**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| PROGRAMA DE FORMACIÓN | Análisis y desarrollo de software |
| --- | --- |

| COMPETENCIA | 220501096 – Desarrollar la solución de software de acuerdo con el diseño y metodologías de desarrollo. | RESULTADOS DE APRENDIZAJE | 220501096-04 Codificar el software de acuerdo con el diseño establecido. |
| --- | --- | --- | --- |

| NÚMERO DEL COMPONENTE FORMATIVO | 30 |
| --- | --- |
| NOMBRE DEL COMPONENTE FORMATIVO | Construcción de aplicaciones con JAVA |
| BREVE DESCRIPCIÓN | Mediante el estudio de este componente, el aprendiz estará capacitado para procesar y aplicar conceptos sobre lenguajes de programación, formularios HTML, elementos de JSP, Java, programación orientada a objetos, entre otros. En el mismo sentido, potenciará sus habilidades para la construcción de aplicaciones con Java y construcción de aplicaciones web básicas. |
| PALABRAS CLAVE | Java, Java Web, JDBC, JSP, *Servlets*. |

| ÁREA OCUPACIONAL | 2 - CIENCIAS NATURALES, APLICADAS Y RELACIONADAS |
| --- | --- |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**

**Introducción**

* 1. **Java con JDBC**
  2. Clases e interfaces JAVA
  3. Conexión a base de datos desde aplicación Java
  4. **Servlets y JSP**
  5. Introducción a servlets
  6. Formularios HTML con Servlets
  7. HTTP Session
  8. Elementos de JSP
  9. MVC

1. **DESARROLLO DE CONTENIDOS:**

**Introducción**

Se da la bienvenida al estudio de este componente del tecnólogo en análisis y desarrollo de software; para comenzar, es necesario explorar el recurso que se muestra a continuación:

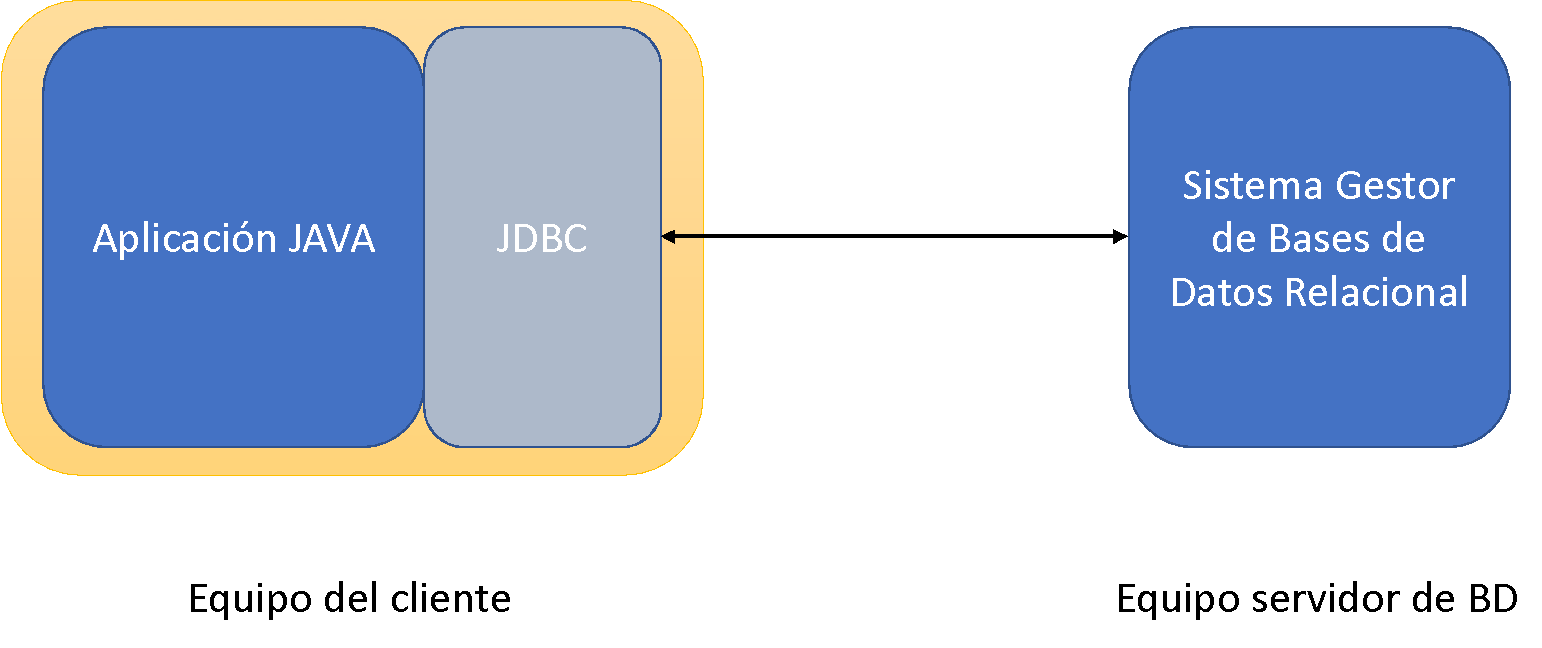


1. **Java con JDBC**

Cuando se habla de JDBC (*Java Data Base Connectivity*) se hace referencia a un *Driver* que encapsula un conjunto de clases e interfaces escritas de JAVA, creadas para establecer conexiones, enviar sentencias SQL y procesar los datos resultantes sobre bases de datos relacionales, ver figura 1.

**Figura 1**

*Java con JDBC*



En términos más generales, JDBC es uno de los puentes que puede ser usado en JAVA para comunicar una aplicación JAVA con un servidor de bases de datos relacionales.

Trabajar conexiones a bases de datos con JDBC implica realizar, cuidadosamente, acciones como las que se enuncian a continuación, ver tabla 1:

* Manejar manualmente las conexiones y desconexiones a la base de datos.
* Ser cuidadosos en la escritura de sentencias SQL a ser ejecutadas en el servidor.
* Conocer las relaciones existentes en el modelo de base de datos a manipular.
* Conocer las características de la base de datos a manipular ya que esto podría afectar las consultas.

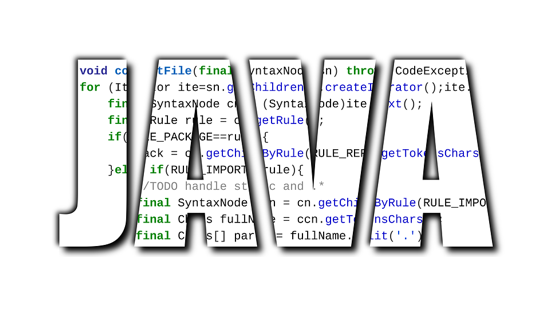
 

**Tabla 1**

*Drivers JDBC para SGBD*

| Base de Datos | Página oficial para descarga de JDBC |
| --- | --- |
| Mysql | <https://dev.mysql.com/downloads/connector/j/> |
| SQL Server | <https://docs.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver15> |
| Oracle | <https://www.oracle.com/database/technologies/jdbc-upc-downloads.html> |
| PostgreSQL | <https://jdbc.postgresql.org/download.html> |
| DB2 | <https://www.ibm.com/support/pages/db2-jdbc-driver-versions-and-downloads> |

* 1. **Clases e interfaces JAVA**



Además de tener a disposición el Driver JDBC correspondiente, para la base de datos que se desea conectar, se requiere de un conjunto adicional de **clases e interfaces Java** para hacer uso del *driver*, para lo cual es necesario importar los paquetes *java.sql* o javax.sql.

A continuación, se describen algunas de las clases más importantes dentro de estas librerías, y otros elementos, para gestionar correctamente una base de datos por medio de JDBC:





* 1. **Conexión a base de datos desde aplicación Java**

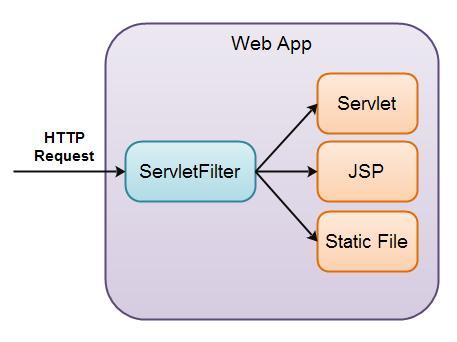
Garantizar la conexión a una base de datos requiere la aplicación concreta de los pasos ya explicados. En este punto, se debe tomar nota de ellos y registrar en su libreta personal de apuntes, las acciones que permiten cumplirlos (explicados en el anexo 1).

Los pasos que deben realizarse para garantizar una conexión a base de datos desde una aplicación Java, utilizando JDBC, son:

* Descarga del driver JDBC (Si no se tiene).
* Importar el Driver descargado en las librerías del proyecto JAVA.
* Importar el paquete java.sql.\*;
* Inicializar el Driver usando la sentencia Class.forName
* Crear un Objeto de tipo Connection por medio del DriverManager
* Crear un objeto Statement que permite configurar las sentencias SQL desde el Objeto de Connection creado en el punto anterior.
* Ejecutar la sentencia SQL usando cualquiera de los métodos del objeto Statement.
* Procesar los resultados de la sentencia ejecutada por medio de un ResultSet y sus métodos de apoyo.



1. **Servlets y JSP**



Los *Servlets* y los JSP (*Java Server Pages*), son tecnologías que se utilizan para realizar ejecuciones en servidores Java, permitiendo extender los servicios de un servidor *web* (Berzal, 2007).

Un ejemplo sencillo de lo que se puede hacer con JSP y Servlet es que por medio de una página JSP se consigue hacer un formulario que contenga una serie de datos como: primer nombre, segundo nombre, edad, correo y, a través de un *servlet*, realizar el envío de estos registros para que sean almacenados en una base de datos y que, finalmente, el usuario pueda visualizar esta información.

A continuación se pueden conocer los procesos principales de estas tecnologías y algunos aspectos generales del uso de JSP y *Servlet*:



Hacer uso de los enlaces que se le ofrecen a continuación, para profundizar en lo relacionado con tecnologías Servlet y JSP:

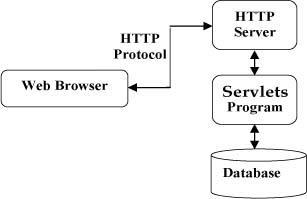


* 1. **Introducción a *servlets***

*Servlet* es un programa que se lleva a cabo en una aplicación o servidor Web; se emplea como capa que intercede entre la base de datos y el servidor de peticiones HTTP, por medio de un navegador Web o cualquier otro cliente HTTP (Gómez, 2017).

**Figura 2**

*Arquitectura de un servlet*



A continuación, se pueden conocer algunas ventajas de utilizar *servlet*:

* Rendimiento: a diferencia de CGI (***Common Gateway Interface)*** el rendimiento de un *servlet* es mucho mejor.
* No necesita procesos independientes para operar las solicitudes del cliente .
* Ya que los *servlet* están escritos en Java los hace una plataforma independiente.
* Creíble: resguarda los recursos del servidor debido a el servidor de seguridad de java.

Las tareas principales de un servlet son:

****

**Ciclo de vida de un Servlet**

A diferencia de las clases en Java, que inician con un método ***main (),*** en los *servlets* son “llamados” unos métodos, al recibir peticiones. A esta metodología se le llama ciclo de vida de un *servlet* y se compone por tres métodos: **init**, **service**, **destroy**.

**Figura 3**

*Ciclo de vida de un Servlets*

Diagrama

Descripción generada automáticamente

Se puede explorar el recurso que se muestra a continuación para conocer el ciclo de vida de un *Servlet*; procurar llevar registro, en la libreta personal de apuntes, de los aspectos más importantes y destacados.



**Ejemplo de *Servlet***

El siguiente código muestra un **Hola Mundo** en un S*ervlet*, con sus métodos de ciclo de vida.

1. public class HolaMundo extends HttpServlet {

2. private String mensaje;

**3.** public void init () throws ServletException

4. { // inicia la asignación de mensaje hola mundo

5. mensaje = “Hola Mundo”;

}

6. public void doGet (HttpServletRequest request,HttpServletResponse response) throws ServletException, IOException

{ // Se especifica el formato de la respuesta utilizando método getWriter

7. salida = response.getWriter ();

// Genera respuesta de la petición

8. salida.println("<h1>" + mensaje + "</ h1>");}

9.public void destroy ()

10. {.}

En el código anterior se hace lo siguiente:

* Se crea la clase **HolaMundo** que hereda de ***HttpServlet*** (línea 1).
* La clase ***HttpServlet*** contiene el método ***doGet*** que se encarga de recibir la petición del usuario que realiza desde el navegador.
* El método **doGet** recibe dos argumentos **request** (para procesar la petición) y **response** (para dar respuesta a la petición).
* Se sobrescribe el método ***doGet*** (línea 6) para crear la lógica de respuesta del usuario, se establece el formato de la respuesta del usuario en formato **html** y se crea el contenido de la respuesta.



* 1. **Formularios HTML con *Servlets***



Las páginas web contienen formularios *html* para la solicitud de información al usuario. El proceso de recolección de datos y respuesta, de este formulario, se debe hacer por medio de un programa. Esto lo hace un *servlet* que recoge las peticiones de un formulario y las procesa.

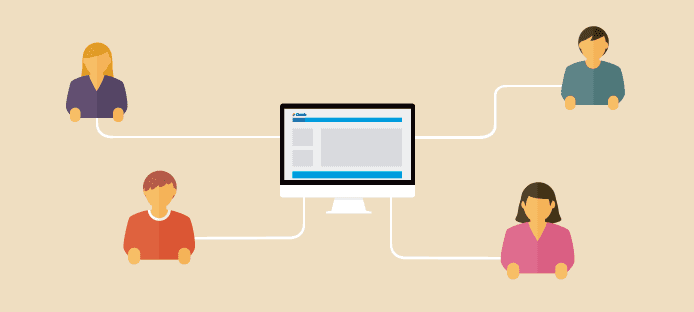
La declaración de un formulario HTML se hace por medio de la etiqueta ***<form>***, donde es primordial que se especifique el método por el que se realizará el envío de los datos, los cuales pueden ser *get* o *post*.

A continuación, se presentan los aspectos más relevantes, en lo relativo a formularios html con servlets:



****

* 1. **HTTP Session**



Las sesiones son creadas para hacer un seguimiento y para memorizar las acciones de un usuario en un sitio web. Las ***java session*** permiten guardar información entre las diversas peticiones HTTP, siendo una clase que se utiliza para realizar seguimiento a las sesiones; son objetos almacenados también del lado del servidor.

Para poder hacer uso de este objeto, se requiere obtener un objeto de la sesión, para poder escribirlo, eliminarlo o leerlo, una vez cerrada la sesión.

Los métodos más utilizados de *HTTPSession* son:

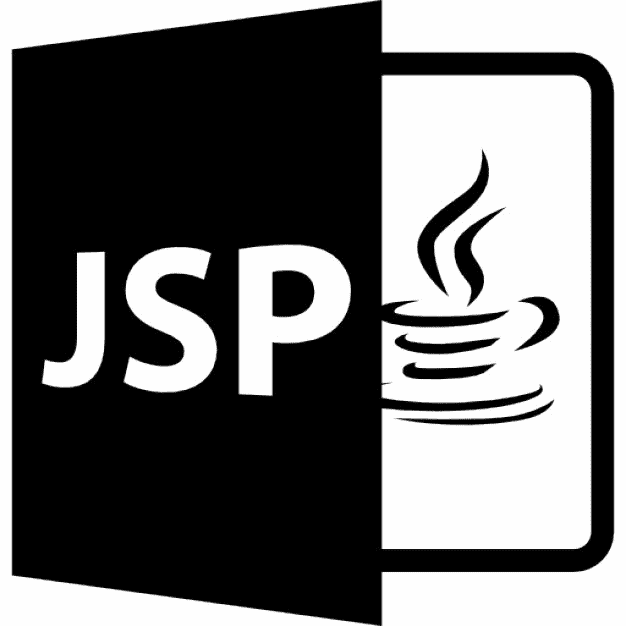
* **getId ()**: retorna el identificador que es asignado a la sesión, se utiliza para la identificación de la sesión.
* **isNew ()**, retorna valor booleano true en caso de que el cliente no tenga una sesión.
* **setAttribute ()**, establece un objeto a la sesión, empleando un nombre explícito.
* **getAttribute ()**, retorna el objeto que fue asignado a la sesión, el cual se identifica con el nombre que le fue asignado.
* **setInactiveInterval ()**, detalla el mayor tiempo que exista entre las peticiones continuas del cliente.
* **invalidate ()**, elimina la sesión actual.



A continuación, se puede conocer el funcionamiento básico de las sesiones; se debe recordar llevar registro de los aspectos más relevantes y destacados en la libreta personal de apuntes:



* 1. **Elementos de JSP**



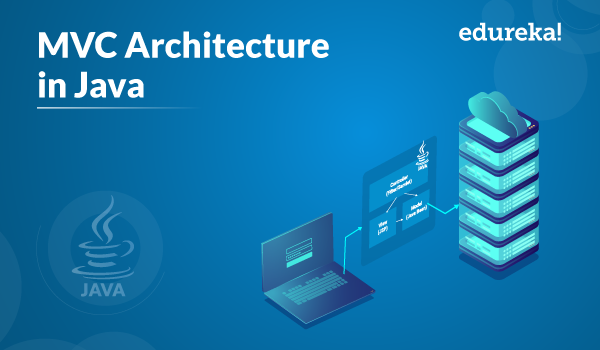
En JSP se encuentran tres tipos de elementos que pueden ser insertados en una página web: el código, las directivas y las acciones. Cada uno de ellos favorece las distintas ejecuciones que se requieren en la construcción de aplicaciones con JAVA.

A continuación, descubra todos los aspectos importantes y destacados de JSP:



****

* 1. **MVC**



El MVC (Modelo-Vista-Controlador) es un patrón de diseño estándar que se usa a la hora de crear aplicaciones dinámicas con **Java**. Consiste en separar la parte lógica de los proyectos de la parte visual y lo hace integrando *Servlets* y *JSP* donde los S*ervlets* se encargan de la parte lógica y los JSP de la parte visual.

**Figura 4**

*Arquitectura MVC*



Imagen en blanco y negro

Descripción generada automáticamente con confianza baja









1. **SÍNTESIS**

En este punto de finalizar el recorrido por el componente 30 del tecnólogo en **Análisis y desarrollo de software**. Se invita a explorar el último recurso didáctico, que mostrará una síntesis del mismo y refuerza en los aspectos más importantes.



1. **ACTIVIDADES DIDÁCTICAS (OPCIONALES SI SON SUGERIDAS)**

| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| --- | --- |
| Nombre de la Actividad | Aplicaciones con Java |
| Objetivo de la actividad | Afianzar las características más importantes asociados con JDBS, JSP, servlets. |
| Tipo de actividad sugerida | Completar espacios en blanco |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | Anexo documento en Word llamado Actividad didáctica 1 |

1. **MATERIAL COMPLEMENTARIO:**

| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| --- | --- | --- | --- |
| N/A |  |  |  |
| N/A |  |  |  |
| N/A |  |  |  |

1. **GLOSARIO:**

| TÉRMINO | SIGNIFICADO |
| --- | --- |
| CGI: | Interfaz de entrada común, se refiere a un sistema de comunicación que le indica al servidor web como enviar y recibir datos de una aplicación de servidor a un cliente. |
| Frameworks: | Marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, una especie de plantilla. |
| Html: | HyperText Markup Language Lenguaje de Marcas de Hipertexto. |
| JDK: | Kit de Desarrollo de Java permite. |
| URL: | Sus siglas (Localizador uniforme de recursos) facilita la recuperación de información determinada en internet se escribe en el navegador para ingresar a una página determinada. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Berzal, F., Cubero, J., y Cortijo, F. (2007). *Desarrollo profesional de aplicaciones web con Asp. net*. <https://books.google.es/books?hl=es&lr=&id=J1d_9l6zlAIC&oi=fnd&pg=PA3&dq=servlests+y+jsp+fernando+berzal&ots=GsR9oMYn8M&sig=YaCnuz6ed8E_kEo_G1CPHH-KjjU>

Gallagher, J. (4 de mayo de 2021). Los lenguajes de programación más populares *Career Karma.* <https://careerkarma.com/blog/top-programming-languages-2021/>

Gómez, M., & Cervantes, J. (2017). Introducción a la Programación Web con Java: JSP y Servlets, JavaServer Faces.

Statistics Times (2021). *Principales idiomas informáticos.* STATISTICSTIMES. <https://statisticstimes.com/tech/top-computer-languages.php>

1. **CONTROL DEL DOCUMENTO**

|  | Nombre | Cargo | Dependencia | | Fecha |
| --- | --- | --- | --- | --- | --- |
| Autor (es) | Jonathan Guerrero Astaiza | Experto temático | Centro de Teleinformática y Producción Industrial | Octubre de 2021 | |
| Zulema León Escobar | Experta temática | Centro de Teleinformática y Producción Industrial | Octubre de 2021 | |
| Fabián Leonardo Correa Díaz | Diseñador Instruccional | Centro agropecuario La Granja, Regional Tolima | Octubre de 2021 | |
| Ana Catalina Córdoba Sus | Revisora metodológica y pedagógica | Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica. | Noviembre 2021 | |
| Rafael Neftalí Lizcano Reyes | Asesor pedagógico | Regional Santander - Centro Industrial del Diseño y la Manufactura. | Noviembre 2021 | |
| Jhon Jairo Rodríguez Pérez | Diseñador y evaluador instruccional | Regional Distrito Capital - Centro para la Industria de la Comunicación Gráfica | Noviembre 2021 | |

1. **CONTROL DE CAMBIOS**

|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| --- | --- | --- | --- | --- | --- |
| Autor (es) |  |  |  |  |  |