Clase grabada

Recibe una cálida:

¡Bienvenida!

Te estábamos esperando 😁

English Always | alkemy

Clase grabada

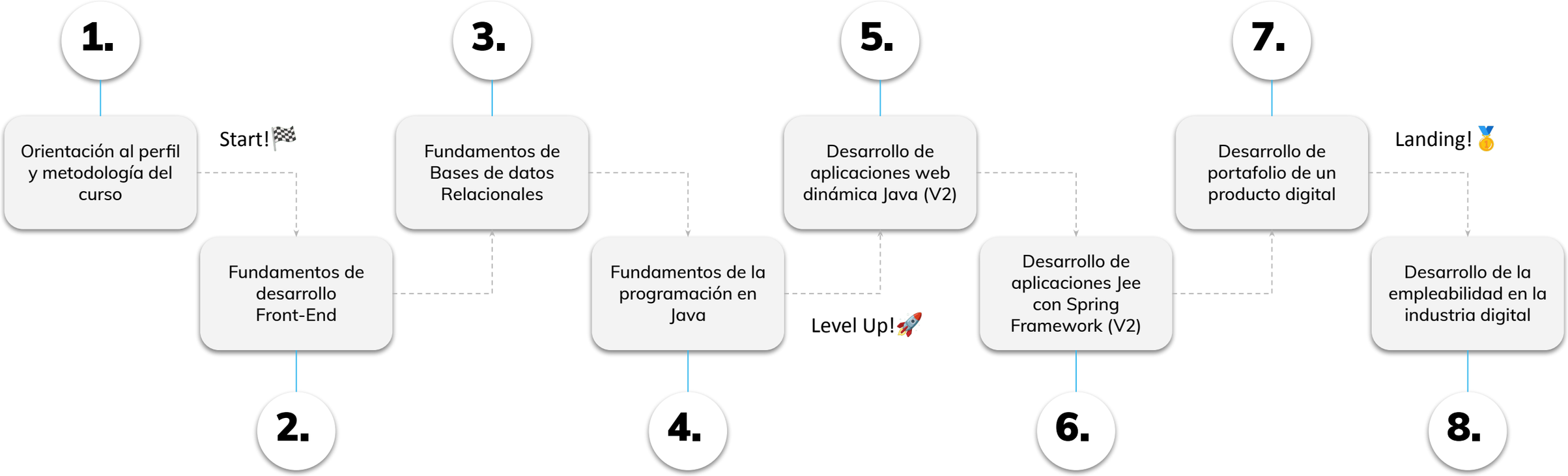
Conociendo los Servlets

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0

English Always | alkemy

HOJA DE RUTA

¿Cuáles **skill** conforman el programa?



English Always | alkemy

REPASO CLASE ANTERIOR

En la clase anterior trabajamos 📚:

- ✓ *Manejo de parámetros de POST request*

English Always | alkemy

LEARNING PATHWAY

¿Sobre qué temas trabajaremos?

5.3 Start!🏁

Conociendo los Servlets

Paso de parámetros de un Servlet hacia una vista JSP Implementando un formulario con validación en el servlet

- Envío de parámetros a JSP
  - Parametros a JSP
- Validación de formulario
  - Validando en el servlet

English Always | alkemy

OBJETIVOS DE APRENDIZAJE

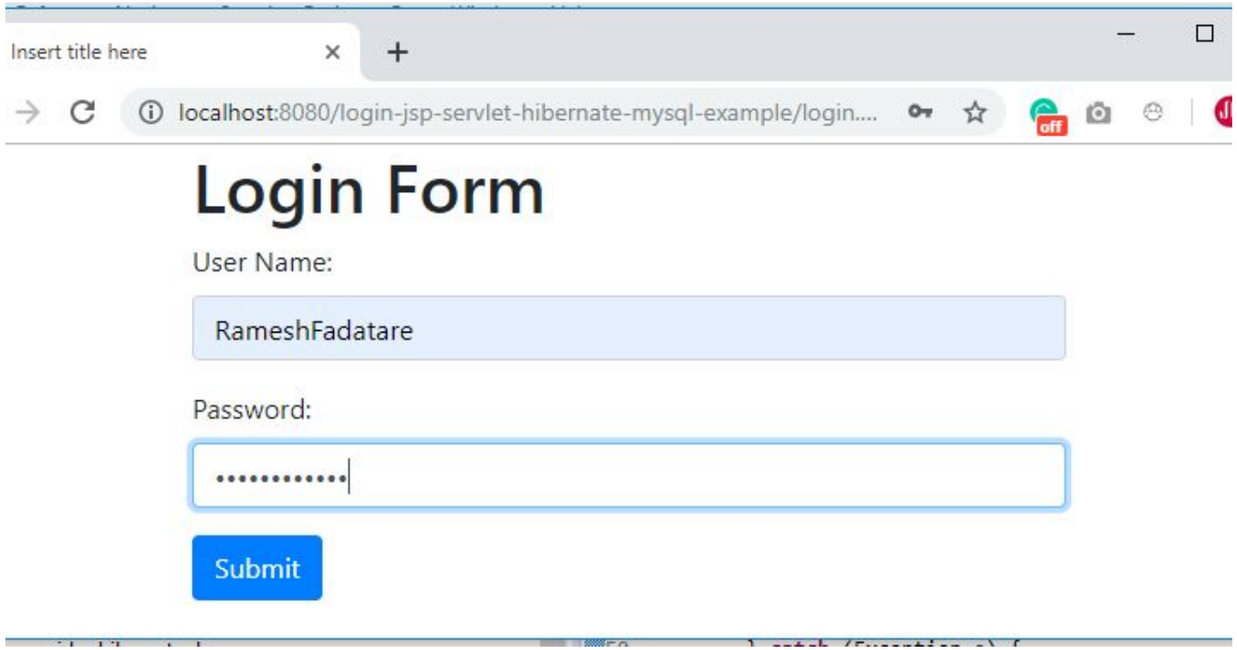
¿Qué aprenderemos?

- *Aprender a pasar parámetros desde un Servlet a una vista JSP*
- *Comprender cómo implementar un formulario con validación en el servlet*

Rompehielo🥶

¿Cómo crees que llegará la información enviada hasta el servlet?: 🙌

Existen varios métodos vistos anteriormente pero, ¿y si alguno de los datos ingresados es erróneo o está vacío? ¿Quién maneja este error?



Paso de parámetros de un Servlet hacia una vista JSP

Paso de parámetros de Servlet a JSP

Para pasar parámetros desde un servlet a una página JSP, puedes utilizar el objeto request o el objeto session. La elección entre uno u otro dependerá de si deseas que los parámetros sean visibles en la URL o si necesitas mantenerlos disponibles a lo largo de la sesión del usuario.

Cuando el cliente (navegador) solicita una URL, esta se envía al servidor. **En el servidor, se ejecutan archivos .jsp y .java**. Cuando, por ejemplo, un index.jsp llamar a un servlet(.java) estos se **comparten variables**, de forma que el resultado de una variable en el servlet, se le pueda mandar al jsp, y viceversa.

Paso de parámetros de Servlet a JSP

Page Scope
Son variables que solo se ven dentro de nuestro JSP o Servlet mientras se está ejecutando. Una vez construída y enviada la respuesta al navegador, desaparecen. Su única utilidad es ayudarnos a construir la página que queremos mostrar en el navegador.

```
<% int variable = 33; %>
...
<p>La variable vale <%= variable %></p>
```

y dentro de un Servlet no serían más que variables locales de nuestros métodos doGet() o doPost().

```
public class UnServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException {
        int variable = 33;
        ...
        resp.getWriter().print("<p> La variable vale " + variable + "</p>");
```

Paso de parámetros de Servlet a JSP

Request Scope

Estas variables **son más útiles**. Las guardamos en la **petición** que nuestro JSP/Servlet van a hacer a otro JSP/Servlet de nuestra aplicación, y este segundo JSP/Servlet puede recogerlos para hacer cosas con ellas.

Dentro de un JSP tenemos predefinido un objeto request, y en él podríamos fijar estas variables con setParameter() o con setAttribute(). y obtener los valores con los getParameter() o getAttribute().

Paso de parámetros de Servlet a JSP

Puedes agregar atributos al objeto request en el servlet y acceder a estos atributos en la página JSP utilizando la expresión ${}.

```java
public class MainServlet extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        // Obtener el valor del parámetro "nombre" de algún lugar (base de datos, etc.)
        String nombre = "Juan";

        // Agregar el parámetro "nombre" al objeto request
        request.setAttribute("nombre", nombre);

        // Redireccionar a la página JSP
        request.getRequestDispatcher("miPagina.jsp").forward(request, response);
    }
```

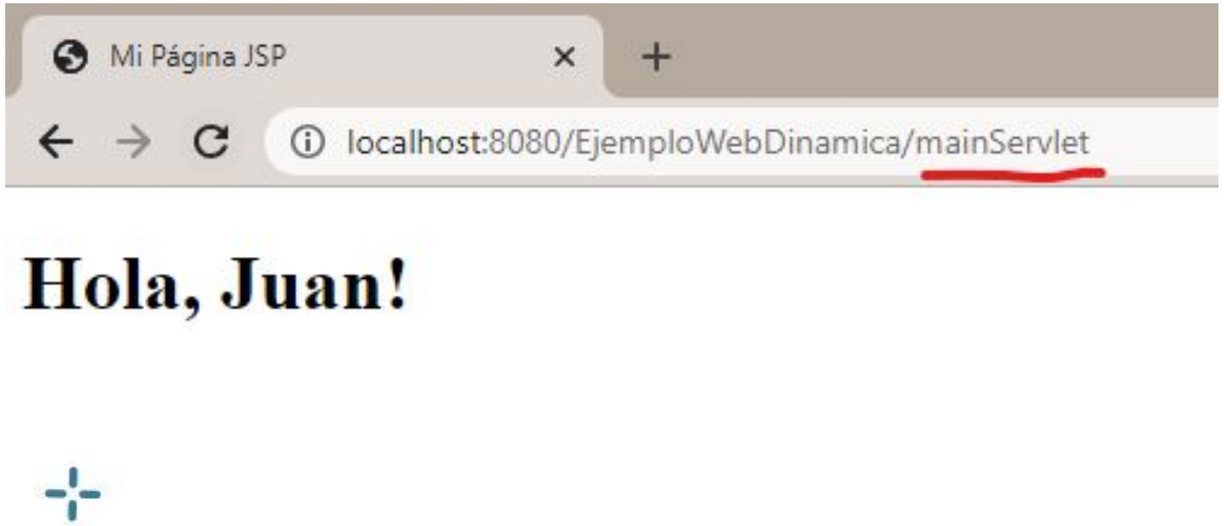
De esta manera en la página JSP puedes acceder al parámetro utilizando ${nombre}:

*miPagina.jsp

```jsp
<!DOCTYPE html>
<html>
<head>
    <title>Mi Página JSP</title>
</head>
<body>
    <h1>Hola, ${nombre}!</h1>
</body>
</html>
```

Paso de parámetros de Servlet a JSP

Con esta configuración, podrás acceder a la vista jsp directamente desde la url del servlet:



Paso de parámetros de Servlet a JSP

Session Scope

Estas variables se mantienen durante lo que se conoce como una sesión. Cuando un usuario visita nuestra aplicación por primera vez, **automáticamente se crea una sesión para ese usuario**. Esta sesión **suele estar abierta mientras el usuario va navegando por las páginas** de nuestra aplicación y desaparece cuando el usuario deja de navegar por nuestra aplicación durante un tiempo predeterminado.

Dicho de otra forma, **cualquier valor que guardemos en la sesión del usuario, sólo será visible por las páginas que visite ese usuario y mientras el usuario esté activo**. Son las variables típicas donde guardar si un usuario ha entrado en sesión con un usuario y password, su nombre, su carrito de la compra si nuestra aplicación es de comercio electrónico, etc.

Paso de parámetros de Servlet a JSP

Si necesitas mantener los parámetros disponibles durante toda la sesión del usuario, puedes utilizar el objeto session para ello. Los atributos del objeto session **estarán disponibles hasta que la sesión del usuario expire** o se invalide.

```
protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    // Obtener el valor del parámetro "nombre" de algún lugar (base de datos, etc.)
    String nombre = "Juan";

    // Obtener la sesión actual o crear una nueva
    HttpSession session = request.getSession(true);

    // Agregar el parámetro "nombre" al objeto session
    session.setAttribute("nombre", nombre);

    // Redireccionar a la página JSP
    response.sendRedirect("miPagina.jsp");
}
```

Paso de parámetros de Servlet a JSP

De esta manera en la página JSP (miPagina.jsp), puedes acceder al parámetro utilizando ${sessionScope.nombre}:

miPagina.jsp | MainServlet.java

```jsp
<!DOCTYPE html>
<html>
<head>
    <title>Mi Página JSP</title>
</head>
<body>
    <h1>Hola, ${sessionScope.nombre}!</h1>
</body>
</html>
```

Paso de parámetros de Servlet a JSP

En ambos casos (request y session), los parámetros pasados desde el servlet estarán disponibles en la página JSP y podrás mostrarlos o utilizarlos según tus necesidades.

Es importante recordar que **los atributos agregados al objeto request solo estarán disponibles durante la ejecución de la solicitud actual, mientras que los atributos agregados al objeto session estarán disponibles durante toda la sesión del usuario**.

LIVE CODING

Ejemplo en vivo

Compartiendo información:

Vamos a compartir información entre los servlets y las JSP:

1- Crear un formulario JSP llamado registro.jsp que pida al usuario un nombre, apellido, RUT y fecha de nacimiento (string)

2- Crear un Servlet que reciba esta información y devuelva un mensaje de bienvenida mostrando todos los datos del usuario en el navegador.

Tiempo: 20 minutos

Evaluación Integradora ✨

¿Listos para un nuevo desafío? En esta clase comenzamos a construir nuestro....

Trabajo Integrador del Módulo 💪🏽

Iremos completándolo progresivamente clase a clase.

English Always | alkemy

Ejercicio N° 1

Parametros a JSP

Parametros a JSP

Manos a la obra: 🙌

Crear un archivo JSP en el proyecto AlkeWallet para registro de usuario.

Consigna: ✍️

1- JSP llamado registro.jsp que pida al usuario un nombre, apellido, RUT, fecha de nacimiento (string) y contraseña.

2- Crear un Servlet que reciba esta información y devuelva un mensaje de bienvenida mostrando todos los datos del usuario en el navegador (excepto la contraseña).

Tiempo🕐: *20 minutos*

Implementando un formulario con validación en el servlet

Formulario

Implementando un formulario con validación en el servlet

Como hemos visto, el método que utilizamos para recuperar los datos de los parámetros es getParameter("X"). Esta recuperación puede generar resultados diferentes:

- Cuando el campo "X" **no existe**, **obtenemos el valor null.**
- Cuando el campo "X" existe, pero **no tiene valor**, **obtenemos el valor vacío (isEmpty)**.
- Cuando el campo "X" **existe y tiene valor**, **obtenemos la cadena** de caracteres que corresponde a dicho valor.

Es habitual requerir que determinados campos tengan valor (campos requeridos), y en muchos casos es deseable que los valores de los campos cumplan determinadas propiedades

Formulario

Implementando un formulario con validación en el servlet

Para asegurarnos de que los datos que se ingresaron en el formulario son correctos o aceptables, vamos a realizar la validación en el servlet.

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

    // Obtener los parámetros enviados desde el formulario
    String nombre = request.getParameter("nombre");
    String email = request.getParameter("email");

    // Realizar la validación

    if (nombre == null || nombre.trim().isEmpty() || email == null || email.trim().isEmpty()) {
        // Si hay campos vacíos, redirigir al usuario de vuelta al formulario con un
        // mensaje de error
        response.sendRedirect("registro.html?error=1");
    } else {
        // Si la validación es exitosa, redirigir al usuario a una página de éxito
        response.sendRedirect("exito.html");
    }
```

Formulario

Implementando un formulario con validación en el servlet

En la vista JSP debemos tener en cuenta marcar con el atributo required los inputs cuyos datos sean requeridos para procesar.

```
<!DOCTYPE html>
<html>
<head>
    <title>Formulario de Registro</title>
</head>
<body>
    <h1>Registro de Usuario</h1>
    <form action="registroServlet" method="post">
        <label for="nombre">Nombre:</label>
        <input type="text" name="nombre" id="nombre" required>
        <br>
        <label for="email">Email:</label>
        <input type="email" name="email" id="email" required>
        <br>
        <input type="submit" value="Registrarse">
    </form>
</body>
</html>
```

LIVE CODING

Ejemplo en vivo

Validando datos en el servlet:

Vamos a validar todos los datos ingresados en el formulario creado en el ejercicio anterior :

1- Validar los datos enviados por registro.jsp.

2- Crear un mensaje de error que dispare el Servlet en caso de no pasar la validación.

Tiempo: 15 minutos

Evaluación Integradora ✨

¿Listos para un nuevo desafío? En esta clase comenzamos a construir nuestro....

Trabajo Integrador del Módulo 💪🏽

Iremos completándolo progresivamente clase a clase.

English Always | alkemy

Ejercicio N° 1

Validando en el servlet

Parametros a JSP

Manos a la obra: 🙌

Vamos a validar todos los datos ingresados en el formulario creado en el ejercicio anterior.

Consigna: ✍️

1- Validar los datos enviados por registro.jsp.

2- Crear un mensaje de error que dispare el Servlet en caso de no pasar la validación.

Tiempo🕛: *15 minutos*

¿Alguna consulta?

RESUMEN

¿Qué logramos en esta clase?

- ✓ *Comprender cómo enviar información de un Servlet a una JSP*
- ✓ *Aprender a realizar validaciones en el servlet*

English Always | alkemy

#WorkingTime

Continuemos ejercitando

¡Antes de cerrar la clase! Te invitamos a: 👇👇👇

1. Repasar nuevamente la grabación de esta clase
2. Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
   a. *Lectura Modulo 5, Lección 3: páginas 16 - 19*
3. Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.

¡Muchas Gracias!

Nos vemos en la próxima clase 👋🏻

English Always | alkemy

Momento:

Time-out!

⏰5 min.