4. Desarrollo de aplicaciones web utilizando código embebido

04 Octubre 2021

Educar en la Verdad
para ser libres

Necesidades del Módulo 4: Desarrollo de aplicaciones web utilizando código embebido

4.1. Mantenimiento del estado

4.2. Sesiones

4.3. Cookies

4.4. Seguridad: usuarios, perfiles, roles

4.5. Autenticación de usuarios

4.6. Herramientas de programación

4.7. Pruebas y depuración

Educar en la Verdad para ser libres

0. PROGRAMACIÓN BASADA EN LENGUAJES DE MARCAS CON CÓDIGO EMBEBIDO

En esta unidad vamos a estudiar las principales características que debemos tener en cuenta a la hora de desarrollar una aplicación web utilizando código embebido.

Empezaremos por estudiar el concepto de estado, ver los diferentes estados que podemos tener y darnos cuenta de que el concepto de estado empieza a cobrar importancia desde la aparición de los objetos en el mundo de la programación.

Veremos el concepto de sesiones, un recurso que permite establecer un vínculo entre el navegador y el servidor durante la navegación de una página web. Las sesiones tienen muchos usos pero son utilizadas, sobre todo, para mantener autenticados a los usuarios durante toda la navegación, de manera que solo tengan que identificarse la primera vez que entran

Educar en la Verdad para ser libres

4.1 Mantenimiento del estado

Para entender el concepto de estado y mantenimiento del estado debemos primero entender cómo se comportan los ordenadores cuando se comunican entre ellos.

Cuando un usuario abre su navegador y escribe la dirección de una página web se realiza una conexión con ese servidor web. Se busca la página y se devuelve al usuario.

Este es el caso “normal” de una conexión donde el servidor web, una vez ha servido la página, se “olvida” de quién se la ha pedido porque es una información que ya no necesita.

Pero en caso de que el servidor necesite recordar quién ha sido el cliente, entonces necesitamos habilitar este nuevo mecanismo de mantenimiento de estado.

El cliente no necesita saber nada de conexiones, ni estados, ni saber qué se debe recordar; para eso ya está el servidor.

4.1 Mantenimiento del estado

Todos los servidores web o servidores de aplicaciones disponen de una compleja estructura de objetos que podemos clasificar, de entre varios modos diferentes (Figura 4.1), como:

- **Utilizables por las páginas web.** Estos son los objetos que tienen que ver con el mantenimiento del estado y las sesiones.

- **No utilizables por las páginas web.** Estos son objetos de uso interno del servidor web y que por supuesto también se mantienen, pero su función es otra.

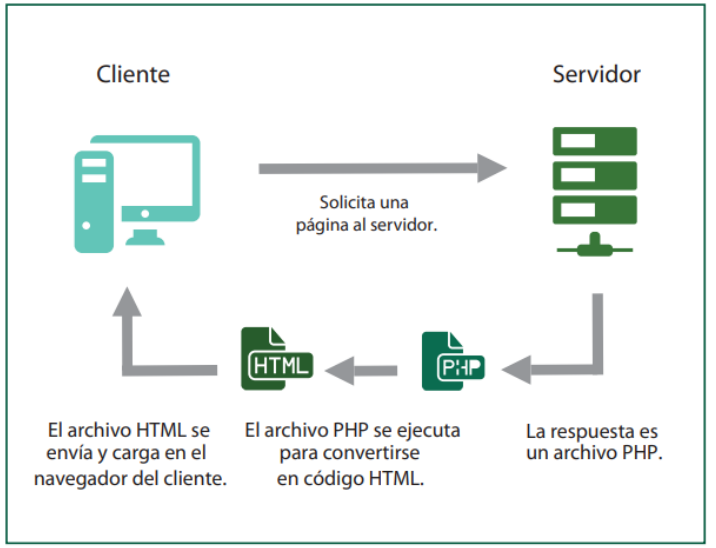

Figura 4.1
Proceso de comunicación entre página PHP y código HTML.

4.2 Sesiones

Una sesión es un conjunto de objetos que están en un estado concreto.

El mantenimiento del estado es responsabilidad del servidor web o del servidor de

aplicaciones pero cada lenguaje de programación habilita ciertas características

diferentes.

Este mantenimiento del estado puede ser:

- **Automático**. Es una característica habitual en los servidores IIS. De forma automática el servidor gestiona el mantenimiento del estado.

- **Bajo demanda.** PHP y JSP así lo utilizan. Debe pedirse el mantenimiento del estado para todas las páginas que lo necesiten.

4.2 Sesiones

La llamada a la función de creación de sesión será:

- **PHP:** <?php session_start() ?>

- **JSP:** <%@page session=”true”%>

Mantener el estado significa almacenar la información relativa a esa sesión. Para eso es necesario también saber de qué sesión se trata, para lo cual se utiliza el identificador de sesión (Session Identifier).

Esta información se almacena en unas variables globales a todas las páginas web. También se suelen llamar variables superglobales.

4.2 Sesiones

Para almacenar la información:

- PHP: $_SESSION[“nombre_de_la_variable”] = valor;
- JSP: session.setAtribute(“nombre_de_la_variable”, valor);
- ASP: Session(“nombre_de_la_variable “) = valor

Para recuperar la información:

- PHP: variable = $_SESSION[“nombre_de_la_variable”];
- JSP: variable = session.getAtribute(“nombre_de_la_variable”);
- ASP: variable = Session(“nombre_de_la_variable “)

4.2 Sesiones

A continuación veremos un ejemplo de código en PHP (Figura 4.2).

```
<!DOCTYPE >
<?
  session_start();
?>
<html>

<body>

<?php
session_name("Sesion");
echo "usuario:".$_SESSION["usuario"];
echo "</br>";
echo "pag".$_SESSION["pag"];
echo "</br>";
echo "status:".$_SESSION["status"];
echo "</br>";

$_SESSION["status"] ="";
$_SESSION["pag"] = "";
$_SESSION["usuario"]="";

session_destroy();
?>


</body>

</html>
```

Figura 4.2
Ejemplo de código PHP para gestionar sesiones en PHP.

4.2 Sesiones

El identificador de sesión es un valor único para cada sesión que la identifica. Además este identificador de sesión está asociado al cliente web que está realizando las peticiones.

La equivalencia entre cliente web e identificador de sesión no es segura. Esto significa que un atacante puede escuchar la conversación entre el cliente web y el servidor, capturar el identificador de sesión y utilizarlo fraudulentamente como si fuese ese otro cliente web.

Aunque esta posibilidad existe es extremadamente difícil que se pueda producir por varios motivos:

- Tiempo. El tiempo necesario para realizar la búsqueda del identificador de sesión es muy grande puesto que hay que encontrar el paquete transmitido en cuestión entre millones de otros paquetes.

4.2 Sesiones

● Conveniencia. Aunque se encontrara el identificador de sesión para un cierto cliente sobre una cierta página web, el atacante debería comprobar la conveniencia de su uso puesto que es muy posible que "no le interese atacar" a esa página web.

● Plazo. En el caso de que se haya encontrado el identificador de sesión y se haya verificado que la página web es la que se desea atacar, el plazo de validez del identificador de sesión puede terminarse, puesto que las sesiones se cierran cuando ya ha finalizado la comunicación entre el cliente web y el servidor, y por tanto ese identificador de sesión que se ha encontrado ya no será válido.

Recuerda

Una sesión se almacena en el servidor y guarda temporalmente datos relativos a una conexión de usuario.

Educar en la Verdad para ser libres

4.3 Cookies

Para saber cuál es nuestra sesión utilizamos el identificador de sesión, pero este valor debe almacenarlo el cliente porque es una información necesaria para que el navegador del cliente pueda comunicarse correctamente con el servidor.

Esta información y algunas otras se almacenan en el cliente mediante cookies(galletas). Una cookie es un pequeño fichero de texto que contiene una serie de entradas atributo - valor al que también se le asocia a una fecha de caducidad.

Lo más interesante de las cookies es que pueden ser generadas en el cliente a partir de las respuestas del servidor, pudiendo el usuario ser totalmente ajeno a que se están usando cookies.

Una de las informaciones importantes que se almacenan en las cookies es justamente el identificador de sesión. Este valor, una vez almacenado en las cookies, se utiliza para las posteriores llamadas hacia el servidor.

4.3 Cookies

Cuando el servidor requiere que se almacenen datos en las cookies, se lo indica al cliente enviándole una instrucción especial para realizar el almacenamiento de la información en una cookie y a partir de ese momento se van a enviar los datos de las cookies de cliente al servidor en las sucesivas llamadas de navegador a ese servidor.

Las cookies tienen una característica importante, que es la fecha de caducidad. Cuando se crea la cookie se hace indicándole una fecha de caducidad.

Es caso de no establecerse fecha de caducidad para la cookie, ésta se elimina al finalizarse la comunicación entre cliente y servidor.

Los navegadores actuales tienen parámetros para activar o desactivar la aceptación de cookies, y por motivos de seguridad solamente deben aceptarse las cookies que sean gestionadas desde sitios conocidos y de confianza, puesto que en ciertos sitios maliciosos se utilizan cookies para recopilar información del usuario.

4.3 Cookies

Cuando el servidor requiere que se almacenen datos en las cookies, se lo indica al cliente enviándole una instrucción especial para realizar el almacenamiento de la información en una cookie y a partir de ese momento se van a enviar los datos de las cookies de cliente al servidor en las sucesivas llamadas de navegador a ese servidor.

Las cookies tienen una característica importante, que es la fecha de caducidad. Cuando se crea la cookie se hace indicándole una fecha de caducidad.

Es caso de no establecerse fecha de caducidad para la cookie, ésta se elimina al finalizarse la comunicación entre cliente y servidor.

Los navegadores actuales tienen parámetros para activar o desactivar la aceptación de cookies, y por motivos de seguridad solamente deben aceptarse las cookies que sean gestionadas desde sitios conocidos y de confianza, puesto que en ciertos sitios maliciosos se utilizan cookies para recopilar información del usuario.

4.3 Cookies

Es conveniente, de todos modos, revisar la configuración de las cookies dentro de nuestro navegador para que las acepte, las bloquee o que pregunte en los casos en que se solicite el almacenamiento de una cookie (Figura 4.3)..

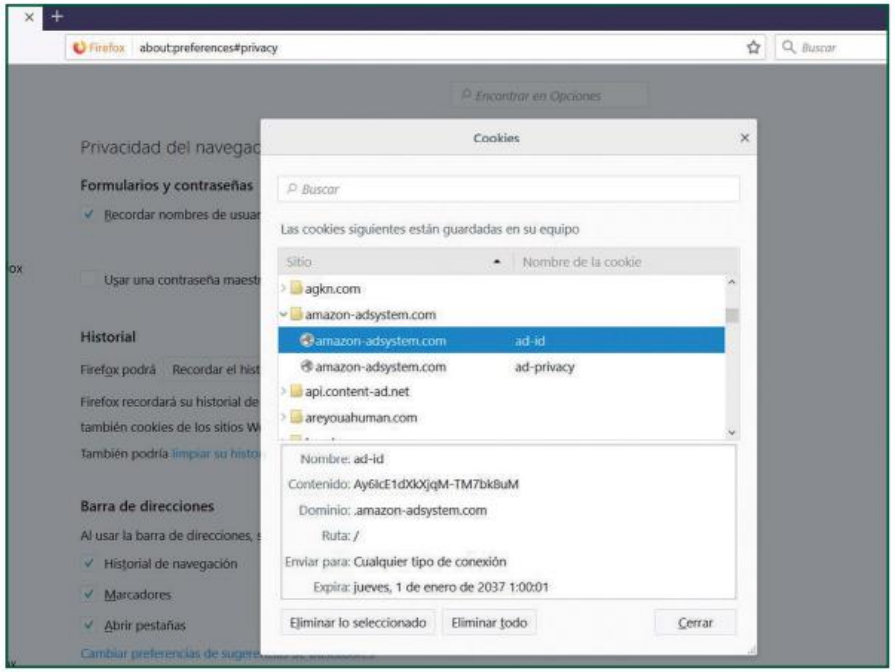

Figura 4.3
Visualización de una cookie en un navegador.

4.3 Cookies

Tu servidor de correo gratuito de Internet utiliza cookies cuando le indicamos la casilla de “recordarme en este equipo”. Esto provocará que el servidor enviará la creación de una cookie con nuestro nombre de usuario y nuestra contraseña para que se almacene en ese equipo y la caducidad será de al menos dos años. Esa misma página nos indica que no marquemos recordar cuando ese sea un equipo público, puesto que entonces cualquier persona que utilizara ese servicio después que nosotros tendría acceso a nuestro correo electrónico.

Para crear una cookie en PHP, tendremos que utilizar la función setcookie(), en la que como mínimo tendremos que enviar el nombre y el valor de cookie. A continuación mostraremos una tabla (Figura 4.4) donde podemos ver los diferentes campos que tiene una cookie.

Para tener control sobre nuestras cookies, es interesante que a la hora de crearlas especifiquemos al menos los primeros tres parámetros: nombre de la cookie, valor y fecha. A continuación mostramos un ejemplo de creación de cookie: Esta cookie dura una hora (1 hora = 3.600 segundos).

4.3 Cookies

A continuación mostramos un ejemplo de creación de cookie: **Esta cookie dura una hora (1 hora = 3.600 segundos).**

```
setcookie("visitas", "1", time() + 3600);
```

La función time() retorna la fecha actual en segundos desde la época Unix (el instante de tiempo cero para sistemas Unix es el 1 de enero de 1970). Podemos expresar también la hora utilizando la función mktime (hora, minuto, segundo, mes, día, año) que permite especificar la hora y la fecha concretas.

Vamos a ver un ejemplo del código necesario para crear una cookie que caduque el 10 de enero de 2013 a las 22:30:00 horas:

```
setcookie("visitas", "1", mktime(22,30,0,1,10,2013));
```

4.3 Cookies

Para recuperar el valor de la cookie, podemos utilizar el array global de PHP $_COOKIE[‘nombre_ cookie’:

CAMPO	FUNCIÓN
Nombre	Nombre que tiene la cookie
Valor	Contenido de la cookie
Fecha	Fecha de expiración de la cookie, es decir el número de segundos que tienen que pasar para que caduque la cookie. Si no ponemos ningún valor la cookie caduca al terminar la sesión.
Camino	Especifica la ruta donde se tiene que guardar la cookie
Dominio	Especifica el dominio en el que es válido la cookie
Segura	Indica si la cookie necesita una conexión segura (HTTPS). Para indicar está condición pondremos un valor de 1 Si ponemos un 0 o nada indicaremos que no necesitamos una conexión segura.

Figura 4.4
Funciones que tienen los campos de las cookies.

Educar en la Verdad para ser libres

4.4 Seguridad: usuarios, perfiles, roles

Como ya sabes, la seguridad es un aspecto muy importante en la comunicación web. Pero la seguridad debe tomarse en cuenta utilizando el sentido común, puesto que mucha seguridad puede provocar lentitud de respuesta.

Entre los muchos aspectos de seguridad se encuentran los usuarios. Es muy importante saber qué acciones pueden realizar los usuarios en nuestras páginas y servicios web. Para completar este título hay que añadir cuatro palabras más: usuarios, perfiles, roles y permisos.

Definamos estos conceptos:

- Permiso. Restricción o garantía de realizar cierta acción.
- Rol. Agrupación de permisos otorgados.
- Perfil. Agrupación de funciones a realizar.
- Usuario. Actor dentro del sistema

4.4 Seguridad: usuarios, perfiles, roles

Sería muy compleja la gestión de permisos y usuarios sin los perfiles y los roles (Figura 4.5). Se deberían asignar muchos permisos a cada uno de los usuarios de nuestro sistema.

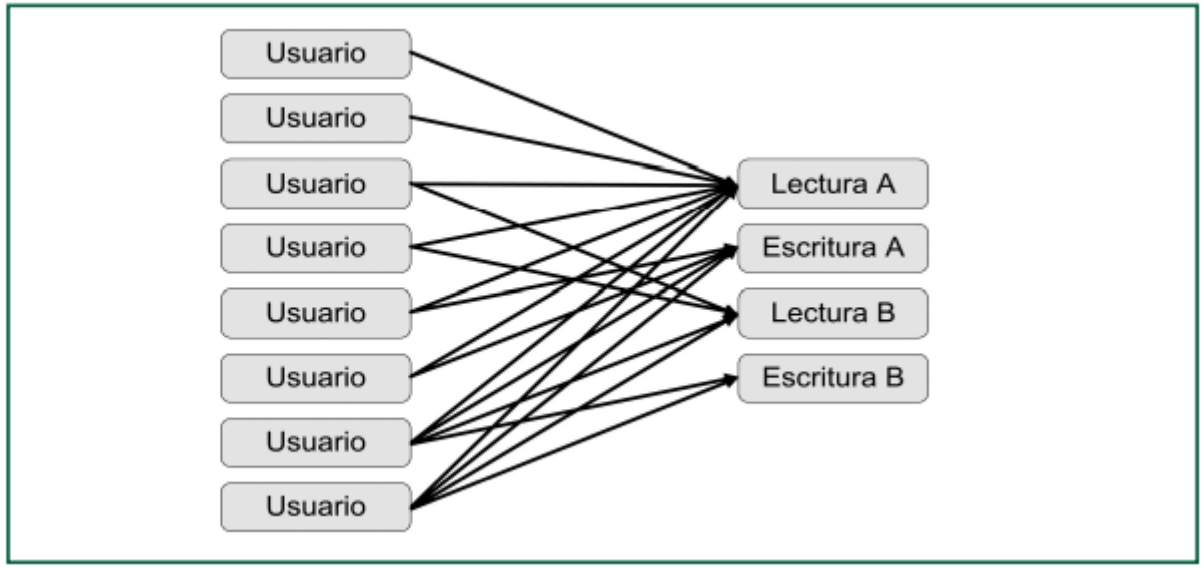

Figura 4.5
Gestión de usuarios y permisos sin roles ni perfiles.

Educar en la Verdad para ser libres

4.4 Seguridad: usuarios, perfiles, roles

La complejidad de este sistema es evidente puesto que es necesario asignar uno a uno los permisos tanto de lectura como de escritura para cada uno de los objetos. Las tareas de administración de usuario suelen ser básicamente tres: Alta, Baja y Modificación.

Cuando se deba dar de alta un nuevo usuario se deberá asignar a ese usuario todos los permisos que deba tener, que probablemente serán muchos. Al darse de baja se deberá realizar la operación inversa de deshacer las asignaciones. Y la modificación será la asignación o des asignación de ciertos permisos

En cambio, es mucho más sencillo de administrar y de comprender un sistema gestionado mediante roles y perfiles (Figura 4.6).

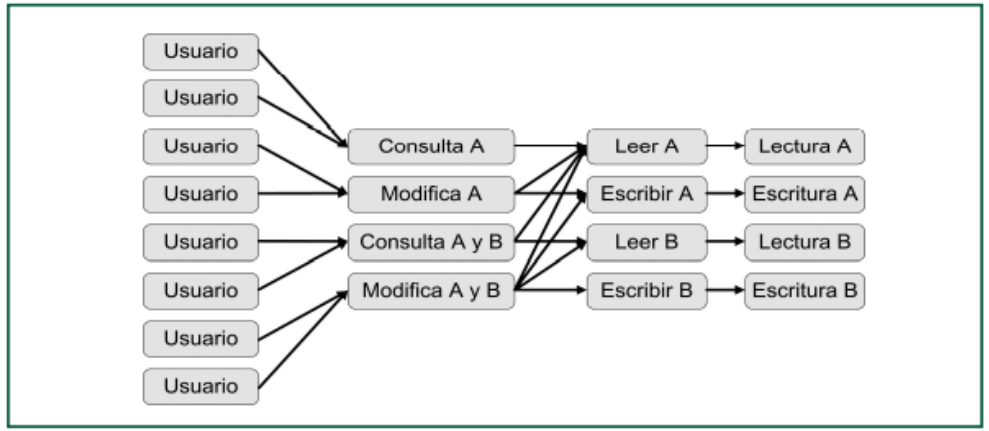

Figura 4.6
Gestión de usuarios con perfiles, roles y permisos.

4.4 Seguridad: usuarios, perfiles, roles

Utilizando roles y perfiles estamos agrupando las necesidades de seguridad utilizando una pila de capas de asignación.

De este modo, garantizamos la seguridad de los datos permitiendo la lectura o la escritura de los datos solamente a los roles que lo necesiten. Esta estructura apilada de seguridad transforma las necesidades físicas de seguridad, que son los permisos, subiendo en las capas hasta la capa lógica final que son las posibles acciones que puedan solicitar los usuarios.

Cuando se deba dar de alta un nuevo usuario será tan sencillo como asignar a ese usuario a un perfil concreto y nada más.

Dar de baja significará desasignar a ese usuario de un perfil. Y la modificación podrá ser incluso totalmente transparente a los usuarios puesto que se pueden modificar los perfiles y los roles sin que los usuarios se vean afectados.

4.4 Seguridad: usuarios, perfiles, roles

Hay servidores de aplicaciones que ya disponen de perfiles predeterminados para agrupar las posibles funcionalidades que se pueden asignar a los usuarios como pueden ser: administrador, operador, desarrollador, usuario, etc.

Otro aspecto de la seguridad se refiere al posible acceso a datos de otros usuarios.

Las sesiones están almacenadas de tal manera que no es posible acceder a los datos de otra sesión. Por tanto, las sesiones también garantizan la seguridad de sus datos.

Únicamente se podría acceder a una sesión diferente si se conociera el identificador de sesión de destino, pero ya hemos visto que esto es prácticamente imposible de conocer en el instante de tiempo en que ese identificador de sesiones válido.

La forma de dar una garantía casi total de seguridad es con la utilización del protocolo seguro **https.** El protocolo seguro utiliza **certificados digitales** para garantizar que la comunicación es segura.

4.4 Seguridad: usuarios, perfiles, roles

En este caso la comunicación viaja encriptada y únicamente los dos extremos de la comunicación, cliente y servidor, son capaces de desencriptarla.

Este aspecto de seguridad se denomina **criptografía de clave asimétrica** y, hoy en día, es la forma que nos garantiza la mayor y mejor seguridad conocida.

4.5 Autenticación de usuarios

Autenticación de usuarios

La validación, o la también llamada autenticación de usuarios, es un problema clásico que, tarde o temprano, tenemos que utilizar en una página web. Hoy en día es casi imposible encontrar una página que no incluya una validación de usuarios.

Existen varias formas de realizar una autenticación de usuarios en una página web. Algunas de ellas son realmente simples aunque tienen el problema de que no consideran aspectos básicos de seguridad. Hay otras formas de autenticar que consideran a la seguridad como la razón de ser de una validación (lo cual es obvio) y, por lo tanto, son mucho más complejas.

La solución de autenticación que se va a mostrar a continuación es bastante simple y sí considera aspectos básicos de seguridad, por lo que es un buen punto de partida a la hora de plantearnos construir un sistema de autenticación para una página web ya que nos sirve para entender muy bien la problemática.

4.5 Autenticación de usuarios

El sistema de autenticación consiste en que el usuario accede a una página que podemos llamar **login.php** e introduce un usuario y una clave o password. Una vez cargados estos dos datos, vamos a una segunda página, **autenticacion.php,** y allí comprobamos que el usuario introducido corresponde con la clave que hayamos utilizado. Generalmente las credenciales están almacenadas en una base de datos, de modo que la autenticación será una comprobación de que existe en una tabla "usuarios", algún registro con el usuario y contraseñas introducidas por el usuario.

Si no existe, querrá decir que los datos introducidos son incorrectos. Todo este proceso se realiza internamente, sin que el usuario deba visualizar en ningún momento la página **autentication.php**. Si la comprobación es errónea salta a la primera página que vimos**, login.php** para que se vuelva a introducir el usuario y la clave. Si la autenticación es correcta, salta a una tercera página, **aplicacion.php,** que será una página de nuestro sitio que ya contiene contenido y que contiene además un bloque de seguridad que sirve para comprobar que el usuario ha pasado por las páginas **login.php** y **autenticacion.php**, y no ha accedido a nuestro contenido de forma directa.

4.5 Autenticación de usuarios

A continuación se muestra una figura (Figura 4.7) donde podemos ver una pantalla típica donde se piden los datos para la autenticación de usuarios.

Figura 4.7
Proceso para la autenticación de usuarios.

4.6 Herramientas de programación

Herramientas de programación

En la unidad 1 viste qué son un editor y un compilador. Desde el inicio de la informática se ha trabajado de esta manera. Sin embargo, con el paso de los años, se han ido modelando otros tipos de programas para ayudar a la programación: los entornos IDE (Integrated Development Environment). Un entorno IDE es un programa informático compuesto por un conjunto de herramientas de programación.

Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios lenguajes.

Estos entornos IDE facilitan la programación proporcionando herramientas de búsqueda de información y de autocompletado del texto conforme se va escribiendo.

4.6 Herramientas de programación

Uno de los entornos IDE más utilizados es NetBeans (Figura 4.8) que cuenta con una gran cantidad de plugins de ayuda a la programación en múltiples lenguajes.

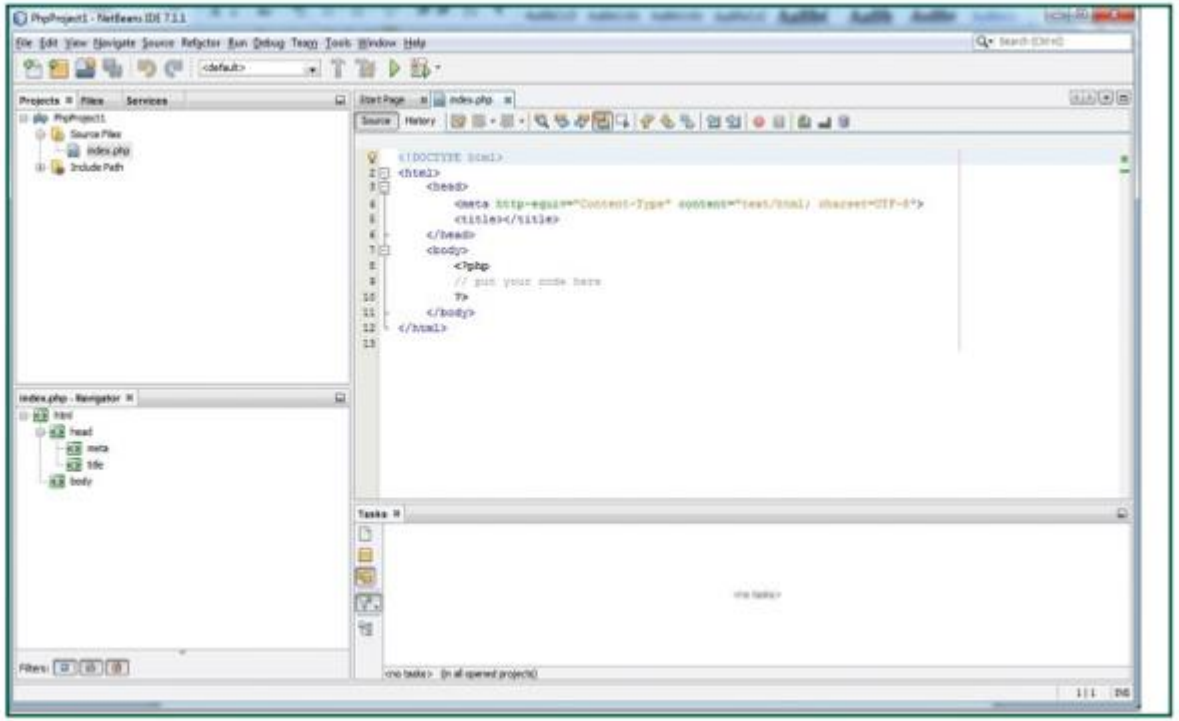

Figura 4.8
Ejemplo de proyecto con NetBeans.

Educar en la Verdad para ser libres

4.6 Herramientas de programación

Otro ejemplo de IDE muy utilizado es el software de Eclipse. Este software es un IDE de código abierto multiplataforma para desarrollar Aplicaciones de Cliente Enriquecido.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge.

Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

4.7 Pruebas y Depuración

Pruebas y depuración

Cuando desarrollamos nuestro sitio web, necesitamos realizar pruebas para verificar que hemos escrito correctamente nuestro código. Pero las pruebas deben hacerse aparte. En programación se definen tres entornos (Figura 4.9):

- **Desarrollo.** Habitualmente será el grupo de PCs del equipo de desarrollo que han diseñado un entorno de trabajo para generar los desarrollos.

- **Test.** Entorno similar al entorno final de producción tanto en arquitectura como en magnitud de servicios y datos. Es el entorno donde se realizan las pruebas y se depura el código

- **Producción.** Entorno real de servicios y datos. No se hacen pruebas. Solo pueden acceder los administradores para realizar cambios

4.7 Pruebas y Depuración

Es importante entender que debe existir esta separación de entornos para que nunca se ponga en peligro la integridad del entorno de producción.

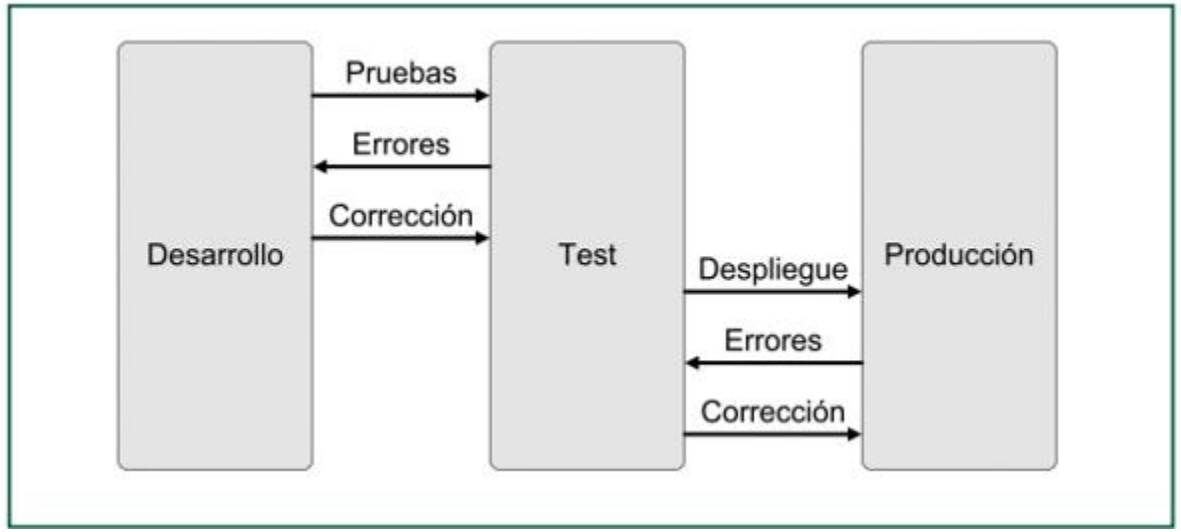

Figura 4.9
Los tres entornos de la programación.

4.8 Resumen

El concepto de estado empieza a cobrar importancia desde la aparición de los objetos en el mundo de la programación. Anteriormente el concepto de estado no tenía mucho sentido en los programas pero la programación orientada a objetos desveló toda su capacidad.

Podemos definir estado como el conjunto de valores que toma un objeto en un momento dado. Dentro de un sistema complejo el estado de ese sistema correspondería al conjunto de estados de todos sus objetos.

Una sesión es el conjunto de objetos que están en un estado concreto. El mantenimiento del estado es responsabilidad del servidor web o del servidor de aplicaciones pero cada lenguaje de programación habilita ciertas características diferentes. Este mantenimiento del estado puede ser automático o bajo demanda.

4.8 Resumen

Una cookie es un pequeño fichero de texto que contiene una serie de entradas atributo - valor al que también se le asocia a una fecha de caducidad. Lo más interesante de las cookies es que pueden ser generadas en el cliente a partir de las respuestas del servidor, pudiendo el usuario ser totalmente ajeno a que se están usando cookies. Una de las informaciones importantes que se almacenan en las cookies es justamente el identificador de sesión.

Los aspectos importantes que tenemos que definir a la hora de implementar la seguridad de nuestras aplicaciones web son la gestión de usuarios, roles, perfiles y permisos. Estos aspectos nos marcarán la forma en la que nuestros usuarios se validarán contra nuestra aplicación web.

Un entorno IDE es un programa informático compuesto por un conjunto de herramientas de programación que facilitan la programación proporcionando herramientas de búsqueda de información y de autocompletado del texto conforme se va escribiendo

Bibliografía

- Vicente, J. (2013) “Desarrollo Web en entorno servidor”, editorial: Ibergaceta publicaciones, S.L. ISBN:978-84-1545-292-8

- Comesaña,L. (2017) “Desarrollo Web en entorno servidor”

- Desarrollo web en entorno servidor. Realización: ITACA (Interactive Training Advanced Computer Aplications, S.L.) Colaboradores: Departamento de Producto de Centro de Estudios CEAC © Planeta DeAgostini Formación, S.L.U. Barcelona (España), 2016 ISBN: 978-84-9063-804-0 (Obra completa) ISBN: 978-84-9128-797-1 (Desarrollo web en entorno servidor)