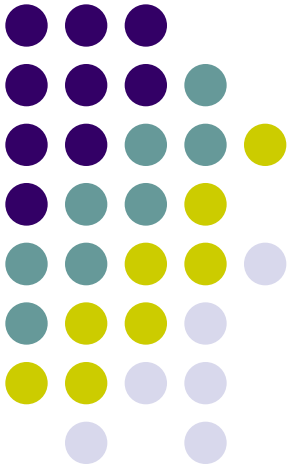
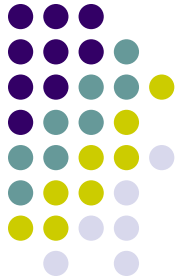
PHP5

PHP5 Básico



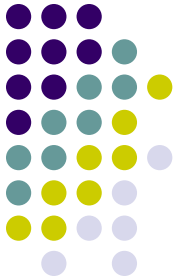
INTRODUCCIÓN



- PHP (acrónimo de PHP: *Hypertext Pre-Processor*) es un lenguaje de programación con las siguientes características:

1. Relativamente nuevo
2. Antecesor PHP/FI a Finales de 1994
3. Desarrollo de aplicaciones Web
4. Diseño dinámico de páginas Web

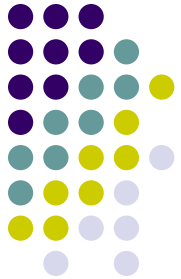
INTRODUCCIÓN



Características:

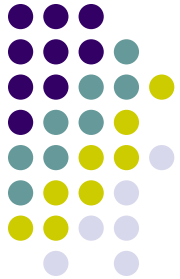
- Lenguaje interpretado de alto nivel
- Embebido en páginas HTML
- Ejecutado en el servidor
- Soporta Unix, WIN32, Mac OS X, etc.
- Se pueden desarrollar aplicaciones sencillas en corto tiempo
- Es *Open Source*

INTRODUCCIÓN



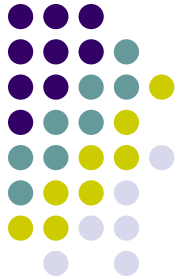
- PHP se ha convertido en el lenguaje más utilizado al lado de ASP, JSP, ColdFusion y Perl.
- Es el complemento ideal para Linux Apache
- Hasta Agosto del 2004, 17 millones de dominios utilizan PHP

Antecedentes



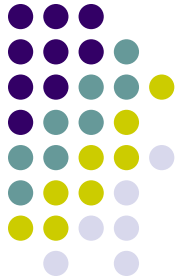
- Su creador fue Rasmus Lerford
- Era un conjunto de *scripts* escritos en Perl
- Se les denominó originalmente como *Personal Home Page Tools.*
- *Actualmente* PHP es un acrónimo recursivo que significa ***P****HP* ***H****ypertext* ***P****re-processor*

UN POCO DE HISTORIA



- En 1995 se libera el código escrito en C
- Después nace PHP/FI
- En 1997 se libera PHP/FI 2.0
- 1998 PHP 3.0
- 2000 PHP 4.0, denominado Zend

UN POCO DE HISTORIA

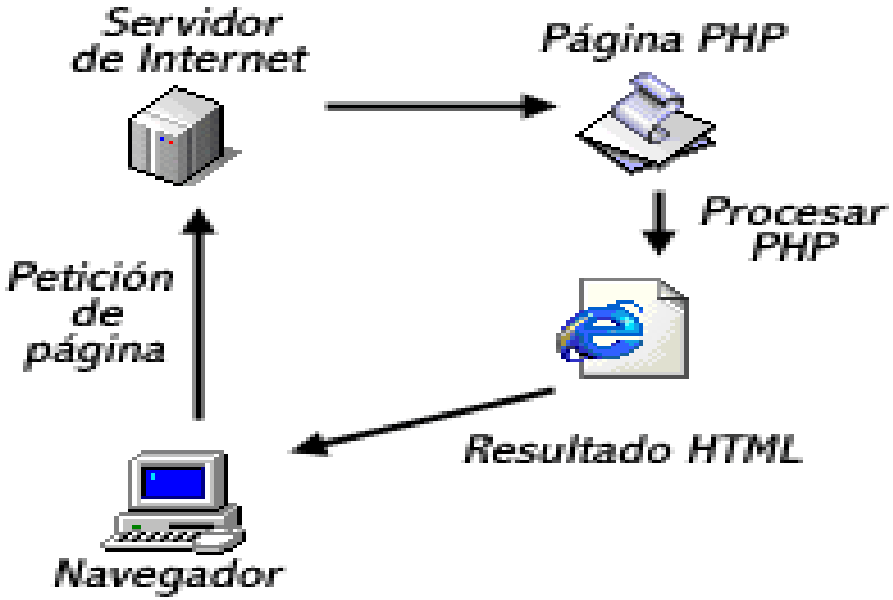


- PHP5
  - El 13 de julio de 2004, fue lanzado PHP 5, utilizando el motor Zend Engine 2.0 (o Zend Engine 2). La versión más reciente de PHP es la 5.3.1 (19 de noviembre de 2009), que incluye todas las ventajas que provee el nuevo Zend Engine 2 como:
  - Mejor soporte para la Programación Orientada a Objetos, que en versiones anteriores era extremadamente rudimentario, con PHP Data Objects
  - Mejoras de rendimiento
  - Mejor soporte para MySQL con extensión completamente reescrita.
  - Mejor soporte a XML (XPath, DOM, etc.).
  - Soporte nativo para SQLite
  - Soporte integrado para SOAP.
  - Iteradores de datos.
  - Manejo de excepciones.
  - Mejoras con la implementación con Oracle.
  - Aún se siguen publicando versiones de la rama 5.2.X, siendo publicada la versión 5.2.12 el 17 de diciembre de 2009, aunque la mayoría son actualizaciones de seguridad

COMO TRABAJA

EL código PHP normalmente aparece insertado dentro de documentos HTML

El documento PHP una vez interpretado en el servidor, genera una página HTML que será enviada al cliente.



LENGUAJE EMBEBIDO

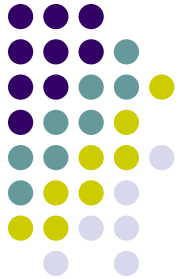
- Para diferenciar ambos lenguajes dentro del mismo documento, se utilizan etiquetas de comienzo y fin del código.

```
<?php
        instrucciones php
?>
```

Otro es:

```
<script languaje="php">
        Instrucciones php
</script>
```

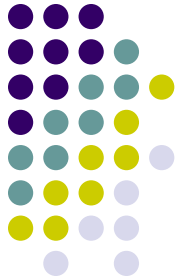
EXTENSIONES



- Las extensiones que indican al servidor HTTP que el archivo contiene código PHP son:

.php3	Código PHP 3.x
.php4	Código PHP 4.x
.php	Código PHP
.phtml	Extensión en desuso

SINTAXIS

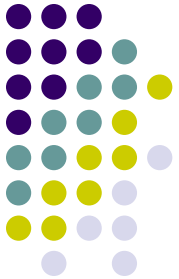


- Comentarios

```
<?php
    // comentario inicial
    echo “Tipos de comentarios”; // Una línea
    # Comentario tipo shell de unix
    /* Comentario
    de bloque*/
?>
```

Nota: Comentarios multilínea (bloque) anidados producen error !!!

VARIABLES



- Variables débilmente tipeadas
- No es necesario declararlas
- Se crean en el instante que son utilizadas
- Pueden almacenar diferente tipo de información

VARIABLES

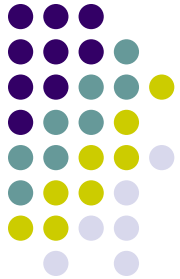
- Se definen así:
  - $ seguido por [a..z,A..Z,_] y [a..z,A..z,0..9,_]

$Suma_Total	OK
$#Datos	NO
$_Suma2	OK
$9Producto	NO

Tipos de Datos

integer	float	string
boolean	array	object

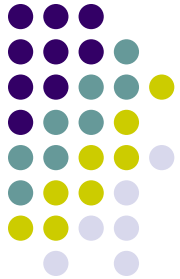
Conversión de Tipos



- Debido a que PHP es un lenguaje débilmente tipeado, si a una variable se le asigna un entero será de tipo *integer*, si después se le asigna una cadena de caracteres pasará a ser de tipo *string*.
- PHP hace una conversión automática de tipos para llevar a cabo operaciones.

Ver programa: variables6.php

Conversión de tipos por casting

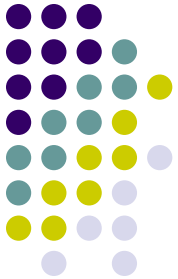


- También es posible convertir las variables de un tipo a otro cuando el programador lo desee. Esto se llama casting y es similar a como se hace en java. Los tipos puede ser:

integer, float, string, boolean, array, object

Ver programa variables7.php

Funciones de variables



- Gettype. Obtener el tipo de una variable

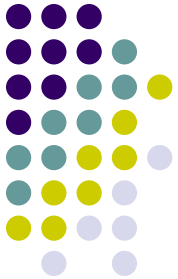
*string **gettype** (mixed var)*

- Settype. Definir el tipo de una variable

*bool **settype** (mixed &var, string tipo)*

Ver programa variables9.php

Funciones de variables



- *isset.* Determinar si una variable está definida.

  *bool **isset** (mixed var [, mixed var [, ...]])*

- *unset.* Remover una variable dada.

  *void **unset** (mixed var [, mixed var [, mixed ...]])*

- *empty.* Determinar si una variable está vacía.

  *bool **empty** (mixed var)*

Ver programa variables10.php

Comprobando el Tipo de Variables

- Las funciones para comprobar tipos de variables utilizadas en un programa php son:

  *bool **is_int** (mixed var)*

  *bool **is_bool** (mixed var)*

  *bool **is_float** (mixed var)*

  *bool **is_numeric** (mixed var)*

  *bool **is_string** (mixed var)*

  *bool **is_object** (mixed var)*

  *bool **is_array** (mixed var)*

  Ver programa variables11.php

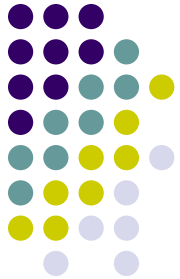
Estructuras de control

if (condición) { [sentencias]}

if (condición) { sentencias}
else { sentencias]}

Ver programa if.php

Estructuras de control



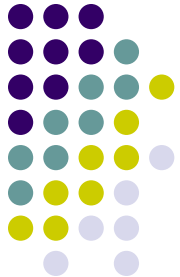
if compacto

<expresión1> ? <expresion2> : <expresion3>

- Son para formar expresiones condicionales que devolverán uno de dos posibles valores.
- Ejemplo:

```
$elmayor=($a>$b)?$a:$b;
```

Bucle *for*



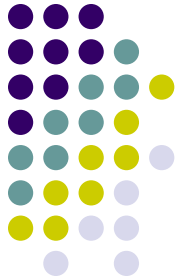
- Los bucles *for* son los bucles más complejos en PHP. Se comportan como en java. La sintaxis de un bucle *for* es:

  for (expr1; expr2; expr3) sentencia

- *La primera expresión (expr1) se evalúa (ejecuta) incondicionalmente una vez al principio del bucle.*
- *Al comienzo de cada iteración, se evalúa expr2 . Si se evalúa como **TRUE**, el bucle continúa y las sentencias anidadas se ejecutan. Si se evalúa como **FALSE**, la ejecución del bucle finaliza.*
- *Al final de cada iteración, se evalúa (ejecuta) expr3.*

Ver programa for.php

Bucle *while*



- Los bucles *while* son los tipos de bucle más simples en PHP. Se comportan como su contrapartida en C. La forma básica de una sentencia *while* es:

  while (expr) sentencia o

  while (expr): sentencia ... endwhile;

  Ver programa while.php

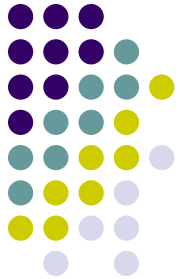
Bucle *do while*

- Aquí las condiciones se comprueban al final de cada iteración en vez de al principio. Aquí se garantiza la ejecución de la primera iteración (la condición se comprueba sólo al final de la iteración) Hay una sola sintaxis para los bucles *do..while*:

```
do {
        sentencia…
} while (expr)
```

Ver programa dowhile.php

Arreglos



- Arreglos predefinidos

```
$ciudad = array("París", "Roma", "Sevilla", "Londres");
```

- Asignando valores a un arreglo

```
$miarreglo[0] = 1;
$miarreglo[1] = "hola!!";
$miarreglo[] = 3; // Tercer elemento del arreglo
```

- Contando los elementos del arreglo

```
$numelentos = count($ciudad);
```

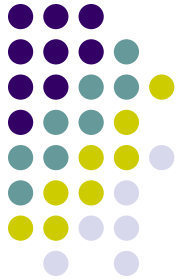
Arreglos

- Imprimiendo el contenido del arreglo

```
for ($i=0; $i < $numelentos; $i++)
{
    print ("La ciudad $i es $ciudad[$i] <BR>\n");
}
```

Ver programa arreglos.php

Bucle *foreach*



- *foreach* funciona solamente con arreglos y devolverá un error si se intenta utilizar con otro tipo de datos ó variables no inicializadas. Una forma de utilizarlo es

foreach(expresion_array as $value) sentencia

Ver programa foreach.php