・このプリントは科目「Web アプリケーション構築 1」の自宅学習用補助プリントです。
プリントの指示に従って、学習を進めてください。不明点については担任に電話確認すること。

★科目「Web アプリケーション構築 2」では、本テキストの 8 章を参考にグループでシステム開発を行い★
★ます。科目「オブジェクト指向プログラミング」の復習も兼ねているのでしっかり取り組んで下さい。★

【学習期間】　4 月 2 4（月）～5 月 6（金）
【内容】テキスト【第 4 章】p107～152　・セッション、フォワード、リダイレクト、スコープ

【学習の流れ】

第 4 章

4-2. セッション(Session)(P.107)

・セッションとは、Web サーバがクライアントとの通信接続状態を管理する情報のこと。
複数のクライアントを識別するために Web サーバはクライアントごとにセッション ID を発行します。

・セッションの発行・送受信については、Web サーバが自動で行います。私たちはセッションを利用するために用意されたクラスやメソッドの使い方を覚えましょう。(P.109)

セッション管理の開始　　：getSession メソッド
リクエストからのセッションの取得　：getSession メソッド
セッション ID の取得　　：getId メソッド

・セッションとクッキーの違い

セッション：サーバがクライアントを識別するための仕組み（サーバに保存）
クッキー　：サーバがクライアントにデータを保存させるための仕組み（クライアントに保存）

・セッションの有効時間

有効時間は、Web サーバ(本テキストでは Tomcat というソフトウェアを使用)で設定する。
「web.xml」ファイルの[session-timeout]の記述を変更することができる。

```
<session-config>
  <session-timeout>30</session-timeout>
</session-config>
```

「30」の単位は「分」です。初期設定では、30 分間クライアントからサーバへの通信がない場合は、サーバは保存していたクライアントの情報を削除します。

・P.110～112 のセッションに関する例題に取り組んでください。

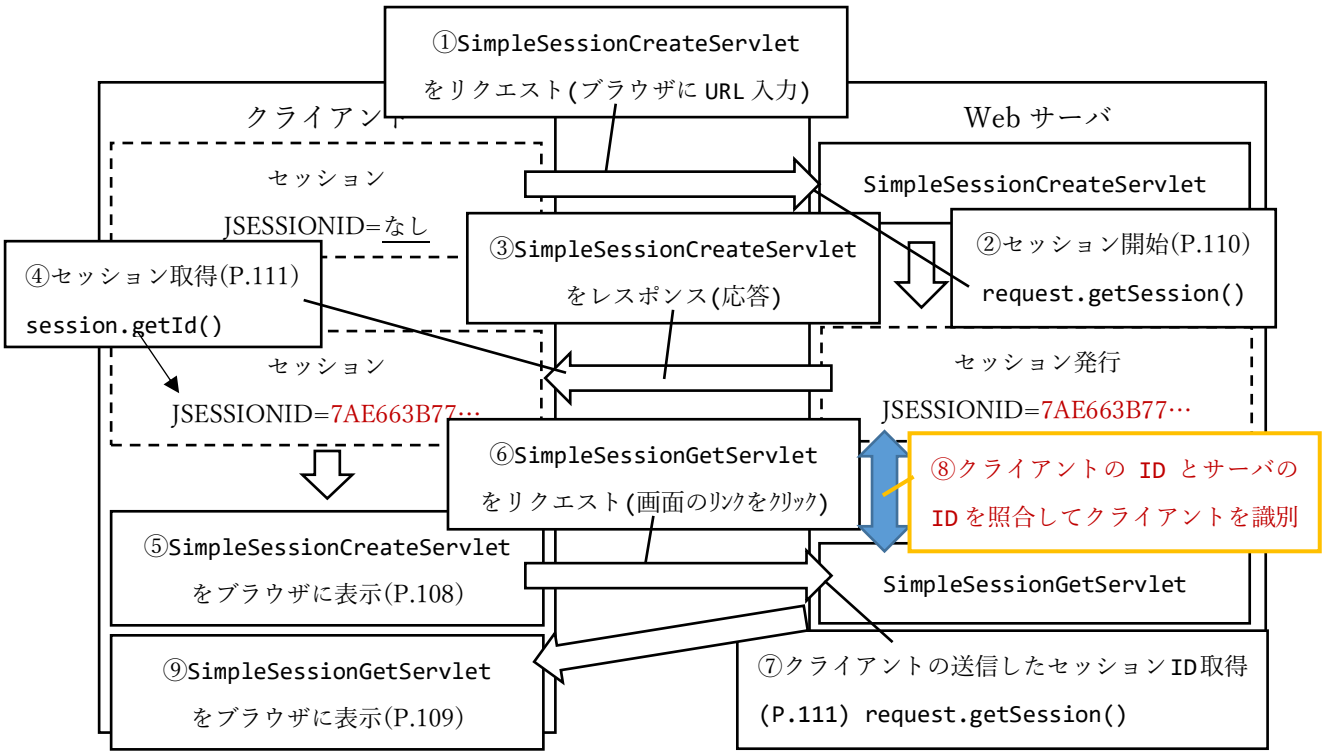
P.108 の画面 図 4.7　→ P.110 SimpleSessionCreateServlet.java
P.109 の画面 図 4.10　→ P.111 SimpleSessionGetServlet.java

・P.110 SimpleSessionCreateServlet.java

```
25 行目　HttpSeesion　session　＝　request.getSeesion();
```

リクエストからのセッションの取得にも getSession メソッドを使用します（P.109）。

・P.110～112 の例題の実行結果で同じセッション ID が表示されたことからセッションを共有出来きていることがわかります。




・P.113～118 セッションによるデータの受け渡し

P.115 の画面 図 4.14　→ P.121 startsessionsample.html
P.116 の画面 図 4.17　→ P.122 SessionCrateServlet.java
P.118 の画面 図 4.20　→　P.123 SessionGetServlet.java
P.119 の画面 図 4.22　→　P.125 SessionRemoveServlet.java

・P.119～120 セッション操作(HttpSession クラス)のメソッド

セッションにデータ(※)を保存　　：setAttribute メソッド
セッションからデータ(※)を取得　　：getAttribute メソッド
セッションの有効期限設定　　：setMaxInactiveInterval メソッド
セッションの無効化　　：invalidate メソッド
※「名前」と「値」の組み合わせ

・P.121～126 の例題では、セッションを使ってデータの受け渡しを行っています。
セッションもクッキーと同様で「名前」と「値」の組み合わせでデータを格納することができます。
データの格納には setAttribute メソッドを使用しています。

P.122　SessionCreateServlet

```
～省略～
String memberName = request.getParameter("membername");
String memberDate = request.getParameter("memberdate");

HttpSession session = request.getSession();
session.setAttribute("MemberName", memberName);
session.setAttribute("MemberDate", memberDate);
～省略～
```

「名前」と「値」の組み合わせ

セッションに格納した値を取り出すときは、getAttribute メソッドを使用します。

P.124 SessionGetServlet

```
～省略～
String memberName ="";
String memberDate = "";

HttpSession session = request.getSession();
session.getAttribute("MemberName");
session.getAttribute("MemberDate");
～省略～
```

取り出す値の名前を引数に指定する

セッションを無効化する場合は、invalidate()メソッドを使用します。

P.125 SessionRemoveServlet

```
～省略～

HttpSession session = request.getSession();
session.invalidate();
～省略～
```

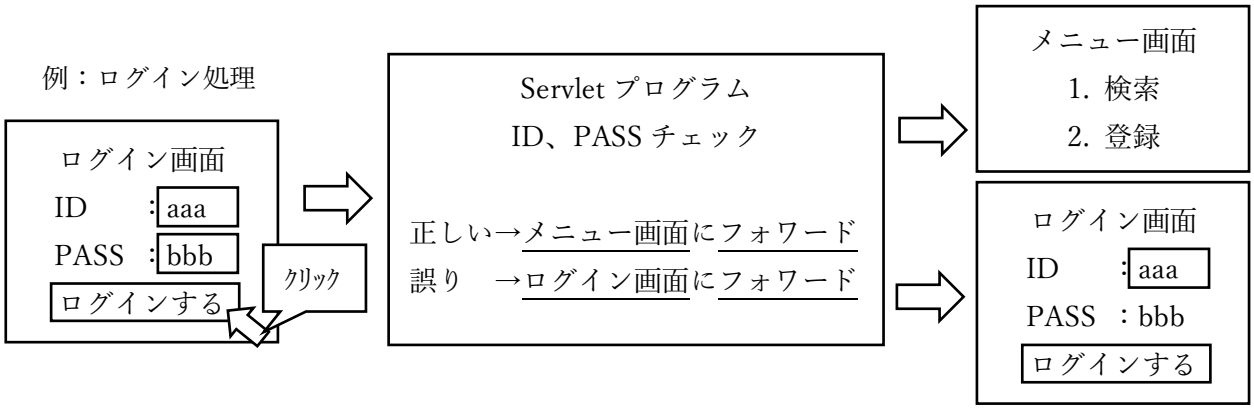
セッションを無効化する

## 4-3. フォワード(forward)(P.127)

・「転送」処理のことです。

クライアント側で、画面移動のボタン押下や、リンクのクリックをさせることなく、サーバ側で画面を移動させることができます。

ただし、同一サーバの異なる Web アプリケーションや、別の Web サーバには転送できません。




・P.128 フォワード処理では、転送先の指定を行います。

その際、getRequestDispatcher メソッドを使用します。引数にフォワード先の相対パスを指定します。その後、forward メソッドを使用して、リクエストの転送を行います。

・P.129～130 フォワードによる画面遷移

P.129 の画面 図 4.25　→ P.131 startreqdispsample.html

P.130（画面なし）　→ P.133 SimpleDispatchServlet1.java

P.130 の画面 図 4.27　→ P.133 SimpleDispatchServlet2.java

P.132.SimpleDispatchServlet.java

```
～省略～
request.setAttribute("SampleAttribute1","Hoge");
RequestDispatcher dispatch
        = request.getRequestDispatcher("SimpleDispatchServlet2");
dispatch.forward(request, response);
～省略～
```

転送をするための準備
転送先の指定をする。

転送処理

P.133.SimpleDispatchServlet2.java

```
～省略～
@WebServlet("/SimpleDispatchServlet2")
～省略～
```

## 4-4.リダイレクト(P.135)

・「転送」処理のことです。
　フォワードとの違いは、「URL」を指定することで、別の Web アプリケーションや Web サーバへの転送ができるところです。

・指定した URL にリダイレクトするには、sendRedirect メソッドを使用します。

```
response.sendRedirect("https://www.kcsk.ac.jp/");
```

・P.136～138 リダイレクトによる画面遷移
　P.136 の画面 図 4.30　→ P.131 redirect.html
　P.136 の画面 図 4.31　→ P.133 Redirect.java で転送した先の Web ページ

## 4-5.スコープと JSP 暗黙オブジェクト(P.142)

・セッションなどのオブジェクトには、有効である範囲（スコープ）が定義されています。
スコープの範囲外に移動すると、保管しておいたデータも消滅します。
セッション情報が多くなると、Web サーバの処理効率（データの保存容量、検索時間など）が悪くなりますし、意図せずに Web ページからセッション情報が参照できてしまうなど、セキュリティ的にもよくないです。P.142「図 4.35 スコープの概念」をよく読んでおいてください。

・P.144～152 リダイレクトによる画面遷移
　P.146 の画面 図 4.37　→ P.148 SetAttributeServlet.java
　P.146 の画面 図 4.38　→ P.150 GetAttributeServlet.java

・P.148 からの例題では、セッションスコープとアプリケーションスコープの比較が行われています。
SetAttributeServlet.java から GetAttributeServlet.java を呼び出す処理を実行した後に、ブラウザを閉じて、再度 GetAttributeServlet にアクセスしてみてください。セッションに格納した something、this、that はセッションが切れたため、値を保持していませんが、any はアプリケーションスコープのため、値を保持し続けています。

P.149 SetAttributeServlet.java

```
～省略～
HttpSession session  = request.getSession();
ServletContext application = getServletContext();
session.setAttribute("something", "贈り物");
session.setAttribute("this", "届け物");
session.setAttribute("that", "袖の下");
application.setAttribute("any", "どれか");
～省略～
```

スコープが異なるため、保持していた値が消滅するタイミングが異なる