

ログイン前の CSS、画像の表示について

1. こんな現象ありませんか？

ログイン前の画面で CSS が反映されない、画像が表示されない

2. 原因

LoginFilter の以下の処理が原因です。

```
//loginページ以外の場合は、ログインチェックを行う
if("/login".equals(servletPath) != true && "/auth".equals(servletPath) != true){
    //ログインチェックを行う（セッションからログイン情報を取得してnullでなければOK）

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

    //ログイン情報をセッションから取得
    LoginInfoBeans loginInfo = (LoginInfoBeans)session.getAttribute("loginInfo");

    //ログイン情報がnullなら未ログイン
    if( loginInfo == null ){
        System.out.println("ログインしていないのでログイン画面へ戻します");
        ((HttpServletResponse)response).sendRedirect("/login");
        return;
    }
}
```

この処理は『URL が"/login"、"/auth"の場合はログインチェックをしない』という処理です。除外する理由は、ログイン前なので当然ログインチェックが NG になるためです。

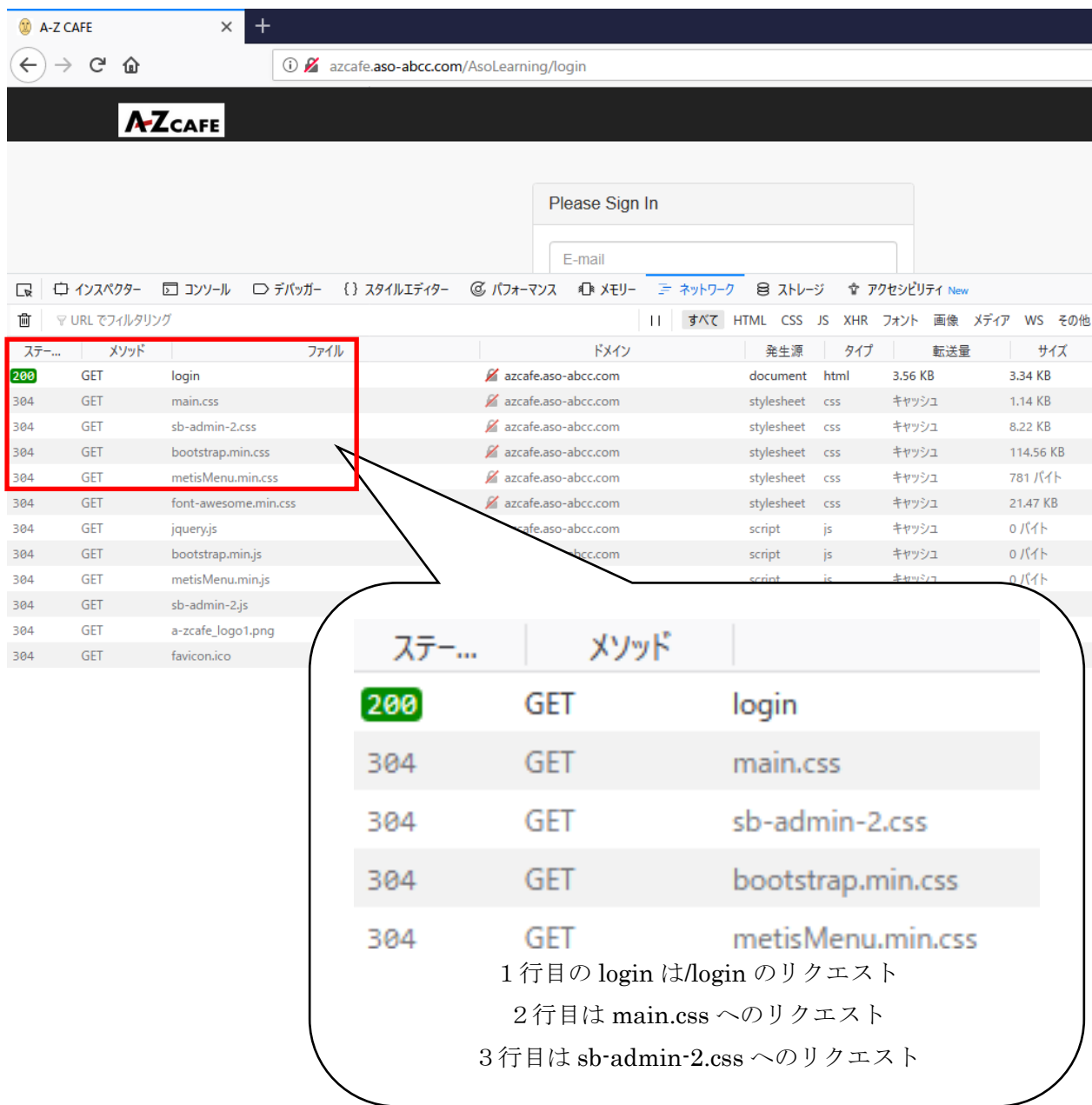
なぜ、この処理があると CSS や画像が表示されないのでしょうか？

理由は以下です。

CSS や画像ファイルもひとつひとつサーバーにリクエストされている為
です。

Firefox のインスペクタを見てください（下図）

AZ-Café のログイン画面を表示した時にリクエストの様子です



The screenshot shows the Firefox Developer Tools Network tab. The URL bar indicates the page is `azcafe.aso-abcc.com/AsoLearning/login`. The page content shows a "Please Sign In" form with an "E-mail" input field. The Network tab displays a list of requests. A red box highlights the first four requests:

ステータス	メソッド	ファイル
200	GET	login
304	GET	main.css
304	GET	sb-admin-2.css
304	GET	bootstrap.min.css

A callout box provides a detailed view of these requests:

ステータス	メソッド	ファイル
200	GET	login
304	GET	main.css
304	GET	sb-admin-2.css
304	GET	bootstrap.min.css
304	GET	metisMenu.min.css

1 行目の login は/login のリクエスト
2 行目は main.css へのリクエスト
3 行目は sb-admin-2.css へのリクエスト

このように CSS や画像ファイルはページのリクエストとは別にリクエストが飛んでいます。一方でサブレットのフィルターはリクエストを受けると必ず動きます。

当然 CSS がリクエストされたときもフィルターが動作します。

フィルターでの処理は「/login、/auth ならばログインチェックをしない」ですから

CSS は該当せず、ログイン前にもかかわらず CSS はログインチェックをされてしまします。

その結果、CSSは「未ログインなのでlogin画面へリダイレクト」されCSSが適用されないという流れになっています。

画像の場合も同様です。

3. 解決方法

LoginFilter に以下の処理を追加すれば OK です。

・リクエストされた URL の拡張子を見て CSS や JPG など特定のファイルの場合もログインチェック除外とする

具体的には以下のような修正をすれば OK

1) LoginFilter に以下のメソッドを追加する

```
/**
 * 拡張子を取得する
 * @param fileName
 * @return
 */
private String getExt(String fileName){
    if (fileName == null)
        return null;
    int point = fileName.lastIndexOf(".");
    if(point == -1){
        return "";
    }
    return fileName.substring(point+1);
}
```

このメソッドは、ファイル名から拡張子を取り出すメソッドです。

2) doFilter メソッドに赤枠の部分を追加します。

```
@Override
public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
    throws IOException, ServletException {

    //除外対象の拡張子
    String excludeExtList[] = {
        "js", "css", "png", "gif", "jpg", "ico"
    };

    //////////////////////////////////////
    //リクエストのサーブレットパスを取得
    String servletPath = ((HttpServletRequest)request).getServletPath();

    System.out.println("servletPath:" + servletPath);

    //////////////////////////////////////
    //拡張子による除外
    if( Arrays.asList(excludeExtList).contains(getExt(servletPath))){
        chain.doFilter(request, response);
        return;
    }

    //////////////////////////////////////
    //loginページ以外の場合は、ログインチェックを行う
    if("/login".equals(servletPath) != true && "/auth".equals(servletPath) != true){
        //ログインチェックを行う (セッションからログイン情報を取得してnullでなければOK)

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

        //ログイン情報をセッションから取得
        LoginInfoBeans loginInfo = (LoginInfoBeans)session.getAttribute("loginInfo");
```

LoginFilter の全体は次ページに記載

```

1 package gatcha.filter;
2
3 import java.io.IOException;
4 import java.util.Arrays;
5
6 import javax.servlet.Filter;
7 import javax.servlet.FilterChain;
8 import javax.servlet.FilterConfig;
9 import javax.servlet.ServletException;
10 import javax.servlet.ServletRequest;
11 import javax.servlet.ServletResponse;
12 import javax.servlet.annotation.WebFilter;
13 import javax.servlet.http.HttpServletRequest;
14 import javax.servlet.http.HttpServletResponse;
15 import javax.servlet.http.HttpSession;
16
17 import gatcha.beans.LoginInfoBeans;
18
19 @WebFilter("/*")
20 public class LoginCheckFilter implements Filter {
21
22     @Override
23     public void destroy() {
24         //無処理
25     }
26
27     @Override
28     public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
29         throws IOException, ServletException {
30
31         //除外対象の拡張子
32         String excludeExtList[] = {
33             "js", "css", "png", "gif", "jpg", "ico"
34         };
35
36         //////////////////////////////////////
37         //リクエストのサーブレットパスを取得
38         String servletPath = ((HttpServletRequest)request).getServletPath();
39
40         System.out.println("servletPath: " + servletPath);
41
42         //////////////////////////////////////
43         //拡張子による除外
44         if (Arrays.asList(excludeExtList).contains(getExt(servletPath))) {
45             chain.doFilter(request, response);
46             return;
47         }
48
49         //////////////////////////////////////
50         //loginページ以外の場合は、ログインチェックを行う
51         if ("/login".equals(servletPath) != true && "/auth".equals(servletPath) != true) {
52             //////////////////////////////////////
53             //ログインチェックを行う (セッションからログイン情報を取得してnullでなければOK)
54
55             HttpSession session = ((HttpServletRequest)request).getSession(true);
56
57             //ログイン情報をセッションから取得
58             LoginInfoBeans loginInfo = (LoginInfoBeans)session.getAttribute("loginInfo");
59
60             //ログイン情報がnullなら未ログイン
61             if (loginInfo == null) {
62                 System.out.println("ログインしていないのでログイン画面へ戻します");
63                 ((HttpServletResponse)response).sendRedirect("login");
64                 return;
65             }
66
67         }
68
69         //処理を続行する
70         chain.doFilter(request, response);
71     }
72 }

```

次ページに続く・・・

```
76 @Override
77 public void init(FilterConfig arg0) throws ServletException {
78     //無処理
79     ;
80 }
81 /**
82  * 拡張子を取得する
83  * @param fileName
84  * @return
85  */
86 private String getExt(String fileName){
87     if (fileName == null)
88         return null;
89     int point = fileName.lastIndexOf(".");
90     if(point == -1){
91         return "";
92     }
93     return fileName.substring(point+1);
94 }
95 }
96 }
97 }
98 }
```