第3章　セッション（session）の利用

3.1　セッションとは

Web サーバと Web ブラウザの間の通信 (HTTP による通信) は本来一回ページを表示するごとに完結するものである。つまり、ユーザーが過去にどのようなページを見ていたか、などといった履歴には関係なく動作する。

そこで、過去の履歴（例えば会員制ページのユーザのログイン情報や、ショッピングサイトの商品の選択（いわゆるショッピングカート）の情報）に依存するような Web アプリケーションを実現するためには、なんらかの形で過去のユーザーのアクセスの情報を保持する必要がある。このようなデータは、Web サイトにアクセスするユーザごとに管理しなければならないので、単純にインスタンス変数やファイルなどに保存することはできない。このためサーバ側にユーザごとにデータを保持し、ブラウザからのページ要求のたびに、何らかの方法でユーザを識別するためのデータを送信してもらう方法を取る。

このユーザを識別するデータを送信する方法としては、クッキー（cookie）という技術を使う方法、フォームの隠し要素（hidden）を使う方法, URL 中の Query String に履歴データを埋め込む方法などがある。

問 3.1.1 クッキー（*cookie*）とは、どういう仕組みか、調べよ。

Java Servlet ではユーザごとのデータを扱うために**セッション（session）**[1]という仕組みを提供している。セッションは裏側ではクッキーなどの仕組みを使っているが、複雑な部分をプログラマが見なくても済むようにして、Servlet のプログラマが簡単にユーザーの履歴データを利用できるインタフェースを提供している。使い方はひじょうに簡単で、`HttpServletRequest` クラスの `getSession` というメソッドでセッションオブジェクトを取り出し、そのオブジェクトに対して、データを関連づけ（`setAttribute`）たり、読み出し（`getAttribute`）たりするだけである。

3.2　Quiz サーブレット

セッションを利用する簡単な Servlet として Quiz サーブレットを例にあげる。これは過去に正解した問題数などによって、表示を変更する Servlet である。

正解した問題数や現在何問めを実行しているかを保存するためにセッションを利用する。（さもないと、複数のユーザが同時にこのサーブレットにアクセスした時に、正解した問題数のデータがごっちゃになってしまう。）

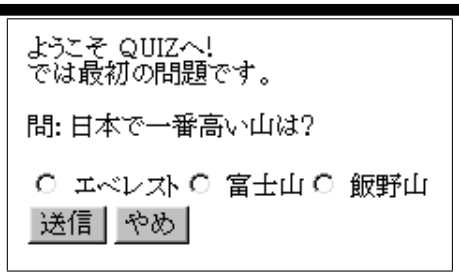
[1]もともとは会期などという意味

例題: QUIZ

ファイル quiz.txt

```
日本で一番高い山は? エベレスト 富士山 飯野山 2
日本で一番広い湖は? 琵琶湖 府中湖 満濃池 1
トリノオリンピックで日本が獲得した金メダルの個数は? 1個 2個 3個 1
工学部の所在地は? 幸町 花園町 林町 3
```

この Servlet は上のようなテキストファイルから右のような QUIZ を表示するページを作成する Servlet である。簡単のため問題は 3 択に固定している。



この Servlet では “現在何番目の問題か？” (number) と “これまでの正解数”（score）というデータがセッションのなかに保持されている。（その 1）の部分は特に変わったところはない。ArrayList を使用するのでこれを import している。

ファイル Quiz.java（その 1）

```
import java.io.*;
import java.util.ArrayList;
import javax.servlet.http.*;

public class Quiz extends HttpServlet {
```

（その 2）の doGet メソッドは最初に Servlet が実行される時に呼ばれる。このメソッドは単に doPost メソッドを呼び出すだけである。

ファイル Quiz.java（その 2）

```
  @Override
  public void doGet(HttpServletRequest request, HttpServletResponse response)
  throws IOException {
    doPost(request, response);
  }
```

（その 3）は doPost メソッドの最初の部分を示す。ここでは、いくつかの局所変数を宣言し、そして、HttpServletRequest クラスの getSession メソッドで現在のセッションオブジェクトを得る。

ファイル Quiz.java（その 3）

```
  @Override
  public void doPost(HttpServletRequest request, HttpServletResponse response)
  throws IOException {
    response.setContentType("text/html; charset=Windows-31J");
    PrintWriter out = response.getWriter();
    out.println("<html><head></head><body>");

    int i, number=0, score=0, answer=0;
    ArrayList<String[]> questions;

    HttpSession session = request.getSession(true) ;
```

session.isNew() が真の時は、セッションができたばかりであることを示す。つまり、このページのアクセスが最初の問であることがわかる。

そのときは、QUIZのデータが入っているファイル（quiz.txt）を開いて、questionsというArrayListに読み込む。このquestionsの値を次の問以降の表示のときに利用するために、setAttributeメソッドを用いて、セッションオブジェクトに保存する。

また、最初の問用のメッセージを表示する。

ファイルQuiz.java（その4）

```
    if (session.isNew()) { // 最初の問
      questions = new ArrayList<String[]>();
      File f = new File(getServletContext().getRealPath("/quiz.txt"));
      BufferedReader in = new BufferedReader(
        new InputStreamReader(new FileInputStream(f), "Windows-31J"));
      String line="";
      while((line=in.readLine())!=null) {
        line = line.trim();
        if(line.trim().equals(""))
          continue;
        questions.add(line.split("¥¥s+"));
      }
      in.close();
      session.setAttribute("questions", questions);
      out.println("<p>ようこそ QUIZ へ!<br>では最初の問題です。</p>");
    }
```

一方、session.isNew()が偽の時は、最初の問ではないので、まず、送られたフォームのデータから、前の問題で解答者が選んだ答の番号（answer）を得る。さらに、セッションデータには最初の問のときに読み込んだ問題のデータ、現在の問題の番号（"number"）とこれまでの正解数（"score"）が記録されているはずなので、Sessionクラスのget Attributeメソッドでその値を取り出す。getAttributeメソッドの引数は取り出したいデータの名前で、戻り値がセッションに保存されていたその名前のデータである。

getAttributeメソッドの戻り値型はObject型（つまりすべてのクラスのスーパークラス）なので、String型に型変換（キャスト）する必要がある。

questionsのnumber番目の最後のトークンを解答と比べる。その結果によってメッセージとscore変数の値を変える。

ファイルQuiz.java（その5）

```
      else { // 最初の問ではない
      answer = Integer.parseInt(request.getParameter("answer"));
      questions = (ArrayList<String[]>)session.getAttribute("questions");
      number = Integer.parseInt((String)session.getAttribute("number"));
      score  = Integer.parseInt((String)session.getAttribute("score"));

      String[] tokens = questions.get(number);
      int a = Integer.parseInt(tokens[tokens.length-1]);
      if (a==answer) { // aは最後の文字
        out.println("正解です。<br>");
        score++;
      } else {
        out.println("残念でした。<br>");
      }
    }
```

次に、numberを一つ増分して、次の問を表示する準備をする。ただし、number番めの問題がないときは、クイズを終了する。このときは、invalidateメソッドでセッションオブジェクトを破棄する。

ファイル Quiz.java（その 6）

```
    number++;
    if (number >= questions.size()) { // 終
      out.println("<br>これで QUIZ は終わりです。<br>");
      out.println("正解数は、"+score+"問でした。");
      session.invalidate();
    }
```

問題が終わりでなければ、questions から次の問の情報を読み取って、フォームとして出力する。そして各選択肢のラジオボタンと送信ボタン、リセットボタンを出力する。

最後に setAttribute メソッドを用いて、セッションオブジェクトにこれまでの問題数と正解数を記録している。setAttribute の引数は保存したいデータの名前（String 型）と保存したいデータ（Object 型）である。Integer.toString メソッドを使って、int 型から String 型に変換してから、setAttribute を呼び出して、セッションオブジェクトに書き込んでいる。（このデータは次の問題を表示するときに利用される。）

ファイル Quiz.java（その 7）

```
      else { // 次の問を表示
      String[] tokens = questions.get(number);
      out.println("次の問: "+tokens[0]+"<br>");
      out.println("<form method='post'>");
      for (i=0; i<3; i++) {
        out.print("<input type='radio' name='answer' value='"+(i+1)+"'>");
        out.print(" "+tokens[i+1]);
      }
      out.println("<br>");
      out.println("<input type='submit' value='送信'>");
      out.println("<input type='reset' value='やめ'>");
      out.println("</form>");

      session.setAttribute("number", Integer.toString(number));
      session.setAttribute("score", Integer.toString(score));
    }
```

ファイル Quiz.java（その 8）

```
    out.println("</body></html>");
    out.close();
  }
}
```

問 3.2.1 *Quiz.java* を *3* 択だけではなく、各問ごとに選択肢の数を変えられるように拡張せよ。

問 3.2.2 （選択肢の記録）
Quiz.java を、 正解数だけではなくて、どの問に対してどの選択肢を選んだかまでわかるように記録し、その結果を「これで *QUIZ* は終わりです。」のメッセージのあとに表示するようにせよ。

キーワード :

クッキー, hidden（隠し要素）, セッション, HttpSession クラス, getSession メソッド, getAttribute メソッド, setAttribute メソッド, Integer.toString メソッド