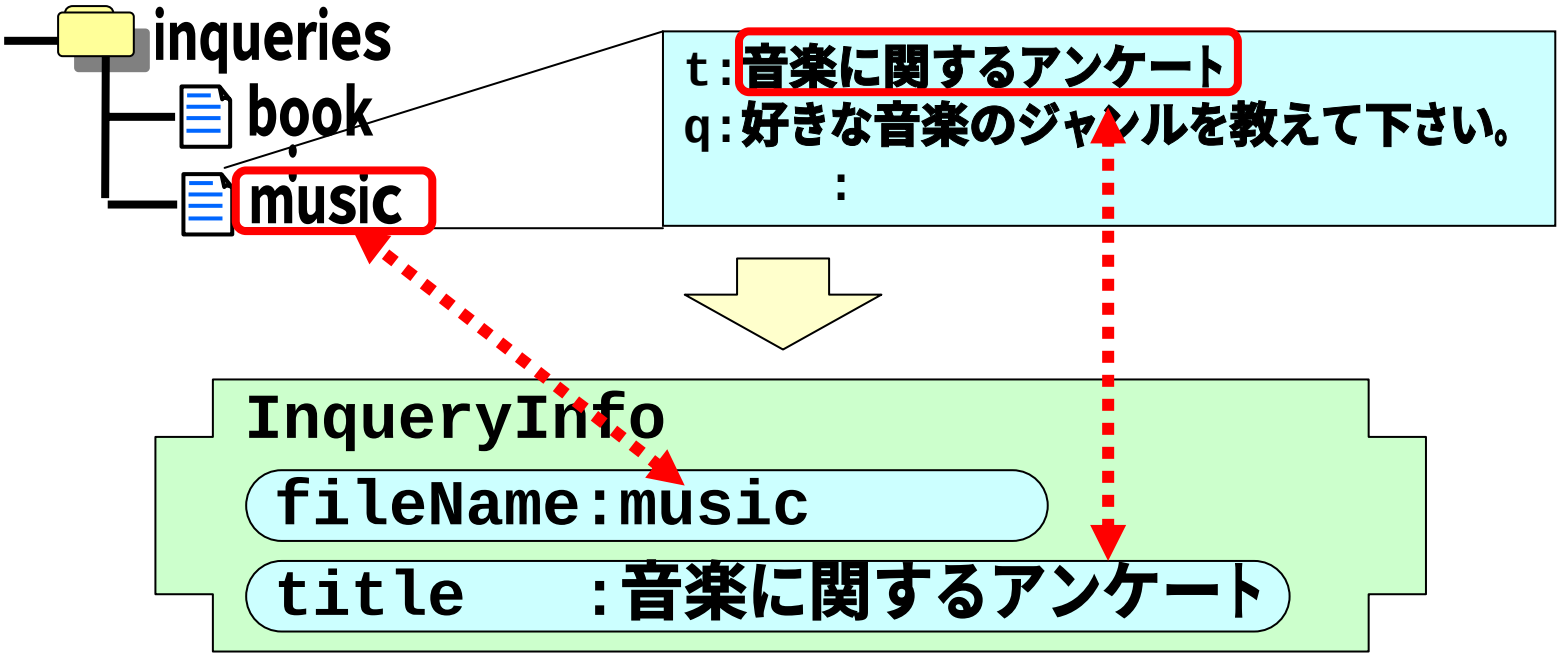
```
public void setUserName(String userName){
  this.userName = userName;
}
public String getUserName(){
  return this.userName;
}
```

(7．2) InqueryInfo (アンケート情報)

■アンケートの情報をまとめたものです。

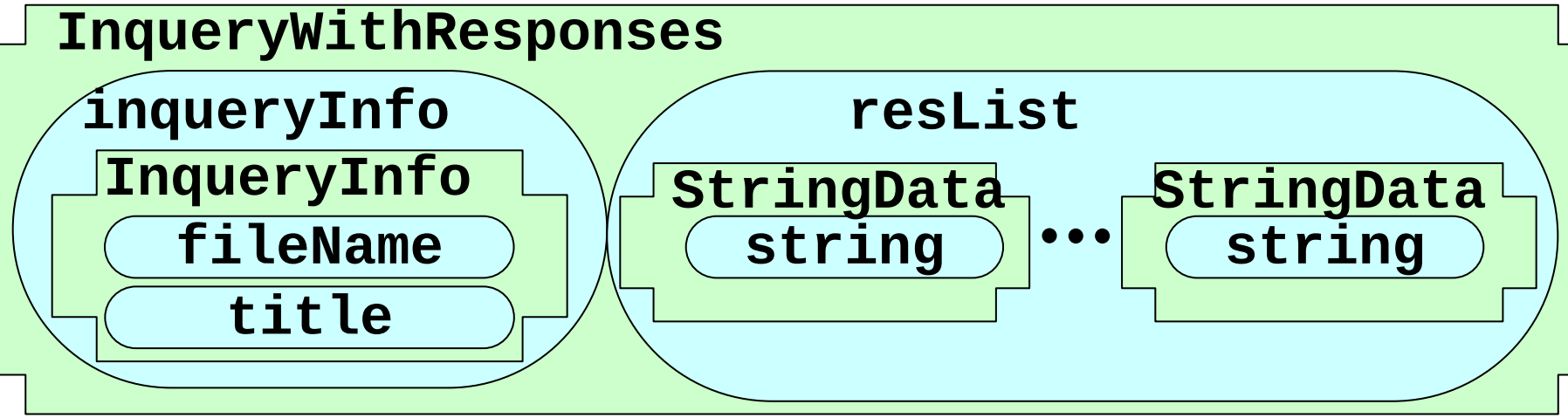


■例として、次のような値が入ります。

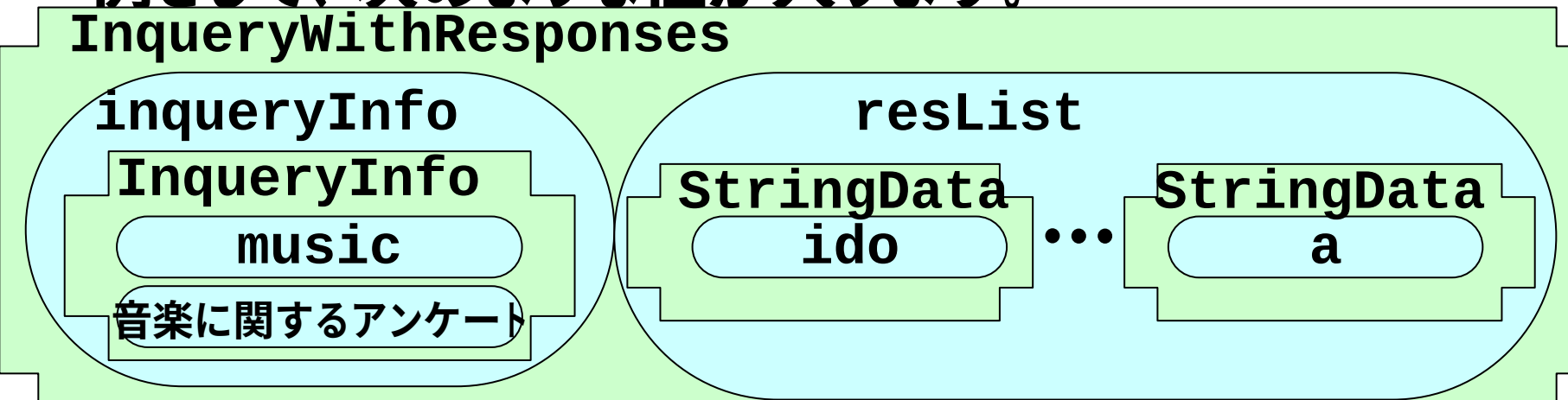


（7．3）InqueryWithResponses

■前スライドのアンケート情報に加えて、そのアンケートに回答した回答者のリストを加えています。

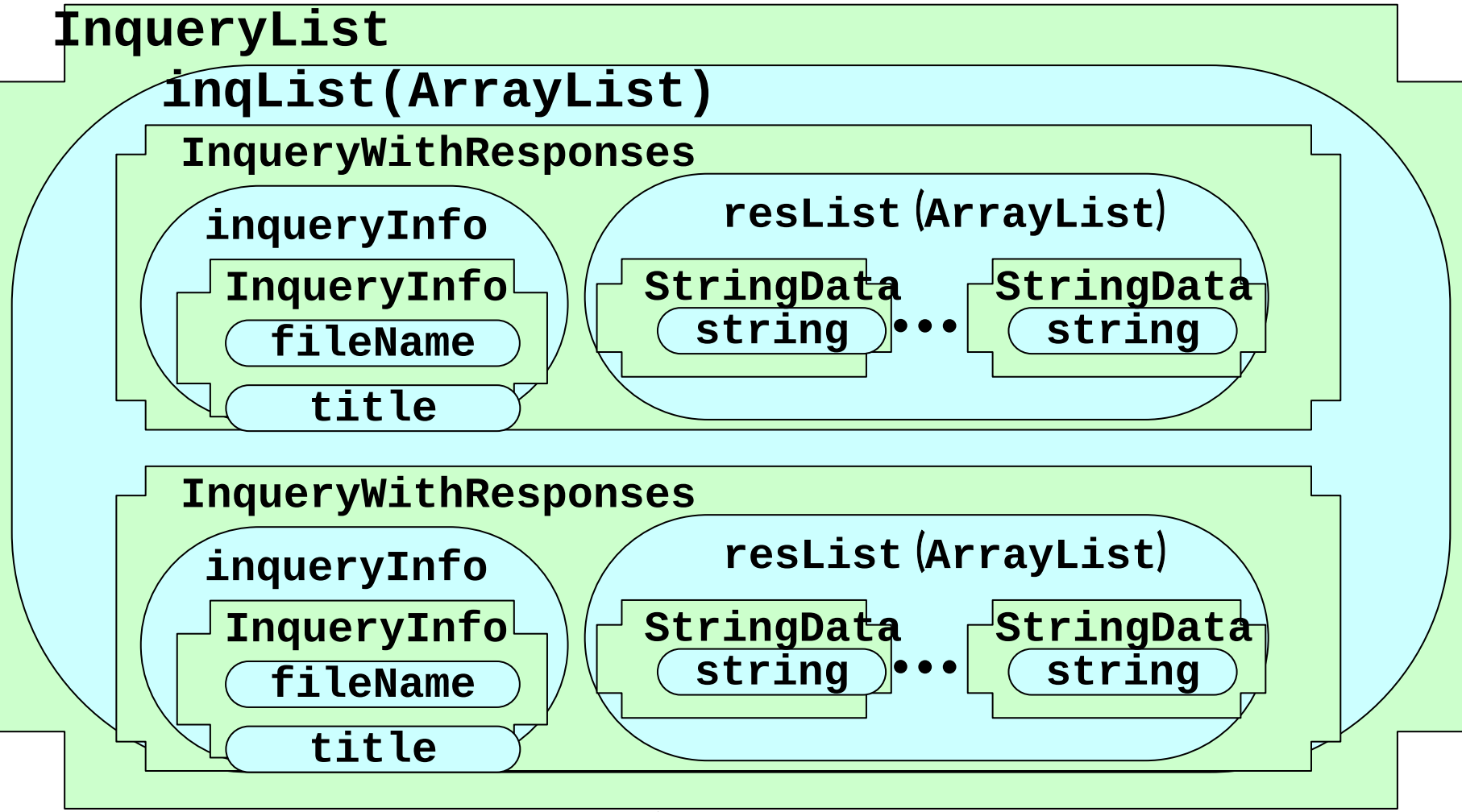


■例として、次のような値が入ります。



(7．4．1) InqueryList (アンケート・リスト)

■前スライドのInqueryWithResponsesを複数まとめたものです。

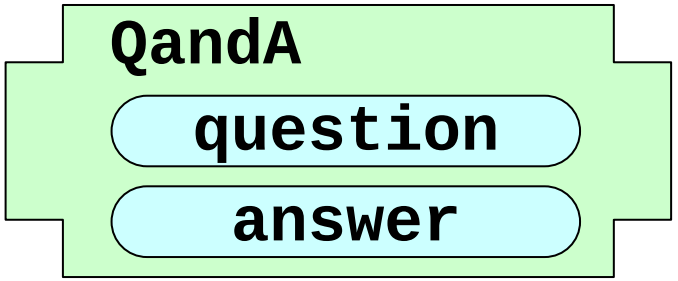


（7．4．2）InqueryListの内容のイメージ

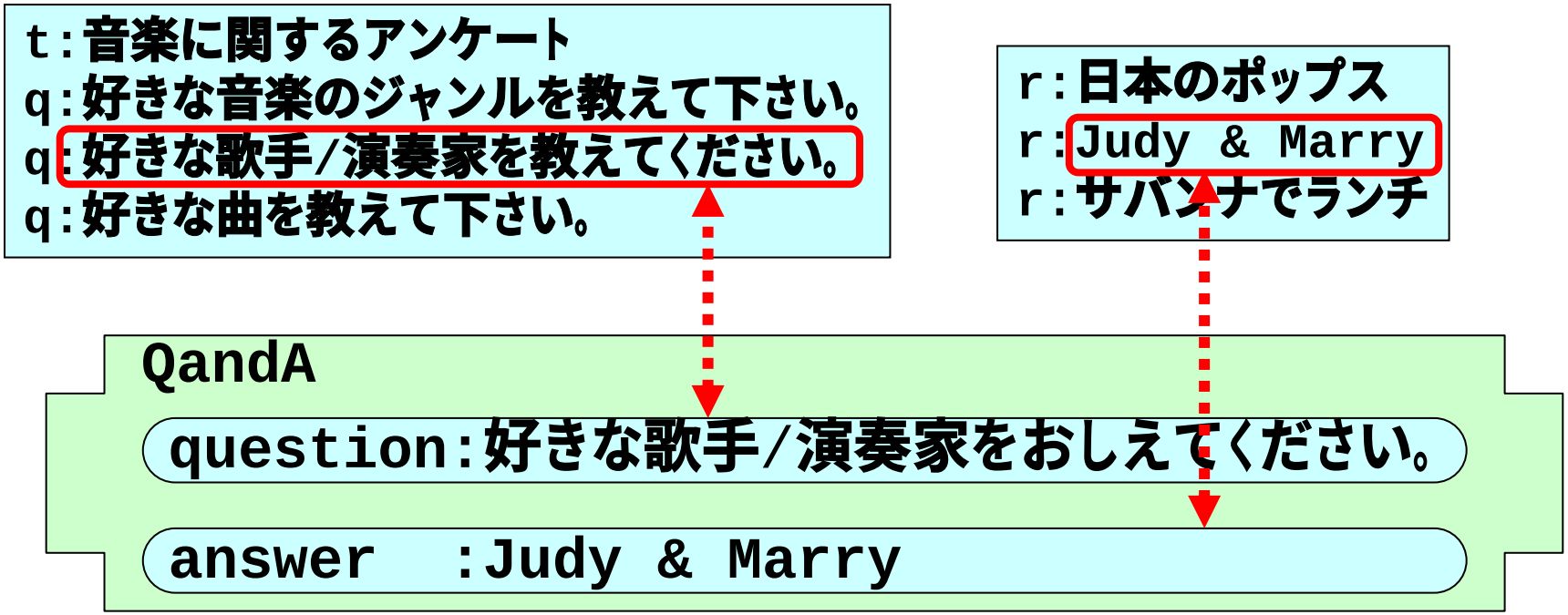
■InqueryListの内容のイメージを示します。

(7．5) QandA (質問と回答の組)

■アンケートにおける質問と回答の組をまとめたものです。

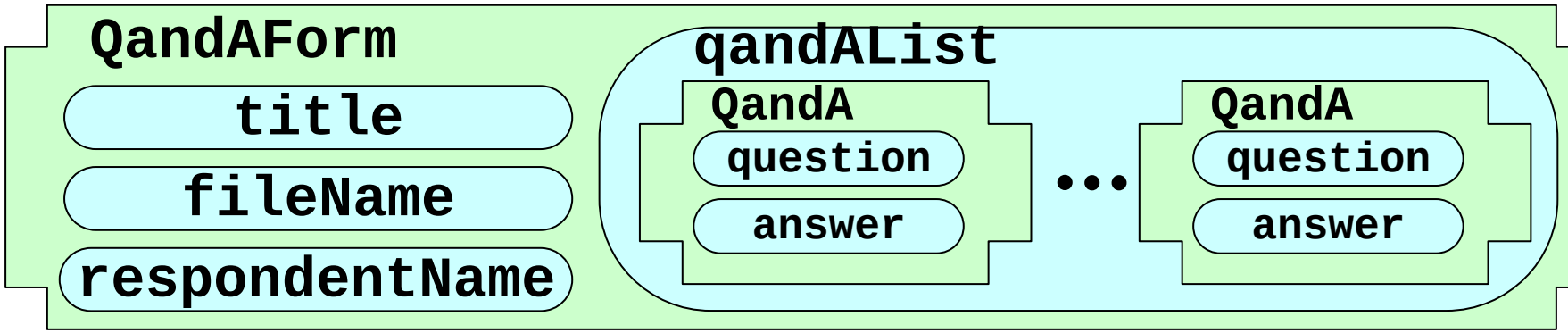


■例として、次のような値が入ります。

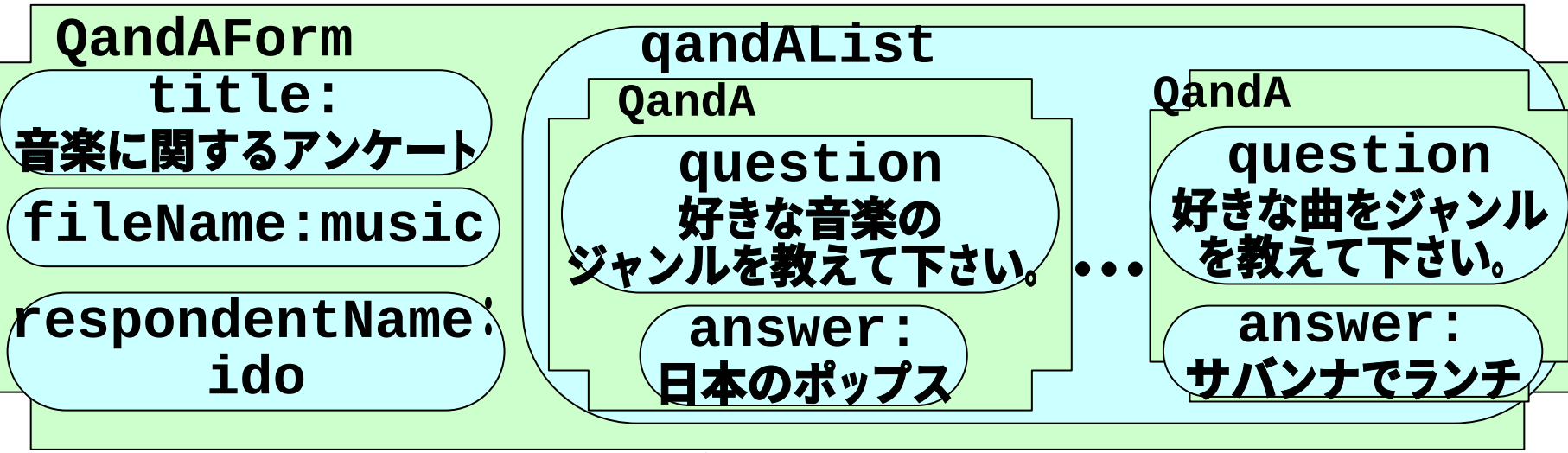


⑺.6) QandAForm(質問･回答に関するフォーム)

■アンケートの質問と回答に関する情報をまとめています。内部にQandA(前スライド)のリストを含んでいます。



■例として、次のような値が入ります。



(8) ファイル操作

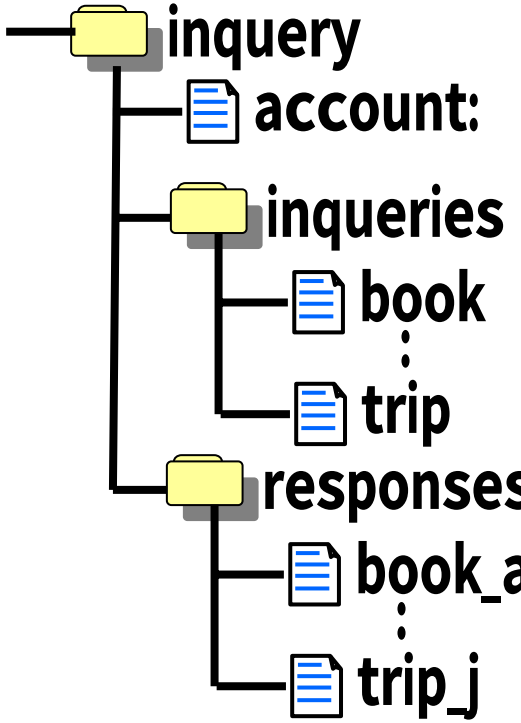
■アンケートシステムは、スライド(3.2)に示した右のようなファイルに基づき動作します。

■Javaでのファイル操作の方法については、次の資料を参照してください。

「シャバドゥビ、ジャバ　－Java覚書－」
(http://www.gifu-keizai.ac.jp/~ido/doc/java/java_text.pdf)

- (14) ファイル入力処理
- (15) ファイル出力処理
- (16) 例外処理
- (17) ディレクトリ・ファイルの操作

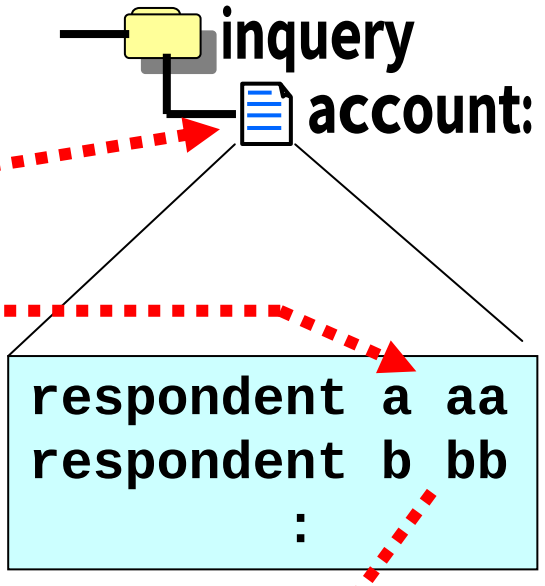
■ファイル操作のプログラムは、“FileAccess.java”にまとめてあります。以下、このクラスのメソッドについて記します。



(8．1) setRespondentInfoFromFile

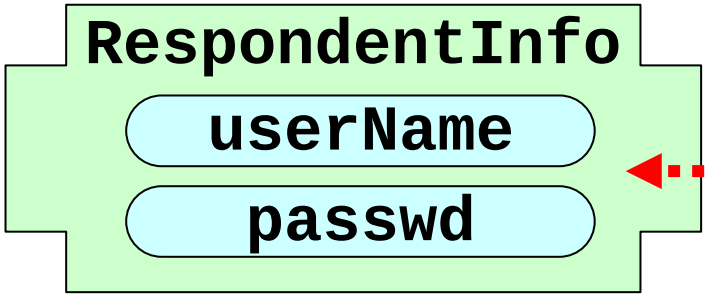
■入力

- String path ：アカウントを記したファイル(スライド(3.2.1)) のパス名
- String inputName ：ユーザ名
- String passwd ：パスワード



■出力

- RespondentInfo ：回答者情報



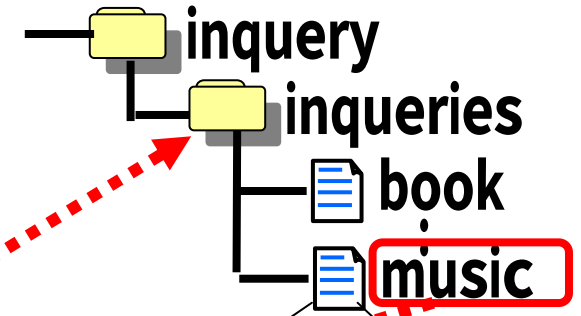
■処理

- アカウントのファイルから、ユーザ名･パスワードで指定されたアカウントを探す。
- 見つかれば、回答者情報にユーザ名･パスワードを設定する。
- 見つからなければ、エラーを返す。

(8．2) setInqueryListFromFile

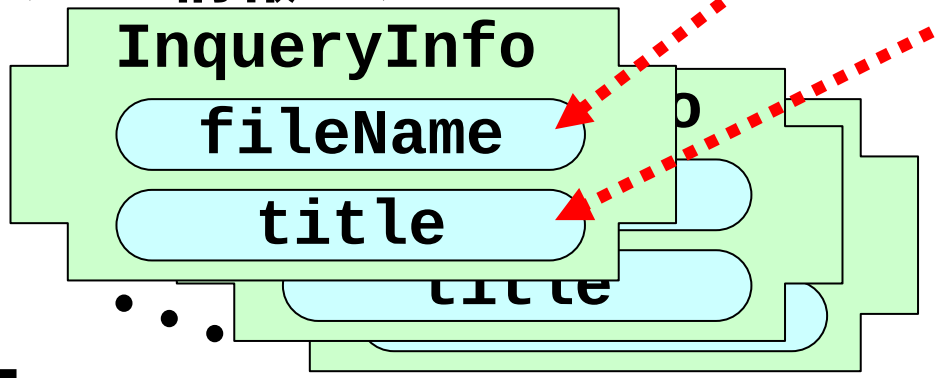
■入力

- `String path` :
  アンケートの質問を記した
  ファイル (スライド(3.2.2)) がある
  ディレクトリのパス名



■出力

- `ArrayList inqueryList` :
  アンケート情報のリスト



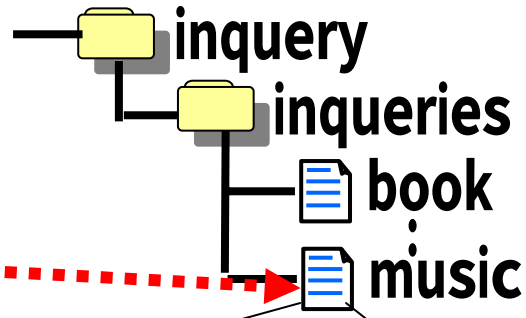
■処理

- それぞれのアンケートにつき、アンケート情報を設定し、一覧を作成する。

(8．3) setQuestionsFromFile

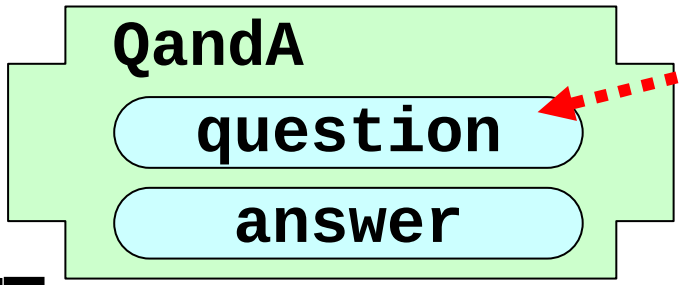
■入力

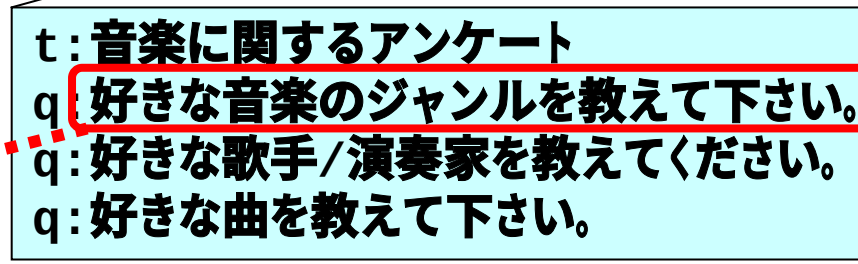
- `String path` :
  アンケートの質問を記した
  ファイル (スライド(3.2.2)) のパス名



■出力

- `QandAForm qandAForm` :
  質問と回答の組のリスト





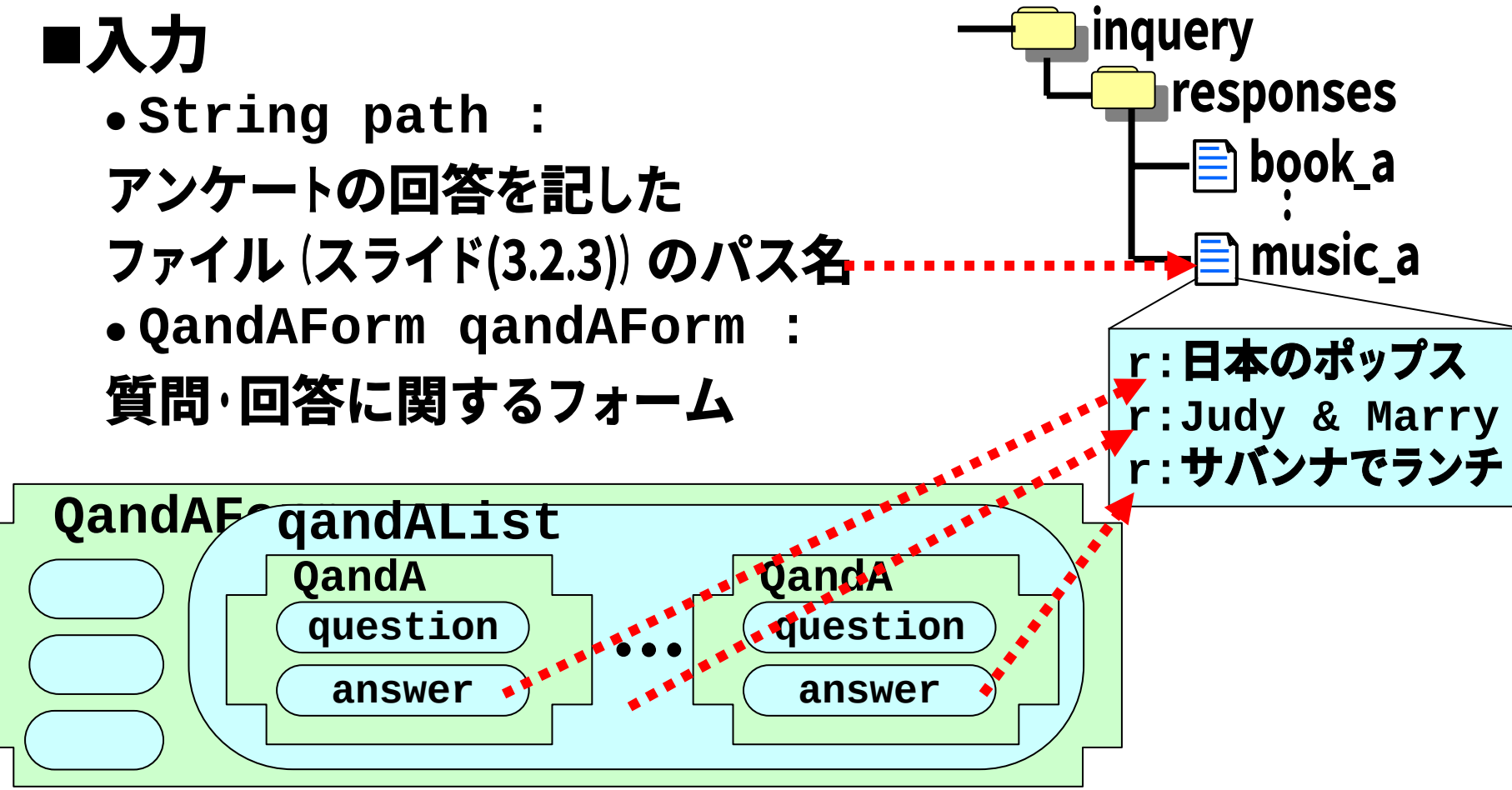
■処理

- 指定されたアンケートファイル内の質問を読み出し、質問と回答の組のリストに設定する。

(8．4) saveResponsesToFile

■入力

- `String path` :
  アンケートの回答を記した
  ファイル(スライド(3.2.3)) のパス名
- `QandAForm qandAForm` :
  質問･回答に関するフォーム



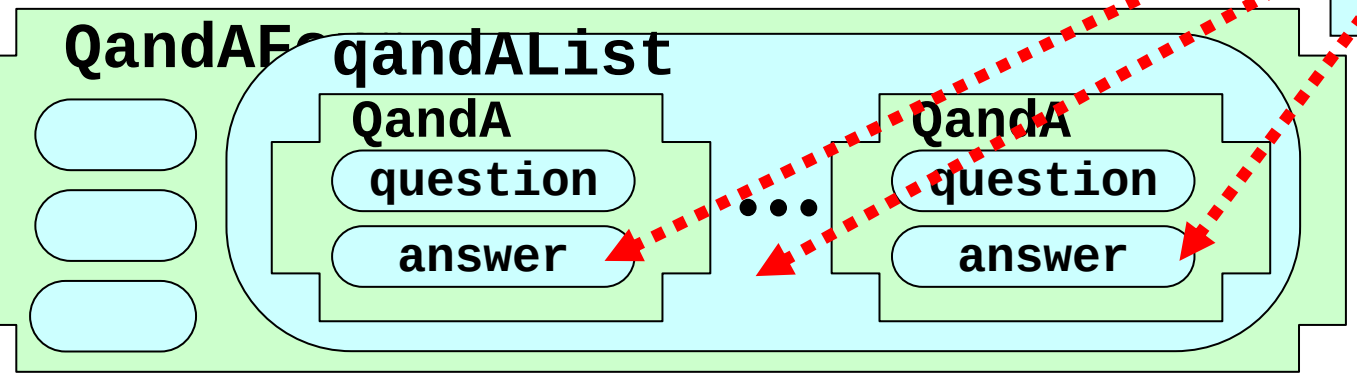
■処理

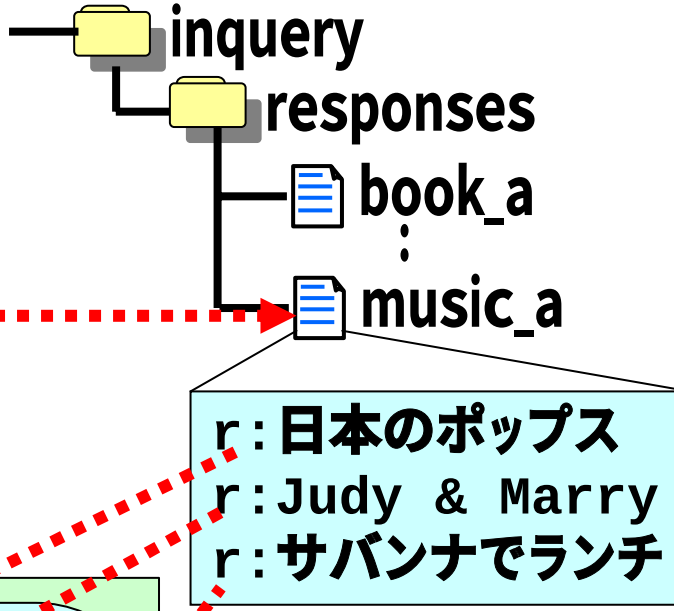
- 指定された回答ファイルへ、質問と回答の組のリストから回答の内容を設定する。

(8．5) setResponsesFromFile

■入力
- String path :
  アンケートの回答を記した
  ファイル (スライド(3.2.3)) のパス名

■出力
- QandAForm qandAForm :
  質問･回答に関するフォーム



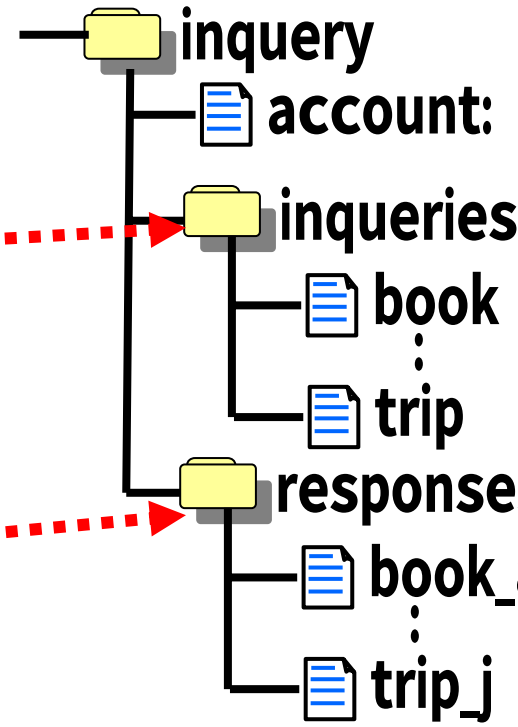


■処理
- 指定された回答ファイルから、質問と回答の組のリストへ、
  回答の内容を設定する。

(8．6) setInqueryListFromFile

■入力

- String inquiriesDirectory :
アンケートの質問を記した
ファイル (スライド(3.2.2)) がある
ディレクトリのパス名
- String responsesDirectory :
アンケートの回答を記した
ファイル (スライド(3.2.3)) がある
ディレクトリのパス名



■出力

- InqueryList inqueryList :
- スライド (7.2) ～(7.4)参照

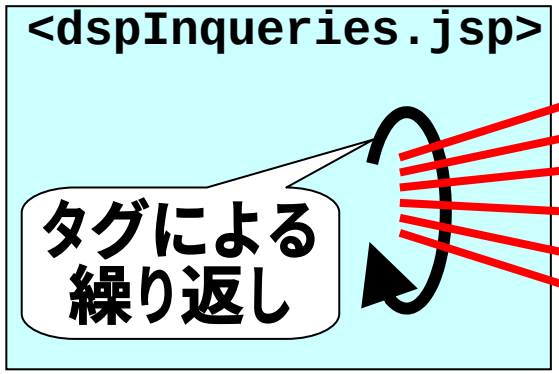
■処理

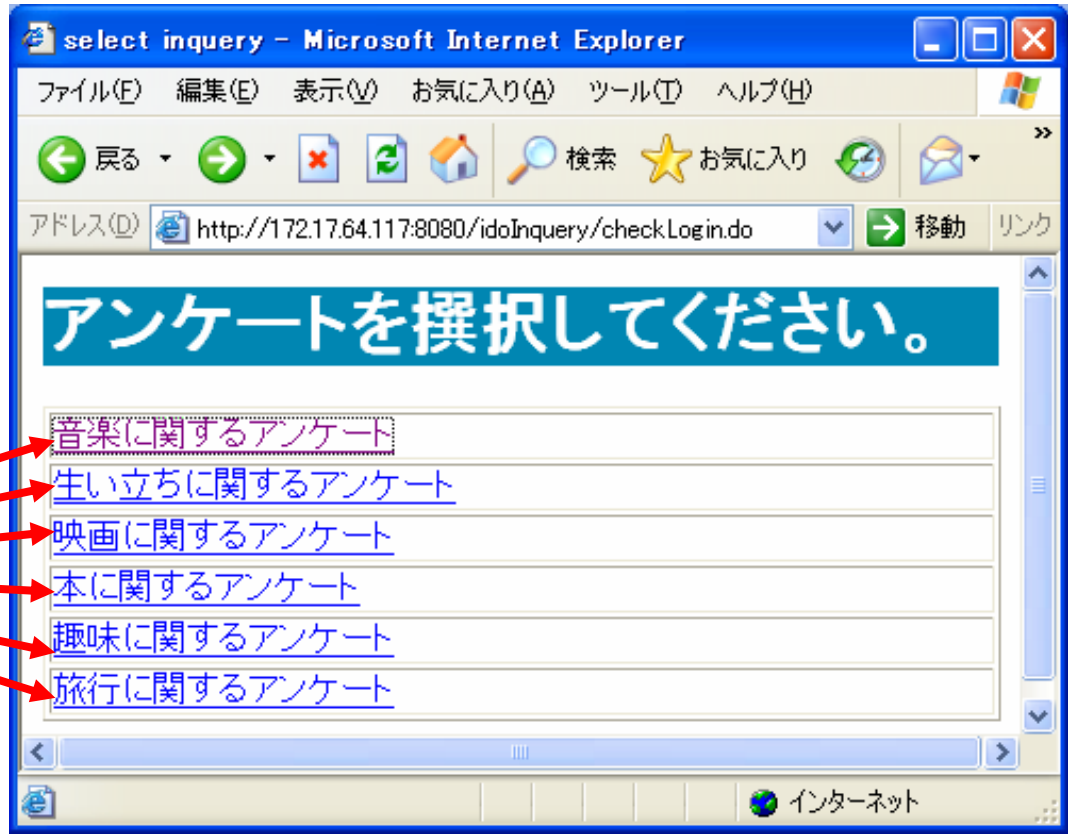
- スライド(7.4)のアンケート・リストを作成します。

(9) タグによる繰り返し

■回答者がアンケートを選択する画面（アンケート選択画面）では、アンケートのタイトルとリンクとが列挙されています。

■この“列挙”は、JSP内のタグにより実現されています。





(9．1)表示のソースコード

■JSPにより作成される表示のソースコートは、次のとおりです。

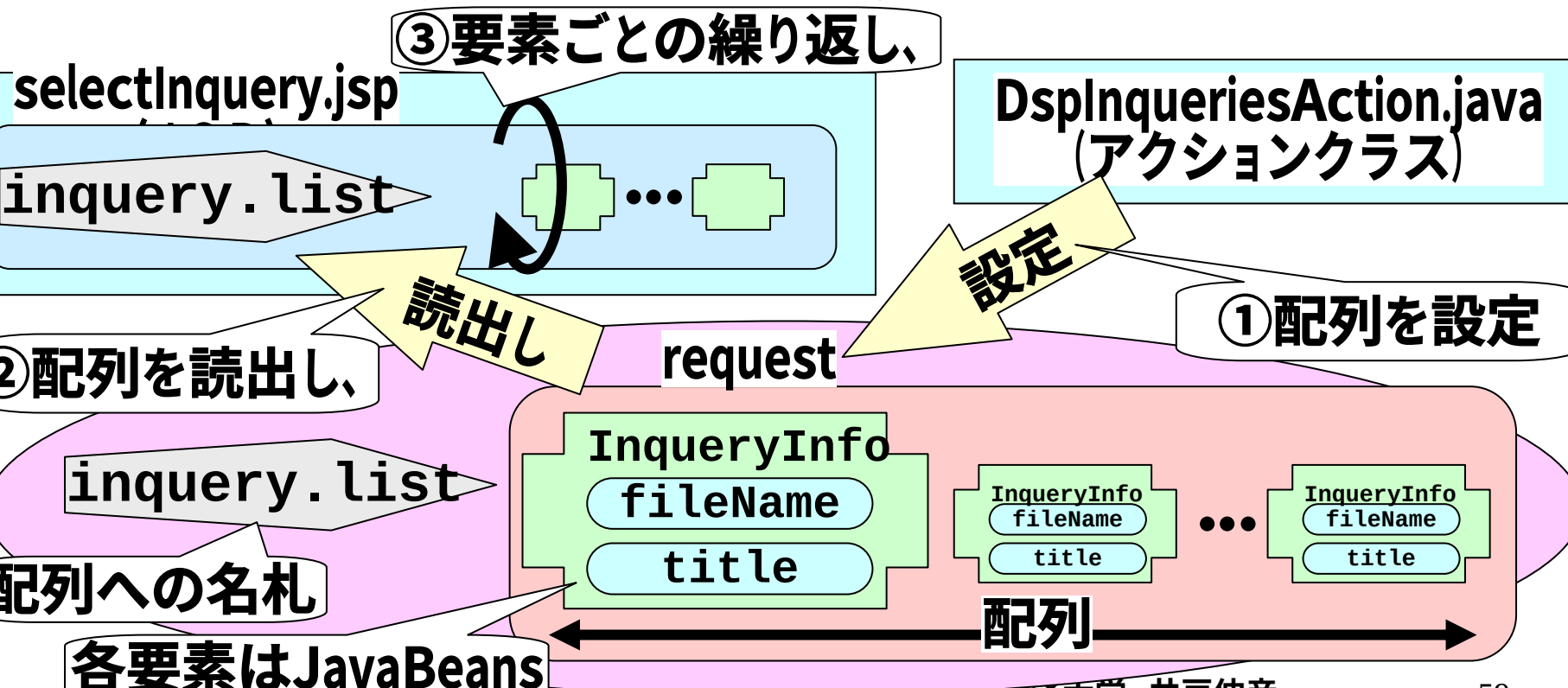
```
(略)
<table border="1" width="450">

  <tr><td>
  <a href="/idoInquery/showInquery.do?fileName=music">
  音楽に関するアンケート</a>
  </td></tr>
  <tr><td>
  <a href="/idoInquery/showInquery.do?fileName=born">
  生い立ちに関するアンケート</a>
  </td></tr>
  <tr><td>
  <a href="/idoInquery/showInquery.do?fileName=movie">
  映画に関するアンケート </a>
  </td></tr>
(略)

</table>
(略)
```

繰り返しによる表示

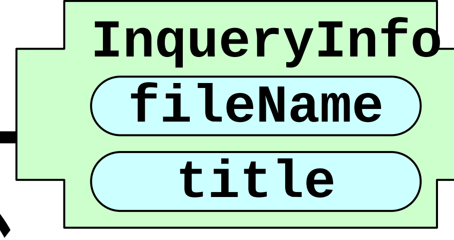
(9．2) アクションクラス、JSP、JavaBeans

■アクションクラス (DspInqueriesAction.java) では 、“inquery.list”という名札をつけて、 request”上に配列を設定します。

■JSP (selectInquery.jsp) では、“inquery.list”というインデックスにより読出した配列の各要素 (JavaBeans) ごとに繰り返しを行います。各要素のフィールドを指定して値を取り出します。



(9．3) Java Beans



■"request"上に設定される配列の要素は、JavaBeansであり、次の条件を満たすようにします。

①引数のないコンストラクタが定義されている。

②プロパティを参照するアクセサ・メソッドが定義されている。

③Serializableインタフェースを実装する。

```
package inquery;
import java.io.Serializable;
public final class InqueryInfo implements Serializable{
    private String fileName;
    private String title;
    public InqueryInfo(){ fileName=null;
                          title=null;}
    public String getFileName(){ return this.fileName; }
    public String getTitle(){ return this.title; }
    public void setFileName(String fileName)
     { this.fileName = fileName; }
    public void setTitle(String title)
     { this.title = title; }
}
```

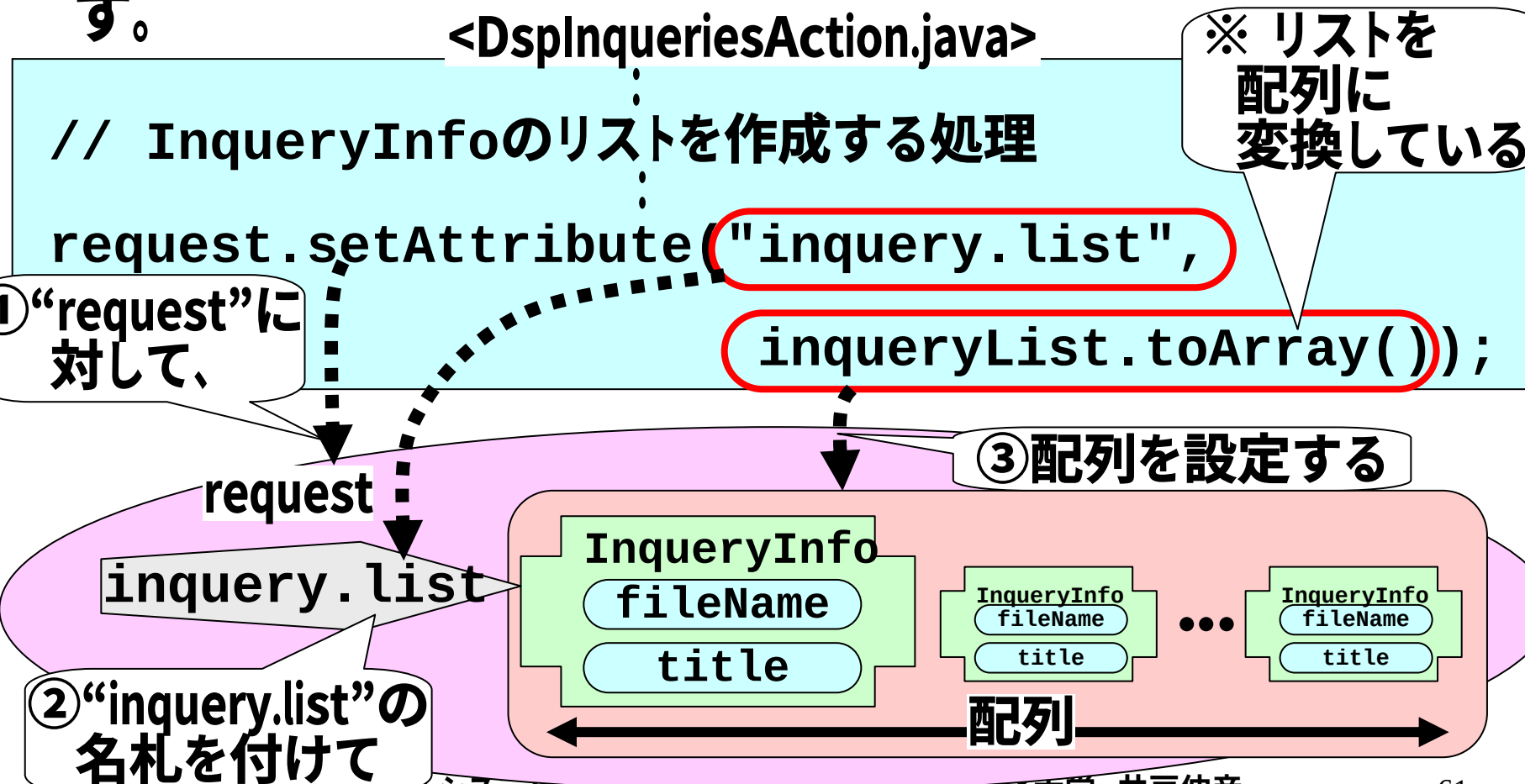
③Serializableインタフェース

①コンストラクタ

②アクセサ・メソッド

(9．4) アクションクラスでの処理

■アクションクラス“DspInqueriesAction”では、“inquery.list”という名札を付けて、“request”に配列を設定します。これは普通のJavaBeansの扱いと同じです。



(9．5) JSPでの繰り返しの概要

<selectInquery.jsp>

```
(略)
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
(略)
<table border="1" width="450">
<logic:iterate id="inqueryData"
    name="inquery.list"
    scope="request">
  <tr><td>
    <html:link page="/showInquery.do"
               paramId="fileName"
               paramName="inqueryData"
               paramProperty="fileName">
      <bean:write name="inqueryData"
                  property="title" />
    </html:link>
  </td></tr>
</logic:iterate>
</table>
(略)
```

logicタグを使用することの宣言

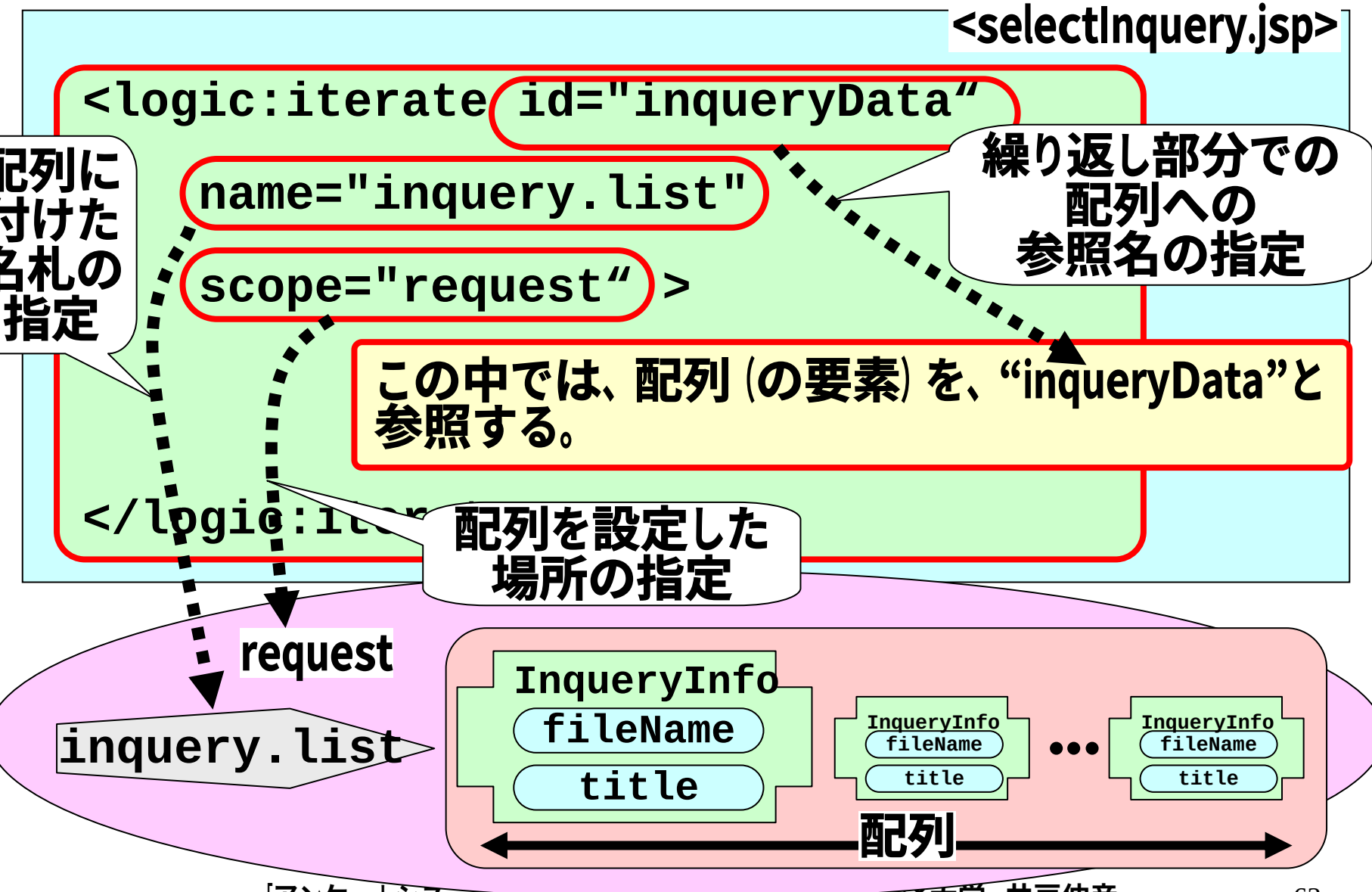
配列を読み出して、繰り返しを指定するタグ

配列の要素（JavaBeans）へアクセスするhtmlタグ

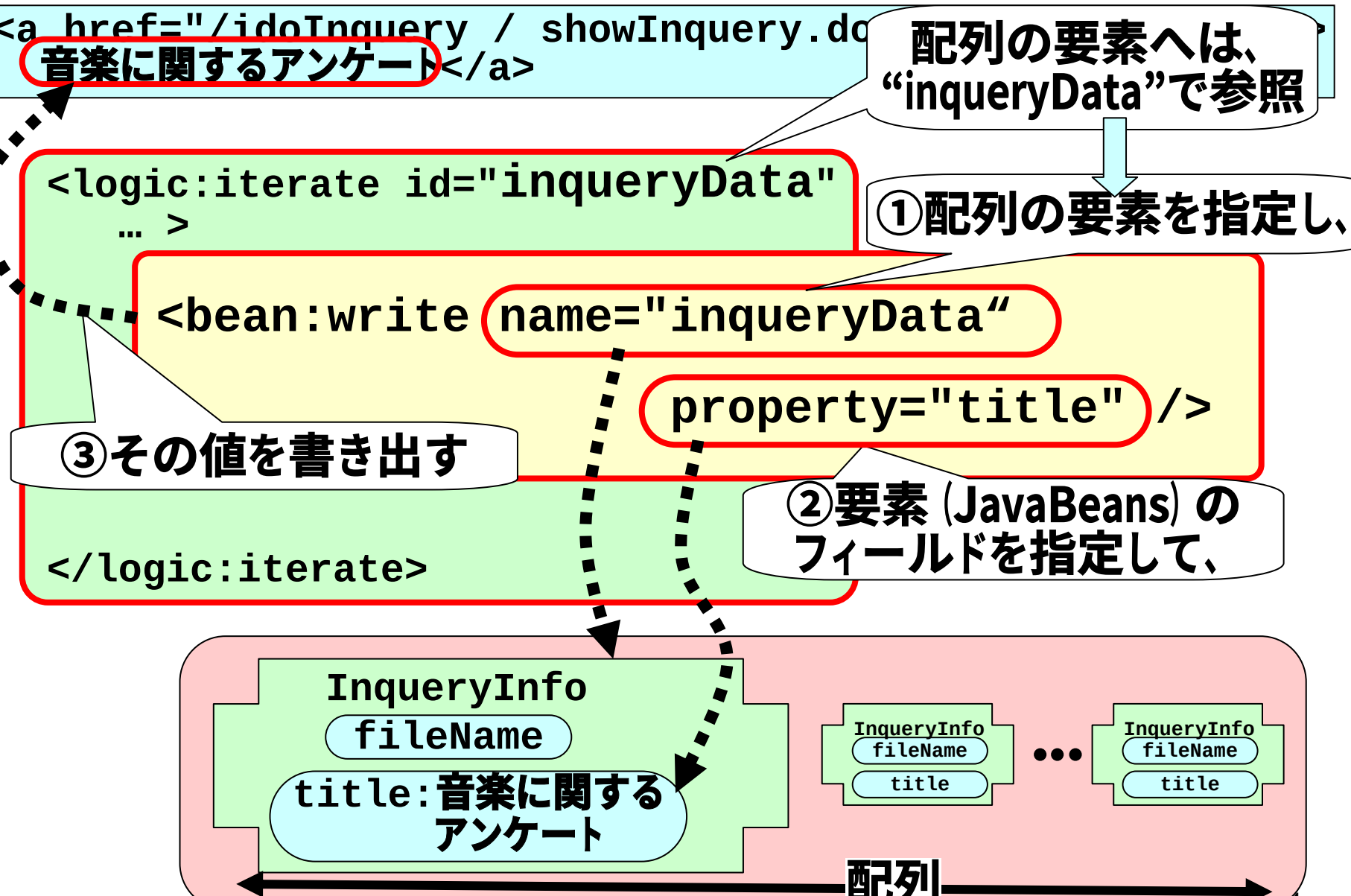
配列の要素（JavaBeans）へアクセスするbeanタグ（リンクのタグの要素）

繰り返し表示される部分

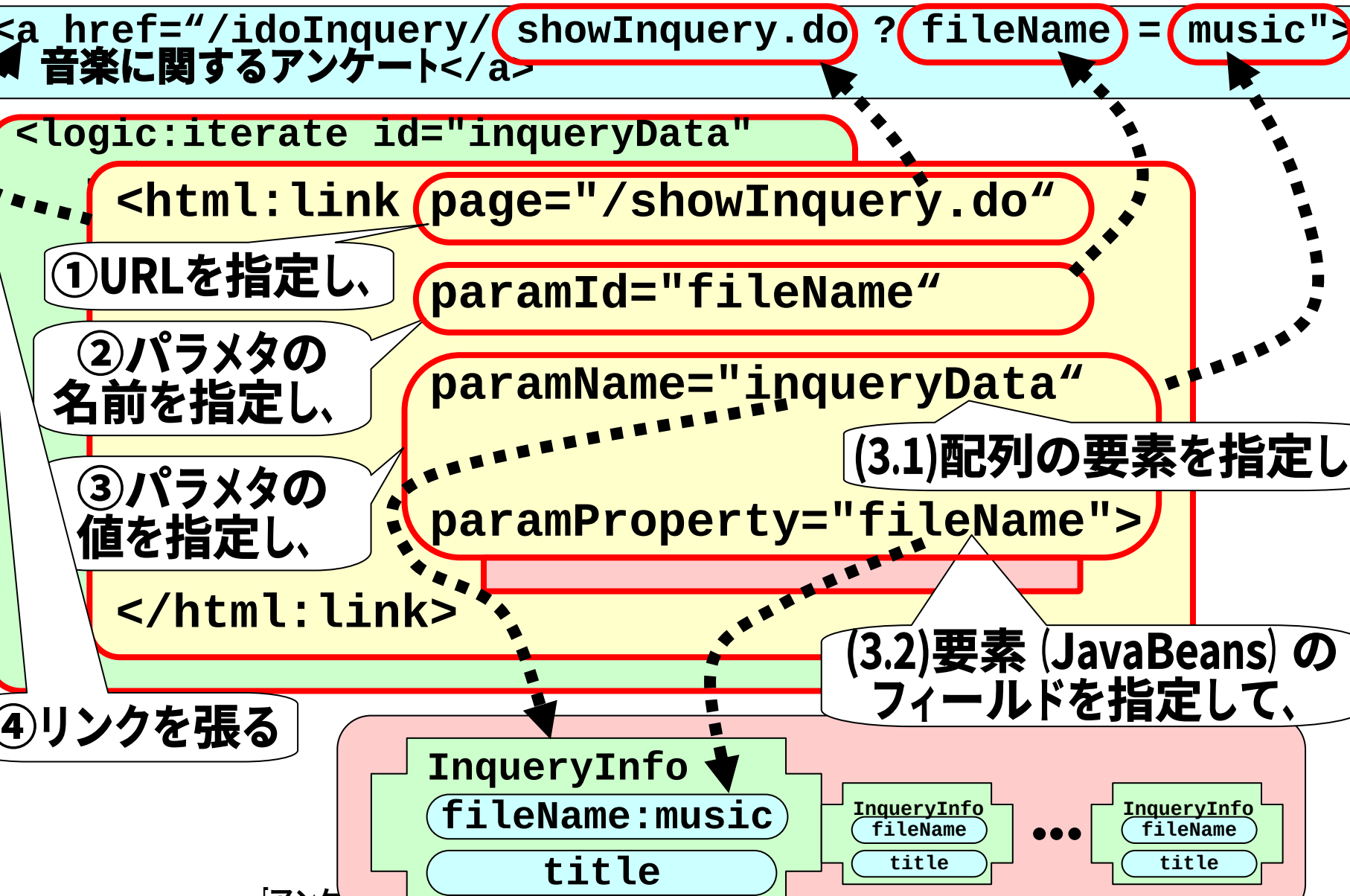
(9.6)<logic:iterate>タグ



(9．7) 繰り返し部分：<bean:write>タグ



(9．8) 繰り返し部分：<html:link>タグ

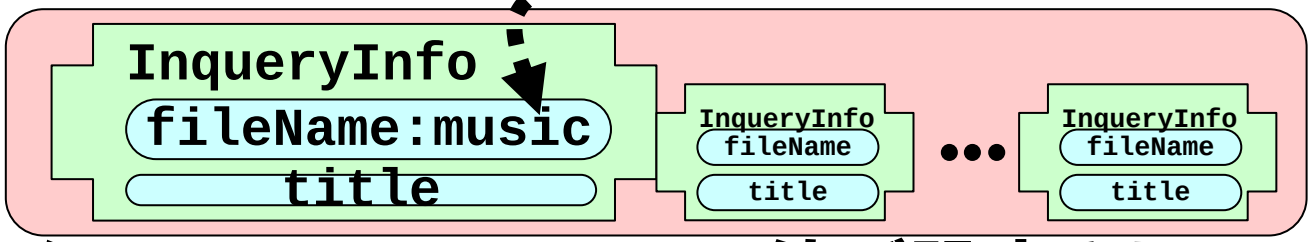


(9.9)"fileName"の名前

■前スライドにて、フォーム上のパラメタ名と、JavaBeans上のフィールド名とが、いずれも"fileName"となっており一致しています。しかしながら、この2つが同じ名前である必然はありません(別でもOKです)。

```
<a href="/idoInquery/ showInquery.do ? fileName = music">
  音楽に関するアンケート</a>
```

```
<html:link (中略)
           paramId="fileName"
(中略)
           paramProperty="fileName">
</html:link>
```

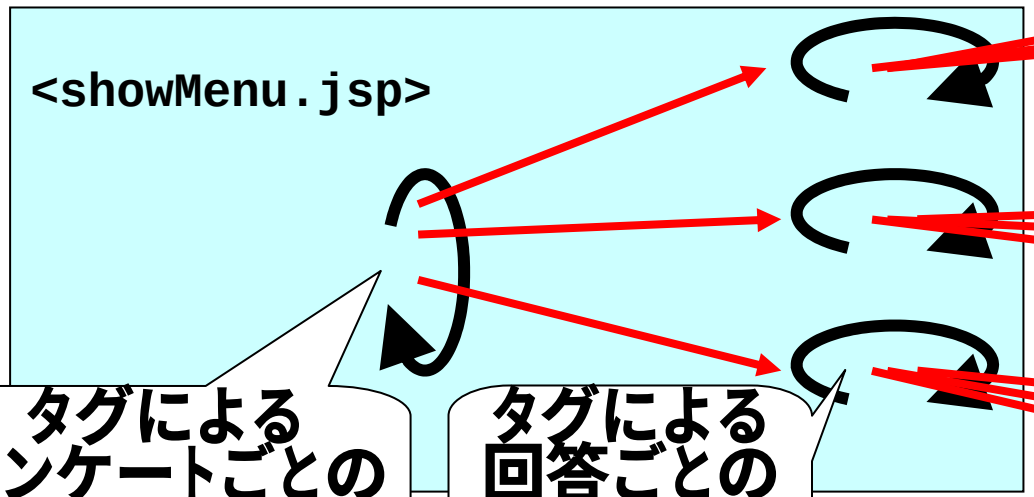


■このパラメタとフィールドとは、同じ値が設定されるので、ここでは同じ名前にしました。

(10) タグによる2重ループ

■ユーザがアンケート回答を選択する画面（アンケート回答選択画面）では、アンケートのタイトルが列挙され、それぞれのタイトルの中でアンケート回答が列挙されています。

■すなわち、この“列挙”は2重ループになっていますが、これもJSP内のタグにより実現されています。



タグによるアンケートごとの繰り返し

タグによる回答ごとの繰り返し

(10．1) 表示のソースコード

```
(略)
<h2>本に関するアンケート</h2>
<table border="1" cellspacing="10">
<tr><td>
<a href="/idoInquery/viewResponse.do?responseFile=book_a">book_a</a>
</td><td>
<a href="/idoInquery/viewResponse.do?responseFile=book_h">book_h</a>
</td><td>
<a href="/idoInquery/viewResponse.do?responseFile=book_ido">book_ido</a>
</td></tr></table>

<h2>生い立ちに関するアンケート</h2>
<table border="1" cellspacing="10">
<tr><td>
<a href="/idoInquery/viewResponse.do?responseFile=born_a">born_a</a>
</td><td>
(略)
</td></tr></table>

<h2>趣味に関するアンケート</h2>
<table border="1" cellspacing="10">
(略)
</table>
```

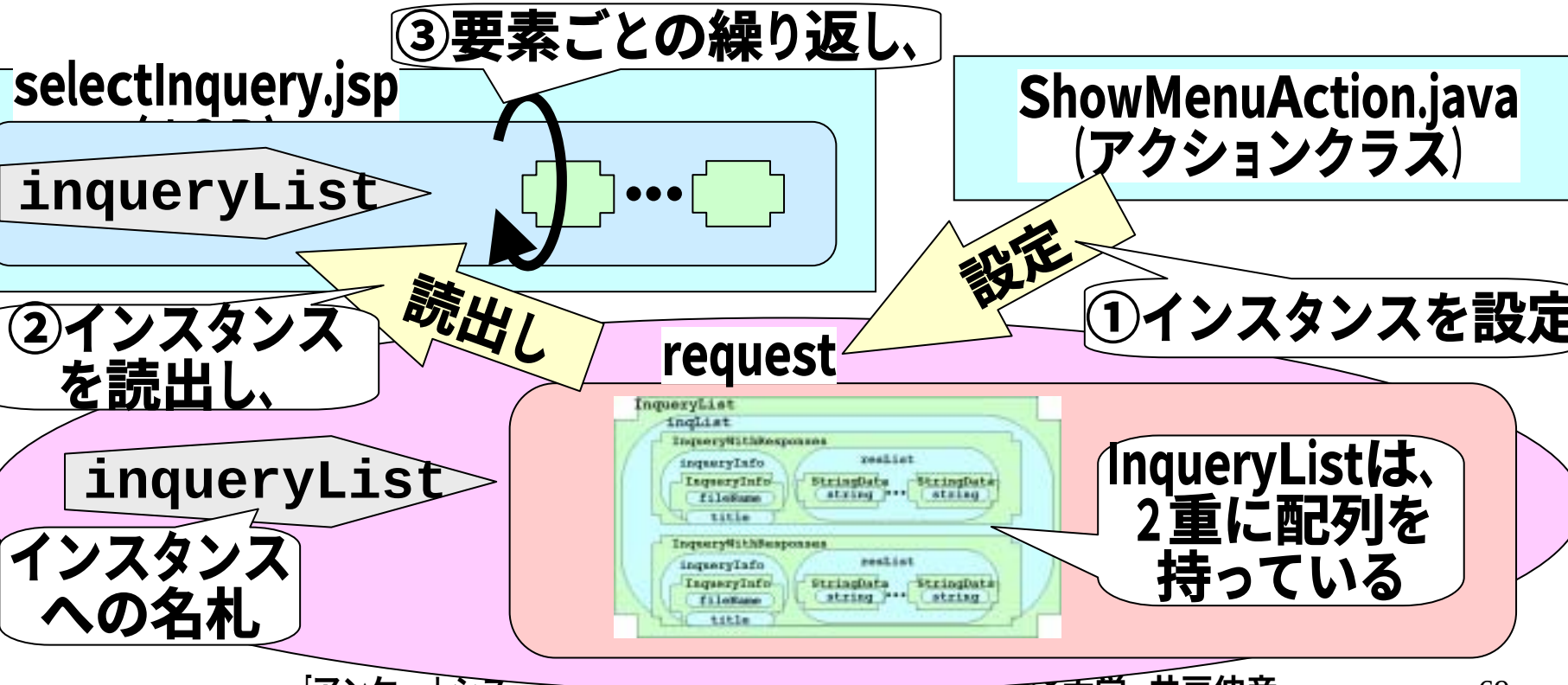
タグによる
回答ごとの
繰り返し

タグによる
アンケートごとの
繰り返し

(10．2) アクションクラス、JSP、JavaBeans

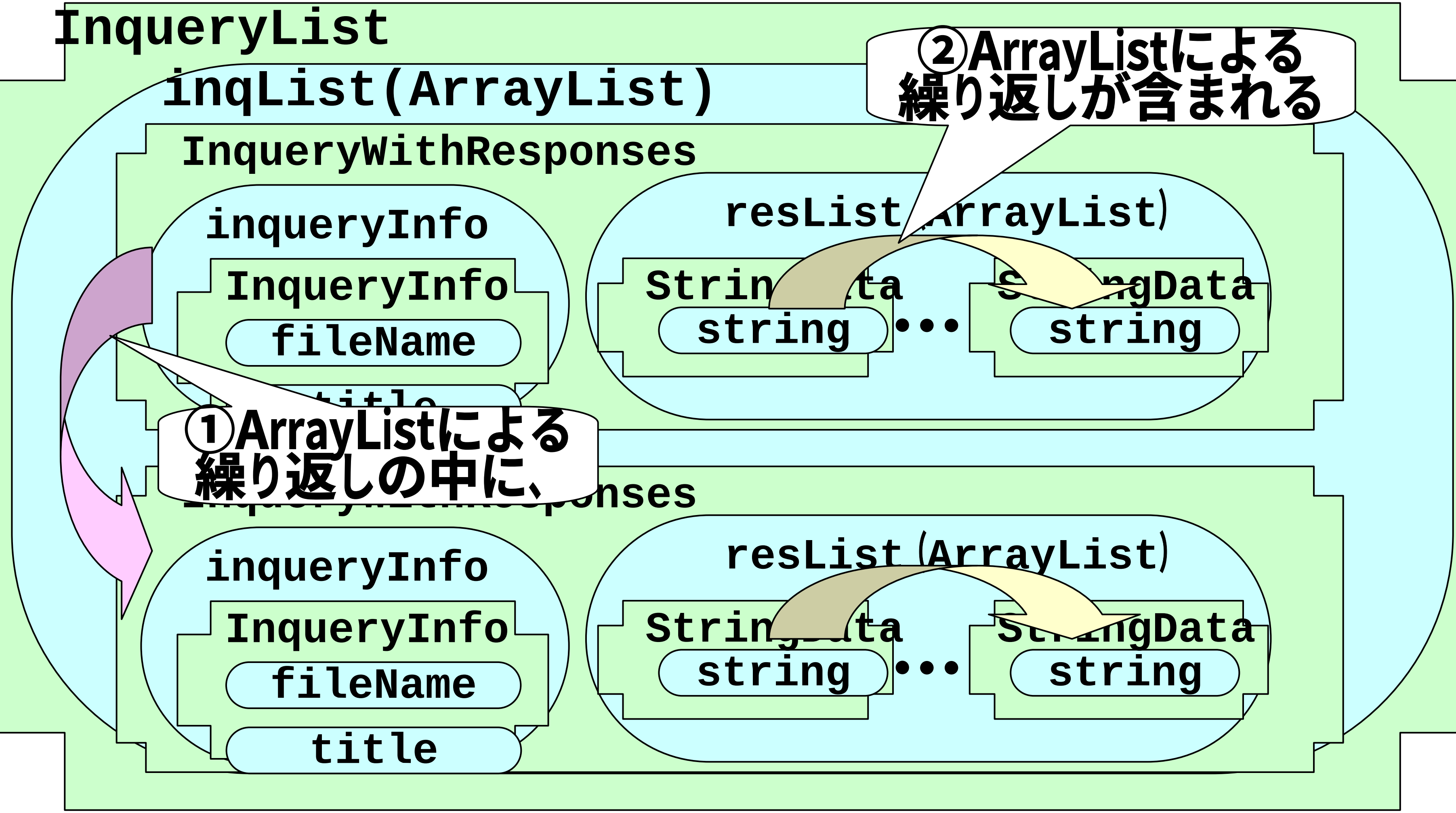
■アクションクラス (DspInqueriesAction.java) では、“inqueryList”という名札をつけて、request”上にInqueryListのインスタンスを設定します。

■JSP (selectInquery.jsp) では、“inqueryList”というインデックスにより読出したInqueryListの内部にある2重の繰り返しを、順次読み出していきます。



（１０．３）InqueryListの２重の繰り返し

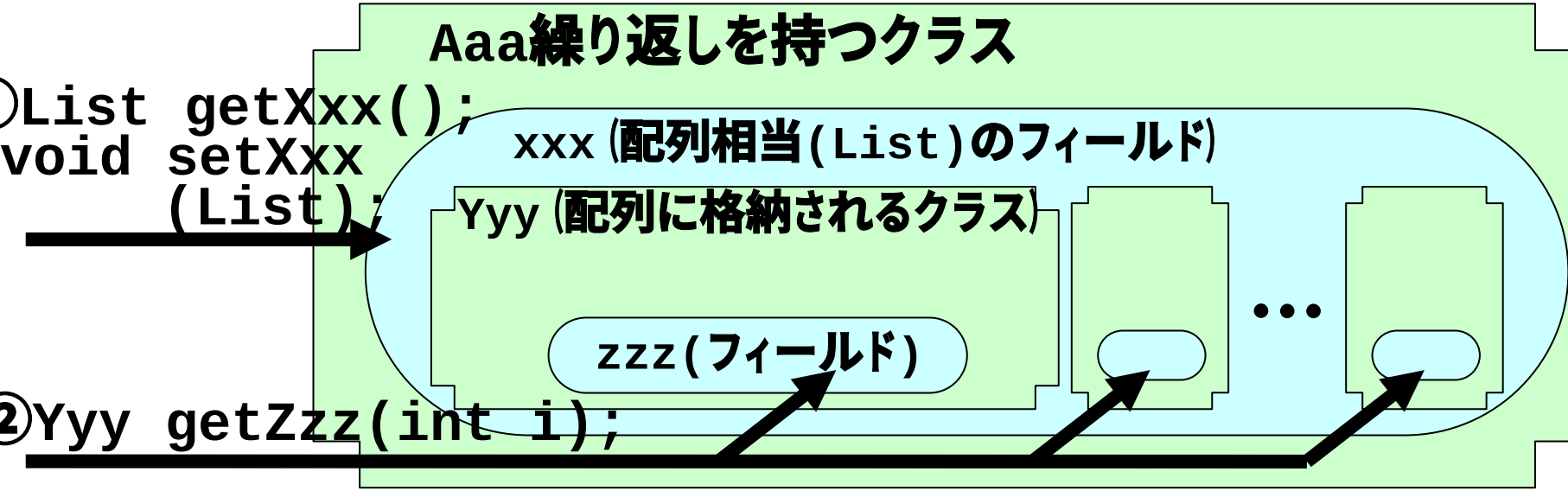
■InqueryList内には、ArrayListが入れ子に宣言されており、２重の繰り返しになっています。



(10．4) 繰り返しを持つクラスの作り方

■繰り返しに用いるクラス、InqueryList(外側)とInqueryWithResponses(内側)とは、ともに次のようなメソッドを持っています。

- ①繰り返しのListのフィールド変数へのアクセサメソッド
- ②繰り返される要素に対する、引数を伴った読出しのメソッド



Aaa	xxx	Yyy	zzz
InqueryList	inqList	InqueyWithResponses	resList
InqueyWithResponses	resList	StringData	string

(10．4．1) 実際のメソッド

■繰り返しを持つクラスは、前スライドに記したようなメソッドをすることにより実現されます。

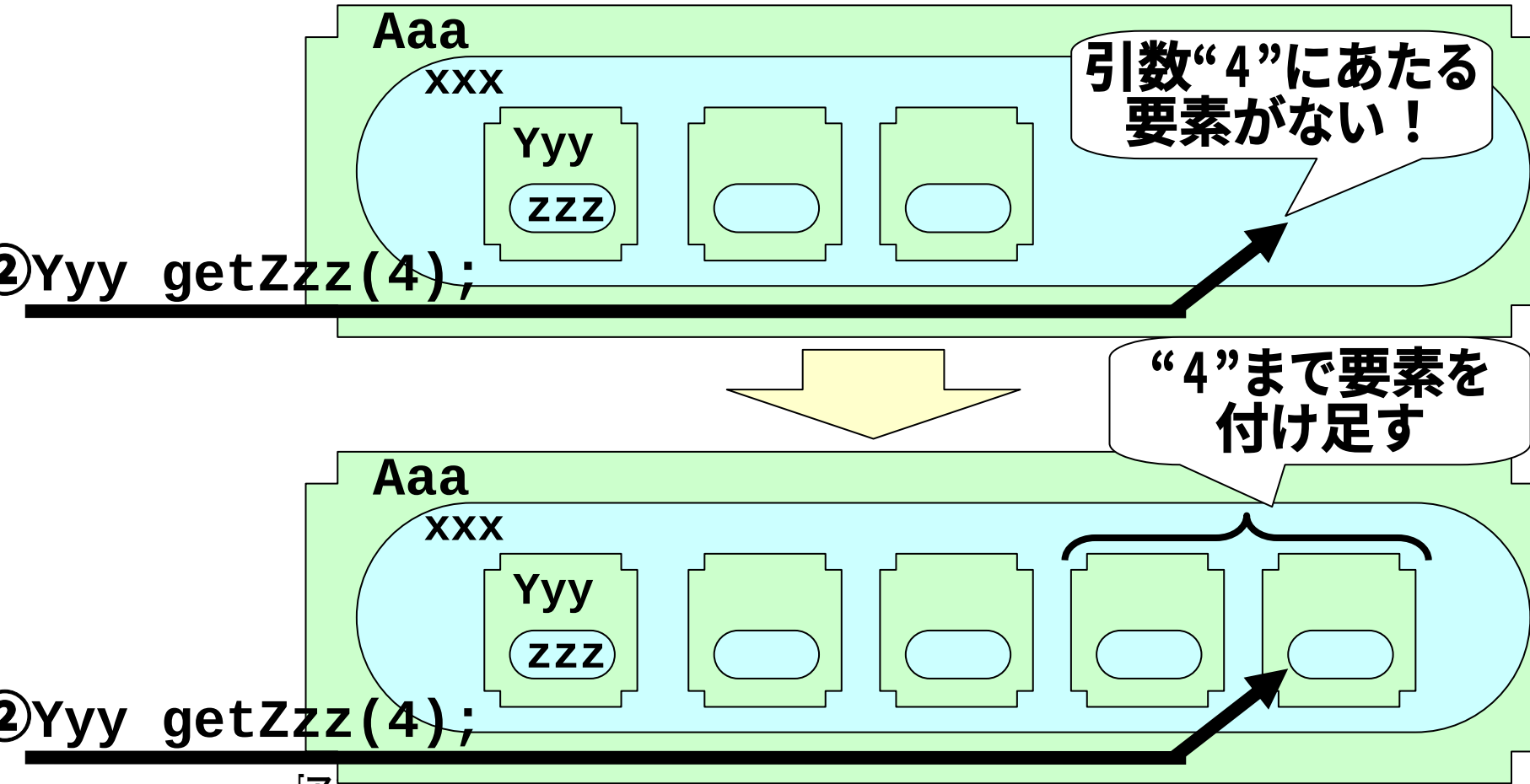
```
public class InqueryList {
  protected List inqList = new ArrayList();
  public void setInqList(List inqList){
    this.inqList = inqList;
  }
  public List getInqList(){
    return this.inqList;
  }
  public InqueryWithResponses getResList
                                  (int index){
    // (略)
    return resList;
  }
}
```

①Listのフィールド変数へのアクセサメソッド

②Listの要素を読み出すメソッド

(10．4．2) 読出しのメソッドでの処理

■ArrayList等で実装された配列への、引数による読出しにおいては、配列長よりも大きな値の引数で呼ばれた際、その長さまで配列を伸ばしています。



(10．5) アクションクラスでの処理

■アクションクラス“DspMenuAction”では、“inqueryList”という名札を付けて、“request”にInqueryListのインスタンスを設定します。これは普通のJavaBeansの扱いと同じです。

<DspMenuAction.java>

```
InqueryList inqueryList
= new InqueryList();

request.setAttribute("inqueryList",
                     inqueryList);
```

①“request”に対して、

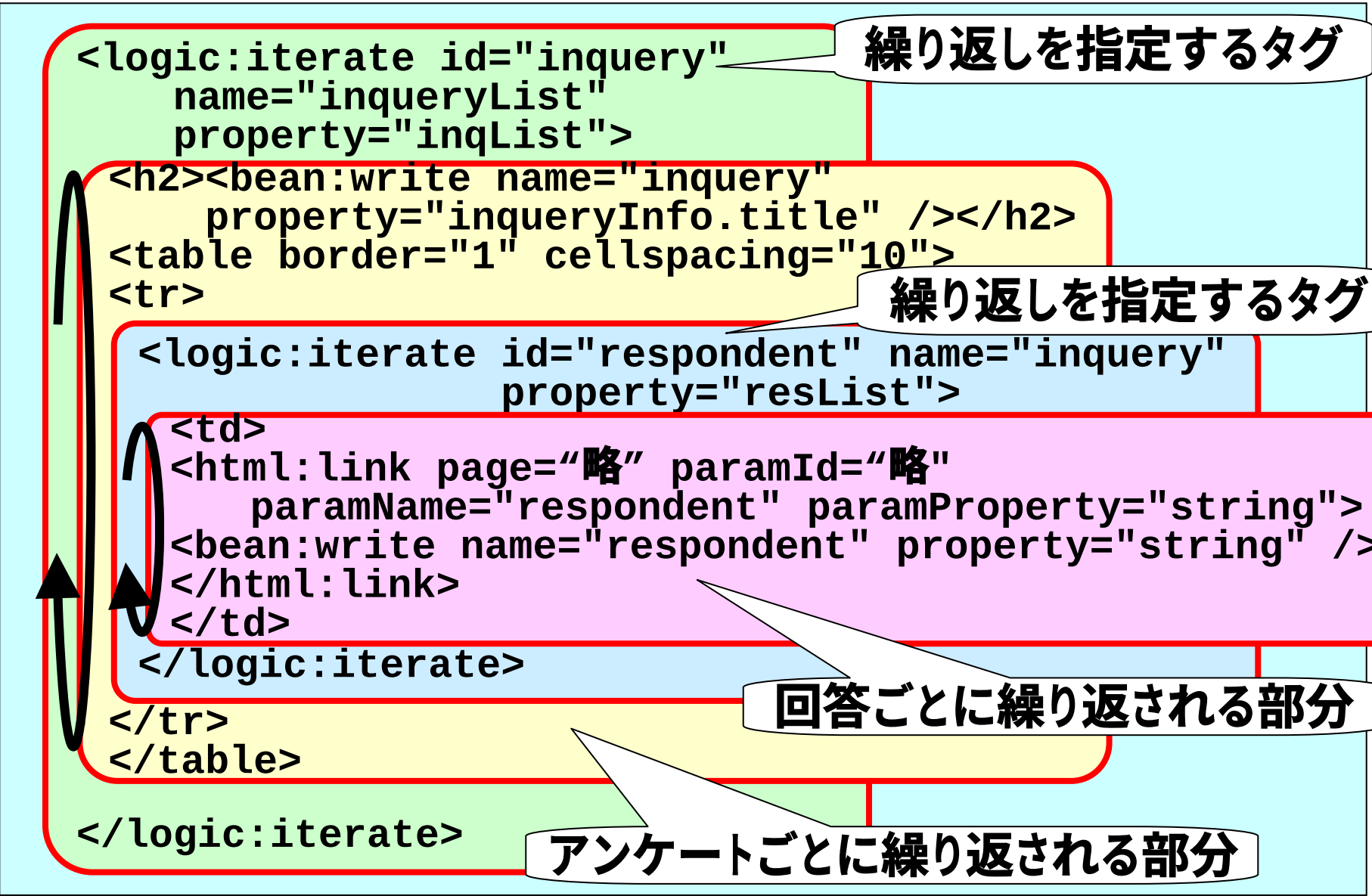
request

inqueryList

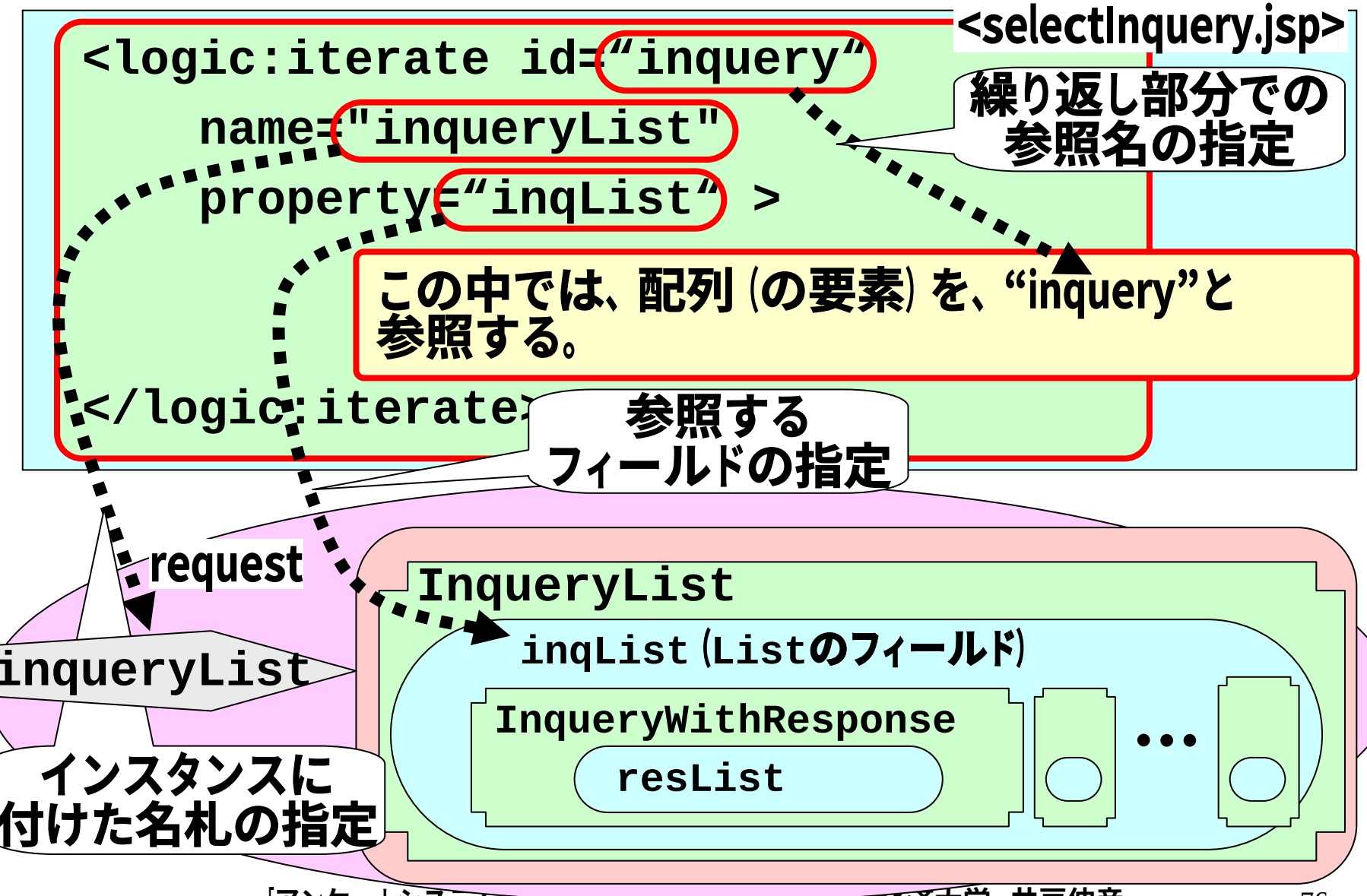
②“inqueryList”の名札を付けて

InqueryListを設定する

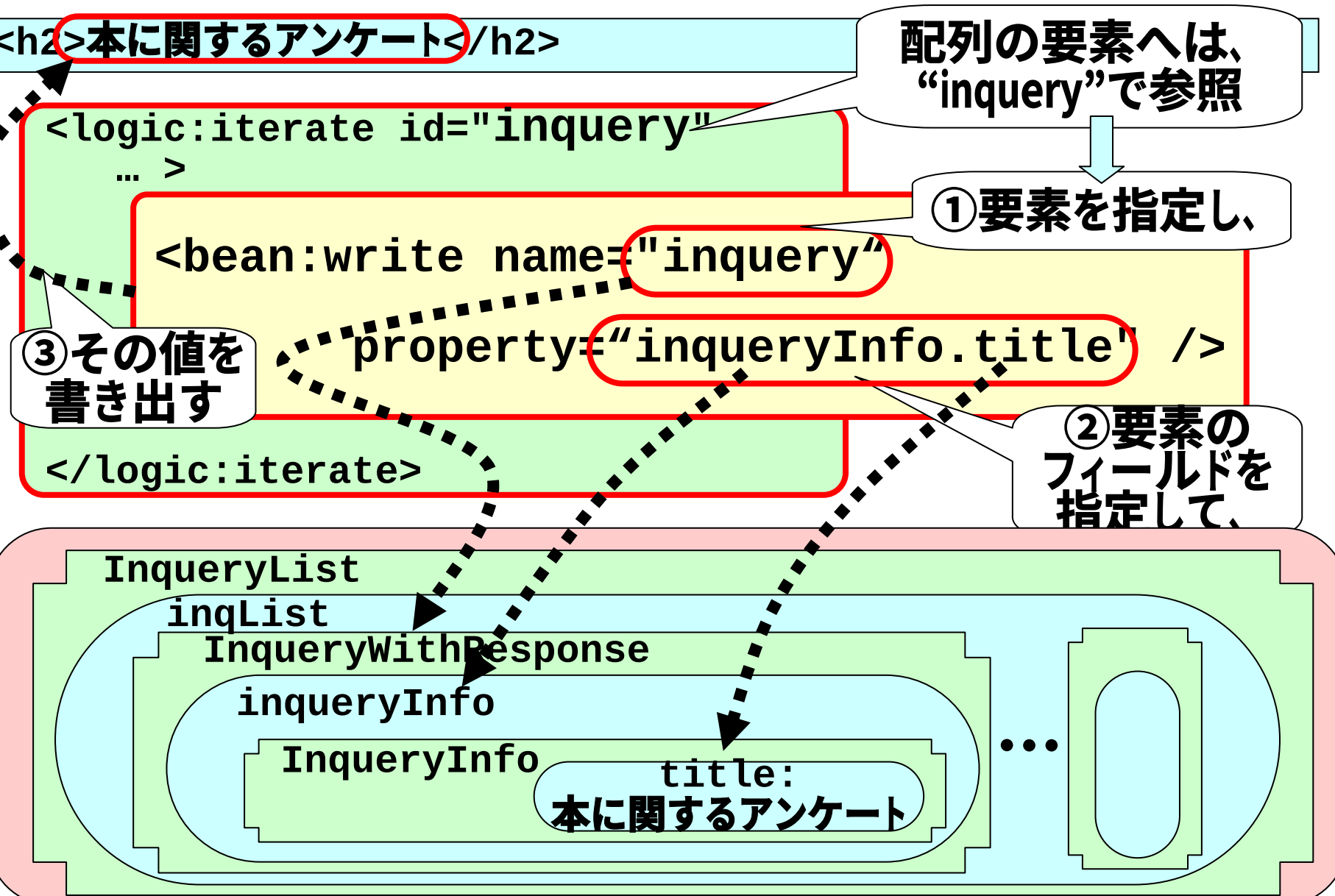
(10.6) JSPでの繰り返しの概要



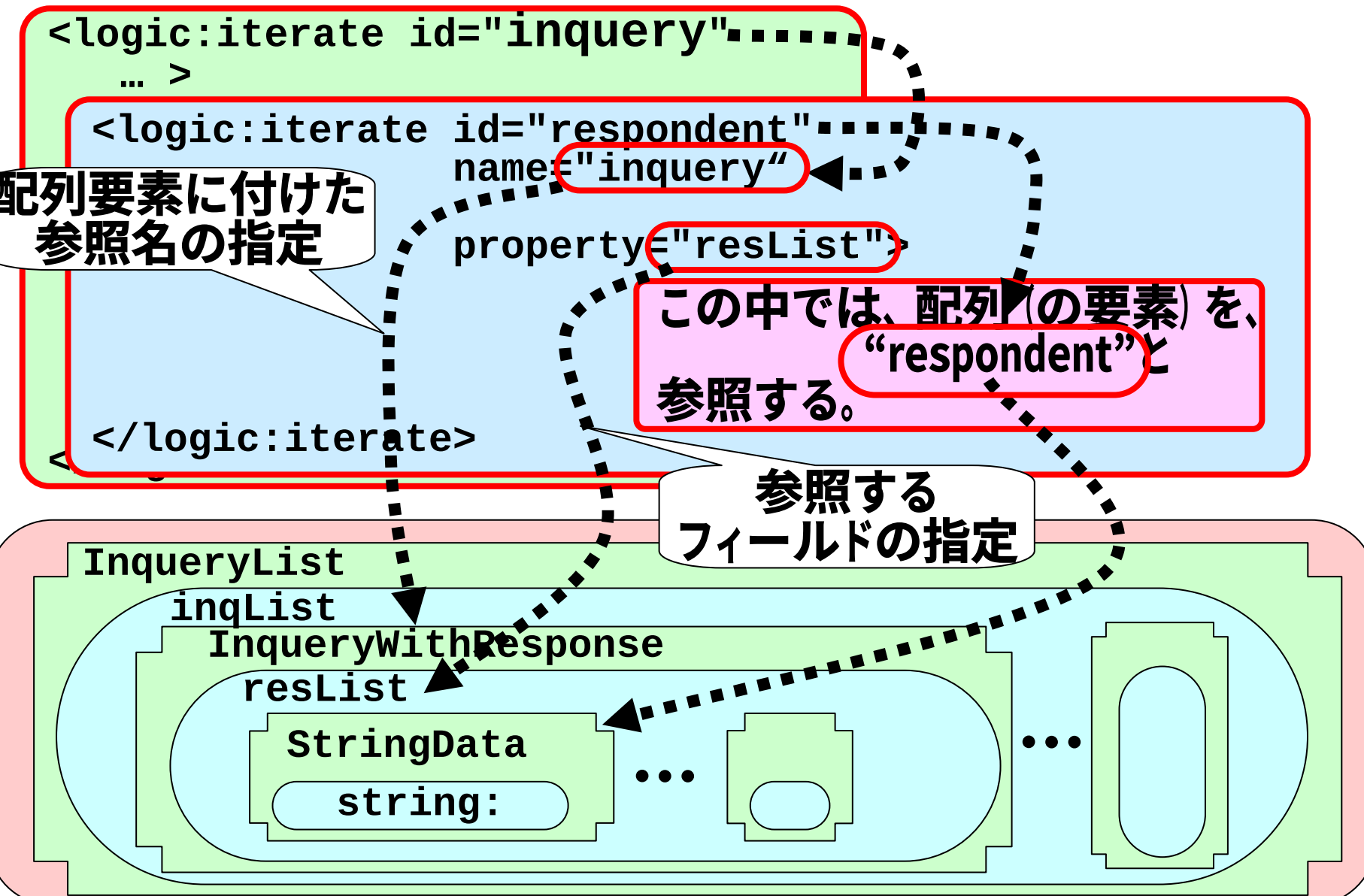
(10.6.1) ひとつめの<logic:iterate>タグ



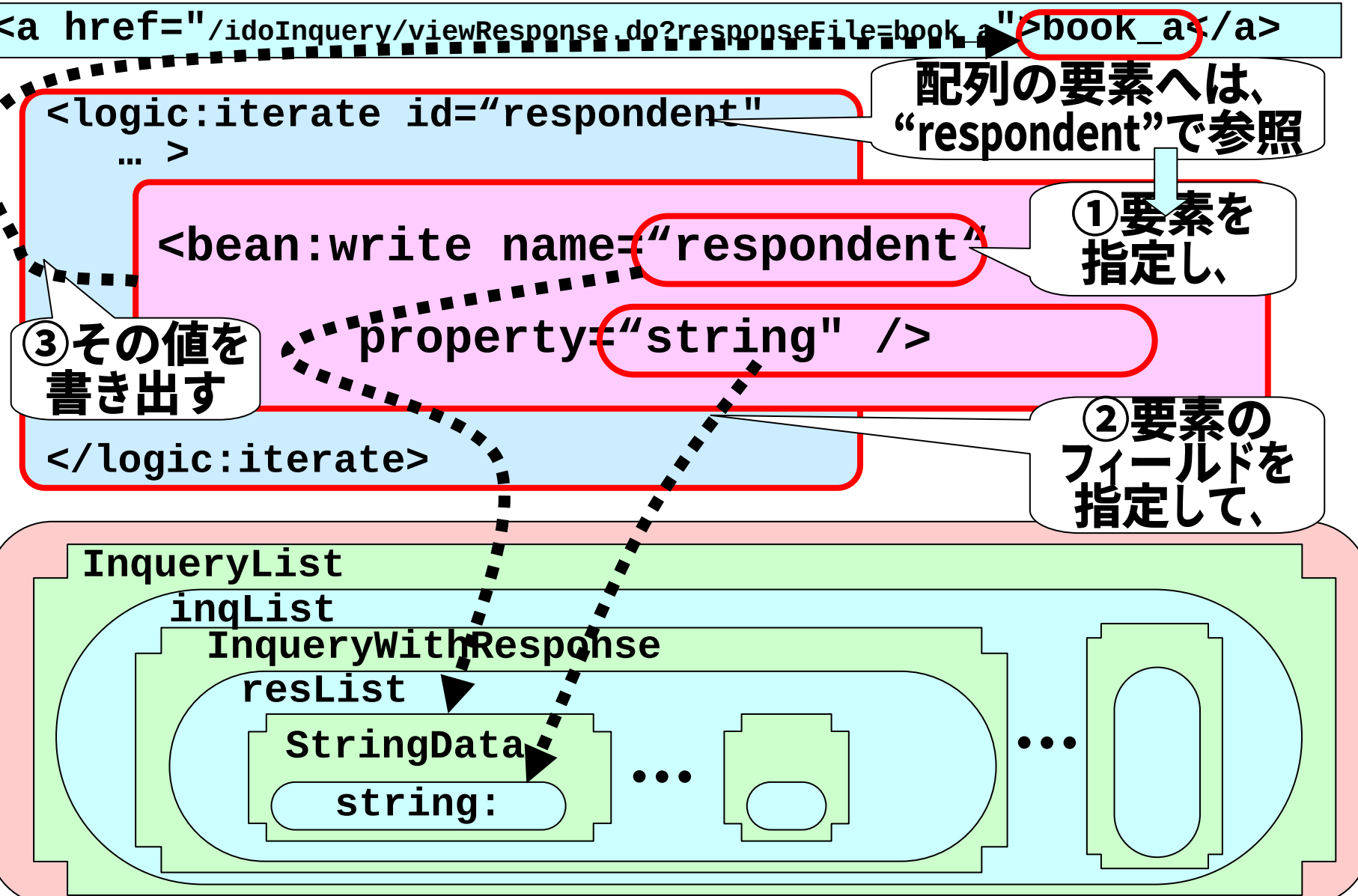
(10.6.2) ひとつめの繰り返し部分



⑽.6.3 ふたつめの<logic:iterate>タグ

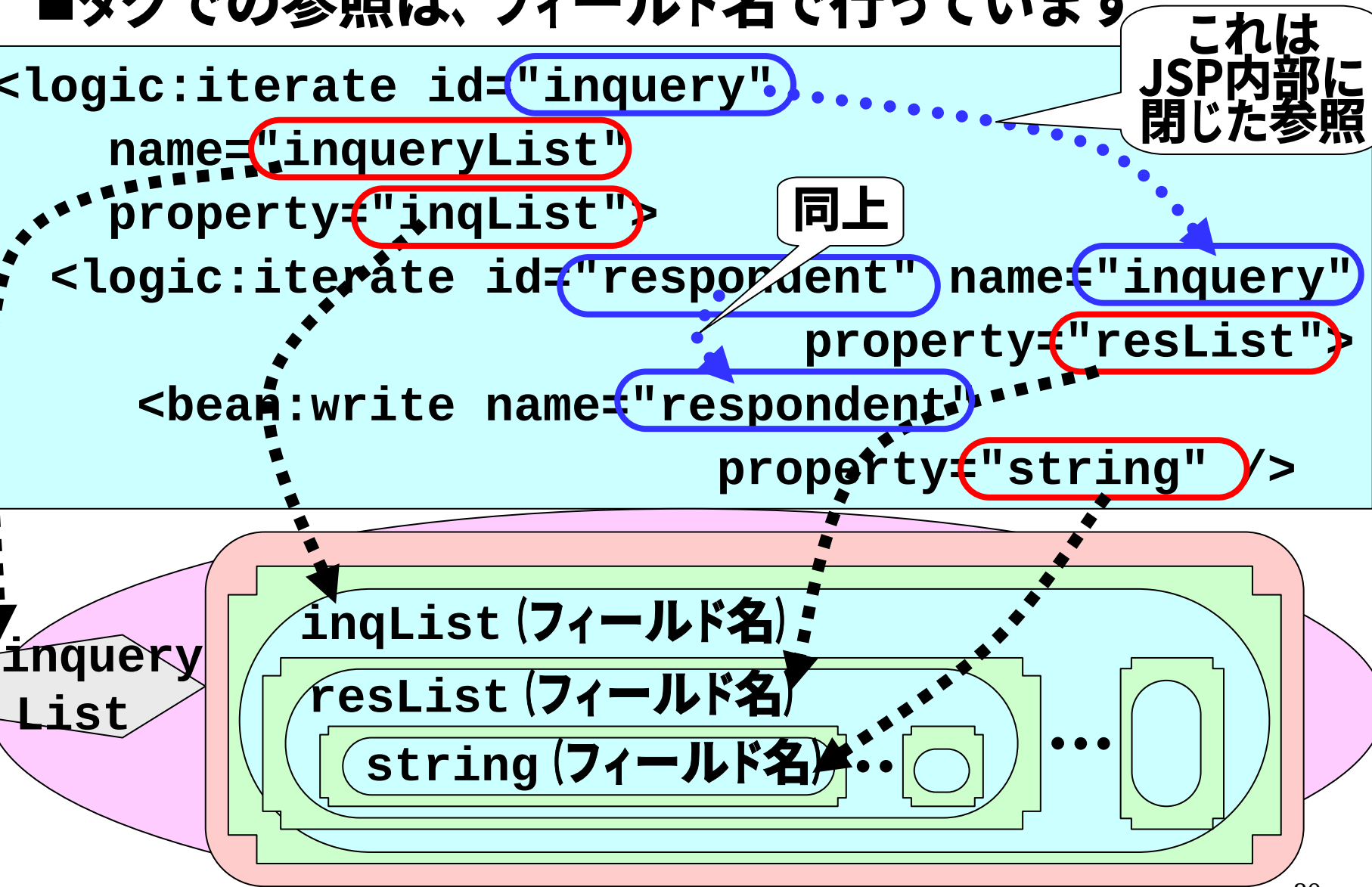


(10．6．4) ふたつめの繰り返し部分



(10.6.5) 概略整理

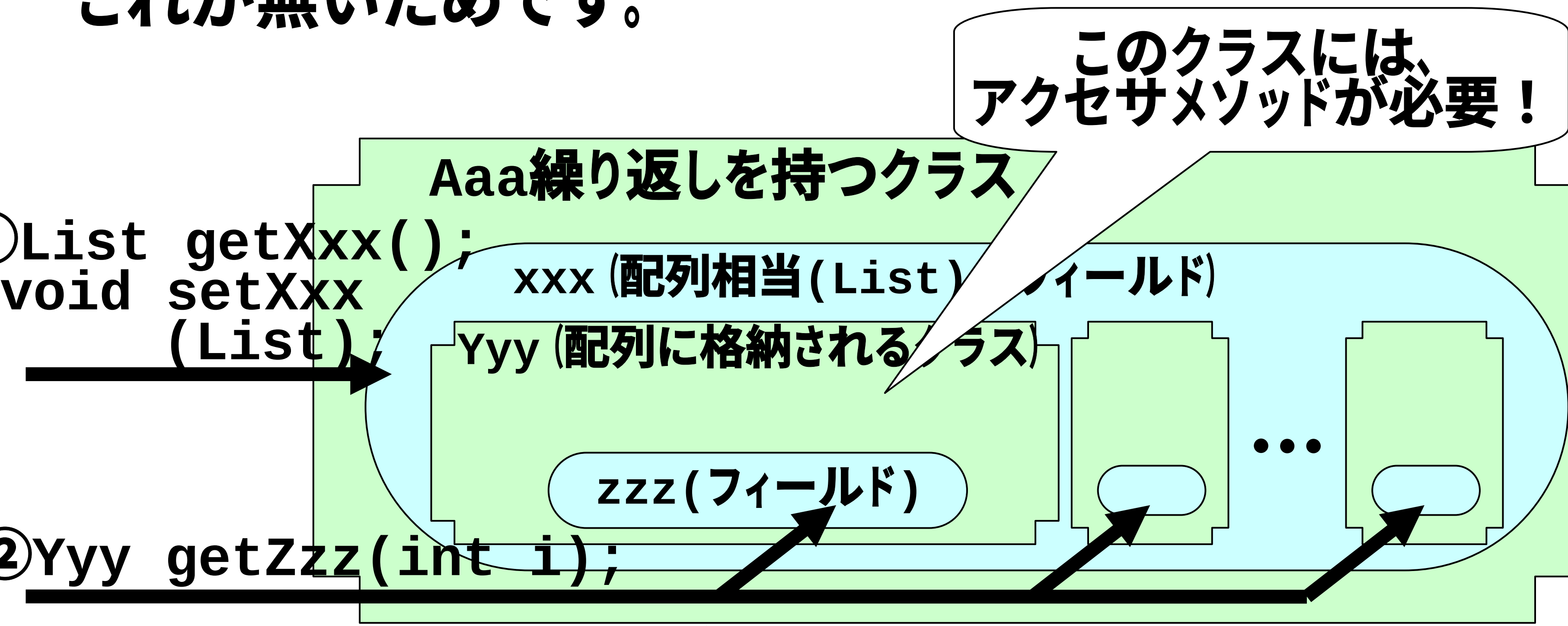
■タグでの参照は、フィールド名で行っています



（10．6．6）StringDataクラス

■StringDataクラスは、文字列を定義しているだけであり、あまり必要がないように見えるかも知れません。

■しかしながら、これをStringクラスで代用することは出来ません。タグでアクセスする配列の要素には、アクセサ・メソッドがある必要がありますが、Stringクラスにはこれが無いためです。



(11) ログ