Example: Servlet Which Destroy Session.

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

public class LogoutController extends HttpServlet
{
        protected void doPost(HttpServletRequest request,
                                                HttpServletResponse response)
                                                Throws ServletException, IOException
        {
                response.setContentType("text/html");
                PrintWriter out = response.getWriter();
                out.println("Thank You. Session was destroyed successfully!!");
                HttpSession session = request.getSession(false);

                synchronized (session)
                {
                        session.setAttribute("user", null);
                        session.removeAttribute("user");
                        session.getMaxInactiveInterval();
                        session.invalidate();
                }
        }
}
```

MVC (Model View Controller)

MVC stands for Model View and Controller. It is a design pattern that separates the business logic and data access layers from the presentation layer.

Controller: Controller is a Servlet that acts as an interface between View and Model. Controller intercepts all the incoming requests.

Model: Model is a Java Bean which represents the state of the application i.e. data. It can also have business logic.

View: View is a JSP page which represents the presentation or User Interface.

Steps required to Implement MVC

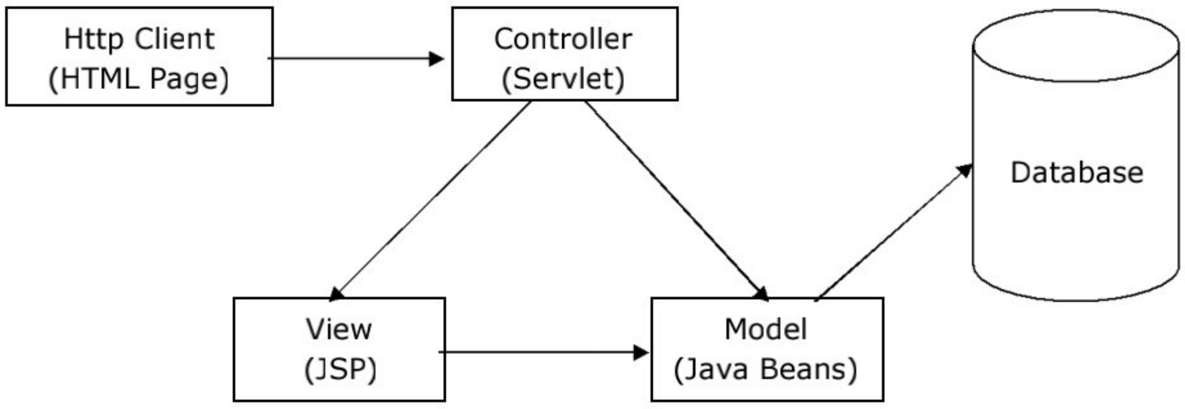
1. Define beans to represent the data of application.
2. Use a Servlet to handle requests from clients.

3. Populate the beans by placing result obtained from accessing business logic and data access code in the beans.
4. Store the bean either in the request, session or Servlet context (application).

Scanned with CamScanner

5. Forward the request to a JSP Page using forward method of RequestDispatcher interface or sendRedirect method of HttpServletResponse interface.

6. Extract data from the beans by accessing beans from JSP page using <jsp:useBean> and <jsp:getproperty> tags of JSP.



MVC Example in JSP

In this example, we are using Servlet as a controller, JSP as a view component, Java Bean class as a model.

In this example, we have created following pages:
index.jsp a page that gets input from the user.
ControllerServlet.java a Servlet that acts as a controller.
login-success.jsp and login-error.jsp files acts as view components.

Example: Login.html

```
<form action="ControllerServlet" method="post"> Name:<input
       type="text" name="name"><br> Password:<input
       type="password" name="password"><br> <input
       type="submit" value="login">
</form>
```

Example: Controller Servlet

Scanned with CamScanner

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

public class ControllerServlet extends HttpServlet
{
protected   void   doPost(HttpServletRequest   request,   HttpServletResponse
response)
                                        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out=response.getWriter();

        String name = request.getParameter("name"); String
        password = request.getParameter("password");

        LoginBean bean = new LoginBean();
        bean.setName(name);
        bean.setPassword(password);
        request.setAttribute("bean", bean);
        boolean status = bean.validate();

        RequestDispatcher rd = null;

        if(status) {
                rd = request.getRequestDispatcher("login-
                success.jsp"); rd.forward(request, response);
        }
        else {
                rd=request.getRequestDispatcher("login-
                error.jsp"); rd.forward(request, response);
        }
}

        protected void doGet(HttpServletRequest req, HttpServletResponse
                                    resp) throws ServletException, IOException
{
        doPost(req, resp);
}
}
```

File: LoginBean.java

```
public class LoginBean
{
```

Scanned with CamScanner

```
        private String name, password;

        public String getName() {
                return name;
        }

        public void setName(String name) {
                this.name = name;
        }

        public String getPassword() {
                return password;
        }

        public void setPassword(String password) {
                this.password = password;
        }

        public boolean validate()
        {
                if(password.equals("admin"))
                        return true;
                else
                        return false;
        }
}
```

File: login-success.jsp

```
<%@page import = "LoginBean" %>
<p>You are successfully logged in!</p>
<%
        LoginBean bean=(LoginBean)request.getAttribute("bean");
        out.print("Welcome, "+bean.getName());
%>
```

File: login-error.jsp

```
<p>Sorry! User Name or Password Error</p>
<%@ include file="Login.html" %>
```

Advantages of MVC

1. Faster and parallel development process.
2. MVC Application is device independent.
3. Easy to maintain the large application.

Scanned with CamScanner

4. Support for asynchronous technique of development.
5. Modification does not affect the entire model.
6. MVC model returns the data without formatting.
7. MVC makes the overall code much easier to maintain, test, debug, and reuse.

Disadvantages of MVC

1. Increased complexity and Need multiple programmers.
2. Inefficiency of data access in view.
3. Difficulty of using MVC with modern user interface.
4. Knowledge on multiple technologies is required.

Scanned with CamScanner