前回は、HTTP リクエストメッセージの可視化用モジュールを作って、その利用を確認した。
以下にその表示例を示します。(長い行の一部を省略して、・・に変更しています)

```
GET http「//118.241.153.53:8080/all/a200/list.jsp HTTP/1.1
host: 118.241.153.53:8080
connection: keep-alive
upgrade-insecure-requests: 1
user-agent: Mozilla/5.0 (Windows NT ・・, like Gecko) Chrome/104.0.0.0 Safari/537.36
accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/web・・
referer: http://118.241.153.53:8080/all/a200/index.jsp
accept-encoding: gzip, deflate
accept-language: ja-JP,ja;q=0.9,en-US;q=0.8,en;q=0.7
cookie: JSESSIONID=C67C429C13360F42C816B3CFF25C0779
if-modified-since: Sun, 04 Sep 2022 02:20:44 GMT
```

上記の「cookie: JSESSIONID=C67C429C13360F42C816B3CFF25C0779」の部分は、このサーバの index.jsp を最初に閲覧した時のレスポンスで、Set cookie メッセージでブラウザ側に保存した情報です。
上記はそれ以降で同じサーバにリクエストメッセージを送る場合のメッセージ例で、このように記憶した（「cookie: JSESSIONID=C67C429C13360F42C816B3CFF25C0779」）と埋め込んで送り返すようになっています。これを受信したサーバでは、自身が送った ID より相手ブラウザを識別して、その相手用に記憶した session 情報を取り出せる仕組みになっています。
このクッキーは一般に有効期間を指定しないので、**ブラウザを閉じると session 情報も消えることになります。**

10 番目課題で行ったように、**cookie を使って有効期間を指定すればブラウザが閉じた後も情報を残すことができます。**

今回の作品は、複数の**文字列**を指定の有効期限で記憶して、それを表示する能力とします。
これを確かめると、買い物カゴに商品を入れるような処理の基本が体験できます。
なお、session 情報は JSESSIONID の値を使って、サーバ側に記憶しますが、今回の作品は、**文字列**を Cookie で記憶するので、ブラウザ側で記憶します。
さて、11 番目の testcookie.jsp を実行した後で、課題進行ページのリクエストメッセージの Cookie ヘッダ情報を確認すると、次のようになります。

```
cookie: ABC=123ab; JSESSIONID=269BF7FB0FEB63937DB30D85FC594E6A
```

つまり、**cookie は、セミコロンで区切られて複数の情報が１行に存在する形式です。**
ABC=123ab の例でわかるように、有効期間や相手を識別する情報は作り手が組み込まなければ存在しません。
そこで、相手の識別に UUID（"Universally Unique Identifier"）を使い、有効時間に UNIX 時間を連結したクッキー名を使います。また他のクッキーと区別するため**"IDCO"**の接頭辞を付けることにします。
クッキー名の例　IDCO6c7e607e-16e7-44f9-8a48-513a53fd5bfd1662871659

1662871659 の部分が UNIX 時間です。このようなクッキー名にすれば、この部分を分離して、Date expires = new Date(1662871659* 1000);とすることで、有効期限の日時が算出でき、文字列に変換すれば、「2022 年 9 月 11 日 13 時 47 分 39 秒」の表現が得られます。このようにクッキー名をネーミングすることで、セッションを使わなくてもクライアント情報を識別できて、ブラウザを閉じても有効期限以内であれば、記憶した情報をとりだせることが出来ます。
サイトを訪れた日時や、訪問回数などの用途で、さまざまな内容の記録に使われています。（ブラウザを完全に閉じても、前の情報を利用する用途のほとんどで、クッキーが利用されます。）
クッキー記憶の欠点は記憶容量のサイズです。１行のヘッダー行にすべてのクッキー情報がセミコロンで区切られ、URL エンコードして埋め込んで送るので、大きなデータの記録に向きません。サーバ側の Session やデータベースと連動させれば大きなデータも扱えます。
クッキー単体のサイズは Cookie ヘッダーのサイズできまり、それはブラウザの仕様で決まります。
クッキー単体利用のおおよそのサイズは、ドメインあたり 50 個で、ドメインあたりの合計が 4KB 以内の程度とされています。

今回の作品は、クッキー名として（“IDCO”＋ UUID＋ “UNIX 時間”）を使い、
任意の文字列をブラウザで記憶、削除、変更ができる次のような作品を作ります。

秒指定の有効時間と、記憶させたい文字列を入力して、ADD ボタンで記憶させることができるようにします。クッキー名は、“IDCO” ＋ UUID＋ “UNIX 時間”なので、
“IDCO” ＋ UUID＋の文字列がクライアントを識別する文字列で、残りの“UNIX 時間”の文字列から有効時間に変換して、上記のように表で列挙しています。
そして、ラジオボタンで選択したクッキーの削除や変更ができる作品を目指します。
以下のヒントのコードを示します、これで追加と表示ができます。削除、変更のコードが空欄なので、**追加しなさい。**

cookieitems.jsp　の名前で作ります。

```
<%@page contentType="text/html; charset=UTF-8" language="java"
import="java.util.*"
import=" java.net.*"
import="java.text.SimpleDateFormat"
%>
<% //java
String cookieId= "IDCO" + UUID.randomUUID();//新しいクライアントに付ける ID
int lenId=cookieId.length();// cookie 名の共通部の長さ

String val = request.getHeader("cookie");//cookie の値
// val に存在する複数のクッキー情報を、分割して、必要な情報だけ items に記憶
Map <String,String> items = new TreeMap<String,String>();//HashMap の順序あり版
String []cookies=val.split(";");
for(String s : cookies ){
    String s2 = s.trim();
    String []a2=s2.split("=");// クッキー名と値を区切る文字列の=で分解
    if(a2 == null || a2.length != 2) continue;
    a2[0]=a2[0].trim();
    if(a2[0].startsWith("IDCO") == false) continue;
    items.put(a2[0], URLDecoder.decode(a2[1].trim(), "UTF-8"));
    cookieId = a2[0].substring(0, lenId);
}
// form 内の受信情報
String cmd = request.getParameter("CMD");//ボタン表示内容
String rbtn = request.getParameter("RB");//ラジオボタンの値
String time = request.getParameter("TIME");// TextField(有効時間)内容
int sec = time != null ? Integer.parseInt(time) : 0;
String data = request.getParameter("DATA");//TextField 内容

// ボタン操作の処理
if(cmd != null && cmd.equals("ADD")){
    Date date = new Date();
    long mt=date.getTime()/1000 + sec;// クッキー有効期限 Unix 時間
    Cookie cookie = new Cookie(cookieId+mt, URLEncoder.encode(data, "UTF-8"));//クッキー生成
    cookie.setMaxAge(sec);//有効期限は秒数で指定します。
    response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

    // 更新操作時で、ADD などの連続処理を防ぐため強制移
    response.sendRedirect(request.getRequestURL().toString());//自身のページへ移動
} else if(rbtn != null && cmd != null && cmd.equals("DELETE")){

}else if(rbtn != null && cmd != null && cmd.equals("CHANGE")){

}
%>
<!DOCTYPE html>
<html>
<head>
<style>
    body ,input {
        font-size: large;
        line-height: 2em;
    }
</style>
</head>
<body>
<h1>クッキー記憶文字列(期限切れ時間とその記憶文字列)</h1>

```

```
58
59  <form action="cookieitems.jsp">
60  <table border>
61  <% for (Map.Entry<String, String> it : items.entrySet()) {
62      String timeArea = it.getKey().substring(lenId);
63      Date expires = new Date(Long.parseLong(timeArea)* 1000);// 有効期限
64      SimpleDateFormat formatter = new SimpleDateFormat("yyyy 年 M 月 d 日 H 時 m 分 s 秒");
65      String strCls = formatter.format(expires).toString();
66  %>
67      <tr>
68      <td><input type="radio" name="RB" value="<%= it.getKey() %>"></td>
69      <td><%= strCls %></td>
70      <td><%= it.getValue() %></td>
71      </tr>
72  <% } %>
73  </table>
74  <br>
75  <input type="text" value="30" size="5" name="TIME">秒
76   <input type="text" value="変更や入力の文字列" size="20em" name="DATA">
77  <br>
78      <input type="submit" name="CMD" value="ADD">
79      <input type="submit" name="CMD" value="DELETE">
80      <input type="submit" name="CMD" value="CHANGE">
81  </form>
82  <%= cookieId %>がクッキー名の共通部で、残りが Unix 時間です。<br>
83  </body>
84  </html>
```

クッキー名の生成で、java.util.UUID.randomUUID();を使いました。

これで得られる例が、6c7e607e-16e7-44f9-8a48-513a53fd5bfd のような文字列で UUID”Universally Unique Identifier”と呼ばれ、「世界中で唯一の識別子」です。この文字列は、同じ文字列が生成される確率が、「ゼロに等しいと言っていいほどに小さくなり、世界中どこを探しても同じものが存在しないので、このような相手ブラウザの ID 名やファイルの識別名や、オブジェクトの識別名などに使われるものです。初めての相手には、これを利用した ID 名にして、すでに名前が付いていれば、それを使って（21 行）います。

14 行の繰り返しで、受信したクッキーの 1 行の文字列から分解、URL デコードしたクッキー群を Map に記憶しています。セッション名をキーにしてその値を記憶する Map ですが、キーとなる時間で昇順の並びになるように、HashMap でなく TreeMap を使いました。

これで記憶した Map 型の items を表として表示する繰り返しが、61 行の for です。

この繰り返しの終わり指定が 72 行の<% } %>です。

今回初めて、<% for ～%> と <% } %>を分けて書く書き方を示しています。

これにより囲まれた範囲の HTML タグも繰り返しの対象になっていることに注目ください。

39 行で response.sendRedirect により、ブラウザの閲覧直後に移動して、更新でボタンの処理連続動作できなくする細工をしています。Response の暗黙オブジェクトは、

javax.servlet.http.HttpServletResponse 型です。

なお、明示的にクッキーの内容を削除する命令はありませんが、削除対象のクッキーID で、有効期間がゼロ秒のクッキーを作って送れば、ブラウザに削除させることができます。

補足：クッキー以外で、ブラウザ側に情報を記憶させる方法では、Web Storage API の Window.localStorage と呼ぶ機能を Javascript で制御する方法もあるようです。

削除処理の解答例

「

```
Cookie cookie = new Cookie(rbtn, "");//クッキー生成
cookie.setMaxAge(0);//0 の有効期限で削除
response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

// 更新操作時で、連続処理を防ぐため強制移
response.sendRedirect(request.getRequestURL().toString());//自身のページへ移動
```

」

変更処理の解答例

「

```
Cookie cookie = new Cookie(rbtn, "");//クッキー生成
cookie.setMaxAge(0);//0 の有効期限で削除
response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

Date date = new Date();
long mt=date.getTime()/1000 + sec;// クッキー有効期限 Unix 時間
cookie = new Cookie(cookieId+mt, URLEncoder.encode(data, "UTF-8"));//生成
cookie.setMaxAge(sec);//有効期限は秒数で指定します。
response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

// 更新操作時で、連続処理を防ぐため強制移
response.sendRedirect(request.getRequestURL().toString());//自身のページへ移動
```

」

```
<%@page contentType="text/html; charset=UTF-8" language="java"
import="java.util.*"
import=" java.net.*"
import="java.text.SimpleDateFormat"
%>
<% //java
String cookieId= "IDCO" + UUID.randomUUID();//新しいクライアントに付ける ID
int lenId=cookieId.length();// cookie 名の共通部の長さ

String val = request.getHeader("cookie");//cookie の値
// val に存在する複数のクッキー情報を、分割して、必要な情報だけ items に記憶
Map <String,String> items = new TreeMap<String,String>();//HashMap の順序あり版
String []cookies=val.split(";");
for(String s : cookies ){
    String s2 = s.trim();
    String []a2=s2.split("=");
    if(a2 == null || a2.length != 2) continue;
    a2[0]=a2[0].trim();
    if(a2[0].startsWith("IDCO") == false) continue;
    items.put(a2[0], URLDecoder.decode(a2[1].trim(), "UTF-8"));
    cookieId = a2[0].substring(0, lenId);
}
// form 内の受信情報
String cmd = request.getParameter("CMD");//ボタン表示内容
String rbtn = request.getParameter("RB");//ラジオボタンの値
String time = request.getParameter("TIME");// TextField(有効時間)内容
int sec = time != null ? Integer.parseInt(time) : 0;
String data = request.getParameter("DATA");//TextField 内容

// ボタン操作の処理
if(cmd != null && cmd.equals("ADD")){
    Date date = new Date();
    long mt=date.getTime()/1000 + sec;// クッキー有効期限 Unix 時間
    Cookie cookie = new Cookie(cookieId+mt, URLEncoder.encode(data, "UTF-8"));//クッキー
生成
    cookie.setMaxAge(sec);//有効期限は秒数で指定します。
    response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

    // 更新操作時で、ADD などの連続処理を防ぐため強制移
    response.sendRedirect(request.getRequestURL().toString());//自身のページへ移動
} else if(rbtn != null && cmd != null && cmd.equals("DELETE")){
    Cookie cookie = new Cookie(rbtn, "");//クッキー生成
    cookie.setMaxAge(0);//0 の有効期限で削除
    response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

    // 更新操作時で、ADD などの連続処理を防ぐため強制移
    response.sendRedirect(request.getRequestURL().toString());//自身のページへ移動
}

else if(rbtn != null && cmd != null && cmd.equals("CHANGE")){
    Cookie cookie = new Cookie(rbtn, "");//クッキー生成
    cookie.setMaxAge(0);//0 の有効期限で削除
    response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

    Date date = new Date();
    long mt=date.getTime()/1000 + sec;// クッキー有効期限 Unix 時間
    cookie = new Cookie(cookieId+mt, URLEncoder.encode(data, "UTF-8"));//クッキー生成
    cookie.setMaxAge(sec);//有効期限は秒数で指定します。
    response.addCookie(cookie);//クッキーを記憶する指示を応答のヘッダに含める指定

    // 更新操作時で、ADD などの連続処理を防ぐため強制移
    response.sendRedirect(request.getRequestURL().toString());//自身のページへ移動
}
%>
```

```
<!DOCTYPE html>
<html>
<head>
<style>
    body ,input {
        font-size: large;
        line-height: 2em;
    }
</style>
</head>
<body>
<h1>クッキー記憶文字列(期限切れ時間とその記憶文字列)</h1>

<form action="cookieitems.jsp">
<table border>
<% for (Map.Entry<String, String> it : items.entrySet()) {
    String timeArea = it.getKey().substring(lenId);
    Date expires = new Date(Long.parseLong(timeArea)* 1000);// 有効期限
    SimpleDateFormat formatter = new SimpleDateFormat("yyyy年 M月 d日 H時 m分 s秒");
    String strCls = formatter.format(expires).toString();
%>
    <tr>
    <td><input type="radio" name="RB" value="<%= it.getKey() %>"></td>
    <td><%= strCls %></td>
    <td><%= it.getValue() %></td>
    </tr>
<% } %>
</table>
<br>
<input type="text" value="30" size="5" name="TIME">秒
 <input type="text" value="変更や入力の文字列" size="20em" name="DATA">
<br>
    <input type="submit" name="CMD" value="ADD">
    <input type="submit" name="CMD" value="DELETE">
    <input type="submit" name="CMD" value="CHANGE">
</form>
<%= cookieId %>がクッキー名の共通部で、残りが Unix 時間です。<br>
</body>
</html>
```