



На странице представлена информация по практическому заданию 10.

- Внимание!!!**  
 Данную практику Jenkins/Sonar собирать/проверять не будут.  
 Необходимо просто закоммитить исходный код в репозиторий.
- При коммите действуют ограничения на заливаемые файлы: class, jar файлы из коммита исключить!  
 В противном случае не сможете сделать коммит.

- [Задание 1](#)
- [Задание 2](#)
- [Задание 3](#)
- [Задание 4](#)
- [Задание 5](#)
- [Задание 6](#)
- [Пример получения DataSource](#)
- [Замечание \(Ant build-script\)](#)

## Задание 1

Создать JSP страницу, которая возвращает таблицу умножения, использовать скриптовые элементы (скриплеты, выражения).

	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

## Задание 2

Создать JSP страницу, которая возвращает таблицу умножения, использовать JSTL.

## Задание 3

Создать JSP страницу, которая содержит форму ввода имени. После сабмита формы введенное имя попадает в список, который хранится в сессии. Ответ сервера - исходная форма, ниже которой распечатано содержимое списка.

Дополнительно сделать гиперссылку **remove**, которая очищает список.

- Реализовать паттерн PRG:** по сабмиту формы на сервер уходит POST-запрос, который обрабатывает сервлет, после чего делает редирект на исходную JSP страницу.

## Задание 4

1. Создать схему базы данных из двух таблиц:

**users (login/password/role\_id)**


**roles (id/name)**

Создать базу данных, наполнить ее как минимум двумя пользователями с двумя разными ролями, например - admin и client.

2. Создать форму аутентификации (логин/пароль).

Если пользователь не аутентифицирован, то идет возврат обратно к форме аутентификации.

Если аутентифицирован, то клиент видит страницу из п.3

 Для получения соединения использовать DataSource, который конфигурировать в META-INF/context.xml проекта

3. Объединить все три задания (Задание 1, Задание 2, Задание-3) в одно: сделать так, чтобы:

справа вверху каждой из страниц было информативное сообщение: **you are logged as ROLE LOGIN**  
(пример: **you are logged as admin ivanov**)

слева вверху каждой из страниц было меню из соответствующих гиперссылок: **Part1 Part2 Part3**

## Задание 5

Добиться правильного отображения login в случае, когда он записан кириллицей.

## Задание 6

1. Добавить в таблицу users поле name.
2. Добавить меню четвертой гиперссылкой: **Part4**
3. По нажатию на Part4 появляется форма редактирования имени текущего пользователя.
4. Добиться правильной работы приложения в случае, когда имя записано кириллицей.

## Пример получения DataSource

[Click here to expand...](#)

Для получения соединения с помощью DataSource объекта, его вначале нужно сконфигурировать в файле /META-INF/context.xml:

**/META-INF/context.xml**

```
<Context>
  <Resource name="jdbc/testdb" auth="Container" type="javax.sql.DataSource"
    username="user"
    password="pass"
    driverClassName="com.mysql.jdbc.Driver"
    url="jdbc:mysql://localhost/testdb"
    maxActive="10"/>
</Context>
```

После этого объект DataSource можно получить следующим образом:

**com.my.MyServlet**

```

package com.my;
import java.io.*;
import java.sql.*;
import javax.naming.*;
import javax.servlet.*;
import javax.servlet.annotation.*;
import javax.servlet.http.*;
import javax.sql.*;
@WebServlet("/MyServlet")
public class MyServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        try {
            // obtain an environment naming context
            Context initCtx = new InitialContext();
            Context envCtx = (Context) initCtx.lookup("java:comp/env");
            // look up a data source
            DataSource ds = (DataSource) envCtx.lookup("jdbc/testdb");
            // obtain a connection from the pool
            Connection con = ds.getConnection();

            // log a connection to a console
            System.out.println(con);

            // send a connection to a client
            response.getWriter().println(con);

            con.close();
        } catch (NamingException | SQLException ex) {
            // log an exception to a console
            ex.printStackTrace();

            // send an exception to a client
            StringWriter sw = new StringWriter();
            PrintWriter pw = new PrintWriter(sw);
            ex.printStackTrace(pw);
            response.getWriter().println(sw);
        }
    }
}

```

**Замечание (Ant build-script)**

Click here to expand...

Для того, чтобы контейнер гарантированно делал трансляцию JSP страницы в сервлет, можно удалять соответствующие файлы в рабочем каталоге контейнера. Ниже ANT build-скрипт, который облегчает эту операцию при разработке в Eclipse IDE.

```

<project default="remove">
    <property name="dest.dir"
value="..\metadata\plugins\org.eclipse.wst.server.core\tmp0\work\Catalina\localhost\Test"/>

    <target name="remove">
        <delete>
            <fileset dir="${dest.dir}"/>
        </delete>
    </target>
</project>

```

Если один раз его выполнить, то далее можно ассоциировать с этим действием сочетание клавиш:

1. **Window - Preferences - Keys;**
2. выбрать **Run Last Launched External Tool.**