情報科学プロジェクト実験レポート課題

S142063 佐藤涼亮

平成 28 年 12 月 22 日

HTTP Cookie

1　課題の内容

HTTP Cookie によるセッション管理

1.1　要点

1. ある HTTP 要求に対してサーバは応答ヘッダに Set-Cookie:を設定する。
2. Web ブラウザは設定されたその値をメモリまたはファイルに保存する。
3. Web ブラウザは指定された URL へのアクセスの際には、要求ヘッダに Cookie:でその値を指定する。
4. サーバは要求ヘッダ内の Cookie:の指定の有無で応答内容を変更する。
   (この値は CGI の環境変数 HTTP_COOKIE に設定される)

2　プログラムの説明

ログイン画面と検索画面を作成し、それぞれのに対応した処理を行う。ログイン画面には、ユーザー名とパスワードの入力欄を設け、ログインに必要な情報の入力を求め、ログインを行う。検索画面では、郵便番号入力欄を設け、検索したい住所の情報の入力を求め、検索を行う。それぞれ以下の 4 つの状況に応じた呼び出しと処理を行う。

- get メソッドでクッキー無しまたは DB に一致しないクッキー付きの場合
  ログイン画面を表示
- get メソッドで、DB に一致するクッキー付きの場合
  検索画面を表示
- post メソッドで、クッキー無しまたは DB に一致しないクッキー付きの場合
  ログイン画面を表示
- post メソッドで、DB に一致するクッキー付きの場合
  pnum がなければ検索画面を表示
  pnum があれば検索結果とともに検索画面を表示

「DB に一致するクッキー」とは有効期限内のもののみとする
有効期限は登録、更新した時刻から 3 分間とする。
入力されたユーザー情報は、データベースに格納される。ユーザー情報を格納するデータベースのテーブル、スキーマは以下の通りである。

```
sqlite> .table
account
sqlite> .schema
CREATE TABLE account (
user text not null,
pass text not null,
expire integer,
cookie text
);
```

2.1 目的

HTTP Cookie によるセッション管理

2.2 方法

HTTP Cookie によるセッション管理を用いて、セッションにタイムアウト機能を設けた Web プログラムの実装

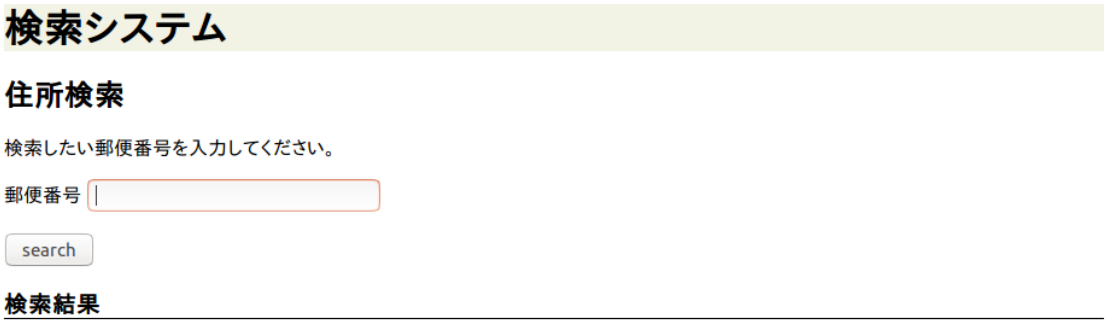
2.3 結果

初期状態

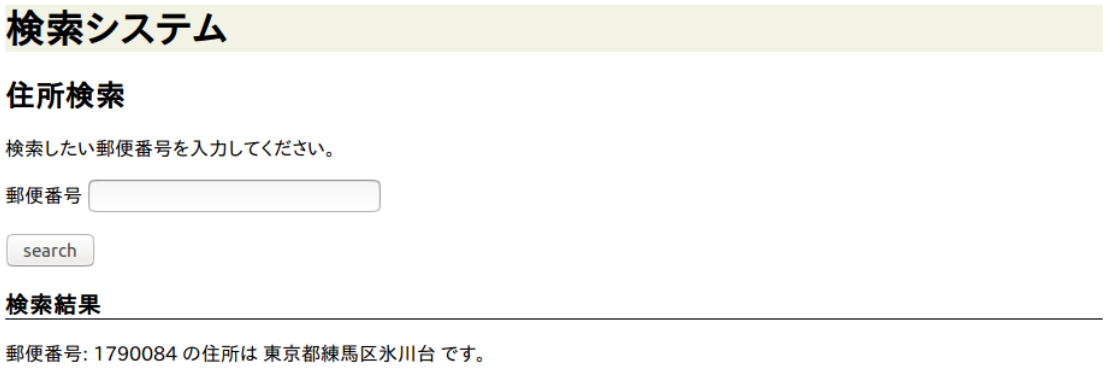
ユーザー名に user、パスワードに password と入力しログインを行う

↓

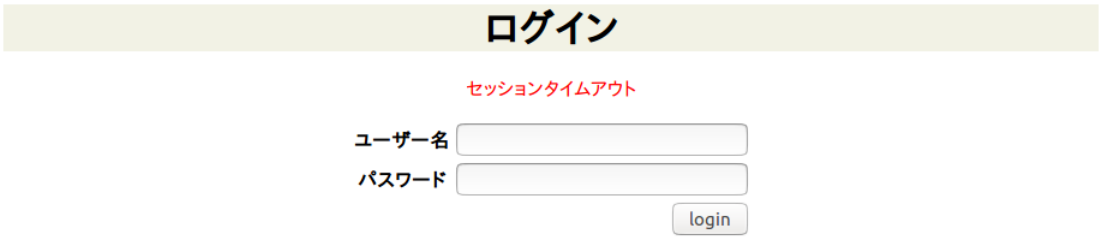
初めてのユーザー名の場合、パスワードと一緒に登録され
検索画面が表示される



検索したい郵便番号を入力し検索をする
↓
検索画面に結果が追加され表示される

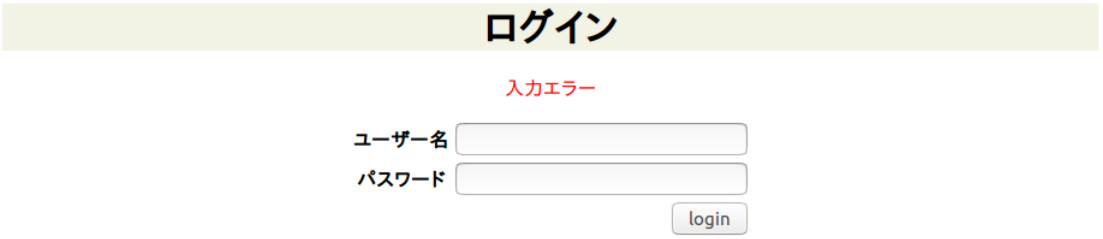


有効期限内は検索が繰り返し行える。
↓
セッションがタイムアウトすると
ページがリロードされるときログイン画面に戻される



再度ログインすると Cookie と有効期限が更新され検索画面が表示される

入力に間違いがあると入力エラーとなる。



2.4 考察

実行前のデータベースの状態は以下のとおりである。

```
sqlite> select * from account;
sqlite>
```

実行後、データベースは以下のとおりになる。

```
sqlite> select * from account;
user|password|1482230002|48727995
sqlite>
```

再度ログイン後のデータベースは以下のとおりになる。

```
sqlite> select * from account;
user|password|1482231268|49741217
sqlite>
```

これらより、データの格納・更新は成功し、セッションのタイムアウトも行われたことから、期待通りの結果が得られた。

3 感想

今回、CGI プログラムにおいて、HTTP Cookie におけるセッション管理の機能を加えたプログラムを作成することができた。この技術を用いて、成蹊ポータルのようなサイトの作成ををしてみたいと思った。その上では、セキュリティの知識が必要となってくる。なので、セキュリティについて今後学んでみたいと思う。また、今回のプログラムに入力されたユーザー名やパスワードなどの使用可能かの検証を行えば、より、いいサイトが作成できると思う。それにあたって、以前学んだ Ajax の技術などを用いれるのではないかと考える。

4 プログラム

ソースコード 1: cookie.cpp

```cpp
#include <random>
#include <sstream>
#include <iomanip>
#include <iostream>
#include <ctime>
#include <string>
#include <MyDBS3.hpp>
#include <CGIinput.hpp>
using namespace std;

int html_login(int s=0);
int html_search(string pnum="");
bool check_cookie(string cookie);
bool check_user(string user);
bool check_account(string user,string pass);
void create_account(string user,string pass);
void update_cookie(string user);

int main() {
  CGIinput tbl;
  cout << "Content-type: text/html; charset=UTF-8\r\n";
  const char *e = getenv("HTTP_COOKIE");
  if (e != nullptr) {
    // クッキーが送られてきた
    string s = e;
    bool valid = check_cookie(s); // DB を検索して s を探す
    if (!valid) return html_login(1); // s がない||期限が無効
    string pnum = tbl["post"]; // POST メソッドの post 指定を取り出す
    return html_search(pnum);
  }
  e = getenv("REQUEST_METHOD");
  if ((e != nullptr && string(e) == "POST")) {

    string user = tbl["user"];
    string pass = tbl["pass"];
    if(tbl["OK"] == "search") return html_login(1); // セッションタイムアウト

    if (user == "" || pass == "") return html_login(2); // form 入力エラー

    if (!check_user(user)) create_account(user,pass); // 新アカウント作成

    else if (check_account(user,pass)) update_cookie(user); // cookie を更新

    else return html_login(2); // pass 間違い

    return html_search(); // 新規または login 成功
  }
  return html_login();
}

// login html
int html_login(int s){
  cout << R"(
  <!doctype html>
  <html>
  <head>
  <meta charset="utf-8">
  <title>Login</title>
  </head>
  <body>
  <h1 align="center" style="background-color: #F2F2E5;">ログイン</h1>
  )";

  // 状況に応じた表示
```

```
65    s==2?cout << R"(<p align="center" style="color: #ff0000;">入力エラー</p>)":
66    s==1?cout << R"(<p align="center" style="color: #ff0000;">セッションタイムアウト</p
          >)":
67          cout << R"(<p align="center"">ログインしてください</p>)";
68
69    cout << R"(
70    <form action="cookie.cgi" method="post">
71    <table align="center">
72    <tr>
73    <th><label for="user">ユーザー名</label></th>
74    <td><input name="user" size="20" type=text /></td>
75    </tr>
76    <tr>
77    <th><label for="pass">パスワード</label></th>
78    <td><input name="pass" size="20" type=password /></td>
79    </tr>
80    <tr>
81    <th></th>
82    <td align="right"><input type="submit" name="OK" value="login" /></td>
83    </tr>
84    </table>
85    </form>
86    </body>
87    </html>
88    )";
89    return 0;
90  }
91
92  // search html and cgi
93  int html_search(string pnum){
94    string search = R"(
95    <!doctype html><html>
96    <head><meta charset="utf-8"><title>Search</title></head>
97    <body><h1 style="background-color: #F2F2E5;">検索システム</h1>
98    <h2>住所検索</h2>
99    <p>検索したい郵便番号を入力してください。</p>
100   <form action="cookie.cgi" method="post">
101   <p>郵便番号 <input name="post" size="20" maxlength='7' /></p>
102   <input type="submit" name="OK" value="search" />
103   </form>
104   <h3 style="border-bottom: solid 1px #000000;">検索結果</h3>
105   )";
106
107   // データベースファイルを開く
108   MyDBS d("db/tokyo.db");
109   if (!d) {cout << search+"open␣error</body></html>";return 1;}
110
111   string sql;
112
113   // 郵便番号での検索
114   if(!pnum.empty()){
115     sql = "select␣kanji1,kanji2,kanji3␣from␣post␣where␣num␣=␣?;";
116     if(d.prepare(sql,pnum) != SQLITE_OK){
117       cout << search + d.error() + "</body></html>";
118       return 1;
119     }
120     string kanji1,kanji2,kanji3;
121     if(d.step(&kanji1,&kanji2,&kanji3) != SQLITE_ROW){
122       search += "<p>郵便番号:␣一致するものがありません</p>";
123     }
124     else{
125       search += "<p>郵便番号:␣" + pnum + "␣の住所は␣"
126         + kanji1 + kanji2 + kanji3 + "␣です。</p>";
127       while(d.step(&kanji1,&kanji2,&kanji3) == SQLITE_ROW){
128         search += "<p>郵便番号:␣" + pnum + "␣の住所は␣"
129           + kanji1 + kanji2 + kanji3 + "␣です。</p>";
130       }
131     }
```

```
132     }
133     cout << search+"</body></html>";
134 
135     return 0;
136 }
137 
138 // cookie の有無と期限の確認
139 bool check_cookie(string cookie){
140     MyDBS d("db/account.db");
141     if(!d) return 0;
142 
143     cookie = cookie.substr(5);
144 
145     string sql = "select␣expire␣from␣account␣where␣cookie␣=␣?";
146     if(d.prepare(sql,cookie) != SQLITE_OK){
147         return 0;
148     }
149     int expire;
150     if(d.step(&expire) == SQLITE_ROW){
151         return expire > (int)time(0);
152     }
153     return 0;
154 }
155 
156 // user の有無
157 bool check_user(string user){
158     MyDBS d("db/account.db");
159     if(!d) return 0;
160 
161     string sql = "select␣count(*)␣from␣account␣where␣user␣=␣?";
162     if(d.prepare(sql,user) != SQLITE_OK){
163         return 0;
164     }
165     int count=0;
166     if(d.step(&count) == SQLITE_ROW){
167         if(count > 0) return 1;
168     }
169     return 0;
170 }
171 
172 // user と pass 対応する 2 つの有無
173 bool check_account(string user,string pass){
174     MyDBS d("db/account.db");
175     if(!d) return 0;
176 
177     string sql = "select␣count(*)␣from␣account␣where␣user␣=␣?␣and␣pass␣=␣?";
178     if(d.prepare(sql,user,pass) != SQLITE_OK){
179         return 0;
180     }
181     int count=0;
182     if(d.step(&count) == SQLITE_ROW){
183         if(count > 0) return 1;
184     }
185     return 0;
186 }
187 
188 // 新しいuser の設定
189 void create_account(string user,string pass){
190     MyDBS d("db/account.db");
191     if(!d) return ;
192 
193     string sql = "insert␣into␣account␣values␣(?,?,?,?)";
194 
195     time_t t = time(0);
196     t = (time_t)(t+180); // 現時刻から 3分後まで有効
197 
198     const int sz = 32; // 30 is enough for this purpose
199     char e[sz];
200     strftime(e, sz, "%a,␣%d␣%b␣%Y␣%T␣%Z", localtime(&t));
```

```
201     string expire = e;
202
203     static default_random_engine dre((unsigned)time(0));
204     uniform_int_distribution<int> di(1,99999999);
205     stringstream ss;
206     ss << setfill('0')<< setw(8) << di(dre);
207     string cookie = ss.str();
208
209     if(d.exec(sql,user,pass,(int)t+(60*3),cookie) == SQLITE_OK){
210       // cookie の設定
211       cout << "Set-Cookie: name=" + cookie + ";"
212                           "expires=" + expire + ";"
213                           "Path=/cgi/"
214            << "\r\n";
215     }
216 }
217
218 // 既存user の更新
219 void update_cookie(string user){
220     MyDBS d("db/account.db");
221     if(!d) return ;
222
223     string sql = "update account set expire = ?,cookie = ? where user = ?";
224
225     time_t t = time(0);
226     t = (time_t)(t+180); // 現時刻から 3分後まで有効
227
228     const int sz = 32; // 30 is enough for this purpose
229     char e[sz];
230     strftime(e, sz, "%a, %d %b %Y %T %Z", localtime(&t));
231     string expire = e;
232
233     static default_random_engine dre((unsigned)time(0));
234     uniform_int_distribution<int> di(1,99999999);
235     stringstream ss;
236     ss << setfill('0')<< setw(8) << di(dre);
237     string cookie = ss.str();
238
239     if(d.exec(sql,(int)t+(60*3),cookie,user) == SQLITE_OK){
240       // cookie の設定
241       cout << "Set-Cookie: name=" + cookie + ";"
242                           "expires=" + expire + ";"
243                           "Path=/cgi/"
244            << "\r\n";
245     }
246 }
```