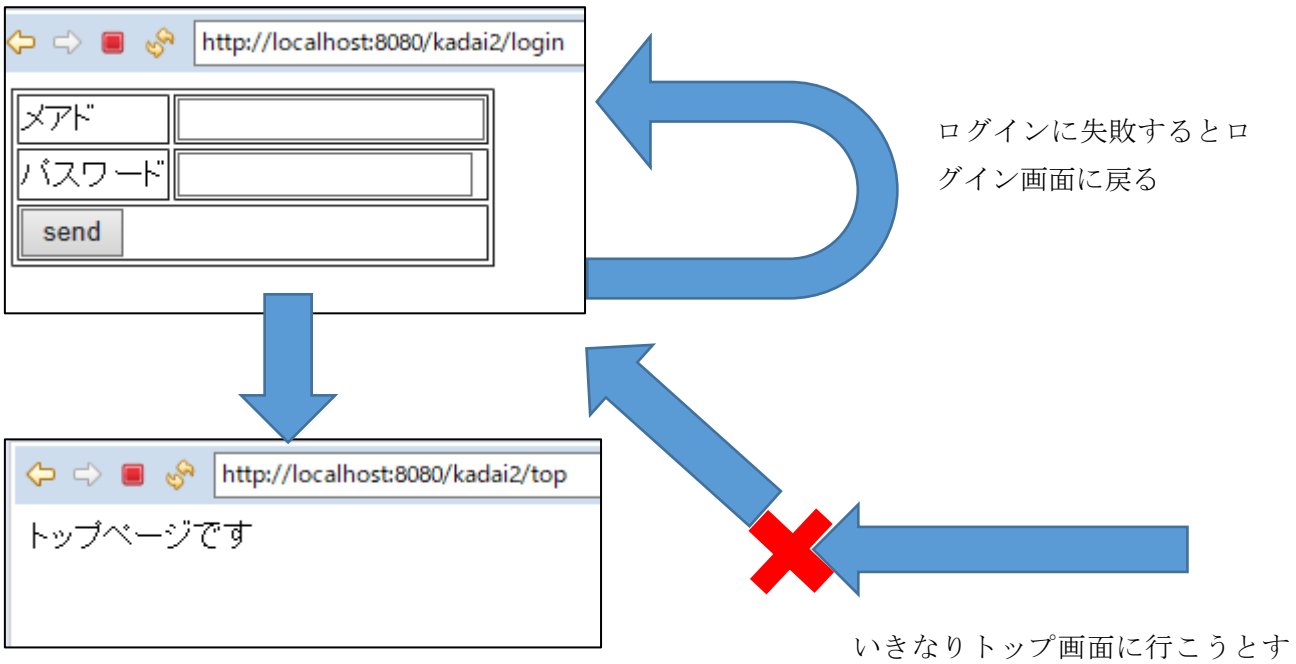
タイトル	
ログイン処理とログインチェックフィルターを作ろう！	
レベル	★★★★★★

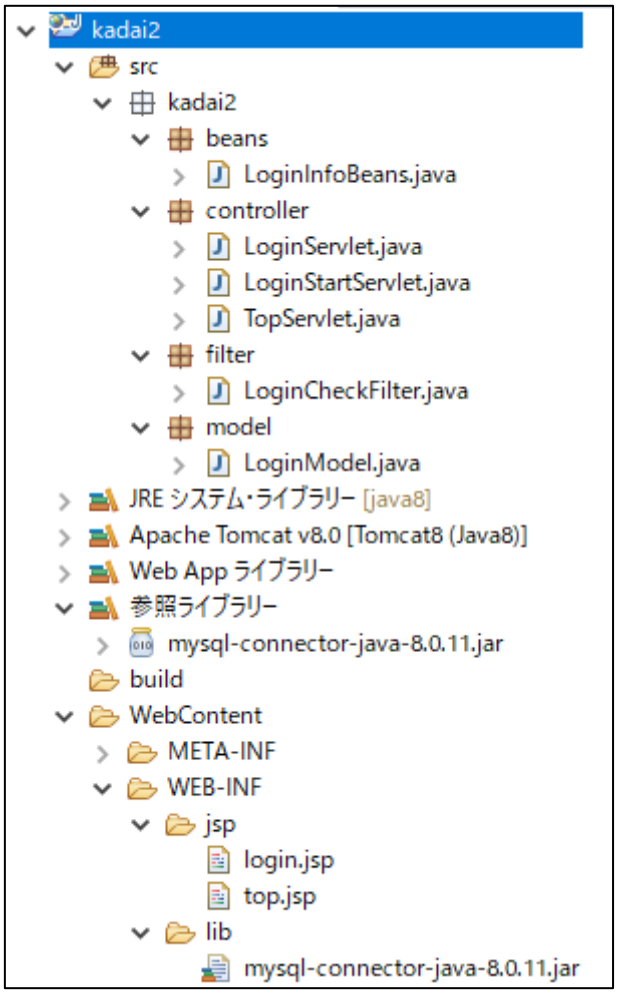
フィルターを実装して、ログインチェックフィルターを作ろう！



早く終わった人は↓もチャレンジ！

内容	難易度
ログイン失敗時にログイン画面にエラーメッセージを表示する。（画面遷移はリダイレクトのまま） ヒント：リダイレクト時は GET 通信になります。GET 通信ではパラメータは URL の一部になるのでしたね	★★★★★★★★

プロジェクトの構成（クラス名）は以下のようにしましょう



ファイルと概要

クラス名 or ファイル名	概要	URL 割り当て
LoginStartServlet	ログイン画面を表示するサーブレット	login
LoginServlet	ログイン処理をするサーブレット	auth
TopServlet	トップページを表示するサーブレット	top
LoginCheckFilter	ログインチェック用のフィルター	すべての URL
LoginModel	DB にアクセスし、メールアドレスとパスワードの照合を行う	
LoginInfoBeans	ログイン情報保持のためのクラス	
login.jsp	ログイン画面の JSP	login
top.jsp	トップ画面の JSP	top

※次ページからのソースコードを元に処理を考えよう！

LoginStartServlet

```java
@WebServlet(                    )
public class LoginStartServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        RequestDispatcher dispatcher = request.getRequestDispatcher("WEB-INF/jsp/login.jsp");
        dispatcher.forward(request, response);
    }
}
```

LoginServlet

```java
@WebServlet(                    )
public class LoginServlet extends HttpServlet {

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        //////////////////////////////////////////
        //JSPからメールアドレスとパスワードを取得
        String mail =                                   ;
        String password =                                       ;

        //////////////////////////////////////////
        //Modelを呼び出しDBの値をメール、パスワードを照合する
        LoginModel loginModel = new LoginModel();

        LoginInfoBeans loginInfo =                                      );

        if( loginInfo != null ){
            //////////////////////////////////////////
            //ログイン結果をセッションに保存する
            HttpSession session =                           ;

                                ("loginInfo", loginInfo);
        }else{
            //ログイン結果がnullの場合はログイン画面に戻す
            response.sendRedirect("login");
            return;
        }

        //////////////////////////////////////////
        //画面を転送する（リダイレクト）
        response.sendRedirect("top");
    }
}
```

TopServlet

```
@WebServlet(        )
public class TopServlet extends HttpServlet {

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        RequestDispatcher dispatcher = request.getRequestDispatcher("WEB-INF/jsp/top.jsp");
        dispatcher.forward(request, response);
    }
}
```

LoginInfoBeans

```
/**
 * ログイン情報ビーンズ
 *
 * @author nishino
 *
 */
public class LoginInfoBeans implements Serializable {

    private String name;
    private String mail;

    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getMail() {
        return mail;
    }
    public void setMail(String mail) {
        this.mail = mail;
    }


}
```

LoginModel

```
public class LoginModel {

	/**
	 * ログイン処理を行う
	 * 引数でもらったmailとパスワード
	 *
	 * @param mail
	 * @param password
	 * @return
	 */
	public LoginInfoBeans login(String mail,String password){
		LoginInfoBeans loginInfo = null;

		Connection con = null;
		PreparedStatement stmt = null;
		ResultSet rs = null;

		try{
			////////////////////////////////////
			//DBの接続
```

```
			////////////////////////////////////
			//SELECT文の発行
```

prepareStatement を使って SQL を組み立てて

executeQuery で実行する

```
			////////////////////////////////////
			//DBから値を取得
```

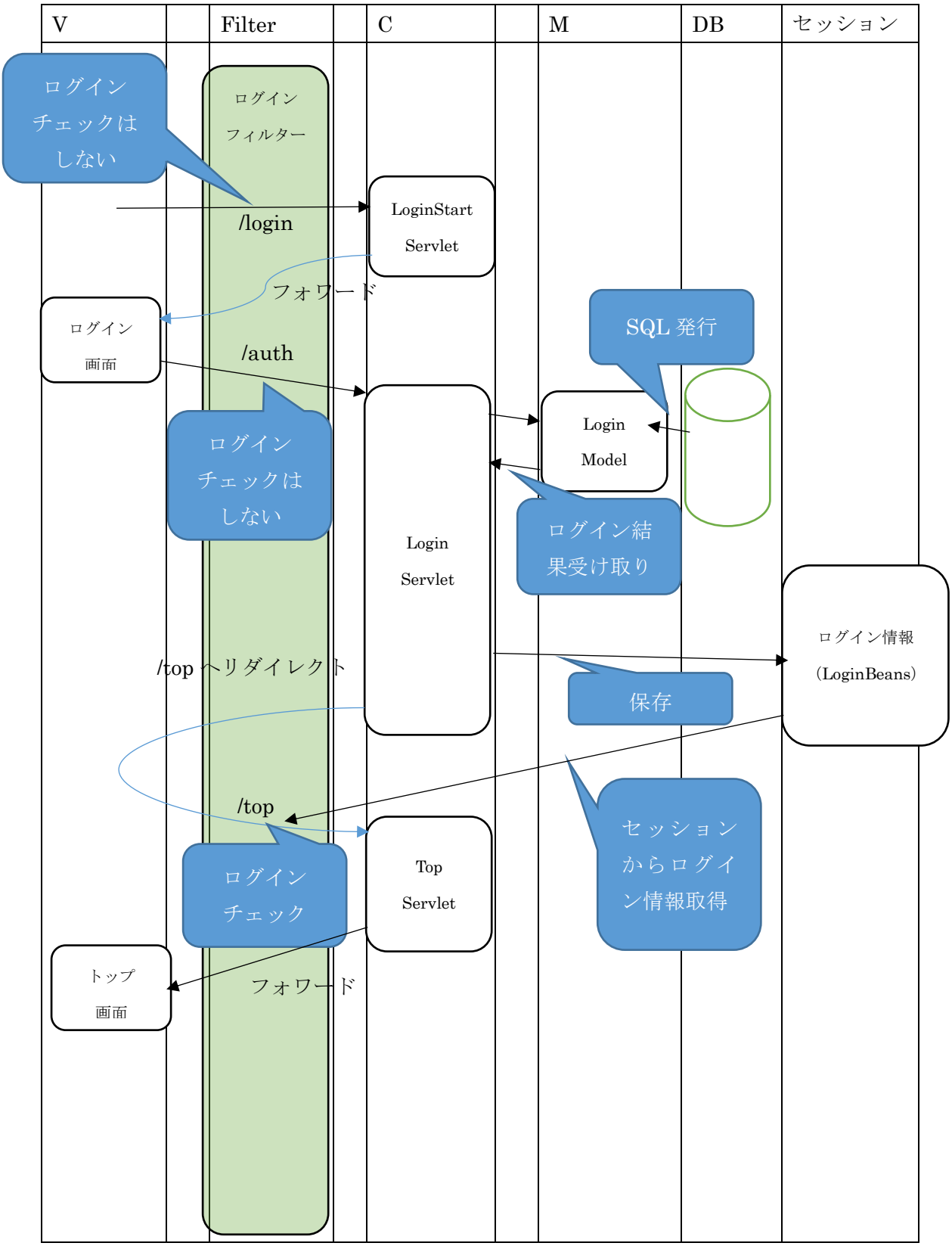
while 文を使って DB から値を取得

```
		}catch(ClassNotFoundException e) {
			//エラー発生した場合にコンソールにログを出力する
			e.printStackTrace();
		}catch(SQLException e) {
			//エラー発生した場合にコンソールにログを出力する
			e.printStackTrace();
		}
		finally {
			//接続（コネクション）を閉じる
			if(con!=null) {
				try {
					con.close();
				}
				catch(SQLException e) {
					e.printStackTrace();
				}
			}
		}


		return loginInfo;
	}
}
```

LoginCheckFilter

```
@WebFilter(          )
public class LoginCheckFilter implements Filter {

    @Override
    public void destroy() {
        //無処理
        ;
    }

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

        ////////////////////////////////////////
        //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);
        ////////////////////////////////////////
        //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 =                                                    ;

            //ログイン情報をセッションから取得
            LoginInfoBeans 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");
                                                                                    ;
                return;
            }

        }

        //処理を続行する
        chain.doFilter(request, response);

    }

    @Override
    public void init(FilterConfig arg0) throws ServletException {
        //無処理
        ;
    }

}
```

リダイレクトでログイン画面へ遷移


V
Filter
C
M
DB
セッション
ログイン
チェックは
しない
ログイン
フィルター
/login
LoginStart
Servlet
フォワード
ログイン
画面
/auth
SQL 発行
ログイン
チェックは
しない
Login
Model
Login
Servlet
ログイン結
果受け取り
ログイン情報
(LoginBeans)
/top へリダイレクト
保存
/top
セッション
からログイ
ン情報取得
ログイン
チェック
Top
Servlet
トップ
画面
フォワード