PROYECTO TIC'S

LENGUAJES DE PROGRAMACIÓN ADECUADOS PARA EL DESARROLLO DE SISTEMAS DE INFORMACIÓN DE LAS PEQUEÑAS Y MEDIANAS EMPRESAS

LUIS FERNANDO AYALA MERLO

LILIANA PATRICIA CARBONO ATENCIO

HELIO RICARDO MARIN ORTIZ

SHIRLEY PAOLA REYES MUÑOZ

CORPORACIÓN EDUCATIVA MAYOR DEL DESARROLLO

SIMON BOLIVAR

INGENIERIA DE SISTEMAS

10 SEMESTRE

BARRANQUILLA 2004

UNIVERSIDAD SIMON BOLIVAR
HEMEROTECA
BARRANQUILLA

TABLA DE CONTENIDO

INTRODUCCION

CAPA DE PRESENTACION 1

BROWSER 1

LOGICA DE PRESENTACION 2

LENGUAJES DE ETIQUETA 2

LENGUAJE HTML 2
LENGUAJE XML 3
CAPAS DE LOS DIEFENTES LENGUAJES DE ETIQUETA 5

LOGICA DE APLICACIÓN 9

LENGUAJES DE SCRIPT 9

LENGUAJES MAS UTILIZADOS ORIENTADOS A LA WEB 11

JSP 11

VENTAJAS DE JSP 12
CARACTERISTICAS DE JSP 13
SERVIDORES PARA JSP 16
CAPAS DE JSP 20
CAPA DE PRESENTACION 20
CAPA DE APLICACIÓN 21

PHP 24

FUNCIONAMIENTO DE PHP 25
SEGURIDAD DE PHP 27
ERRORES EN PHP 27
COOKIES EN PHP 29
MANEJO DE CONEXIONES 30
VENTAJAS DE PHP 31

ASP 32

CARACTERISTICAS DE ASP 32
FUNCIONAMIENTO DE ASP 33
COOKIES EN ASP 45
CONEXION A BASES DE DATOS 53
FORTALEZAS Y DEVILIDADES DE PHP, ASP Y ASP.NET 55

PERL 57

FORMATO DE PROGRAMAS REALIZADOS EN PERL 57
TIPO BASICO DE DATOS EN PERL 58
VENTAJAS DE PERL 59

MOTORES DE BASES DE DATOS MAS UTILIZADOS 60

MYSQL 60
INSTALACION 60
TIPOS DE DATOS QUE MANEJA MYSQL 64
MYSQL CONTROL SERVER 72

POSTGRE 78

INSTALACION DE POSTGRESQL 78

MYSQL VS POSTGRE 86

ANEXOS 90

INTRODUCCION

Algunos lenguajes de programación fueron diseñados para una sola persona como un concepto único, mientras que otros son el producto desarrollado de varios años realizados por comités de diseño de lenguajes.

Hay al menos dos formulas fundamentales desde las que se pueden ver o clasificar los lenguajes de programación por su nivel y por sus principales aplicaciones. Estas están condicionadas por la visión histórica por la que trascurrió el lenguaje.

Los lenguajes de programación son tomados de diferentes perspectivas, es importante para un programador decidir cuales conceptos omitir o cuales incluir en la programación. Con frecuencia el programador es osado a usar combinaciones de conceptos que hacen al lenguaje duro de usar de entender implementar en las diferentes áreas cada programador tiene en mente un estilo particular de programación, la decisión de incluir u omitir ciertos tipos de datos que pueden tener una significativa influencia en la forma en que son usados algunos conceptos de programación o modelos que se adapten en este caso a las Pequeñas y medianas empresas (PYmes).

Capa de presentación

Browser

Son los programas que permiten ver las páginas de la Malla Mundial o WWW se llaman en inglés browsers. El verbo to browse viene de una antigua palabra francesa que significa "brote de una planta", y su sentido inicial era "mordisquear, ramonear", que es lo que hacen los herbívoros. De ahí pasó a significar "echar una ojeada" (por ejemplo, a las cosas de una tienda) u "hojear" las páginas de un libro.

Como la Malla Mundial está compuesta de "páginas" parecía sensato que un programa permitiera "hojearlas", que fue lo que pasó en el inglés. En castellano se usan distintos nombres: por una parte, nunca falta quien utiliza directamente browser (pronunciado más o menos "brauser"). En ocasiones se usa "hojeador", traducción directa del inglés; también se lee "visualizador" o mejor "visor", que son acertados, porque recogen la esencia del programa: permitir ver las páginas tal y como fueron creadas. Bastante menos se utiliza "lector" (que por otra parte también es el dispositivo de acceso al CD-ROM, aunque rara vez hay posibilidad de equivocación porque aparecen en contextos muy distintos).

Pero los que están extendiéndose más son "navegador" y "explorador", porque los usan los programas dominantes en el mercado, pero que tienen el problema de estar ligados a sus nombres comerciales (Navigator

UNIVERSIDAD SIMÓN BOLÍVAR
BIBLIOTECA
CARACAS

y Explorer). Usan dos metáforas distintas: que la Malla Mundial es un océano (que por lo tanto se puede surfear o navegar), y que es un territorio desconocido (que hay que explorar).

Lógica presentación

Html

HTML es un lenguaje para la definición de estilos lógicos en documentos de hipertexto, siendo el medio principal para la diseminación de información en World Wide Web. HTML se limita a describir la estructura y el contenido de un documento, nunca el formato de la página y su apariencia, ya que éstos son muy dependientes del visualizador utilizado.

Los documentos HTML se dividen en tres partes bien diferenciadas:

1. *La Cabecera de tipo de documento*
   La usa el software para saber la versión de HTML que se está usando (no visible).
2. *La Cabecera del documento (document HEADer)*
   Usada para dar información sobre el documento (no visible).
3. *El Cuerpo del documento (document BODY)*
   Es la parte principal del documento, la parte que el usuario ve.

Estas tres partes pueden verse en la plantilla mostrada a continuación.

```
<HTML>
 <HEAD>
 </HEAD>

 <BODY>
 </BODY>
</HTML>
```

La primera linea indica que es un documento HTML:

```
<HTML>
```

La segunda y tercera linea delimitan la cabecera del documento:

```
<HEAD>
</HEAD>
```

La cuarta y quinta linea delimitan el cuerpo del documento:

```
<BODY>
</BODY>
```

La última linea indica que finaliza el documento HTML:

```
</HTML>
```

XML

Durante el último año, y sobre todo estos últimos meses se ha estado hablando del Lenguaje Extensible de "Etiquetado", eXtensible Markup Language, o XML, y en todos los foros de Internet que se precien de estar a la última no se puede tratar ningún tema sin hacer mención a este

nuevo estándar. Pero muchas veces se habla de él con poco o ningún conocimiento de causa, y son demasiados los que alaban sus virtudes sin entenderlas ni alcanzarlas por completo.

Antes de nada conviene repasar su historia y precedentes. La versión 1.0 del lenguaje XML es una recomendación del W3C desde Febrero de 1998, pero se ha trabajado en ella desde un par de años antes. Está basado en el anterior estándar SGML (Standard Gene ralized Markup Language, ISO 8879), que data de 1986, pero que empezó a gestarse desde principios de los años 70, y a su vez basado en el GML creado por IBM en 1969. Esto significa que aunque XML pueda parecer moderno, sus conceptos están más que asentado s y aceptados de forma amplia. Está además asociado a la recomendación del W3C DOM (Document Object Model), aprobado también en 1998. Éste no es más que un modelo de objetos (en forma de API) que permite acceder a las diferentes partes que pueden componer un documento XML o HTML.

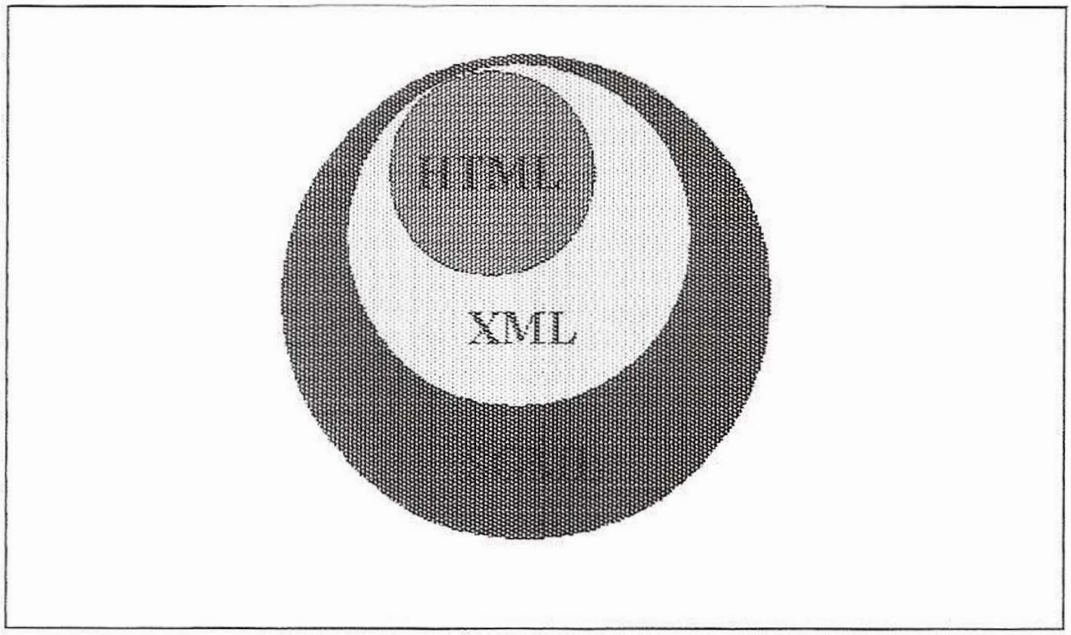
SGML proporciona un modo consistente y preciso de aplicar etiquetas para describir las partes que componen un documento, permitiendo además el intercambio de documentos entre diferentes plataformas. Sin embargo, el problema que se atribuye a SGML es su e xcesiva dificultad; baste con pensar que la recomendación ocupa unas 400 páginas.

Así que, manteniendo su misma filosofía, de él se derivó XML como subconjunto simplificado, eliminando las partes más engorrosas y menos útiles. Como su padre, y este es un aspecto importante sobre el que se incidirá después, XML es un metalenguaje: es un lenguaje para definir

lenguajes. Los elementos que lo componen pueden dar información sobre lo que contienen, no necesariamente sobre su estructura física o presentación, como ocurre en HTML.

Usando SGML, por otro lado, se definió precisamente el HTML, el lenguaje que nos es tan conocido. Entonces, ¿cuál es la diferencia entre ambos?

En una primera aproximación se puede decir que mediante XML también podríamos definir el HTML, con lo que podríamos considerar los siguientes conjuntos:



CAPAS DE LOS DIFERENTES LENGUAJES DE ETIQUETAS MÁS UTILIZADOS EN LA CREACION DE LAS PÁGINAS WEB

De hecho, HTML es simplemente un lenguaje, mientras que XML como se ha dicho es un metalenguaje, esto es, un lenguaje para definir lenguajes. Y esa es la diferencia fundamental.

Desde su creación, XML ha despertado encontradas pasiones, y como para cualquier tema en Internet, hay gente que desde el principio se ha

dejado iluminar por sus expectativas, mientras que otras muchas lo han denostado o simplemente ignorado.

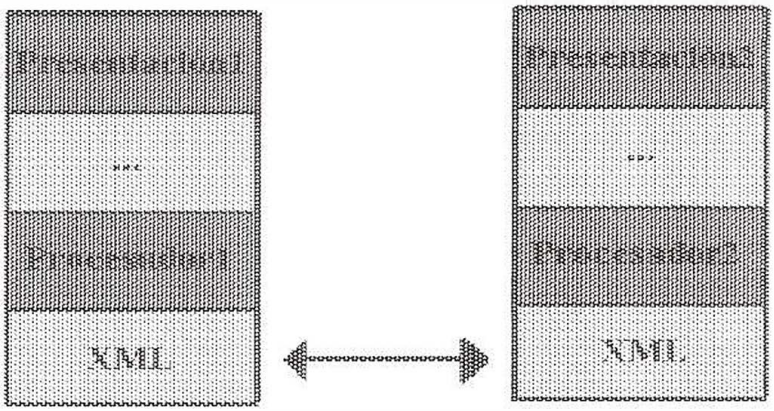
Durante el año 1998 XML ha tenido un crecimiento exponencial, y con ello se refiere sobre todo a sus apariciones en los medios de comunicación de todo tipo, menciones en páginas web, soporte software, tutórales, etc. y ahora XML por fin ha despegado. Y esto significa simplemente que desde noviembre del año pasado hasta hoy ha pasado de mera especulación a ser una realidad empresarial palpable y mesurable: los programas que lo soportan han crecido del mismo modo exponencial, y a día de hoy no hay empresa de software que se precie que no anuncie la compatibilidad de sus productos más vendidos con este nuevo estándar: Micros oft (Office 2000), Oracle (Oracle 8i, Web Application Server) o Lotus (Notes) son tres claros ejemplos de ello. Aún más increíble es pensar que hay empresas que se han creado entorno a él, u otras que han movido su actividad hacia su ámbito (de SGML a XML , por ejemplo, como ArborText).

Una pregunta que ha acudido a muchos de nosotros es: ¿será XML el sustituto de HTML, que tan bien conocemos, dominamos, amamos y odiamos? No. Esa es la respuesta, e intentaré explicar porqué: básicamente XML no ha nacido sólo para su aplicación en Inter net, sino que se propone como lenguaje de bajo nivel (a nivel de aplicación, no de programación) para intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo, y casi cualquier cosa que podamos pensar. Sin ir más lejos, algunos lenguajes de los que hablaremos, definidos en XML,

recorren áreas como la química y la física, las matemáticas, el dibujo, tratamiento del habla, y otras muchas.

Pues bien, no lo sustituirá, pero, aplicado en Internet, sí va a mejorar algo de lo que HTML empezaba a adolecer desde hace un tiempo: establece un estándar fijo al que atenerse, y separa el contenido de su presentación. Esto significa que a partir de ah ora, o mejor desde el momento que se aplique definitivamente, para ver un documento web no estaremos sujetos a la parte del estándar de hojas de estilo (CSS) que soporte el Navigator de Netscape o el IExplorer de Microsoft, ni al lenguaje de script del servidor y al modelo de objetos definido por MS, Nestcape, etc. Además tampoco estaremos atados a la plataforma: podremos ver la misma información desde nuestro PC o desde un Hand-HeldPC, un navegador textual, una lavadora, un microondas o un reloj con acceso a Internet, con presentaciones adecuadas a cada entorno. Ya veremos más adelante el porqué de estas afirmaciones.

Se puede suponer de este modo que XML constituye la capa más baja dentro del nivel de aplicación, sobre el que se puede montar cualquier estructura de tratamiento de documentos, hasta llegar a la presentación.



COMPARTICIÓN DE DOCUMENTOS ENTRE DOS APLICACIONES COMO INTERCAMBIO DE DATOS

Pero ahora, y antes de seguir divagando sobre sus posibilidades, expliquemos los conceptos en los que se basa y la teoría de su aplicación, para que podamos escribir documentos XML en poco tiempo.

Tipos de documentos XML: ***válidos y bien formados****. Éste es uno de los aspectos más importantes de este lenguaje, así que hace falta entender bien la diferencia:*

- ***Bien formados:*** *son todos los que cumplen las especificaciones del lenguaje respecto a las reglas sintácticas que después se van a explicar, sin estar sujetos a unos elementos fijados en un DTD (luego veremos lo que es un DTD). De hecho lo s documentos XML deben tener una estructura jerárquica muy estricta, de la que se hablará más tarde, y los documentos bien formados deben cumplirla.*

- ***Válidos:*** *Además de estar bien formados, siguen una estructura y una semántica determinada por un DTD: sus elementos y sobre todo la estructura jerárquica que define el DTD, además de los atributos, deben ajustarse a lo que el DTD dicte.*

Lógica aplicación

Lenguajes de script

Los lenguajes de guiones son los controladores de HTML Dinámico. La combinación de HTML y de las Hojas de Estilo en Cascada proporciona un control sin precedentes con respecto al aspecto de nuestras páginas para la red; sin embargo, por sí mismas las tecnologías son estáticas e inmóviles, de forma muy parecida a un coche sin conductor. Los lenguajes de guiones nos capacitan para aportar la interactividad y el movimiento a nuestro HTML y a las HEC.

HTML Dinámico es bastante abierto en términos de los lenguajes de guiones que admite. Los futuros diseñadores de navegadores para la red podrían utilizar cualquier lenguaje de guiones que resulte popular en su momento, y seguirán al mismo tiempo beneficiándose de HTML Dinámico.

Sin embargo, Internet Explorer 4.0 contiene dos lenguajes de guiones que se pueden utilizar con HTML Dinámico: JavaScript y VBScript. El hecho de que IE 4.0 venga con dos lenguajes de guiones solicita inmediatamente una respuesta a la pregunta: ¿qué lenguaje de guiones debo utilizar?

Existen muchos argumentos en pro y en contra de cada uno de estos lenguajes de guiones. La versión comercial de Visual Basic (el lenguaje predecesor de VBScript), por ejemplo, es uno de los entornos de programación más vendidos de todos los tiempos y hay enormes cantidades de programadores que lo utilizan a diario. De este modo, lo mejor para una gran cantidad de personas es tener una versión de guiones de Visual Basic que esté a su disposición.

JavaScript, por otra parte, es en la actualidad el lenguaje de guiones de facto en la red, debido a su inclusión en Netscape Navigator 2.0 y siguientes y también en Internet Explorer 3.0 y siguientes. Aun cuando hay más programadores que conocen Visual Basic, hay más programadores de la red que conocen JavaScript. Además, en la actualidad VBScript solamente está disponible en Internet Explorer.

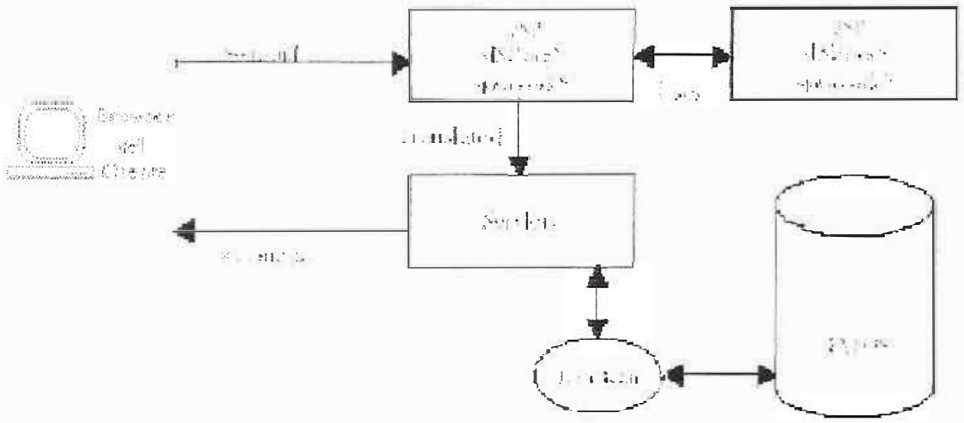
Algunos lenguajes orientados a la web mas utilizados son:

- ***JSP***
- ***ASP***
- ***PERL***
- ***PHP***

JSP

Java serer pack, es una tecnología que permite mezclar HTML estático con HTML generado dinámicamente. Muchas paginas Web que están construidas con programas CGI son casi estáticas, con la parte dinámica ,imitada a muy pocas localizaciones pero muchas variaciones CGI, incluyendo los Servlets, hacen que generemos la pagina completa mediante nuestro programa, incluso aunque la mayoría de ellas siempre sea lo mismo. JSP nos permite crear dos partes de forma separada..

- *Usar código **HTML junto a** código **Java***
- *Parecido al resto de lenguajes en sintaxis (PHP y ASP)*
- *Separación entre **presentación** y **contenido***
- *Reuso de componentes basado en JavaBeans*
- *Uso de XML en los scripts.*



VENTAJAS DE JSP

Contra Active Server Pages(ASP). Asp es una tecnología similar a ala de Microsoft, las ventajas de JSP están duplicadas, primero la artes dinámica esta escrita en java no en Visual Basic otro lenguaje especifico de MS, por eso es mucho mas poderosa y fácil de usar. Segundo es portable a otros sistemas operativos y servidores web.

Contra los Servlets. JSP no nos da nada que en un principio no pudiéramos hacer con un Servlets pero es mucho mas conveniente escribir y modificar HTML normal que tener que hacer un billón de sentencias print Ln que generan HTML. además separado el formato del contenido podemos poner diferentes personas en diferentes tareas.

Contra Java Script. Java script puede generar HTML dinámicamente en el cliente. Este en una capacidad útil, pero solo maneja situaciones donde la información dinámica esta basada en el entorno del cliente. Con la excepción de las COOKIES , el http y el envió de formularios no están disponibles con Java Script y como se ejecuta en el lado del cliente, Java Script no puede acceder a recursos en el lado del servidor, como bases de datos, catálogos, información de precios, etc.

Otras características de JSP

- ***Escalabilidad:***

  La ejecución de Servlets está basada en motores (engines) que mantienen en memoria tanto el bytecode de las clases que implementan los Servlets, como los componentes o variables que pueden ser compartidos entre solicitudes y/o clientes (como conexiones a bases de datos).

- ***Sesiones:***

  Una sesión, encapsulada una o más solicitudes de un cliente en una

sola unidad lógica, donde se pueden mantener valores de variables entre solicitudes.

- ***Estándar:***

  JSP y Java Servlet hacen parte de J2EE de Sun Microsystems, garantizando que es soportada y portable entre diferentes servidores de aplicaciones.

- ***Presentación vs lógica del negocio:***

  JSP permite diseñar aplicaciones modulares, donde la lógica del negocio (reglas, validaciones, procesos) se desarrollan en clases Java y la presentación (interfaz de usuario) en HTML O XML.

- *Separación entre generación de contenido y presentación. Mediante etiquetas HTML o XML se da formato a la página, y mediante etiquetas*

- *JSP, código script o JavaBeans se añade el componente dinámico. De esta forma, encapsulando la lógica, se puede modificar la presentación de la página JSP sin afectar a la generación del contenido.*

- *Reutilización de componentes software. La descomposición en componentes que caracteriza a esta tecnología facilita el que objetos de una aplicación vuelvan a utilizarse en dominios diferentes. Por ejemplo, objetos como el analizador léxico o el extractor de raíces definidos para el glosario, pueden ser reutilizados para desarrollar*

otro tipo de aplicaciones en las que se necesite integrar técnicas de RI.

- *Utilización de etiquetas XML para simplificar el desarrollo de las páginas. Así se ocultan detalles de implementación y se facilita la construcción de páginas que integren código JSP, incluso a desarrolladores web poco acostumbrados a trabajar con lenguajes script. A través del mecanismo de extensibilidad de las librerías de etiquetas, estas pueden personalizarse para las necesidades de cada sistema.*
- *Independencia de la plataforma tanto cliente como servidor. Por un lado, la utilización de código Java garantiza la portabilidad de la aplicación para su ejecución en cualquier servidor que contenga una maquina virtual Java.*

Esta es una ventaja sustancial frente a otras tecnologías similares, como:

- *Active Server Pages (ASP) . En el caso del cliente, al recibir sólo páginas HTML hace que sea compatible con cualquier navegador.*
- *Cuando se produce una solicitud de una página JSP por parte de un navegador, se comienza comprobando si se trata de la primera solicitud de dicha página. En ese caso se compila a un servlet, que es ejecutado y cuya salida es devuelta al usuario que realizó la petición. Las siguientes solicitudes son más eficientes al no necesitar que dicha página sea de nuevo compilada, ya que sólo se invoca al servlet*

que se generó en la petición inicial. En este sentido esta tecnología es más ventajosa que otras como ASP, que requieren cada vez una nueva compilación.

- *El proceso de compilación de las páginas consiste en analizar su contenido buscando etiquetas JSP y traduciendo éstas a código Java equivalente. El contenido estático de las páginas (código html) es traducido a cadenas de caracteres en lenguaje Java. Las etiquetas de componentes JavaBeans son traducidas a su correspondiente objeto, mientras que los elementos script son transferidos tal cual. Todo este código fuente es usado para formar los métodos de servicio del servlet que se va a generar*
- *La potencia y flexibilidad, así como la portabilidad que supone utilizar código Java, hace que JSP sea una tecnología en auge para la construcción de servicios de información en Internet.*

SERVISORES PARA JSP

"Servlet Engines" y "Java Application Servers"

Servlet Engines

Quizás el nombre que más salga a relucir con "Servlet Engines" es Tomcat o Jakarta Apache. Tomcat surgió de Sun Microsystem's cuando desarrollaban un "Servidor de Páginas" que utilizara "Java",y posteriormente cedieron el código fuente a la fundación Apache.

A pesar del nombre Apache-Tomcat; Tomcat no requiere de Apache para su funcionamiento (solo requiere de un JDK ("Java Development Kit") y es aquí donde dista un poco de las previas implementaciones:

En la figura anterior las líneas demuestran que el ***Servlet engine*** *(también llamado "Web Container") es capaz de responder a requisiciones de Internet, en efecto actuando como "Servidor de Páginas", sin embargo, aunque esto sea posible la gran mayoría de las implementaciones de Servlet Engines no funcionan tan eficiente como un "Servidor de Páginas", es por esto (y otras razones que a continuación se mencionan) que se opta por utilizar un "Servidor de Páginas" (Apache,Aol,Netscape..) en conjunción con un "Servlet Engine".*

La linea verde demuestra la utilización de un "Servidor de Páginas" con un "Servlet Engine" (Web-Container), esta implementación suele utilizarse cuando se requiere utilizar encriptación o se tiene un sitio que contiene documentos puros en HTML , las cuales son dos areas que un "Servidor de Páginas" supera en desempeño comparado con un "Servlet Engine". La comunicación entre el "Servidor de Páginas" y el "Servlet Engine" se lleva

acabo mediante el protocolo denominado ajp12 y recientemente con su sucesor ajp13 .

Que hace el Servlet Engine ?

*El "Servlet Engine" ofrece un "**Ambiente**" donde habitan los JSP y Servlets, es ahí donde se contemplan una gran cantidad de funcionalidades como: threading, manutención de sesiones, conectividad con el "Servidor de Páginas", es por esto al "Servlet Engine" también se le denomina "Web-Container".*

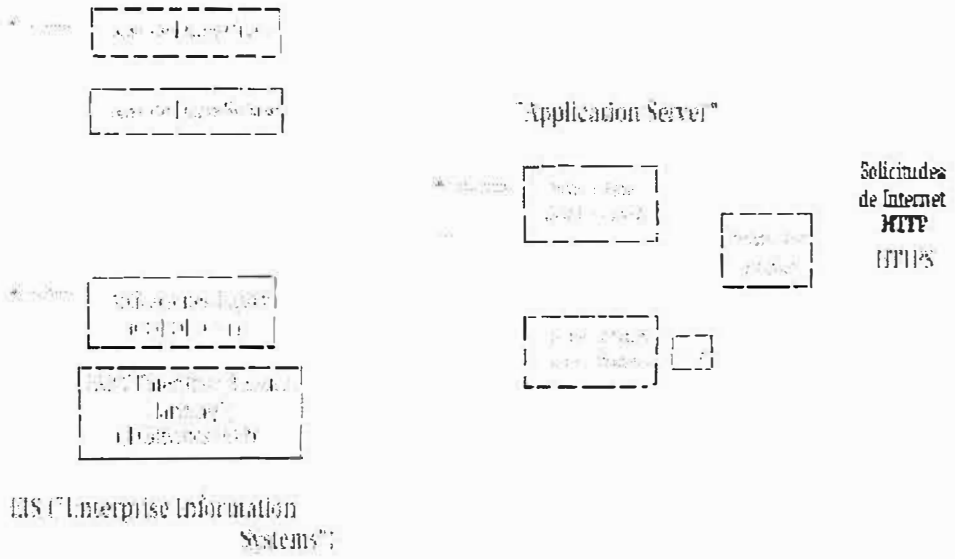
Dos "Servlet Engines" (Web-Containers) que están en amplio uso y son utilizados con "Servidores de Páginas" son: Tomcat y ServletExec, donde el primero es open-source y el último es un producto cerrado; otro "Servlet Engine" es Resin (Open-Source) el cual permite: utilizar JavaScript como "Scripting Language" dentro de JSP's y acceso a XSL una extensión de XML .

Como se observa en el diagrama también se requiere de un JDK ("Java Development Kit") , el cual llevará acabo la ejecución de los programas ("Servlets" y "JSP's") en Java; como toda otra implementación existen diversas versiones de JDK's, esto se debe a que cada JDK debe ser diseñado alrededor de un Sistema Operativo (para ser más exactos es el JVM "Java Virtual Machine" el que debe ser diseñado alrededor del Sistema Operativo), algunos JDK's son: J2SE's (Java 2 Standard Edition) de Sun y JDK's de IBM

Java Application Servers

"Java Application Servers" hoy en día ya denominados "Application Servers" ofrecen una manera de Integrar y ofrecer las funcionalidades requeridas por la gran mayoría de sistemas empresariales, una de las razones por las cuales el mercado ha sido inundado con estos "Application Servers" es que están diseñados alrededor de J2EE , que es solo un grupo de especificaciones definidas por Sun.

Estos "Application Servers" comunmente llamados ***Middleware*** *se encuentran compuestos de la siguiente manera:*



Como su denominación lo implica ("***Middleware***"*) se encuentran en la parte media de una arquitectura de sistema, su flexibilidad reside en la posibilidad de accesar información en sistemas empresariales (EIS) como*

SAP, JdEdwards, bases de datos o alguna aplicación escrita en COBOL u otro lenguaje.

Dependiendo de la empresa que desarrolle el "Application Server" éste puede contener inclusive hasta un "Servidor de Páginas" o algún otro desarrollo propietario, sin embargo, los dos elementos primordiales (aunque no sean comercializados como tal) son el ***"Servlet Engine" (Web-Container) y "Enterprise Bean Engine" (Bean-Container).***

Que hace el Servlet Engine y Enterprise Bean Engine?

- *El Servlet Engine (Web-Container) en un "Application Server" realiza las mismas funcionalidades que fueron mencionadas anteriormente.(Ofrecer un ambiente para JSP y Servlets).*
- *El "Enterprise Bean Engine" (Bean-Container) ofrece un* ***"ambiente"*** *donde residen EJB's ("Enterprise Java Beans") , es mediante "Enterprise Java Beans" que se ejecuta la lógica de negocios sobre la información que reside en los sistemas empresariales ("EIS"). En el "Bean Container" (al igual que en el "Web Container") se contemplan varias funcionalidades: "Pooling" hacia bases de Datos (JDBC), control de transacciones(JTA-JTS), conectividad con ERP(Connectors), aplicaciones legacy(CORBA), entre otras cosas.*

La mayor ventaja de este tipo de arquitectura se debe a la separación de funcionalidades y uso de protocolos de red como RMI/ CORBA , esto facilita que puedan existir 4 o 5 "Hosts" en diferentes regiones geográficas, cada

*uno empleando cualquiera de los componentes antes mencionados. Por último, existen diversos "Application Servers" que son denominados "**Fully J2EE Compliant**" esto indica que cumplen con todas las especificaciones J2EE indicadas por Sun. (Vea J2EE)*

*Algunos "Application Servers" "**Fully J2EE Compliant**" son:*

- *WebLogic*
- *Websphere*
- *JRun*
- *Oracle 9i Application Server*
- *iPlanet (Previamente Netscape Enterprise o Kiva)*

En los últimos tiempos, para el desarrollo de aplicaciones web, se viene imponiendo el empleo de tecnologías de componentes: el más conocido JAVA y el más novedoso .NET. En cualquiera de ellos, pero sobre todo en el más extendido (JAVA), es habitual realizar una división de la arquitectura de las aplicaciones en tres capas funcionales:

1. *Capa de Presentación*
2. *Capa de Aplicación*
3. *Capa de Datos*

*La **capa de presentación** es la encargada de interaccionar con el usuario y se corresponde con lo que tradicionalmente se conoce como Interfaz de Usuario.*

*En la **capa de aplicación** es donde se localiza la lógica de negocio. La capa de aplicación recibe la petición del usuario a través de la capa de presentación, y se encarga de darle curso atendiendo, si es necesario, a los repositorios de información de la aplicación (bases de datos, LDAP, sistemas propietarios....).*

*Sin embargo, el acceso a los repositorios de información no se realiza directamente, sino que se hace a través de la **capa de datos**, encargada de proporcionar dicho servicio.*

El diseño de una aplicación de acuerdo con esta arquitectura de capas presenta la gran ventaja de producir código modular en el que la modificación de uno de sus componentes, por ejemplo la interfaz de usuario (la forma en la que el usuario navega por el web o la forma en la que se presentan los datos) no requiere modificar ninguno de los elementos de las otras capas. Asimismo, si por cualquier razón se migrase el repositorio de información y se alojase en una base de datos diferente, únicamente sería necesario modificar la capa de datos.

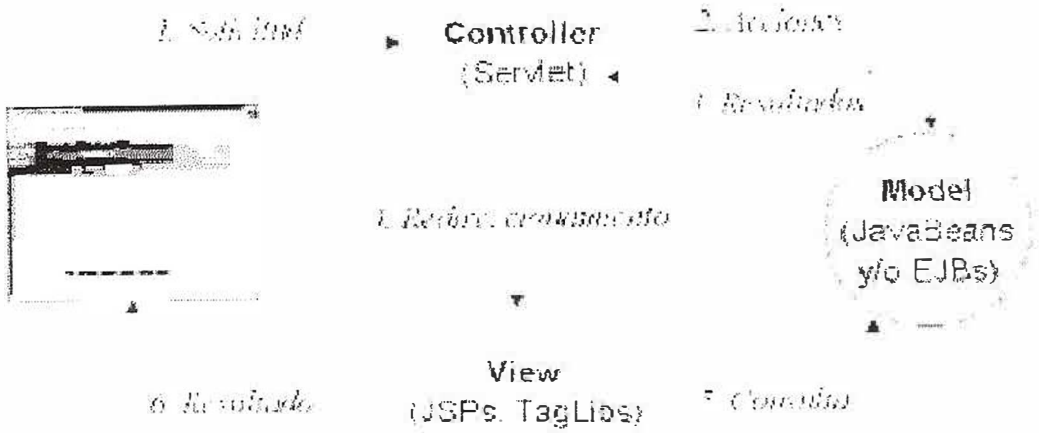
Patrón de Diseño de la capa de presentación

El diseño de aplicaciones web siguiendo la arquitectura funcional antes descrita se refleja posteriormente en el modelo lógico. La capa de presentación se implementa según el patrón de diseño Modelo-Vista-Controlador (MVC), cuyo principal objetivo es separar la presentación de la lógica de negocio.

El patrón MVC descompone una aplicación interactiva en tres grandes bloques:

- *El Modelo (Model) contiene los datos y la funcionalidad de la aplicación. Es independiente de la representación de los datos.*
- *Las Vistas (Views) muestran la información al usuario de una cierta forma.*
- *Un Controlador (Controller) asociado, generalmente un servlet, que recibe como entrada las peticiones de los usuarios y determina la vista a usar para presentar los resultados.*
- *El navegador del usuario, en su petición HTTP, genera una solicitud que es atendida por el controlador. Éste analiza sus entradas y llama a los objetos correspondientes del Modelo. El Modelo se encarga de ejecutar la solicitud y generar los resultados que mostrarán posteriormente las Vistas.*

Según el resultado que retorne el Modelo, el Controlador derivará la generación de la página interfaz a una o más JSPs (Vistas), que podrán consultar los Modelos con el fin de realizar su tarea adecuadamente.



Controlador

Este componente recibe todas las peticiones realizadas a la aplicación. Cada petición se identifica mediante un parámetro. En base a esta identificación, el Controlador decide qué objeto u objetos de negocio (Modelo) debe ejecutar para resolver la petición.

Tras la ejecución de los objetos de negocio, y en función del resultado devuelto por estos, el Controlador determina qué JSP se usará para visualizar el resultado, generando una redirección que concluirá con la generación del código HTML de la página.

Objetos de negocio

Los objetos de negocio son los objetos del Modelo que implementan la lógica de negocio. Sus funciones fundamentales son:

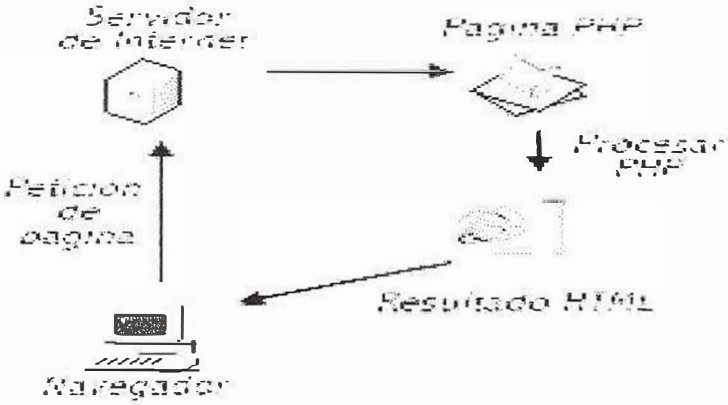
- *Realizar la validación de los datos introducidos por el usuario, tanto sintáctica (numérico, fecha, etc.) como lógica (importe menor que saldo, etc.)*
- *Ejecutar la petición realizada por el usuario. Para ello podrá valerse de transacciones contra Sistemas Host, consultas a bases de datos, consultas a proveedores de contenidos Etc.*
- *Generar los objetos que utilizarán las Vistas para mostrar los resultados obtenidos*
- *Garantizar la navegación y el flujo de pantallas correcto.*

PHP

El lenguaje PHP es un lenguaje de programación de estilo clásico, con esto quiero decir que es un lenguaje de programación con variables, sentencias condicionales, bucles, funciones.... No es un lenguaje de marcas como

podría ser HTML, XML o WML. Está mas cercano a JavaScript o a C, para aquellos que conocen estos lenguajes.

Pero a diferencia de Java o JavaScript que se ejecutan en el navegador, PHP se ejecuta en el servidor, por eso nos permite acceder a los recursos que tenga el servidor como por ejemplo podría ser una base de datos. El programa PHP es ejecutado en el servidor y el resultado enviado al navegador. El resultado es normalmente una página HTML pero igualmente podría ser una pagina WML.



FUNCIONAMIENTO DE PHP

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que su navegador lo soporte, es independiente del navegador, pero sin embargo para que sus páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP.

La ventaja que tiene PHP sobre otros lenguajes de programación que se ***ejecutan en el servidor*** *(como podrían ser los script CGI Perl), es que nos*

permite intercalar las sentencias PHP en las paginas HTML, es un concepto algo complicado de entender si no se ha visto nunca como funciona unas paginas PHP o ASP.

Todos los que nos hemos enfrentado con el diseño de páginas Web hemos echado de menos un poco más de dinamismo en ellas. Representar una página repleta de gráficos y nada más, deja de ser suficiente para ciertas aplicaciones en Internet. Estas aplicaciones requieren de cierta interactividad con el usuario, y han sido muchas las tecnologías aplicadas a este fin (formularios, CGI, etc.).

Ahora está disponible la versión 3 de PHP, o la versión 4 no definitiva, Profesional Home Pages, la solución para la construcción de Webs con independencia de la Base de Datos y del servidor Web, válida para cualquier plataforma. El objetivo final es conseguir la integración de las paginas HTML con aplicaciones que corran en el servidor como procesos integrados en el mismo, y no como un proceso separado, como ocurría con los CGIs. Igualmente interesa que dichas aplicaciones sean totalmente independientes del navegador (lo que no ocurría con otros lenguajes basados en scripts, como JavaScript o VisualBasic Script), independientes de la plataforma y de la Base de Datos.

Perl ha sido el lenguaje que ha servido como estándar para construir CGIs durante mucho tiempo, y aún sigue siendo una de las mejores soluciones para desarrollar aplicaciones Web portables, ya que trabaja sobre cualquier

servidor Web que soporte CGIs, y sobre cualquier plataforma que soporte Perl, que son la mayoría, incluso ha servido para desarrollar módulos que extienden la funcionalidad de los servidores. Pero nada tiene que envidiar PHP. PHP, está más orientado a conexiones entre páginas Web y servidores donde se almacenan toda clase de Bases de Datos.

Soporta un único "Safe Mode", es decir, un mecanismo de seguridad que permite que varios usuarios estén corriendo scripts PHP sobre el mismo servidor. Este mecanismo está basado en un esquema de permisos de ficheros, permitiendo el acceso a aquellos ficheros que son apropiados por el mismo identificador de usuario que el del script que está intentando acceder a ese fichero, o bien cuando el fichero está en el directorio que es propiedad del mismo identificador de usuario que el del script que está intentando acceder.

PHP es un lenguaje de programación soportado por HTML. La sintaxis está heredada de C, Java y Perl. Este lenguaje está orientado para los constructores de páginas Webs, permitiendoles crear páginas dinámicamente generadas de forma rápida.

Seguridad y PHP

PHP es un interprete que puede ser incluido en un servidor Web como un módulo o como un CGI binario.

Con él se pueden realizar accesos a ficheros, conexiones de red, etc. PHP está diseñado para ser más seguro que cualquier otro lenguaje de programación de CGIs, como Perl o C.

Manejo de Errores

En PHP hay cuatro tipos de errores:

- *Funciones de error*
- *Warning*
- *Errores de parse*
- *Notices*

El nivel de error por defecto es 7 (1+2+4), pero esto puede ser modificado en el fichero de configuración php3.ini con la directiva error_reporting
Cualquier expresión en PHP se puede llamar con la "@" al principio que invocará a la función de manejo de errores, y si track_errors está activada, el error podremos encontrarlo en la variable $php_errormsg.

Problema de la autentificación de un Usuario con PHP

Para resolver este problema, se utiliza la función Header(). Esta función debe ser llamada antes de que se haya producido ninguna salida del fichero

html que se está generando. Esta función envía una línea a la cabecera HTML que se está generando. Y con esta función es posible enviar un mensaje de "Autentificación Requerida " que forzará al usuario a completar los datos solicitados.

El funcionamiento es muy sencillo. Una vez que el usuario ha completado los datos que se le piden, se fuerza una nueva llamada al script con las variables $PHP_AUTH_USER, $PHP_AUTH_PW y $PHP_AUTH_TYPE que contendrán el nombre del usuario, el pasword y el tipo de autentificación.

Hay que tener mucho cuidado con el orden en el que se envian los diferentes tipos de cabeceras. Esta cabecera de autentificación, debe ir siempre delante de la del tipo HTTP/1.0 401.

Cookies

Para PHP las cookies son fáciles de utilizar. Para esto se utiliza la función setcookie() y al igual que la función anterior, Header, tiene que ser invocada antes de producir ninguna salida.

La función posee el siguiente prototipo:

```
int setcookie(string name, string value, int expire, string path,
string domain, int secure);
```

Esta función define la cookie que se va a enviar y posee todos los argumentos opcionales a excepción del nombre, aunque los no especificados

deben ser indicados "", menos los parametros expire que es un entero regular que puede devolver la función mktime(), y secure, que indica si el cookie deberia ser utilizado sólo sobre conexiones seguras, enteros que deben ser sustituidos por un 0.

Se pueden utilizar vectores de cookies.

```
setcookie ("cookie[three]", "cookiethree");

setcookie ("cookie[two]", "cookietwo");

setcookie ("cookie[one]", "cookieone");

if(isset($cookie))
  {
    while (list($name,$value) = each ($cookie))

      {
        echo "$name == $value\n";

      }
  }
```

Manejo de conexiones

En PHP las conexiones que se mantinen pueden tener tres estados, Normal (0), Aborted (1) y Timeout (2).

En un script normal, el estado es NORMAL, cuando el cliente desconecta, el estado pasa a ser ABORTED y si el límite impuesto por PHP-imposed ha transcurrido, (set_time_limit(), el tiempo por defecto es 30 segundos) el estado es TIMEOUT.

Una función muy util para estos casos, es connection_status() que devuelve el estado de la conexión.

Conexiones a bases de datos

Las conexiones persistentes son enlaces SQL que no se cierran cuando la ejecución del script termina. El comportamiento de estas conexiones es el siguiente.

Cuando se invoca una conexión de este tipo, PHP comprueba si existe una conexión de este mismo tipo o por el contrario, se trata de una nueva conexión. En el caso de que exista, se procede a su uso, y en el caso de que no exista, la conexión se crea. Dos conexiones se consideran iguales cuando están realizadas sobre el mismo servidor, con el mismo usuario y la misma contraseña.

Pero en realidad, estas conexiones permanentes, no proporcionan ningún tipo de función habilidad adicional frente a conexiones temporales, debido a la forma en que los servidores Web funcionan.

Aún así se utilizan debido a la eficiencia, debido al tiempo de establecimiento de la conexión, y debido a que si tienes una sóla conexión

sobre el servidor, irá mucho más rápido que si tienes 10 conexiones temporales, puesto que la carga que soporta es diferente.

Ventajas de php

- *Libre.*
- *Abierto.*
- *Código fuente disponible.*
- *Diseñado para la web.*
- *Multiplataforma HW.*
- *Multisistema Operativo.*
- *Soporte para varios servidores web.*
- *Soporte nativo para practicamente cualquier Base de Datos.*
- *Buena documentación.*
- *Miles de ejemplos y código fuente disponible.*
- *Perfecta integración del Apache-PHP-MySQL.*
- *Sintaxis clara y bien definida.*
- *Bastante sencillo de aprender y utilizar.*
- *Modulable.*
- *Seguro (evidentemente tiene errores pero se solucionan mucho antes que otros sistemas propietarios)*
- *Amplia base de usuarios (Ahora es el número 1, como lo es también* ***Apache****).*

- *No dependes de un único proveedor de servicios.*

ASP

Caracteristicas

Veremos algunas de las caracteristicas mas importantes de esta tecnologia, las otras las descubriremos a medida que se avanza en este tutorial.

- *La sigla ASP significa Active Server Pages.*
- *Microsoft es el creador.*
- *Se ejecuta en el servidor donde esta alojada la pagina web.*
- *El cliente (navegador del usuario) no nota diferencias con una pagina normal (HTML).*
- *Por medio de ASP se puede tener acceso a bases de datos.*
- *Para su implementacion se utiliza lenguaje script como VBScript o JScript.*
- *Se puede utilizar HTML y ASP en una misma pagina.*
- *Es totalmente ampliable gracias a que es compatible con la tecnologia ActiveX.*

Como Funciona

Aqui estan pasa a paso como funciona una pagina ASP.

1. *Un usuario por medio del navegador solicita una pagina ASP.*

2. *Esta solicitud llega al servidor el cual tiene alojada la pagina pedida.*
3. *Este servidor procesa la pagina ASP y devuelve codigo HTML.*
4. *El usuario visualiza la pagina en su navegador.*

Noten que para el usuario no existe diferencia en ASP y HTML por que a su navegador siempre llega codigo HTML puro, el unico que requiere un trabajo extra es el servidor el cual tiene que procesar el codigo ASP y tranformarlo en HTML para su posterior envio al cliente.

Como utilizarlo

Para utilizarlo nesecitamos un servidor que procese nuestras paginas ASP, para ello existe el Internet Information Sever bajo Windows NT o 2000 y bajo Windows 95/98 esta el Personal Web Server que viene el CD de Instalacion de Windows 98 o en el CD del Visual Studio.

Personal Web Server

El Personal Web Server es un servidor que nos permite trabajar con paginas ASP en nuestra PC sin necesidad de subir las paginas ASP al servidor para realizar las pruebas, es decir, armamos todo el sitio en nuestra PC, lo probamos con el Personal Web Server (PWS) y si todo va bien lo subimos.
Para utilizar el PWS necesitamos tener Windows 95/98/Me, si tenemos como SO Windows NT o 2000 tambien podemos utilizar el PWS pero en este caso es preferible utilizar Internet Information Server (IIS) dado que es mas funcional y robusto. Pueden conseguir el PWS desde el sitio de Microsoft o en los CDs de Instalacion de Windows 98, Visual Basic o Visual Studio.

A la hora de instalar el PWS hay que seguir los pasos de una instalacion tipica de un programa, pero hay que prestar atencion en el momento que nos pide el directorio de publicacion web, es decir el directorio en donde se guardaran nuestras paginas web se puede dejar el que nos ofrese el PWS por defecto o lo podemos modificar, si lo modificamos, por ejemplo ponemos que el directorio de publicacion web es "C:\Paginas Webs" tenemos que guardar nuestras pagina dentro de este directo.

Una ves instalado reiniciamos la PC y al volver tendremos un icono en la barra de notificacion de Windows desde este icono podemos iniciar el PWS, aclaremos que el PWS tiene dos estados Iniciado y Detenido, obviamente nuestro sitio solo funcionara cuando este el PWS iniciado.

Bueno una ves abierto el PWS nos vamos al sector llamado "Avanzada" desde aca configuramos las opciones avanzadas del servidor, lo mas importante es la configuracion del directorio, seleccionamos "Home" y hacemos click en "Modificar Propiedades", aqui seleccionamos el directorio que habiamos elegido como publicacion web, apretamos "Aceptar" y volvemos a las opciones avanzadas. Ahora destildamos "Habitar Documento Predeterminado" y habilitamos "Permitir Exploracion de directorios", volvemos a la seccion "Principal" y hacemos click en el link para abrir nuestro site.

VBScript - Introduccion

Por medio de VBScript podemos crear nuestras paginas ASP, debemos tener en cuenta que antes de escribir cualquier codigo ASP debemos abrir nuestro codigo con el simbolo '<%' y al finalizar cerrarlo con '%>'

```
<%
'este es un comentario
Response.Write ("Mi Primer Pagina ASP")
%>
```

Podemos abrir archivo nuevo agregarle el codigo anterior guardarlo como "hola.asp" dentro de nuestro directorio de publicacion web abrir el PWS y ver el resultado en el navegador.

VBScript es un lenguaje script derivado de Visual Basic asi que tambien hereda sus funciones basicas a continuacion veremos algunas de las mas utilizadas.

VBScript - Funciones Basicas

Si estas familiarizado con Visual Basic podes pasar esta parte dado que los funciones basicas son iguales en ambos lenguajes, y si no conoces Visual Basic aca va una ayuda.

Variables

La forma de declaran variables es por medio de la palabra 'Dim', aclaremos que la declaracion de variables no es necesaria, salvo que al inicio de nuestra pagina ASP pongamos la palabra clave 'Option Explicit' que hace que la declaracion de variables sea necesaria si no nos daria un error.

```
<%
Option Explicit

Dim I 'declaramos la variable

I= 14 ' le asignamos un valor
Response.Write (I) 'mostramos su valor en el navegador
%>
```

Condicionales

El condicional mas utilizado en cualquier lenguaje es el famoso IF, VBScript no podia quedarse afuera, el modo de uso es poner la palabra clave 'If' despues la condicion a evaluar, luego 'Then' y el codigo a ejecutar si se cumple la condicion dada, al finalizar se cierra el bloque 'If' con la palabra clave 'End If'

```
<%
Dim I

I = 14
```

```
If I = 14 Then
Response.Write ("La variable I es igual a 14")
End If
%>
```

Tambien extiste la alternativa 'Else' (sino), es decir sino se cumple la condicion hacer lo siguiente, veamos un ejemplo

```
<%
Dim I
I = 14

If I = 14 Then
Response.Write ("La variable I es igual a 14")
Else
Response.Write ("La variable I es distinta a 14")
End If
%>
```

Fijense que pusimos 'I = 14' tambien podemos hacer otro tipo de condicion como puede ser distinto 'I <> 14', mayor 'I > 14', menor 'I < 14', mayor o igual 'I > = 14', menor o igual 'I <= 14'< /FONT >

Bucles

VBScript posee un bucle llamador 'For' que repite una accion la cantidad de veces indicada

```
<%
Dim I

For I = 1 to 5
Response.Write ("Esto se repite 5 veces")
Next
%>
```

Podemos modificar el numero 5 y poner un 10 y lo repetira 10 veces, tambien podemos modificar el numero 1 por el 3 y el 5 por el 6, esto hara que el bucle se repita 3 veces, '3 to 6' (desde 3 hasta 6), tambien podemos indicar el incremento que tendra por medio de la palabra clave 'step'

```
<%
Dim I

For I = 1 to 10 step 2
Response.Write ("Esto se repite de dos en dos, i = " & ij
Next
%>
```

Gracias a esto podemos hacer un bucle inverso por ejemplo de 10 a 1

```
<%
Dim I
```

```
For I = 10 to 1 step -1
Response.Write ("i = " & i)
Next
%>
```

La palabra clave 'Next' indica el final del bucle, es decir se repite todo lo que esta entre el 'For' y 'Next'

While..Wend

Este bucle a diferencia del 'For' repite una accion siempre que se cumpla una condicion, veamos un ejemplo

```
<%
Dim I

I = 0 'inicializa i igual a 0

While I <> 10 'repita mientras i sea distinto a 10
Response.Write ("i = " & i) 'muestra por pantalla el valor de i
I = I + 1 'incrementa i en unc
Wend
%>
```

Cuando agregamos la palabra clave 'Wend' estamos diciendo que repita todo el codigo incluido entre While y Wend

Objeto Response

Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, su funcion es enviar informacion al cliente (navegador).

PROPIEDADES

Buffer

Esta propiedad es del tipo boolean (true/false) e indica si la pagina se alamacena en el buffer del servidor antes de ser enviada, es decir si esta true envia la pagina al cliente recien cuando este toda cargada y si esta false la envia a medida que la procesa. Se debe utilizar antes de haber enviado algun resultado al diente. Si el servidor utilizado tiene la version de ASP 2.0 el valor predeterminado de esta propiedad es false y en cambio la version es la 3.0 (Windows 2000) el valor predeterminado es true.

Ejemplo

<%Response.Buffer = TRUE%>

Expires

Esta propiedad indica el tiempo, en minutos, que la pagina se guarda el ei

cache del navegador; si este valor se pone a 0 no se utiliza el cache del navegador

Ejemplo

<% Response.Expires = 5%>

ExpiresAbsolute

Esta propiedad indica hasta que dia y hora la pagina se guarda en el cache del navegador.

Ejemplo

<% Response.ExpiresAbsolute=#May 5, 2001 14:30:00# %>

Cookies

Crea una cookie en la maquina del usuario. Una cookies es un archivo con formato de texto en el cual el se puede guardar informacion. Las cookies por lo general se utilizan para controles de acceso, personalizacion de paginas y estadisticas. El siguiente ejemplo crea una cookie llamada 'fecha' la cual contiene el valor devuelto por la funcion 'date' que nos devuelve la fecha

actual

Ejemplo

<% Response.Cookies("fecha") = date %>

Esta cookie se mantiene en la maquina del usuario hasta que termine su session, borre las del navegador o se cumpla el tiempo establecido para esa cookies, a continuacion veremos la forma de establecer la duracion de una cookie

<% Response.cookies("fecha").expires="2/12/2001"%>

Esto hace que la cookie se borre en la fecha indicada, si creamos otra cookie llamada 'fecha' esta sobreescribira a la otra.

METODOS

End

Este metodo detiene el proceso de la pagina ASP y envia lo ya procesado al navegador

Ejemplo

<%Response.End%>

Flush

Este metodo envia el contenido del buffer al navegador siempre y cuando la propiedad Buffer se true, si la propiedad Buffer es igual a false se producira un error

Ejemplo

<%Response.Flush%>

Redirect (cadena_url)

Este metodo redirecciona la pagina actual a la pasada como parametro.

Ejemplo

<%Response.Redirect "otra_pagina.asp"%>

Write (cadena)

Este metodo escribe en el navegador la cadena padada como parametro, tengan en cuenta que tambien se pueden enviar etiquetas HTML y concatenar variable con el operador & (VBScript).

Ejemplo

<%Response.Write ("hola!!!%>

Objeto Request

Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, por medio de este objeto podes tener acceso a la informacion que el navegador pasa al servidor.

PROPIEDADES (COLECCIONES)

Form

Esta propiedad es una coleccion que contiene todos los elementos de un formulario (<form>) pasado al archivo ASP con el metodo POST (<form method= 'post'>),la forma de acceder a el valor de un elemento del formularic es la siguiente...</FONT></FONT></FONT>

<%=Request.Form("nombre_elemento")%>

Ejemplo

```
-------formulario.htm-------
<HTML>
<HEAD>
</HEAD>
<BODY>
<form action="procesa_formulario.asp" method="POST">
Nombre:<input type="Text" name="nombre" maxlength="20"><br>
Apellido:<input type="Text" name="apellido" maxlength="30"><br>
<input type="Submit" name="Submit" value="Enviar">
</form>
</BODY>
</HTML>
--------procesa_formulario.asp----
<HTML>
<HEAD>
</HEAD>
<BODY>
<% Response.Write ("nombre:" & Request.Form("nombre"))%>
<% Response.Write ("apellido:" & Request.Form("apellido")) %>
</BODY>
</HTML>
```

Cookies

Recupera una cookie almacenada en la maquina del usuario. Una cookies es un archivo con formato de texto en el cual el se puede guardar informacion. Las cookies por lo general se utilizan para controles de acceso, personalizacion de paginas y estadisticas. El siguiente ejemplo recupera la cookie llamada 'fecha' que creamos anteriormente.

Ejemplo

```
<% Response.Write Request.Cookies("fecha") %>
```

Debido a la natualeza de las cookies y a que nunca podemos garantizar que una cookie almazenada anteriormente todavia exista debemos manejarnos con cautela y no almacenar en ellas datos criticos.

QueryString

Esta coleccion contiene todos los valores pasados como parametros a una pagina ASP. La formas de pasar parametros a una pagina ASP son varias una es por medio de formularios utilizando el metodo 'GET' y la otra es generandolo 'manualmente'...

http://www.servidor_xxx.com/pagina_asp.asp?nombre=jose&apellido=perez

Esa es la manera de pasar parametros a una pagina ASP, luegop del la direccion URL se pone el signo '?' y a continuacion todos los parametros con sus respectivos valores separados por el simbolo '&'.

Ejemplo

```
----------pasa_parametros.asp----------
<HTML>
<HEAD>
</HEAD>
<BODY><BR><AHREF="lee_parametros.asp?nombre=jose">click aca! </A>
</BODY>
</HTML>

----------lee_parametros.asp----------
<%
Response.Write (Request.QueryString("nombre") & "<br>")
%>
```

ServerVariables

Esta coleccion nos da acceso a todas las variables de entorno del servidor.

Ejemplo

```
<%
Response.Write (Request.ServerVariables("REMOTE_ADDR"))
Response.Write (Request.ServerVariables("SERVER_SOFTWARE"))
%>
```

Objeto Server

Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, por medio de este objeto podes tener acceso a funciones utiles del servidor.

PROPIEDADES

ScriptTimeout

Esta propiedad determina el tiempo maximo en segundos que se espera para ejecutar la pagina ASP. Si la pagina tarda mas segundos en ser procesada se produce un error. El valor predeterminado es 90 segundos.

Ejemplo

```
<% Server.ScriptTimeout = 25 %>
```

METODOS

CreateObject (id_objeto)

Este es uno de los metodos mas importantes y nos permite crear objetos a partir de los cuales vamos a trabajar. Como unico parametro se especifica el nombre del objeto a crear. Al crear el objeto este se le asigna a la variable establecida por medio de la palabra clave SET, esta variable hereda todos los metodos, propiedades y eventos del objeto creado.

Ejemplo

```
<% Set BDD = Server.CreateObject("ADODB.Connection") %>
```

Execute (path)

Este metodo ejecuta una pagina ASP dentro de otra, su unico parametro es la ruta de la pagina a ejecutar. Este metodo solo esta disponible en la version 3.0 de ASP.

Ejemplo

------pagina_llama.asp----

<HTML>

<BODY>

Hola <%Server.Execute("respuesta.asp")%> !

</BODY>

</HTML>

--------respuesta.asp------

<%

Response.Write "Pedro

"

%>

HTMLEncode (cadena)

Este metodo evita que la cadena pasada como parametro sea interpretada como lenguaje HTML.

Ejemplo

<% Response.Write Server.HTMLEncode("sin negrita") %>

MapPath (path)

Devuelve la ruta completa del archivo pasado como parametro.

Ejemplo

<% Response.Write Server.MapPath("calendario\dias.asp") %>

Objeto Session

Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, el mismo sireve para compartir informacion de un usuario entre las paginas del sitio. Por defecto el objeto session se crea al ingresar el usuario a nuestro sitio y se destruye cuando lo abandona o cuando pasan 20 minutos sin actividad del usuario.

PROPIEDADES

SessionID

Esta propiedad devuelve un ID que identifica a un usuario. Este ID es unico, no se puede modificar y lo crea automaticamente el servidor.

Ejemplo

<% Response.Write "ID Session = " & Session.SessionID %>

Timeout

Esta propiedad determina el tiempo en minutos que durara la sesion del usuario antes de ser destruida, por defecto su valos es de 20 minitos.

Ejemplo

<% Session.Timeout = 30 %>

METODOS

Abandon

Este metodo proboca el abandono forzado de la session, destruye todas la variables de sesion .

Ejemplo

<%Session.Abandon %>

VARIABLES DE SESION

Una utilidad muy buena que nos permite el objeto session es crear variables y poder compartirlas en todas la paginas de nuestro sitio algo asi como una variable global pero con una diferencia bastante grande dado que cada usuario que ingrese a nuestra pagina tendra un valor distinto en cada variable. En el ejemplo que sigue a continuacion se muestra la forma de uso.

Ejemplo

-------pagina_1.asp-----

```
<HTML>
<BODY>

<%Session("Dia") = "Lunes"%>

</BODY>
</HTML>
```

-------pagina_2.asp-------

```
<%
Response.Write                    Session("Dia")
%>
```

UNIVERSIDAD SIMON BOLIVAR
HEMEROTECA
BARRANQUILLA

Bases de Datos - Utilizando ASP y Bases de Datos Access.

Esta es una breve explicacion de como utilizar ASP para tener acceso a una base de datos, lo primero que debemos crear es la base de datos, en este caso de Access con el nombre de 'temas.mdb', ahora dentro de esta base de datos crearemos una tabla llamada 'TemasMusicales' y que tenga los siguientes campos...

CAMPOS	TIPOS DE DATOS
Id	Autonumerico
Titulo	Texto
Genero	Texto

Ingresale algunos datos para poder utilizarla. Bueno ya tenemos la base de datos con informacion adentro, lo que vamos a hacer ahora es desde ASP ver todo el contenido de la tabla que hemos creado. Para ello creamos un archivo ASP con el nombre 'muestra_temas.asp' y en su interior agregamos el siguiente codigo...

```
<%

Dim BDD ' variable que se encargara de la base de datos
Dim TABLA ' variable que se encarga de la tabla
```

```
'creamos los objetos que manejan la base de datos y la tabla
Set BDD = Server.CreateObject("ADODB.Connection")
Set TABLA = Server.CreateObject("ADODB.Recordset")

'abrimos la base de datos
BDD.Open ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA SOURCE=" +
Server.MapPath("temas.mdb"))

'abrimos la tabla
TABLA.Open "TemasMusicales", BDD ' indicamos el nombre de la tabla y la
base de datos

'ahora recorremos toda la tabla y mostramos los datos
While NOT TABLA.EOF 'recorre la tabla hasta que EOF sea verdadero
Response.Write "titulo:" & TABLA("Titulo") & "<br>"
Response.Write "genero:" & TABLA("Genero") & "<br>"
Wend

'cerramos todo y liberamos los objetos
TABLA.Close
BDD.Close

Set BDD = Nothing
Set TABLA = Nothing

%>
```

Las Fortalezas Y Debilidades De Php, Asp Y Asp .Net

para la construcción de sitios dinâmicos. Resumiendo, PHP tiene todo un momento gracias a la enorme cantidad de productos libres con los que puede interactuar fácilmente y su gran cantidad de librerías disponibles. Contra ésto ASP no queda muy bien parado, pero es salvado por su omnipresente disponibilidad en todos los servidores Windows NT 4 existentes. ASP . NET es otra historia...

Casi tan incompatible con ASP como con PHP, ASP . NET es literalmente una nueva plataforma atada íntimamente al sistema operativo para todas sus tareas, lo que lo hace potencialmente tan rápido de ejecutar como cualquier otro código compilado. Eso y su innegable y fácil acceso al resto de los productos Microsoft puede ser una ventaja competitiva, si el interesado está preparado para dominar una tecnología que exige su precio en consesiones.

Pero peor es el hecho de que aún dominando .NET en general, la disponibilidad de servidores remotos para esta plataforma es todavía escasa, y nada barata..

PERL

El lenguaje Perl (Practical Extraction and Report Language), cuyas siglas significan lenguaje práctico de extracción y generación de informes, es una de las utilidades de programación que más rápidamente se ha extendido en los últimos años. Fue creado por Larry Wall, como una mezda entre C y los intérpretes de comandos de UNIX. Se utiliza mucho en los sistemas UNIX, para procesar ficheros de texto, o automatizar tareas comunes de la administración de estos equipos. Perl se distribuye gratuitamente, con su código fuente completo, según la licencia pública de GNU, que permite utilizar y distribuir copias modificadas de Perl, siempre que se haga disponible su código fuente, en las mismas condiciones.

Perl utiliza una sintaxis similar a la de C, por lo que es muy fácil de aprender para los programadores que conocen C. Las principales ventajas sobre C se encuentran en el manejo de memoria (no es necesario utilizar punteros y el propio Perl se encarga de gestionar la memoria) y el procesamiento de cadenas (integra las funciones de búsqueda y substitución de sed y awk); las últimas versiones induyen la posibilidad de realizar programación orientada a objetos.

Se trata de un lenguaje interpretado, pero esto no significa que sea lento; el interprete Perl lee el código fuente, lo comprueba, y lo traduce a un formato adecuado para su rápido procesamiento. Si bien no es tan rápido como una aplicación compilada, obtiene tiempos de ejecución mucho mejores que los tradicionales intérpretes de comandos de UNIX.

Formato de un programa Perl

Los programas Perl son ficheros de texto plano, que procesa el intérprete Perl, tras realizar una serie de comprobaciones sencillas sobre la corrección del código y compilarlo a un formato que permite una ejecución eficiente.

Para ***comentar una línea****, se utiliza el carácter* ***#****. El carácter # puede aparecer en cualquier punto de una línea y su efecto se aplica desde el punto de inserción hasta el final de la línea. No existe una forma de comentar bloques completos.*

Para ***ejecutar un programa****, se realiza una llamada al intérprete Perl, pasándole el nombre del fichero Perl.*

perl nombre_de_programa.pl

En los sistemas UNIX es común convertir el fichero en ejecutable, e indicar al shell de UNIX cuál es el intérprete de ficheros Perl. Para ello, la primera línea del programa debería contener la línea #!/usr/local/bin/perl (o el directorio en que esté instalado Perl).

En otros entornos es necesario indicar de otra forma la asociación entre los ficheros Perl y su intérprete. En el caso de los sistemas MS-Windows, por ejemplo, su puede asignar una extensión común para los ficheros Perl (p.e. .pl), y asociar el intérprete de Perl con esta extensión.

El intérprete Perl dispone de numerosas opciones en la línea de comandos, para controlar la forma en que se ejecutan los programas. Las más importantes son:

El intérprete Perl realiza comprobaciones adicionales sobre la corrección del código, avisándonos de muchos problemas comunes que suelen afectar al comportamiento del programa.

Eejecuta el programa desde un entorno de depuración.

Muestra las opciones de llamada disponibles para este entorno.

Tipos básicos de datos

*Perl dispone de un conjunto reducido de tipos de datos, orientados principalmente al manejo de cadenas. Todas las variables comienzan con un carácter especial (**$**, **@**, **%**) que denota su tipo. Las variables se definen en su primer uso, de forma automática, o a través de alguno de los operadores de declaración.*

Los nombres de variables son sensibles a las combinaciones de mayúsculas y minúsculas. Pueden contener letras, números y subrayados. Sin embargo, no deben comenzar por un número. Algunos aspectos de especial interés son:

No hay que preocuparse del tamaño de las variables. Perl asigna memoria automáticamente en función de las necesidades.

Las variables se definen automáticamente en su primer uso, y son globales, salvo que se diga lo contrario.

*La variable **$_** es especial. Se utiliza como valor por defecto en la llamada*

a funciones en las que falta un parámetro.

El operador de asignación es el **=**.

Para manejar literales alfanuméricos se utilizan los separadores ' ' y " ". El significado de los mismos es equivalente al que tienen en UNIX. Dentro de las comillas simples no se realiza ninguna interpretación del contenido. Dentro de las comillas dobles, el intérprete de Perl puede buscar e interpretar determinadas expresiones, por ejemplo, nombres de variables.

Todas las sentencias Perl deben terminar con un carácter **;**.

Ventajas de perl

- *Construcción de pequeños programas que pueden ser usados como filtros para obtener información de archivos, realizar búsquedas, ...*
- *Se puede utilizar en varios entornos, como puede ser desde Windows 95, ..., sin realizar cambios de código, siendo únicamente necesario la introducción del interprete PERL correspondiente a cada sistema operativo.*
- *También es uno de los lenguajes más utilizados en la programación de CGI scripts, que son guiones o scripts que utilizan el interface CGI (Common Gateway Interface), para intercambio de información entre aplicaciones externas y servicios de información.*
- *El mantenimiento y depuración de un programa en PERL es mucho más sencillo que la de cualquier programa en C.*

MOTORES DE BASES DE DATOS MAS UTILIZADOS CON OLOS LENGAUJES DE PROGRAMACION ORIENTADOS A LA WEB

MYSQL

Instalación de mysql en Windows

Pasos a seguir para la instalación de la base de datos MySQL y algunas ayudas básicas para comenzar con buen pie.

Uno de los puntos fuertes de las páginas en PHP es la posibilidad de explotar bases de datos mediante funciones de una simplicidad y potencia muy agradecidas. Estas bases de datos pueden servir a nuestro sitio para almacenar contenidos de una forma sistemática que nos permita clasificarlos, buscarlos y editarlos rápida y fácilmente.

Una base de datos es sencillamente un conjunto de tablas en las que almacenamos distintos registros (artículos de una tienda virtual, proveedores o clientes de una empresa, películas en cartelera en el cine...). Estos registros son catalogados en función de distintos parámetros que los caracterizan y que presentan una utilidad a la hora de clasificarlos. Así, por ejemplo, los artículos de una tienda virtual podrían catalogarse a partir de distintos campos como puede ser un número de referencia, nombre del

artículo, descripción, precio, proveedor...

La base de datos más difundida con el tandem UNIX-Apache es sin duda MySQL. Como para el caso de Apache, una versión para Windows está disponible y puede ser ***descargada*** *gratis.*

Su puesta a punto no entraña mucha dificultad. Una vez instalado el programa podemos ejecutar nuestras ordenes en modo MS-DOS. Para ello abrimos una ventana MS-DOS y nos colocamos en el directorio bin de mysql. En este directorio se encuentran los archivos ejecutables. Aquí habrá que encontrar un archivo llamado mysqld. En el caso de la versión más actual durante la redacción de este articulo este archivo es llamado mysqld-shareware. Una vez ejecutado este archivo podemos ejecutar el siguiente: mysql.

Llegados a este punto veremos cómo un mensaje de bienvenida aparece en nuestra pantalla. En estos momentos nos encontramos dentro de la base de datos. A partir de ahí podemos realizar todo tipo de operaciones por sentencias SQL.

No vamos a entrar en una explicación pormenorizada del funcionamiento de esta base de datos ya que esto nos daría para un manual entero. Daremos como referencia nuestro ***tutorial de SQL*** *a partir del cual se puede tener una idea muy práctica de las sentencias necesarias para la* ***creación*** *y*

edición de las tablas. También existe una documentación extensa en inglés en el directorio Docs de MySQL. A modo de resumen, aquí os proponemos además las operaciones más básicas que, combinadas nuestro ***tutorial de SQL*** *pueden dar solución a gran parte de los casos que se os presenten:*

Instrucción	***Descripción***
Show databases;	*Muestra el conjunto de bases de datos presentes en el servidor*
Use nombre_de_la_base	*Determina la base de datos sobre la que vamos a trabajar*
Create Database nombre_de_la_base;	*Crea una nueva bd con el nombre especificado*
Drop Database nombre_de_la_base;	*Elimina la base de datos del nombre especificado*
Show tables;	*Muestra las tablas presentes en la base de datos actual*
Describe nombre_de_la_tabla;	*Describe los campos que componen la tabla*
Drop Table nombre_de_la_tabla;	*Borra la tabla de la base de datos*
Load Data Local Infile "archivo.txt" Into Table	*Crea los registros de la tabla a partir de un fichero de texto en el que separamos por*

nombre_de_la_tabla;	*tabulaciones todos los campos de un mismo registro.*
Quit	*Salir de MySQL*

Para evitarnos el tener que editar nuestras tablas directamente sobre archivos de texto, puede resultar muy práctico usar cualquier otra base de datos con un editor y exportar a continuación la tabla en un archivo de texto configurado para dejar tabulaciones entre cada campo. Esto es posible en Access por ejemplo pinchando con el botón derecho sobre la tabla que queremos convertir y eligiendo la opción exportar. Una ventana de dialogo aparecerá en la que elegiremos guardar el archivo en tipo texto. El paso siguiente será elegir un formato delimitado por tabulaciones sin cualificador de texto.

Otra posibilidad que puede resultar muy práctica y que nos evita trabajar continuamente tecleando órdenes al estilo de antaño es servirse de programas en PHP o Perl ya existentes y descargables en la red. El más popular sin duda es ***phpMyAdmin****. Este tipo de scripts son ejecutados desde un navegador y pueden ser por tanto albergados en nuestro servidor o empleados en local para, a partir de ellos, administrar MySQL de una forma menos sufrida.*

Asimismo, dentro del directorio bin de MySQL, podemos encontrar una pequeña aplicación llamada MySqlManager. Se trata de una interface windows, más agradable a la vista y al uso que la que obtenemos ejecutando el archivo mysql. En este caso, las sentencias SQL deben realizarse sin el punto y coma final.

Tipos de datos que maneja MYsql

Listado y descripción de los distintos tipos de datos de MySQL.

Después de la fase de diseño de una base de datos, y una vez se ha realizado el paso a tablas del mismo, en necesario crear las tablas correspondientes dentro de la base de datos. Para cada campo de cada una de las tablas, es necesario determinar el tipo de datos que contiene, para de esa forma ajustar el diseño de la base de datos, y conseguir un almacenamiento óptimo con la menor utilización de espacio. El presente artículo describe cada uno de los tipos de datos que puede tener un campo en Mysql, para la versión 4.xx.xx

Los tipos de datos que puede haber en un campo, se pueden agrupar en tres grandes grupos:

1. *Tipos numéricos*

2. *Tipos de Fecha*

3. *Tipos de Cadena*

1 Tipos numéricos:

Existen tipos de datos numéricos, que se pueden dividir en dos grandes grupos, los que están en coma flotante (con decimales) y los que no.

TinyInt: *es un número entero con o sin signo. Con signo el rango de valores válidos va desde -128 a 127. Sin signo, el rango de valores es de 0 a 255*

Bit ó Bool: *un número entero que puede ser 0 ó 1*

SmallInt: *número entero con o sin signo. Con signo el rango de valores va desde -32768 a 32767. Sin signo, el rango de valores es de 0 a 65535.*

MediumInt: *número entero con o sin signo. Con signo el rango de valores va desde -8.388.608 a 8.388.607. Sin signo el rango va desde 0 a16777215.*

Integer, Int: *número entero con o sin signo. Con signo el rango de valores va desde -2147483648 a 2147483647. Sin signo el rango va desde 0 a 429.4967.295*

BigInt: *número entero con o sin signo. Con signo el rango de valores va desde -9.223.372.036.854.775.808 a 9.223.372.036.854.775.807. Sin signo el rango va desde 0 a 18.446.744.073.709.551.615.*

Float: *número pequeño en coma flotante de precisión simple. Los valores válidos van desde -3.402823466E+38 a -1.175494351E-38, 0 y desde 1.175494351E-38 a 3.402823466E+38.*

xReal, Double: *número en coma flotante de precisión doble. Los valores permitidos van desde -1.7976931348623157E+308 a -2.2250738585072014E-308, 0 y desde 2.2250738585072014E-308 a 1.7976931348623157E+308*

Decimal, Dec, Numeric: *Número en coma flotante desempaquetado. El número se almacena como una cadena*

Tipo de Campo	***Tamaño de Almacenamiento***
TINYINT	*1 byte*
SMALLINT	*2 bytes*
MEDIUMINT	*3 bytes*

INT	*4 bytes*
INTEGER	*4 bytes*
BIGINT	*8 bytes*
FLOAT(X)	*4 ú 8 bytes*
FLOAT	*4 bytes*
DOUBLE	*8 bytes*
DOUBLE PRECISION	*8 bytes*
REAL	*8 bytes*
DECIMAL(M,D	*M+2 bytes sí D > 0, M+1 bytes sí D = 0*
NUMERIC(M,D)	*M+2 bytes if D > 0, M+1 bytes if D = 0*

2 Tipos fecha:

A la hora de almacenar fechas, hay que tener en cuenta que Mysql no comprueba de una manera estricta si una fecha es válida o no. Simplemente comprueba que el mes esta comprendido entre 0 y 12 y que el día esta

INT	*4 bytes*
INTEGER	*4 bytes*
BIGINT	*8 bytes*
FLOAT(X)	*4 ú 8 bytes*
FLOAT	*4 bytes*
DOUBLE	*8 bytes*
DOUBLE PRECISION	*8 bytes*
REAL	*8 bytes*
DECIMAL(M,D	*M+2 bytes sí D > 0, M+1 bytes sí D = 0*
NUMERIC(M,D)	*M+2 bytes if D > 0, M+1 bytes if D = 0*

2 Tipos fecha:

A la hora de almacenar fechas, hay que tener en cuenta que Mysql no comprueba de una manera estricta si una fecha es válida o no. Simplemente comprueba que el mes esta comprendido entre 0 y 12 y que el día esta

comprendido entre 0 y 31.

Date: *tipo fecha, almacena una fecha. El rango de valores va desde el 1 de enero del 1001 al 31 de diciembre de 9999. El formato de almacenamiento es de año-mes-dia*

DateTime: *Combinación de fecha y hora. El rango de valores va desde el 1 de enero del 1001 a las 0 horas, 0 minutos y 0 segundos al 31 de diciembre del 9999 a las 23 horas, 59 minutos y 59 segundos. El formato de almacenamiento es de año-mes-dia horas:minutos:segundos*

TimeStamp: *Combinación de fecha y hora. El rango va desde el 1 de enero de 1970 al año 2037. El formato de almacenamiento depende del tamaño del campo:*

Tamaño	***Formato***
14	*AñoMesDiaHoraMinutoSegundo aaaammddhhmmss*
12	*AñoMesDiaHoraMinutoSegundo aammddhhmmss*
8	*ñoMesDia aaaammdd*
6	*AñoMesDia aammdd*
4	*AñoMes aamm*

2	Año aa

Time: *almacena una hora. El rango de horas va desde -838 horas, 59 minutos y 59 segundos a 838, 59 minutos y 59 segundos. El formato de almacenamiento es de 'HH:MM:SS'*

Year: *almacena un año. El rango de valores permitidos va desde el año 1901 al año 2155. El campo puede tener tamaño dos o tamaño 4 dependiendo de si queremos almacenar el año con dos o cuatro dígitos.*

Tipo de Campo	***Tamaño de Almacenamiento***
DATE	*3 bytes*
DATETIME	*8 bytes*
TIMESTAMP	*4 bytes*
TIME	*3 bytes*
YEAR	*1 byte*

3 Tipos de cadena:

Char(n): *almacena una cadena de longitud fija. La cadena podrá contener desde 0 a 255 caracteres.*

VarChar(n): *almacena una cadena de longitud variable. La cadena podrá contener desde 0 a 255 caracteres.*

Dentro de los tipos de cadena se pueden distinguir otros dos subtipos, los tipo Test y los tipo BLOB (Binary large Object)

La diferencia entre un tipo y otro es el tratamiento que reciben a la hora de realizar ordenamientos y comparaciones. Mientras que el tipo test se ordena sin tener en cuenta las Mayúsculas y las minúsculas, el tipo BLOB se ordena teniéndolas en cuenta.

Los tipos BLOB se utilizan para almacenar datos binarios como pueden ser ficheros.

TinyText y TinyBlob: *Columna con una longitud máxima de 255 caracteres.*

Blob y Text: *un texto con un máximo de 65535 caracteres.*

MediumBlob y MediumText: *un texto con un máximo de 16.777.215 caracteres.*

LongBlob y LongText: *un texto con un máximo de caracteres*

4.294.967.295. Hay que tener en cuenta que debido a los protocolos de comunicación los paquetes pueden tener un máximo de 16 Mb.

Enum: *campo que puede tener un único valor de una lista que se especifica. El tipo Enum acepta hasta 65535 valores distintos*

Set: *un campo que puede contener ninguno, uno ó varios valores de una lista. La lista puede tener un máximo de 64 valores.*

Tipo de campo	***Tamaño de Almacenamiento***
CHAR(n)	*n bytes*
VARCHAR(n)	*n +1 bytes*
TINYBLOB, TINYTEXT	*Longitud+1 bytes*
BLOB, TEXT	*Longitud +2 bytes*
MEDIUMBLOB, MEDIUMTEXT	*Longitud +3 bytes*
LONGBLOB, LONGTEXT	*Longitud +4 bytes*
ENUM('value1','value2',...)	*1 ó dos bytes dependiendo del número*

	de valores
SET('value1','value2',...)	*1, 2, 3, 4 ó 8 bytes, dependiendo del número de valores*

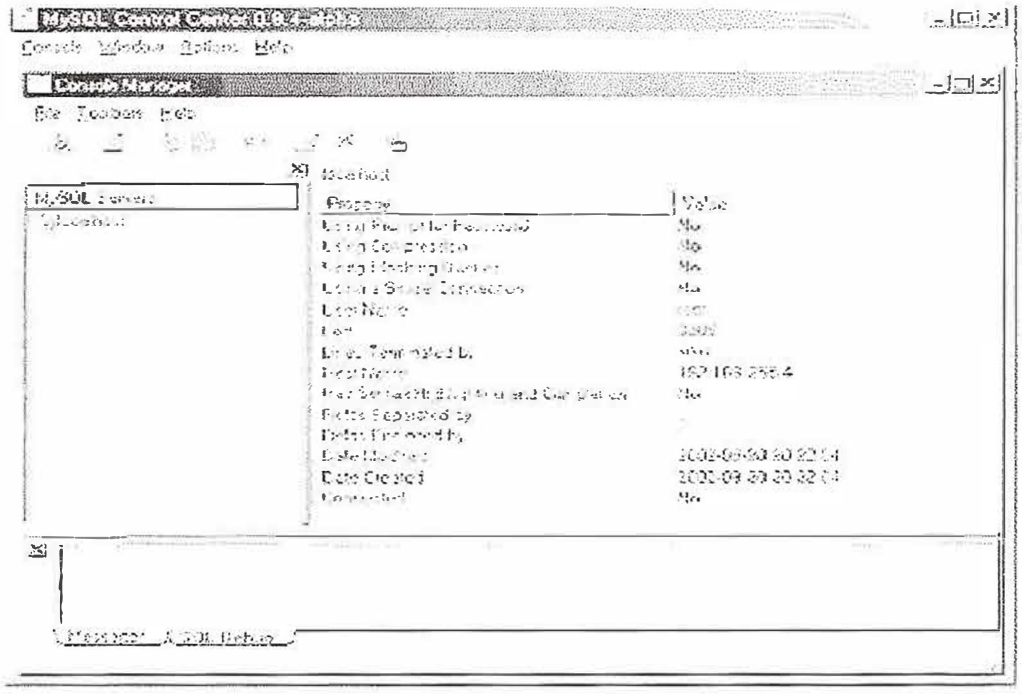
Diferencia de almacenamiento entre los tipos Char y VarChar

Valor	***CHAR(4)***	***Almacenamiento***	***VARCHAR(4)***	***Almacenamiento***
''	''	*4 bytes*	''	*1 byte*
'ab'	*'ab '*	*4 bytes*	*'ab'*	*3 bytes*
'abcd'	*'abcd'*	*4 bytes*	*'abcd'*	
'abcdefgh'	*'abcd'*	*4 bytes*	*'abcd'*	*5 bytes*

Mysql control server

Una consola de Administración de Mysql creada por Mysql AB desde la que se pueden administrar las bases de datos, los usuarios y el servidor de bases de datos

MyCC es una potente consola de Administracíon para Mysql, que permite realizar todas las tareas de administración y trabajo de Mysql mediante un interface gráfico, y que viene a cubrir una de las mayores objeciones que los usuarios tenían de Mysql.

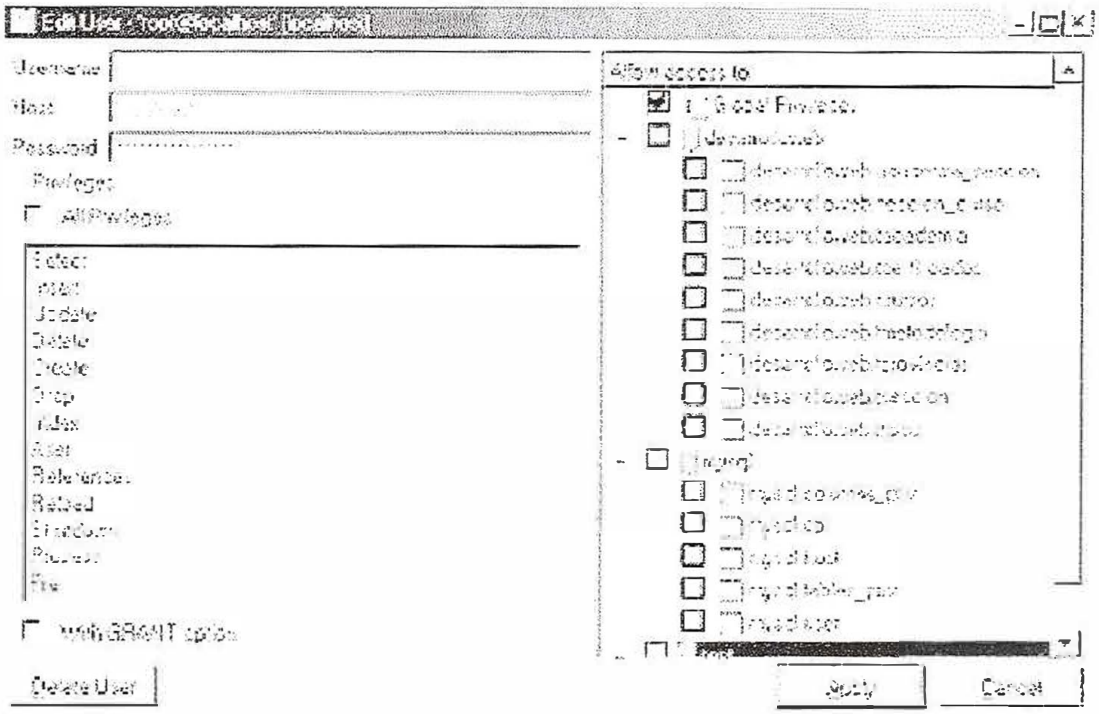


Para Utilizarlo, primero lo tendrás que descargar de Mysql.com. Existen actualmente versiones disponibles para Linux y para Windows. Para poder trabajar con un servidor de Mysql, primero lo tendrás que dar de alta mediante el icono de Resgitrar servidor. Aparece un cuadro de diálogo en el cúal se pueden configurar todos los parámetros de acesso al servidor, como son nombre del servidor, nombre de usuario, contraseña, puerto etc. Una vez conectado a un servidor se podrá acceder a las tareas de administración de

los usuarios de Mysql, administrador del gestor, y administración de las tablas.

Administrador de Usuarios

Desde el se pueden añadir, borrar y modifcar las propiedades de los usarios de Mysql. Al hacer click sobre un usuario aparece la ventana de configuración del usuario.

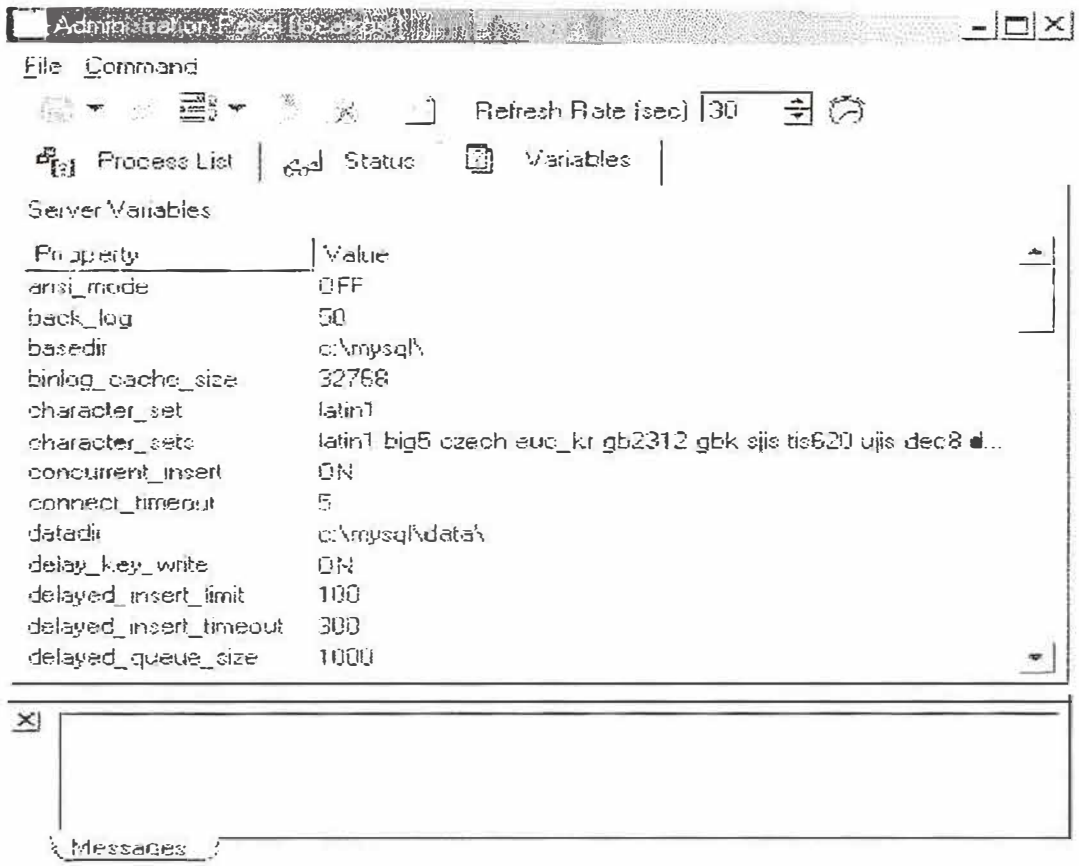


En ella se pueden configurar el nombre de usuario, la contraseña. Respecto a los permisos, permite establecer un sistema de permisos general o individualizado para cada una de las tablas. Además permite darle permiso

al usuario para que se lo ceda a otros (GRANT OPTIONS). Uno de los puntos fuertes respecto a la seguridad es que al definir un usuario, hay que especificar el servidor desde al cual accederá, limitando de esta forma, los riesgos de acceso indebido.

Administración del servidor

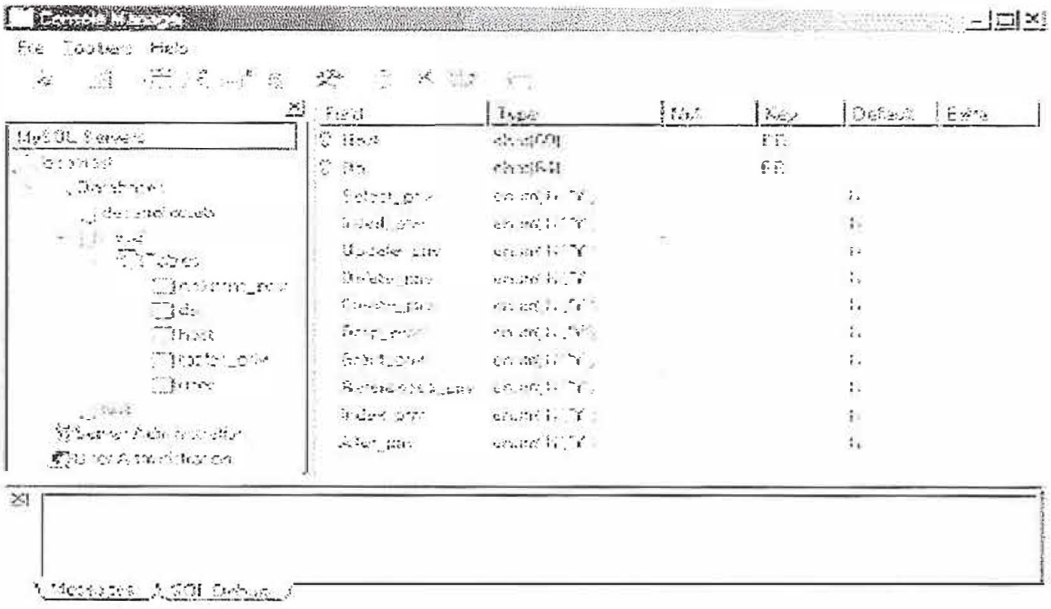
La ventana de Administración del servidor consta de tres pestañas desde la cual se pueden acceder a la lista de procesos, las variables de estado del servidor y las variables de configuración.



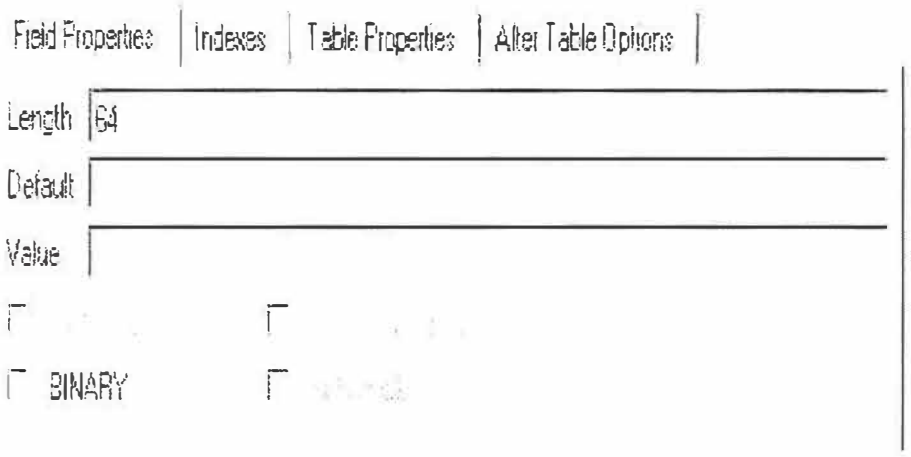
Desde esta ventana además se puede hacer un Ping al servidor, detenerlo, o guardar el contenido de las variables de estado. Desde la pestaña de procesos se puede ver el estado de cada uno de los procesos activos, viendo sus propiedades y con la opción de detenerlos mediante un Kill. Desde la pestaña de estado, se puede ver el contenido de cada una de las variables de estado, y se puede personalizar para mostrar simplemente las que te interesn. Desde la pestaña de variables se puede acceder al valor de cada una de las variables de configuración del servidor de Mysql.

Ventana de Base de Datos

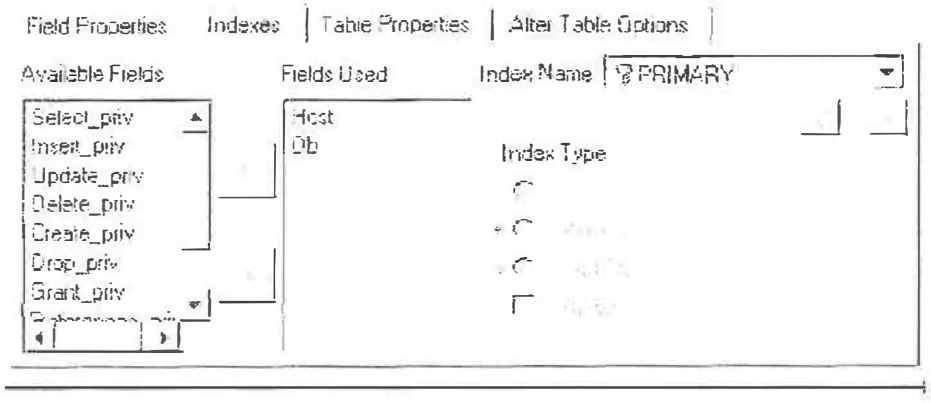
Al seleccionar una base de datos dentro de la sección de bases de datos, aparece la ventana de base de datos, en ella se puede ver información relacionada con la base de datos, como puede ser número de tablas, el número de consultas por segundo, el tiempo que lleva en funcionamiento etc. Al seleccionar una base de datos, se pueden ver los nombres de las tablas que contiene, el número de registros que contiene. Las tablas se pueden eliminar, vaciar y renombrar. Cuando se selecciona una tabla, se pueden ver el nombre de los campos, los tipos de datos y los parámetros opcionales de cada uno de ellos.



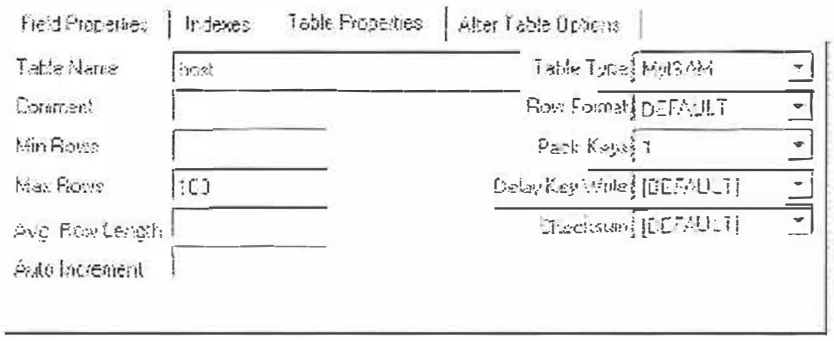
Esta ventana permite crear nuevos campos, eliminarlos o modificar sus propiedades. Además mediante el icono de herramientas, se puede analizar la tabla, optimizarla y repararla. Al hacer doble clic sobre una tabla se muestran en la ventana los datos que contiene. Haciendo clic sobre cualquiera de los campos, se accede a la estructura de la misma. Para cada campo se puede elegir el tipo de datos, el tamaño, si es clave o no, el valor predeterminado del mismo.



Dentro de la tabla, existe un cuadro para configurar los índices que tiene.



Desde ella se pueden seleccionar los campos que lo contienen, así como el tipo de índice que se desea crear. Desde la pestaña de propiedades de la tabla se acceden a los atributos de la misma, como pueden ser el tipo de tabla, el modo de escritura, el tamaño máximo de cada fila etc.



POSTGRE

Instalación de PostgreSQL

Instala PostgreSQL ejecutando los siguientes comandos:

./configure --prefix=/usr &&

make &&

make install

Inicializa un grupo de bases de datos con los siguientes comandos:

mkdir -p /var/pgsql/data &&

useradd -d /var/pgsql/data postgres &&

chown postgres /var/pgsql/data &&

su - postgres -c '/usr/bin/initdb -D /var/pgsql/data'

Arranca el servidor de bases de datos con el siguiente comando:

***su - postgres -c '/usr/bin/postmaster -D /var/pgsql/data > ***

/var/pgsql/data/logfile 2>&1 &'

Ahora podemos crear una base de datos y verificar la instalación:

su - postgres -c '/usr/bin/createdb test'

echo "create table t1 (name varchar(20), state_province varchar(20)

***);" ***

| (su - postgres -c '/usr/bin/psql test ')

***echo "insert into t1 values ('Billy', 'NewYork');" ***

| (su - postgres -c '/usr/bin/psql test ')

***echo "insert into t1 values ('Evanidus', 'Quebec');" ***

| (su - postgres -c '/usr/bin/psql test ')

***echo "insert into t1 values ('Jesse', 'Ontario');" ***

| (su - postgres -c '/usr/bin/psql test ')

echo "select * from t1;" | (su - postgres -c '/usr/bin/psql test')

Explicación de los comandos

useradd -d /var/pgsql/data postgres *: Crea un usuario sin privilegios para lanzar el servidor de bases de datos. Ejecutar el servidor como root es peligroso, y por otro lado, simplemente no funcionará.*

su - postgres -c '/usr/bin/initdb -D /var/pgsql/data' *: Inicializa el espacio de tablas de las bases de datos. Este comando no debe ejecutarse como root.*

su - postgres -c '/usr/bin/postmaster -D /var/pgsql/data > /var/pgsql/data/logfile 2>&1 &' *: Inicia el servidor de bases de datos. Este comando debe ejecutarse también por el usuario postgres.*

createdb test, create table t1 , insert into t1 values..., select * from t1 *: Crea una base de datos, le añade una tabla, inserta varias filas en la tabla y las selecciona para comprobar que la instalación funciona correctamente.*

Configuración de PostgreSQL

Ficheros de configuración

$PGDATA/pg_ident.con, $PGDATA/pg_hba.conf, $PGDATA/postgresql.conf

La variable de entorno PGDATA se usa para distinguir un grupo de bases de datos de otro al establecer su valor al directorio que contiene el grupo deseado. Los tres ficheros de configuración se encuentran en cada directorio PGDATA/. Los detalles sobre el formato de los ficheros y las opciones que pueden usarse se encuentran en file:///usr/share/doc/postgresql/html/index.html.

Crea el guión de arranque con lo siguiente:

cat > /etc/rc.d/init.d/postgresql << "EOF"

```
#!/bin/bash
# Inicio de $rc_base/init.d/postgresql

# Basado en el guión sysklogd de LFS-3.1 y anteriores.
# Reescrito por Gerard Beekmans  - gerard@linuxfromscratch.org

source /etc/sysconfig/rc
source $rc_functions

case "$1" in
        start)
                echo "Iniciando el demonio PostgreSQL..."
                su - postgres -c '/usr/bin/pg_ctl start -W -D /var/pgsql/data \
                        -l /var/pgsql/data/logfile -o "-i" '
                evaluate_retval
                ;;

        stop)
                echo "Parando el demonio PostgreSQL..."
                /usr/bin/pg_ctl stop -m smart -D /var/pgsql/data
                evaluate_retval
                ;;
```

UNIVERSIDAD SIMON BOLIVAR
HEMEROTECA
BARRANQUILLA

```
        restart)
                $0 stop
                sleep 1
                $0 start
                ;;

        status)
                /usr/bin/pg_ctl status -D /var/pgsql/data
                ;;

        *)
                echo "Use: $0 {start|stop|restart|status}"
                exit 1
                ;;
esac

# Fin de $rc_base/init.d/
```

EOF

chmod 755 /etc/rc.d/init.d/postgresql

Crea los enlaces simbólicos a este fichero en los directorios pertinentes de rc.d con los siguientes comandos:

cd /etc/rc.d/init.d &&

ln -sf ../init.d/postgresql ../rc0.d/K26postgresql &&

ln -sf ../init.d/postgresql ../rc1.d/K26postgresql &&

ln -sf ../init.d/postgresql ../rc2.d/K26postgresql &&

ln -sf ../init.d/postgresql ../rc3.d/S34postgresql &&

ln -sf ../init.d/postgresql ../rc4.d/S34postgresql &&

ln -sf ../init.d/postgresql ../rc5.d/S34postgresql &&

ln -sf ../init.d/postgresql ../rc6.d/K26postgresql

Contenido

El paquete PostgreSQL contiene ***clusterdb***, ***createdb***, ***createlang***, ***createuser***, ***dropdb***, ***droplang***, ***dropuser***, ***ecpg***, ***initdb***, ***initlocation***, ***ipcclean***, ***pg_config***, ***pg_controldata***, ***pg_ctl***, ***pg_dump***, ***pg_dumpall***, ***pg_encoding***, ***pg_id***, ***pg_resetxlog***, ***pg_restore***, ***postgres***, ***postmaster***, ***psql***, ***vacuumdb***, *libecpg, libpq y varios módulos de conjuntos de caracteres (charsets).*

Descripciones

clusterdb

clusterdb *es una utilidad para reordenar las tablas de una base de datos PostgreSQL.*

createdb

createdb *crea una nueva base de datos PostgreSQL.*

createlang

createlang *define un nuevo lenguaje de procedimientos PostgreSQL.*

createuser

createuser *crea una nueva cuenta de usuario PostgreSQL.*

dropdb

dropdb *elimina una base de datos PostgreSQL.*

droplang

droplang *elimina un lenguaje de procedimientos PostgreSQL.*

dropuser

dropuser *elimina una cuenta de usuario PostgreSQL.*

ecpg

ecpg *es el preprocesador SQL incorporado.*

initdb

initdb *crea un nuevo grupo de bases de datos.*

initlocation

initlocation *crea un área secundaria de almacenamiento de bases de datos.*

ipcclean

ipcclean *elimina la memoria compartida y los semáforos dejados por un servidor de bases de datos abortado.*

pg_config

pg_config *recupera información de la versión de PostgreSQL.*

pg_controldata

pg_controldata *devuelve información inicializada por* ***initdb****, como la versión del catálogo y el locale del servidor.*

pg_ctl

pg_ctl *controla la parada e inicio del servidor de bases de datos.*

pg_dump

pg_dump *vuelca los datos y metadatos de una base de datos en guiones que son usados para regenerar la base de datos.*

pg_dumpall

pg_dumpall *invoca a* ***pg_dump*** *reiteradamente para cada base de datos de un grupo.*

pg_resetxlog

pg_resetxlog *borra el fichero de registro y opcionalmente pone a cero algunos campos del fichero pg_control.*

pg_restore

pg_restore *crea bases de datos a partir de los ficheros de volcado creados por* ***pg_dump***.

postgres

postgres *es un servidor de bases de datos monousuario, utilizado normalmente para depuración.*

postmaster

postmaster *es el demonio de bases de datos multiusuario.*

psql

psql *es un intérprete de comandos de consola para las bases de datos.*

vacuumdb

vacuumdb *compacta bases de datos y genera estadísticas para el analizador de consultas*

MYSQL VS POSTGRE

- ***MySQL:***
  - *Su principal objetivo de diseño fue la VELOCIDAD. Se sacrificaron algunas características esenciales en sistemas más "serios" con este fin.*
  - *Otra característica importante es que consume MUY POCOS RECURSOS, tanto de CPU como de memoria.*
  - *Licencia GPL a partir de la versión 3.23.19.*
  - ***Ventajas:***
    - *Mayor rendimiento. Mayor velocidad tanto al conectar con el servidor como al servir selects y demás.*
    - *Mejores utilidades de administración (backup, recuperación de errores, etc).*
    - *Aunque se cuelgue, no suele perder información ni corromper los datos.*
    - *Mejor integración con PHP.*
    - *No hay límites en el tamaño de los registros.*

- *Mejor control de acceso, en el sentido de qué usuarios tienen acceso a qué tablas y con qué permisos.*
- *MySQL se comporta mejor que Postgres a la hora de modificar o añadir campos a una tabla "en caliente".*

- *Inconvenientes:*
  - *No soporta transacciones, "roll-backs" ni subselects.*
  - *No considera las claves ajenas. Ignora la integridad referencial, dejándola en manos del programador de la aplicación.*

- ***PostgreSQL:***
  - *Postgres intenta ser un sistena de bases de datos de mayor nivel que MySQL, a la altura de Oracle, Sybase o Interbase.*
  - *Licencia BSD.*
  - ***Ventajas:***
    - *Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y la cantidad de RAM.*
    - *Soporta transacciones y desde la versión 7.0, claves ajenas (con comprobaciones de integridad referencial).*
    - *Tiene mejor soporte para triggers y procedimientos en el servidor.*
    - *Soporta un subconjunto de SQL92 MAYOR que el que soporta MySQL. Además, tiene ciertas características orientadas a objetos.*

- ***Inconvenientes:***
  - *Consume BASTANTES más recursos y carga más el sistema.*
  - *Límite del tamaño de cada fila de las tablas a 8k!!! (se puede ampliar a 32k recompilando, pero con un coste añadido en el rendimiento).*
  - *Es de 2 a 3 veces más lenta que MySQL.*
  - *Menos funciones en PHP.*

En cuanto a consideraciones de estabilidad del servidor, cada comparativa da datos contradictorios. En general parece que MySQL es más estable (aunque también hay gente que opina lo contrario), y que Postgres tiende a desperdiciar memoria y sobrecargar bastante el sistema (aunque de nuevo, hay opiniones distintas).

Como conclusión a la comparación entre MySQL y Postgres, parece aceptado que MySQL junto con Apache y PHP forman un buen equipo para servir páginas web con contenido dinámico, discusiones, noticias, etc., por ejemplo al estilo de SlashDot. En general, sistemas en los que la velocidad y el número de accesos concurrentes sea algo primordial, y la seguridad no sea muy importante (pueda bastar con hacer backups periódicos que se restaurarán tras una caida del servidor). En cambio, para sistemas más serios en las que la consistencia de la BD sea ***fundamental*** *(BD con información realmente importante, bancos, etc.) PostgreSQL es una mejor opción pese a su mayor lentitud.*

Por último, señalar que hace poco Inprise/Borland ha liberado bajo una licencia Open Source (modificación de la MPL de Mozilla) su BD.

SINTEXIS DE COMERCIO EN LINEA

Para la elaboración de la aplicación que simula la compra de productos a través de la Web Hemos utilizado tres herramientas básicas que son MYSQL como manejador de base de datos, PHP como lenguaje de programación para desarrollar los diferentes scripts necesario para el manejo de las tablas y HTML para el manejo de la presentación de las diferentes pantallas.

UTILIZACION DE MYSQL

Hemos creado tres tablas bases en las cuales encontramos la información necesaria para las diferentes simulaciones que vamos a utilizar en esta aplicación.

CREACION DE UNA TABLA EN MYSQL

Esquema de creación de una tabla en mysql utilizando la consola de cliente.

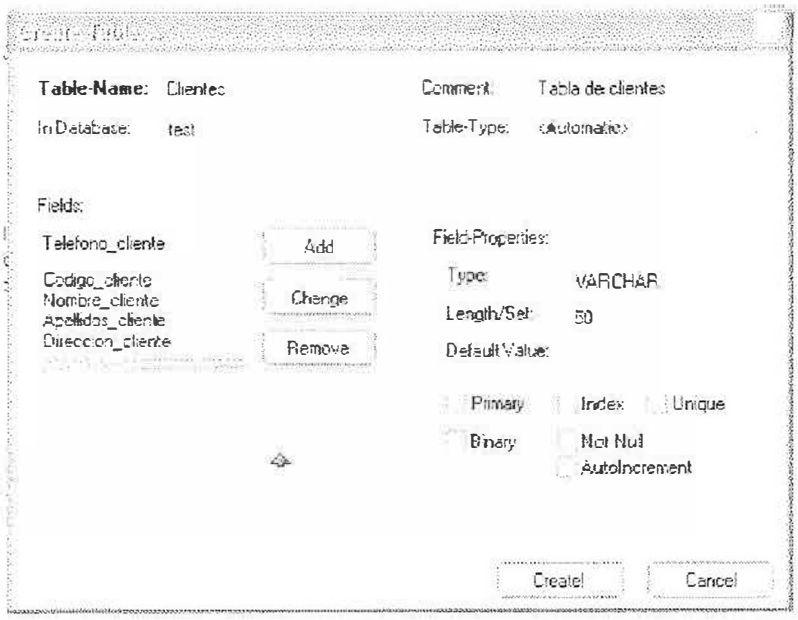


Tabla Clientes: en esta tabla almacenamos los datos basicos correspondiente a un cliente o a un posible cliente.

```
CREATE TABLE Clientes
(Codigo_cliente int (30)  NOT NULL,
 Nombres_cliente varchar (50)  NOT NULL,
 Apellidos_cliente varchar (50)  NOT NULL,
 Direccion_cliente varchar (50)  NOT NULL,
 Barrio_cliente varchar (50)  NOT NULL,
 Telefono_cliente varchar (50)  )
```

GRAFICA DE LA ESTRUCTURA DE LA TABLA CLIENTES VISTA DESDE EL INTERPRETADOR DE MYSQL

UNIVERSIDAD SIMON BOLIVAR
HEMEROTECA
BARRANQUILLA

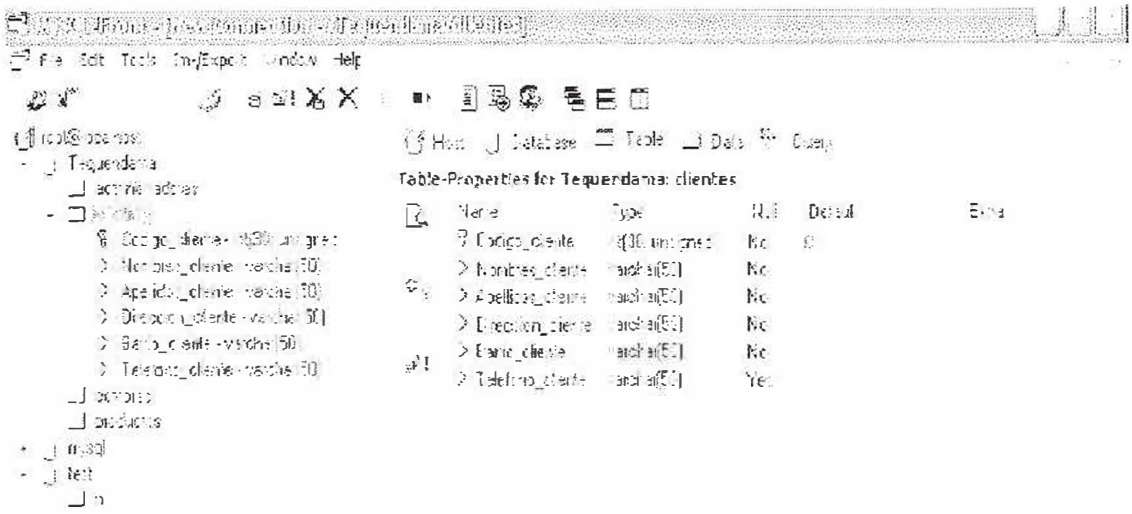
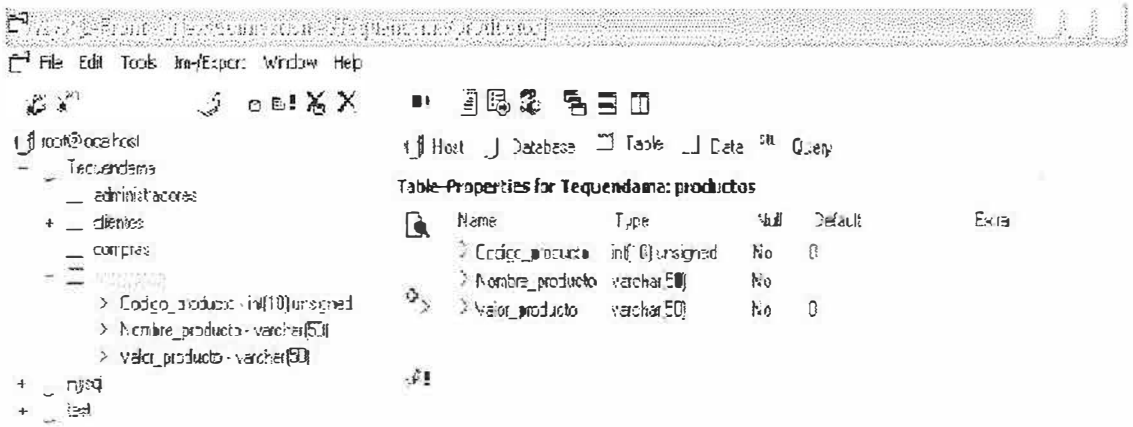


Tabla productos: En esta tabla encontramos la información correspondiente a los diferentes producto ofrecidos por medio de la pagina.

CREATE TABLE Productos
(Codigo_producto int (10) NOT NULL,
Nombre_producto varchar (50) NOT NULL,
Valor_producto varchar (50) NOT NULL)

GRAFICA DE LA ESTRUCTURA DE LA TABLA PRODUCTOS VISTA DESDE EL INTEPRETADOR DE MYSQL

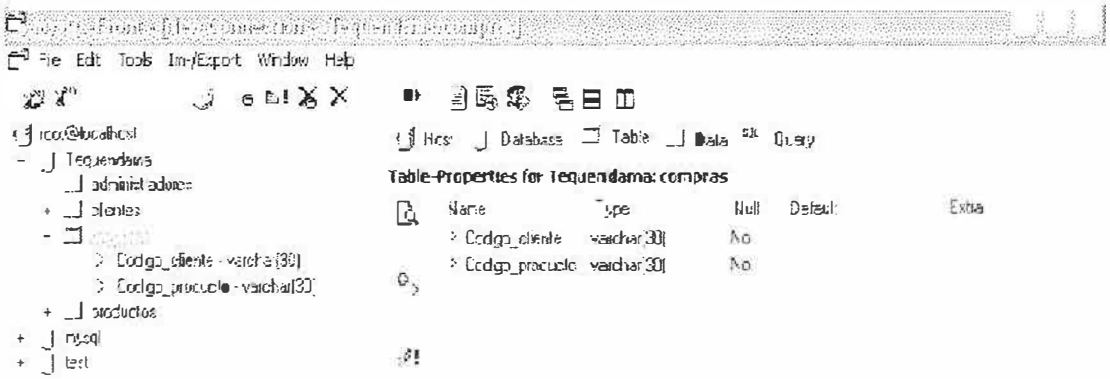


SQL

Tabla compras: En esta tabla encontraremos las diferentes transacciones de compra, utilizamos dos campos el campo de cedula del cliente o codigo_cliente y codigo del producto.

CREATE TABLE Compras
(Codigo_cliente int (10) NOT NULL,
Codigo_producto int (10) NOT NULL,)

GRAFICA DE LA ESTRUCTURA DE LA TABLA COMPRAS VISTA DESDE EL INTEPRETADOR DE MYSQL



SQL

DESARROLLO DE LAS APLICACIONES EN PHP

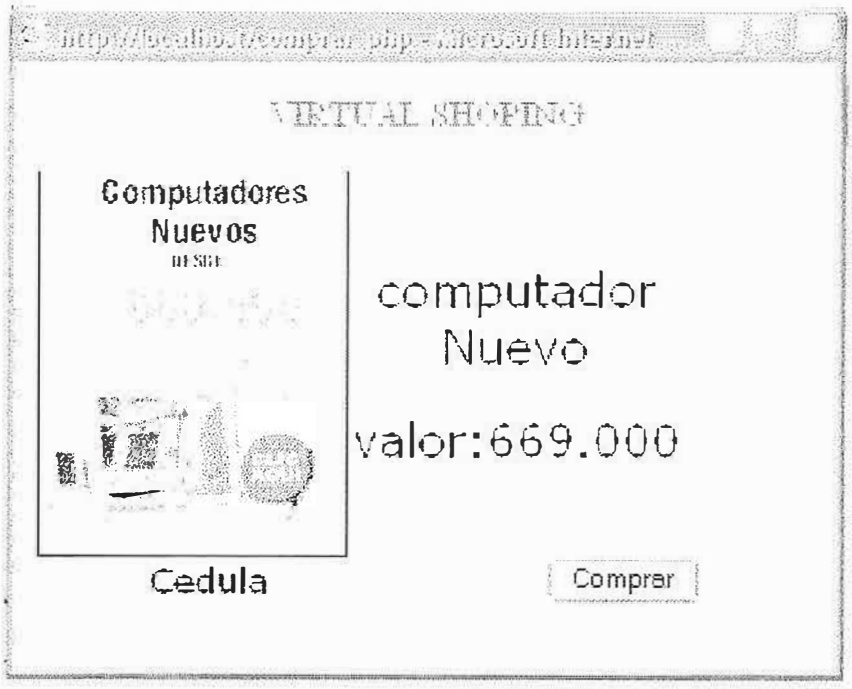
Para el desarrollo de las aplicaciones web hemos utilizado cinco scripts, en los cuales combinamos html, java y php de una manera practica sencilla y facil de entender.

INDEX.PHP: El script de la pagina principal en el cual utilizamos el codigo html encriptado en php, para los diferentes enlaces (ver codigo en anexo1) en este grafico podemos ver la presentación principal en su forma final.

REGISTRAR_PRODUCTO.PHP: El script de la pagina registrar producto utilizada para recopilar datos correspondientes a los diferentes productos que se ofrecerían en nuestra pagina.(ver código en anexo3).

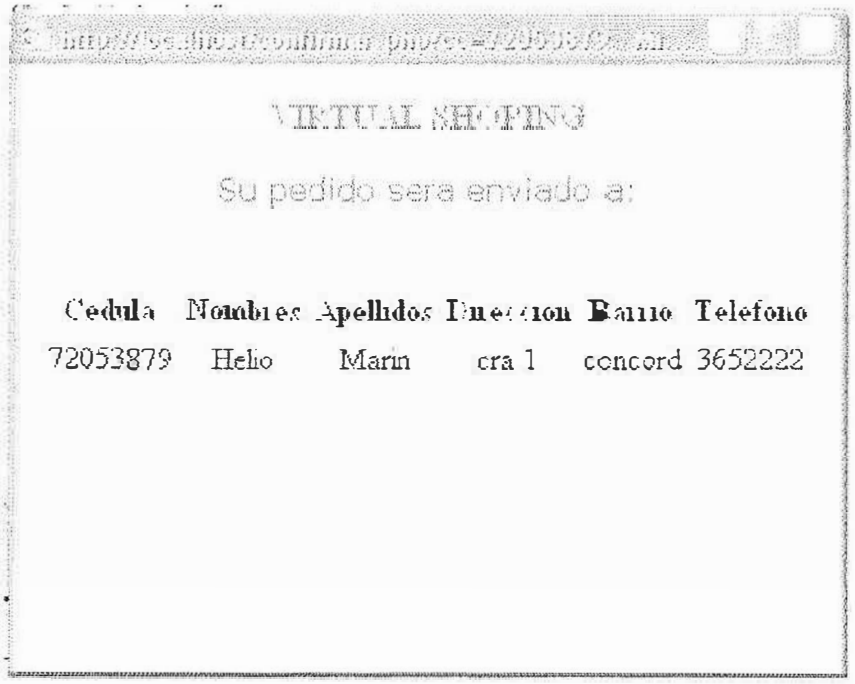
COMPRAR.PHP: El script de la pagina comprar donde tu cedula sera la forma de imdentificarte para poder adquirir un producto si estas

registrado en caso contrario hará un enlace a una pagina donde puedes escoger registrarte o salir sin hacer la transacción. (ver codigo en anexo 4).

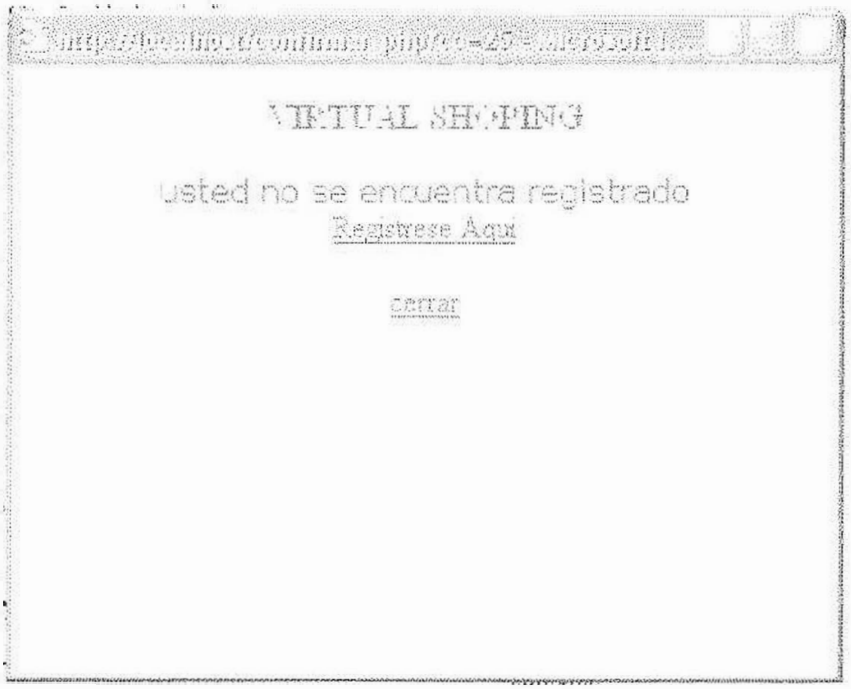


COFIRMAR.PHP: El script de la pagina confirmar utiliza dos facetas en una utilizada en caso de que el usuario este registrado para desplegar sus datos en pantalla y la otra faceta es en el caso que no lo este lo envia a la pagina de registro en ambos casos se usa como complemento del script de comprar. (ver codigo en anexo 5).

CASO I



CASO II



DESARROLLO EN ASP

El manejo de html y java en ASP es el mismo utilizado en PHP solo con algunas pequeñas modificaciones al igual que las bases de datos en MYSQL por eso hemos preferido hacerles una pequeña inducción con las conexiones con una base de datos en ACCESS que se haria de la siguiente forma:

_Bases de Datos - Utilizando ASP y Bases de Datos Access.
Esta es una breve explicacion de como utilizar ASP para tener acceso a una base de datos, lo primero que debemos crear es la base de datos, en este caso de Access con el nombre de 'temas.mdb', ahora dentro de esta base de datos crearemos una tabla llamada 'TemasMusicales' y que tenga los siguientes campos...

UNIVERSIDAD SIMÓN BOLÍVAR
HEMEROTECA
BARRANQUILLA

CAMPOS_
TIPOS DE DATOS__Id_Autonumerico__Titulo_Texto__Genero_Texto_____

Ingresale algunos datos para poder utilizarla. Bueno ya tenemos la base de datos con informacion adentro, lo que vamos a hacer ahora es desde ASP ver todo el contenido de la tabla que hemos creado. Para ello creamos un archivo ASP con el nombre 'muestra_temas.asp' y en su interior agregamos el siguiente codigo...

```
<%
Dim BDD ' variable que se encargara de la base de datos
Dim TABLA ' variable que se encarga de la tabla
'creamos los objetos que manejan la base de datos y la tabla
Set BDD = Server.CreateObject("ADODB.Connection")
Set TABLA = Server.CreateObject("ADODB.Recordset")
'abrimos la base de datos

BDD.Open ("PROVIDER=Microsoft.Jet.OLEDB.4.0;DATA SOURCE=" +
Server.MapPath("temas.mdb"))
'abrimos la tabla
TABLA.Open "TemasMusicales", BDD ' indicamos el nombre de la tabla y
la base de datos
'ahora recorremos toda la tabla y mostramoslos datos
While NOT TABLA.EOF 'recorre la tabla hasta que EOF sea verdadero
Response.Write "titulo:" & TABLA("Titulo") & "<br>"
Response.Write "genero:" & TABLA("Genero") & "<br>"
Wend
'cerramos todo y liberamos los objetos
TABLA.Close
BDD.Close
Set BDD = Nothing
```

```
Set TABLA = Nothing
%>_
```

como podrán darse cuenta no es gran diferencia con lo que se venia trabajando en los otros le aguajes de programación que habíamos utilizado para este tipo de conexiones con Access.

Además para que tenga una visión mas clara de la forma en que debería hacerlo en ASP solo necesita una leve inducción que le daremos con el siguiente manual donde se le explica las funciones básicas en las cuales puede diferir PHP de ASP.
Características.

Microsoft es el creador.
Se ejecuta en el servidor donde esta alojada la pagina web.
El cliente (navegador del usuario) no nota diferencias con una pagina normal (HTML).
Por medio de ASP se puede tener acceso a bases de datos.
Para su implementación se utiliza lenguaje script como VBScript o JScript.
Se puede utilizar HTML y ASP en una misma pagina.
Es totalmente ampliable gracias a que es compatible con la tecnología ActiveX.
Como Funciona
Aquí están pasa a paso como funciona una pagina ASP.
Un usuario por medio del navegador solicita una pagina ASP.
Esta solicitud llega al servidor el cual tiene alojada la página pedida.
Este servidor procesa la pagina ASP y devuelve código HTML.
El usuario visualiza la página en su navegador.
Noten que para el usuario no existe diferencia en ASP y HTML por que a su navegador siempre llega código HTML puro, el único que requiere un trabajo extra es el servidor el cual tiene que procesar el código ASP y trasformarlo en HTML para su posterior envió al cliente.
Como utilizarlo
Para utilizarlo necesitamos un servidor que procese nuestras paginas ASP, para ello existe el Internet Información Server bajo Windows NT o 2000 y bajo Windows 95/98 esta el Personal Web Server que viene el CD de Instalación de Windows 98 o en el CD del Visual Studio.

Personal Web Server
El Personal Web Server es un servidor que nos permite trabajar con paginas ASP en nuestra PC sin necesidad de subir las paginas ASP al servidor para realizar las pruebas, es decir, armamos todo el sitio en

nuestra PC, lo probamos con el Personal Web Server (PWS) y si todo va bien lo subimos.
Para utilizar el PWS necesitamos tener Windows 95/98/Me, si tenemos como SO Windows NT o 2000 también podemos utilizar el PWS pero en este caso es preferible utilizar Internet Information Server (IIS) dado que es mas funcional y robusto. Pueden conseguir el PWS desde el sitio de Microsoft o en los CDs de Instalación de Windows 98, Visual Basic o Visual Studio.
A la hora de instalar el PWS hay que seguir los pasos de una instalación típica de un programa, pero hay que prestar atención en el momento que nos pide el directorio de publicacion web, es decir el directorio en donde se guardaran nuestras paginas web se puede dejar el que nos ofrece el PWS por defecto o lo podemos modificar, si lo modificamos, por ejemplo ponemos que el directorio de publicacion web es "C:\Paginas Webs" tenemos que guardar nuestras pagina dentro de este directo.
Una ves instalado reiniciamos la PC y al volver tendremos un icono en la barra de notificación de Windows desde este icono podemos iniciar el PWS, aclaremos que el PWS tiene dos estados Iniciado y Detenido, obviamente nuestro sitio solo funcionara cuando este el PWS iniciado.

Bueno una ves abierto el PWS nos vamos al sector llamado "Avanzada" desde acá configuramos las opciones avanzadas del servidor, lo mas importante es la configuración del directorio, seleccionamos "Home" y hacemos click en "Modificar Propiedades", aquí seleccionamos el directorio que habíamos elegido como publicacion web, apretamos "Aceptar" y volvemos a las opciones avanzadas. Ahora destildamos "Habitar Documento Predeterminado" y habilitamos "Permitir Exploración de directorios", volvemos a la sección "Principal" y hacemos click en el link para abrir nuestro site.

VBScript - Introducción
Por medio de VBScript podemos crear nuestras paginas ASP, debemos tener en cuenta que antes de escribir cualquier código ASP debemos abrir nuestro código con el símbolo '<%' y al finalizar cerrarlo con '%>'

```
<%
'este es un comentario

Response.Write ("Mi Primer Pagina ASP")

%>
```

_Podemos abrir archivo nuevo agregarle el código anterior guardarlo como "hola.asp" dentro de nuestro directorio de publicacion web abrir el PWS y ver el resultado en el navegador.

VBScript es un lenguaje script derivado de Visual Basic así que también hereda sus funciones básicas a continuación veremos algunas de las mas utilizadas.

VBScript - Funciones Básicas
Si estas familiarizado con Visual Basic podes pasar esta parte dado que los funciones básicas son iguales en ambos lenguajes, y si no conoces Visual Basic acá va una ayuda.
Variables
La forma de declaran variables es por medio de la palabra 'Dim', aclaremos que la declaración de variables no es necesaria, salvo que al inicio de nuestra pagina ASP pongamos la palabra clave 'Option Explicit' que hace que la declaracion de variables sea necesaria si no nos daria un error.

```
<%
Option Explicit


Dim I 'declaramos la variable
I= 14 ' le asignamos un valor

Response.Write (I) 'mostramos su valor en el navegador

%> _Condicionales
```

El condicional mas utilizado en cualquier lenguaje es el famoso IF, VBScript no podia quedarse afuera, el modo de uso es poner la palabra clave 'If' despues la condicion a evaluar, luego 'Then' y el codigo a ejecutar si se cumple la condicion dada, al finalizar se cierra el bloque 'If' con la palabra clave 'End If'

```
<%
Dim I
I = 14

If I = 14 Then
Response.Write ("La variable I es igual a 14")

End If

%> _Tambien extiste la alternativa 'Else' (sino), es decir sino se cumple la condicion hacer lo siguiente, veamos un ejemplo
<%
Dim I
I = 14

If I = 14 Then
```

Response.Write ("La variable I es igual a 14")

Else
Response.Write ("La variable I es distinta a 14")

End If

%> _Fijense que pusimos 'I = 14' tambien podemos hacer otro tipo de condicion como puede ser distinto 'I <> 14', mayor 'I > 14', menor 'I < 14', mayor o igual 'I > = 14', menor o igual 'I <= 14'< /FONT >
Bucles
VBScript posee un bucle llamador 'For' que repite una accion la cantidad de veces indicada
<%
Dim I
For I = 1 to 5

Response.Write ("Esto se repite 5 veces")

Next
%> _Podemos modificar el numero 5 y poner un 10 y lo repetira 10 veces, tambien podemos modificar el numero 1 por el 3 y el 5 por el 6, esto hara que el bucle se repita 3 veces, '3 to 6' (desde 3 hasta 6), tambien podemos indicar el incremento que tendra por medio de la palabra clave 'step'
<%
Dim I
For I = 1 to 10 step 2

Response.Write ("Esto se repite de dos en dos, i = " & i)

Next
%> _Gracias a esto podemos hacer un bucle inverso por ejemplo de 10 a 1
<%
Dim I
For I = 10 to 1 step -1

Response.Write ("i = " & i)

Next
%> _La palabra clave 'Next' indica el final del bucle, es decir se repite todo lo que esta entre el 'For' y 'Next'
While..Wend
Este bucle a diferencia del 'For' repite una accion siempre que se cumpla una condicion, veamos un ejemplo

```
<%
Dim I
I = 0 'inicializa i igual a 0
While I <> 10 'repita mientras i sea distinto a 10
Response.Write ("i = " & i) 'muestra por pantalla el valor de i

I = I + 1 'incrementa i en uno

Wend
```

%> _Cuando agregamos la palabra clave 'Wend' estamos diciendo que repita todo el codigo incluido entre While y Wend

Objeto Response

Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, su funcion es enviar informacion al cliente (navegador).

PROPIEDADES

Buffer Esta propiedad es del tipo boolean (true/false) e indica si la pagina se almacena en el buffer del servidor antes de ser enviada, es decir si esta true envia la pagina al cliente recien cuando este toda cargada y si esta false la envia a medida que la procesa. Se debe utilizar antes de haber enviado algun resultado al cliente. Si el servidor utilizado tiene la version de ASP 2.0 el valor predeterminado de esta propiedad es false y en cambio la version es la 3.0 (Windows 2000) el valor predeterminado es true. Ejemplo <% Response.Buffer = TRUE%>

Expires Esta propiedad indica el tiempo, en minutos, que la pagina se guarda el el cache del navegador, si este valor se pone a 0 no se utiliza el cache del navegador Ejemplo<% Response.Expires = 5%>

ExpiresAbsolute Esta propiedad indica hasta que dia y hora la pagina se guarda en el cache del navegador. Ejemplo <% Response.ExpiresAbsolute=#May 5, 2001 14:30:00# %>

Cookies Crea una cookie en la maquina del usuario. Una cookies es un archivo con formato de texto en el cual el se puede guardar informacion. Las cookies por lo general se utilizan para controles de acceso, personalizacion de paginas y estadisticas. El siguiente ejemplo crea una cookie llamada 'fecha' la cual contiene el valor devuelto por la funcion 'date' que nos devuelve la fecha actual Ejemplo<% Response.Cookies("fecha") = date %> Esta cookie se mantiene en la maquina del usuario hasta que termine su session, borre las del navegador o se cumpla el tiempo establecido para esa cookies, a continuacion veremos la forma de establecer la duracion de una cookie <% Response.cookies("fecha").expires="2/12/2001"%>Esto hace que la cookie se borre en la fecha indicada, si creamos otra cookie llamada 'fecha' esta sobreescribira a la otra.

METODOS

End Este metodo detiene el proceso de la pagina ASP y envia lo ya procesado al navegador Ejemplo<% Response.End%>

Flush Este metodo envia el contenido del buffer al navegador siempre y cuando la propiedad Buffer se true, si la propiedad Buffer es igual a false se producira un error Ejemplo <% Response.Flush%>
Redirect (cadena_url) Este metodo redirecciona la pagina actual a la pasada como parametro. Ejemplo<% Response.Redirect "otra_pagina.asp"%>
Write (cadena) Este metodo escribe en el navegador la cadena padada como parametro, tengan en cuenta que tambien se pueden enviar etiquetas HTML y concatenar variable con el operador & (VBScript). Ejemplo<% Response.Write ("hola!!!%>
Objeto Request
Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, por medio de este objeto podes tener acceso a la informacion que el navegador pasa al servidor.
PROPIEDADES (COLECCIONES)
Form Esta propiedad es una coleccion que contiene todos los elementos de un formulario (<form>) pasado al archivo ASP con el metodo POST (<form method= 'post'>),la forma de acceder a el valor de un elemento del formulario es la siguiente...< /FONT>< /FONT>< /FONT> <%=Request.Form("nombre_elemento")%>Ejemplo----------
formulario.htm----------

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<form action="procesa_formulario.asp" method="POST">

Nombre:<input type="Text" name="nombre" maxlength="20"><br>

Apellido:<input type="Text" name="apellido" maxlength="30"><br>

<input type="Submit" name="Submit" value="Enviar">


</form>
</BODY>
</HTML>
```

----------procesa_formulario.asp--------

```
<HTML>
<HEAD>
</HEAD>
<BODY>
<% Response.Write ("nombre:" & Request.Form("nombre"))%>

<% Response.Write ("apellido:" & Request.Form("apellido")) %>
```

</BODY>
</HTML>__
Cookies__ __Recupera una cookie almacenada en la maquina del usuario. Una cookies es un archivo con formato de texto en el cual el se puede guardar informacion. Las cookies por lo general se utilizan para controles de acceso, personalizacion de paginas y estadisticas. El siguiente ejemplo recupera la cookie llamada 'fecha' que creamos anteriormente. __ __Ejemplo__ __<% Response.Write Request.Cookies("fecha") %> __ __Debido a la natualeza de las cookies y a que nunca podemos garantizar que una cookie almazenada anteriormente todavia exista debemos manejarmos con cautela y no almacenar en ellas datos criticos. __
QueryString__ __Esta coleccion contiene todos los valores pasados como parametros a una pagina ASP. La formas de pasar parametros a una pagina ASP son varias una es por medio de formularios utilizando el metodo 'GET' y la otra es generandolo 'manualmente'... __ __http://www.servidor_xxx.com/pagina_asp.asp?nombre=jose&apellido =perez__ __Esa es la manera de pasar parametros a una pagina ASP, luegop del la direccion URL se pone el signo '?' y a continuacion todos los parametros con sus respectivos valores separados por el simbolo '&'. __ __Ejemplo__ __----------pasa_parametros.asp----------
<HTML>
<HEAD>
</HEAD>
<BODY>
<AHREF="lee_parametros.asp?nombre=jose">click aca! </A>
</BODY>
</HTML>

----------lee_parametros.asp----------
<%
Response.Write (Request.QueryString("nombre") & "<br>")
%> __
ServerVariables__ __Esta coleccion nos da acceso a todas las variables de entorno del servidor. __Ejemplo__ __<%
Response.Write (Request.ServerVariables("REMOTE_ADDR"))
Response.Write (Request.ServerVariables("SERVER_SOFTWARE"))
%>__
Objeto Server
Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, por medio de este objeto podes tener acceso a funciones utiles del servidor.
PROPIEDADES
ScriptTimeout __ __Esta propiedad determina el tiempo maximo en segundos que se espera para ejecutar la pagina ASP. Si la pagina tarda mas segundos en ser procesada se produce un error. El valor

predeterminado es 90 segundos. __ __Ejemplo__ __<% Server.ScriptTimeout = 25 %>___

METODOS

CreateObject (id_objeto) __ __Este es uno de los metodos mas importantes y nos permite crear objetos a partir de los cuales vamos a trabajar. Como unico parametro se especifica el nombre del objeto a crear. Al crear el objeto este se le asigna a la variable establecida por medio de la palabra clave SET, esta variable hereda todos los metodos, propiedades y eventos del objeto creado. __ __Ejemplo__ __<% Set BDD = Server.CreateObject("ADODB.Connection") %> __

Execute (path) __ __Este metodo ejecuta una pagina ASP dentro de otra, su unico parametro es la ruta de la pagina a ejecutar. Este metodo solo esta disponible en la version 3.0 de ASP. __ __Ejemplo__ __---------- pagina_llama.asp-------

```
<HTML>
<BODY>
Hola <% Server.Execute("respuesta.asp")%> !
</BODY>
</HTML>
----------respuesta.asp----------
<%
Response.Write "Pedro
"
%>
```

__

HTMLEncode (cadena) __ __Este metodo evita que la cadena pasada como parametro sea interpretada como lenguaje HTML. __ __Ejemplo__ __<% Response.Write Server.HTMLEncode("sin negrita") %> __

MapPath (path) __ __Devuelve la ruta completa del archivo pasado como parametro. __ __Ejemplo__ __<% Response.Write Server.MapPath("calendario\dias.asp") %>__

Objeto Session

Este objeto ASP es del tipo integrado, es decir no tenemos que crear una instancia del mismo para poder utilizarlo, el mismo sireve para compartir informacion de un usuario entre las paginas del sitio. Por defecto el objeto session se crea al ingresar el usuario a nuestro sitio y se destruye cuando lo abandona o cuando pasan 20 minutos sin actividad del usuario.

PROPIEDADES

SessionID __ __Esta propiedad devuelve un ID que identifica a un usuario. Este ID es unico, no se puede modificar y lo crea automaticamente el servidor. __ __Ejemplo__ __<% Response.Write "ID Session = " & Session.SessionID %> __

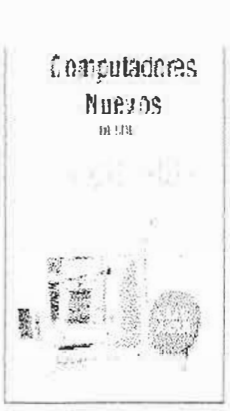
Timeout __ __Esta propiedad determina el tiempo en minutos que durara la sesion del usuario antes de ser destruida, por defecto su valos es de 20 minitos. __ __Ejemplo__ __<% Session.Timeout = 30 %> ___

METODOS
Abandon __ __Este metodo proboca el abandono forzado de la session, destruye todas la _ HYPERLINK "http://www.zonagratuita.com/microsoft/asp/" \l "variables_de_sesion" _variables de sesion _. __ __Ejemplo__ __<% Session.Abandon %> ___

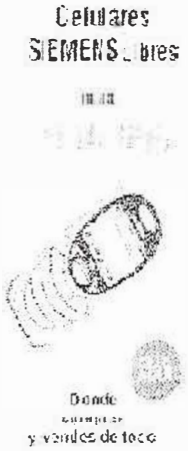
VARIABLES DE SESION __ __Una utilidad muy buena que nos permite el objeto session es crear variables y poder compartirlas en todas la paginas de nuestro sitio algo asi como una variable global pero con una diferencia bastante grande dado que cada usuario que ingrese a nuestra pagina tendra un valor distinto en cada variable. En el ejemplo que sigue a continuacion se muestra la forma de uso. __ __Ejemplo__ __---------- pagina_1.asp-------

```
<HTML>
<BODY>
<% Session("Dia") = "Lunes"%>
</BODY>
</HTML>
```

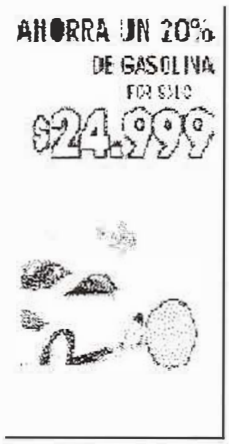
VIRTUAL SHOPING



Cod 1122



Cod 1123



Cod 1124

para acceder a comprar tienes que estar registrado

REGISTRAR_USUARIO.PHP: El script de la pagina registrar usuario utilizada para recopilar datos correspondientes a los clientes y posibles compradores que visiten nuestra pagina. En esta utilizamos las combinaciones html, javascripts y php en un mismo script (ver codigo en anexo2).

	RESUMEN TRABAJO DE INVESTIGACION	F-GI-01
		Fecha vigencia: Oct. 17 / 02

Investigación ● Tesis ○ Fecha Inicio Marzo del 2003

Programa: Ingenieria de Sistemas Fecha Finalización Noviembre del 2004

Tutor:

Estudiantes:	Luis Fernando Ayala Merlo	
	Liliana Patricia Carbonó Atencio	
	Helio Ricardo Marin	
	Shirley Paola Reyes Muñoz	

Titulo

Lenguajes de Programación adecuados para el desarrollo de sistemas de información de las pequeñas y medianas empresas

Palabras Claves

Pymes, Sistema de información, aplicaciones, lenguajes de programación.

Línea de Investigación

La línea de investigación utilizada es de sistemas de gerencia informática ya que se observaran los resultados de las pruebas pilotos que se llevaran a cabo en las pequeñas y medianas empresas, se evaluará el impacto de estas en las personas que trabajan en ellas.

Eje Temático

Desarrollo de una investigación, utilizando tecnologia, que permita brindar soluciones a los requerimientos de desarrollos de sistemas de información.

Pregunta Problema

¿De que manera se podria seleccionar un lenguaje de programación para el desarrollo de sistemas de información en las pequeñas y medianas empresas.

Justificación (Importancia e Impacto de la Investigación)

Con el paso del tiempo las pequeñas y medianas empresas (PYMES) se consolidan en el mercado y con ello aumentan el volumen de sus operaciones, produciendo grandes cantidades de información la cual, si no es organizada convenientemente, puede ser difícil de manejar. Con el advenimiento de los computadores se han creado sistemas de información para el procesamiento de información en una empresa.

Los sistemas de información están constituidos por varios componentes : hardware software, bases de datos, telecomunicaciones.

El software de un sistema de información lo conforman programas de computador construidos en algún lenguaje de programación.

Existen muchos lenguajes de programación, cada uno con características particulares, algunos de ellos diseñados para aplicaciones especificas

Esta investigación pretende analizar los diversos tipos de lenguajes de programación, buscando los más apropiados para el desarrollo de sistemas de información empresariales, y por sus caracteristicas de costo, flexibilidad y facilidad de uso sean adaptables a las pequeñas y medianas empresas.

Objetivo General

Realizar un estudio comparartivo de lso lenguajes de programación convenientes para el desarrollo de sistemas de información en las pequeñas y medianas empresas.

Objetivos Específicos

. Relizar una exploración de los lenguajes de programación existentes, con el fin de conocer sus caracteristicas.

. Establecer cuáles son los lenguajes de programación más adecuados para el desarrollo de aplicaciones empresariales, así como los más usados en la actualidad .

.Analizar el desempeño de los lenguajes de programación adecuados para aplicaciones empresariales en diversas plataformas operativas.
. Realizar un análisis comparativo de los lenguajes de programación antes establecidos con el fin de encontrar aquellos que más se adapten a las características de las pequeñas y medianas empresas.

Marco Teórico (Enfoque Conceptual: Teorías - Autor Guía)

Los lenguajes de programación son esquemas codificados con los cuales se escribe software, tanto de sistemas de aplicaciones. La principal función del lenguaje de programación es proporcionar instrucciones al sistema del computador para que pueda realizar una actividad de procesamiento.
sintaxis) utilizados para escribir códigos de programas. La programación incluye la traducción de lo que quiere lograr el usuario a un código que puede entender y ejecutar el computador

Hipotesis (Si Aplica)

El proyecto que estamos realizando se basa en un sistema de información.

Tipo de Estudio

Investigación Cualitativa — Investigación Cuantitativa

tipo de estudio es exploratorio
ya que el se examinaran problemas que han sido poco estudiados anteriormente, lo cual llevara al descubrimiento de fenómenos desconocidos. En esta investigación se recurrirá a diferentes pruebas de lo lenguajes de programación observando su comportamiento para determinar cuales serán la tendencias de los lenguajes de programación mas convenientes para las pequeña y medianas empresas.

Población y Muestra

Técnica de Recolección de la información y/o Procedimientos

el medio mas utilizado fue Internet, también fue apoyadas en revistas, libros, etc.

Análisis de Resultados

Conclusiones

Recomendaciones

Análisis de la Bibliografía

Número total de fuentes:	
Libros:	sethi, ravi lenguajes de programacion conceptos y constructores.addison-weslwy iberoamerica stair, ralph y reinolds, george. Proncipios de sistemas de informacion. International thomdson editores deitel, deitel como programar en c/ c++ prentice Hall segunda edicion
Revistas:	
Periódicos:	
Internet:	www.es.tldp.org/COMOINSFLUG/COMOs/lenguajesdeprogramacion-minicomo www.tunes.org www.comstro.org www.cs.buap.mx www.lawebdelprogramador
Otros:	
Documentos especializados:	
Documentos generales:	
Clasificación de fuentes por año de publicación	
2002	
2001	
2000	
1999	
1998	
1997	
1996	
etc	

TITULO

LENGUAJES DE PROGRAMACIÓN ADECUADOS PARA EL DESARROLLO DE SISTEMAS DE INFORMACIÓN DE LAS PEQUEÑAS Y MEDIANAS EMPRESAS

AUTORES:

LUIS FERNANDO AYALA MERLO

LILIANA PATRICIA CARBONO ATENCIO

HELIO RICARDO MARIN ORTIZ

SHIRLEY PAOLA REYES MUÑOZ

En el 2004, ***de acuerdo con los postulados teóricos de*** *PATTY PEDROSA, se realizó una investigación sobre LENGUAJES DE PROGRAMACIÓN ADECUADOS PARA EL DESARROLLO DE SISTEMAS DE INFORMACIÓN DE LAS PEQUEÑAS Y MEDIANAS EMPRESAS. Para lo cual utilizaron (Internet asesoráis de profesores expertos, libros, etc). Los autores encontraron o lograron realizar un estudio comparativo de los lenguajes de programación convenientes para el desarrollo de sistemas*

de información en la pequeñas y medianas empresas, realizar una exploración existentes, con el fin de reconocer sus características, establecer cuales son los lenguajes de programación mas adecuados para el desarrollo de aplicaciones empresariales así como los mas usados en la actualidad, analizar el desempeño de los lenguajes de programación adecuados para las aplicaciones empresariales en diversas plataformas operativas, realizar un análisis comparativo de los lenguajes de programación antes establecidos con el fin de encontrar aquellos que mas se adapten a las características de las pequeñas y medianas empresas. Los autores concluyen – que después de realizada la investigación se procederá a sacra un modelo de sistemas de información mas adecuados para las pequeñas y medianas empresas.

Ejemplo:

1.1.1. Desarrollo de una investigación para la elección de LENGUAJES DE PROGRAMACIÓN ADECUADOS PARA EL DESARROLLO DE SISTEMAS DE INFORMACIÓN DE LAS PEQUEÑAS Y MEDIANAS EMPRESAS..

AUTORES:

LUIS FERNANDO AYALA MERLO

LILIANA PATRICIA CARBONO ATENCIO

HELIO RICARDO MARIN ORTIZ

SHIRLEY PAOLA REYES MUÑOZ

En el 2004, de acuerdo con los postulados teóricos de diferentes autores sobre investigación se empieza a realizar una investigación para solucionar problemas que se presentaron en el sector de las pequeñas y medianas empresas referentes a sistemas de información, por lo cual se ha llevado a cabo dicha investigación acerca de la creación de un modelo que permita crear sistemas de información convenientes para este sector surgiendo para lo que se investigo los diferentes tipos de lenguajes de programación y sus respectivos motores de bases de datos lo cual servirá a seguir con investigaciones futuras para generar así modelos de sistemas de información convenientes para las pequeñas y medianas empresas ya que estos sistemas facilitarian el manejo de la información en las pymes y haciendo que estas tengan mas competitividad empresarial