Capítulo 3

Aplicación WEB

Aplicación web

- Aplicaciones web modernas
  - Muchas componentes trabajan juntas para cumplir una meta
    - Modelos, controladores, vistas
    - Como se pueden unir las componentes?
    - Como se comparte información entre ellas?
    - Como se protege la información?
    - Como se asegura la información para los threads?

Aplicación web

- Imagine que se desea imprimir la dirección de email desde un servlet:

```
PrintWriter out = response.getWriter();
out.println("soporte@empresa123.cl");
```

  - Si cambia el email:
    - Modificar el código del servlet
    - Volver a compilar, copiar al contenedor y recargar la aplicación
  - Lo mejor es utilizar el deployment descriptor para mantener ese dato
    - No es necesario modificar el código fuente

Aplicación web

- Parámetros Init
  - Los servlets pueden tener parámetros de inicialización
  - Se deben definir en el deployment descriptor (web.xml)

```
<init-param>
        <param-name>email-soporte</param-name>
        <param-value>soporte@empresa123.cl</param-value>
</init-param>
```

  - Y en el servlet:

```
out.println(getServletConfig().getInitParameter("email-soporte"));
```

    - Todos los Servlet heredan el método getServletConfig()

Aplicación web

- Los parámetros “init” no se pueden utilizar hasta que el servlet esté inicializado
  - No se puede obtener ServletConfig en el constructor del servlet
  - Cuando el contenedor inicializa el servlet, genera un único ServletConfig para el servlet
    - El contenedor lee los parámetros init desde el deployment descriptor
    - Para cada parámetro crea un par nombre-valor y entrega la referencia a ServletConfig
    - Luego crea una nueva instancia de servlet
    - Luego entrega ServletConfig al método init
  - El contenedor lee una vez los parámetros desde DD
    - Se debe recargar la aplicación cuando cambien los valores

Aplicación web

- Parámetros de inicialización de Context
  - Están disponibles para toda la aplicación web

```
<context-param>
        <param-name>email-soporte</param-name>
        <param-value>soporte@empresa123.cl</param-value>
</context-param>
```

  - Como son para toda la aplicación no se deben ubicar dentro de los tag <servlet>
    - Pero si deben estar dentro del tag <web-app>
  - En el servlet:

```
out.println(getServletContext().getInitParameter("email-soporte"));
```

    - También se pueden acceder desde los JSP

Aplicación web

- Con ServletConfig y ServletContext sólo podemos trabajar con Strings
  - ¿Como se puede hacer que toda la aplicación tenga acceso a una conexión compartida a una BD?
    - Se puede colocar el nombre de búsqueda en el contexto
    - Pero, ¿quien hace el trabajo de llevar ese String a un Datasource?
  - ¿Como se podrá inicializar una aplicación web con un objeto?
    - Se necesita que un método se ejecute antes de cualquier servlet o JSP
    - Es necesario escuchar el evento de inicialización del contexto y tomar los parámetros de inicialización
    - Luego, ejecutar algún código antes de atender al cliente

Aplicación web

- ServletContextListener
  - Una clase aislada, no es un Servlet
  - Escucha dos eventos en la vida de ServletContext
    - Inicialización y destrucción
  - Implementa javax.servlet.ServletContextListener
  - Este objeto será notificado cuando el contexto es inicializado
    - Obtiene los parámetros desde ServletContext
    - Usa el parámetro de búsqueda del nombre de la base de datos para hacer la conexión
    - Almacena la conexión a la base de datos como un atributo, que todas las componentes de la aplicación pueden acceder
  - Cuando el contexto es destruido
    - Cierra la conexión a la base de datos