Tema 1

Introducción

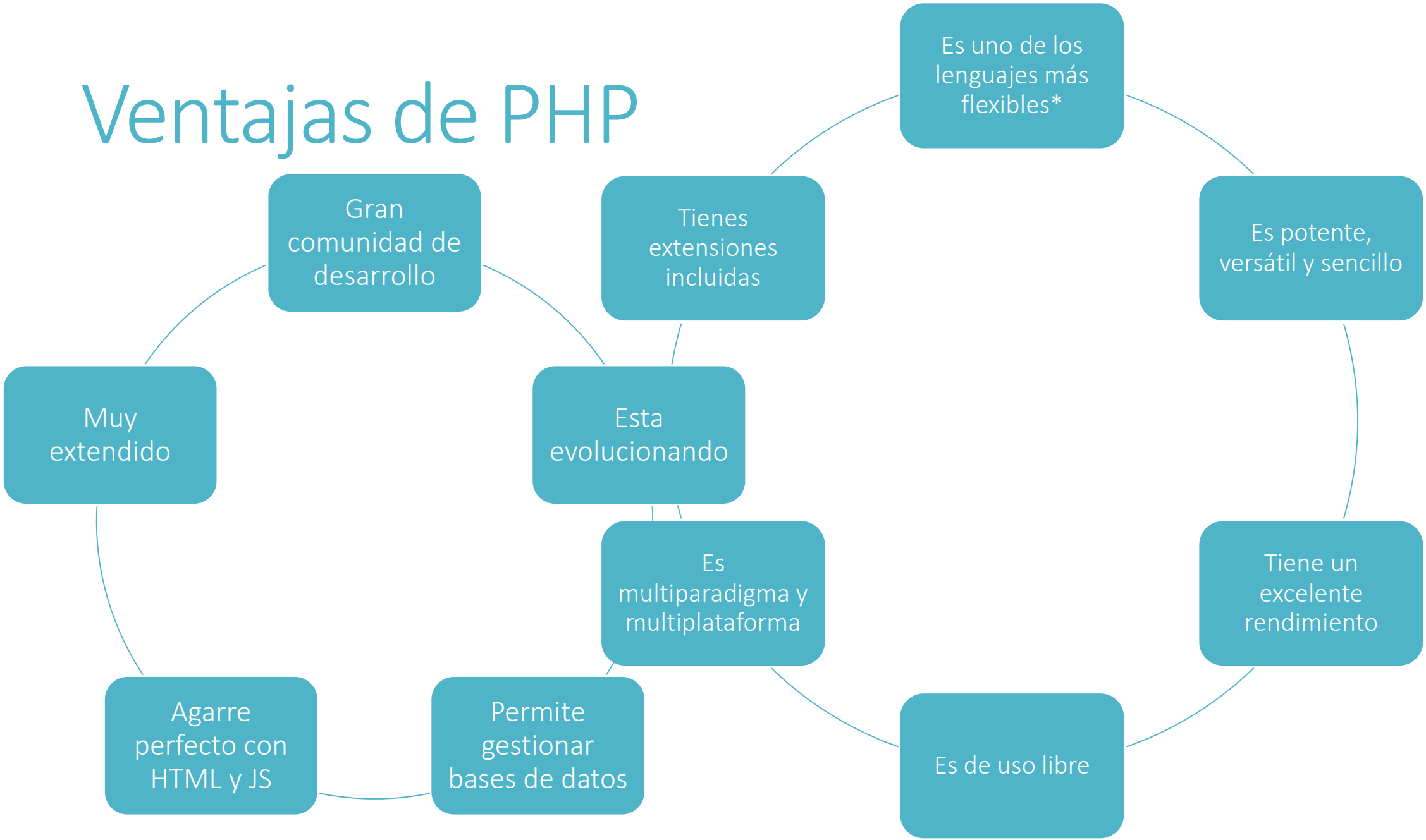
¿Qué es PHP?

Es un lenguaje de propósito general

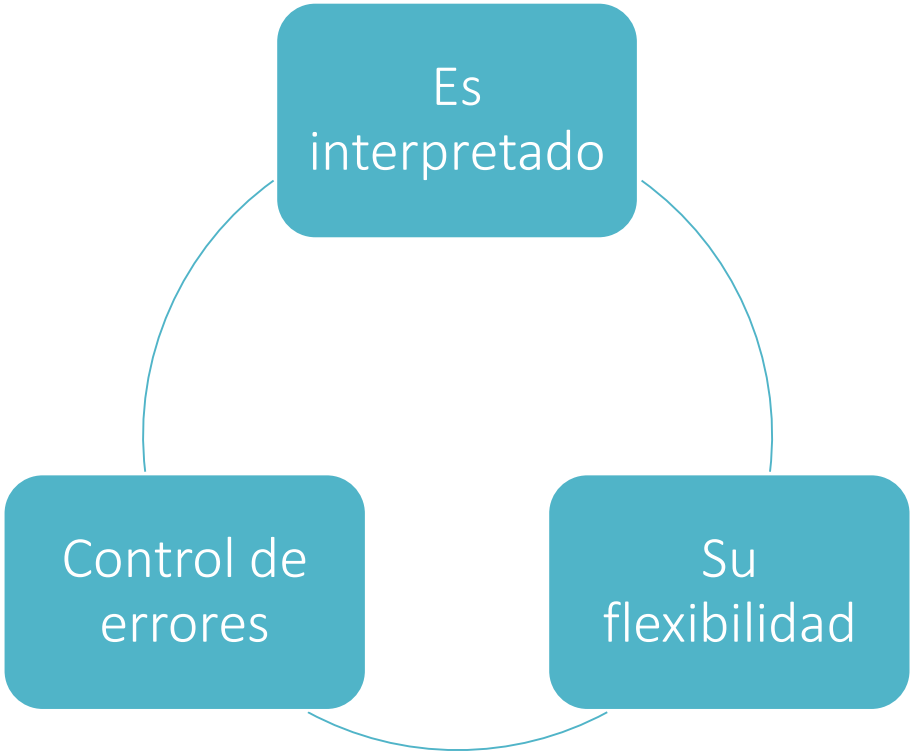
Se diseño para la programación web del lado del servidor

Es un lenguaje interpretado

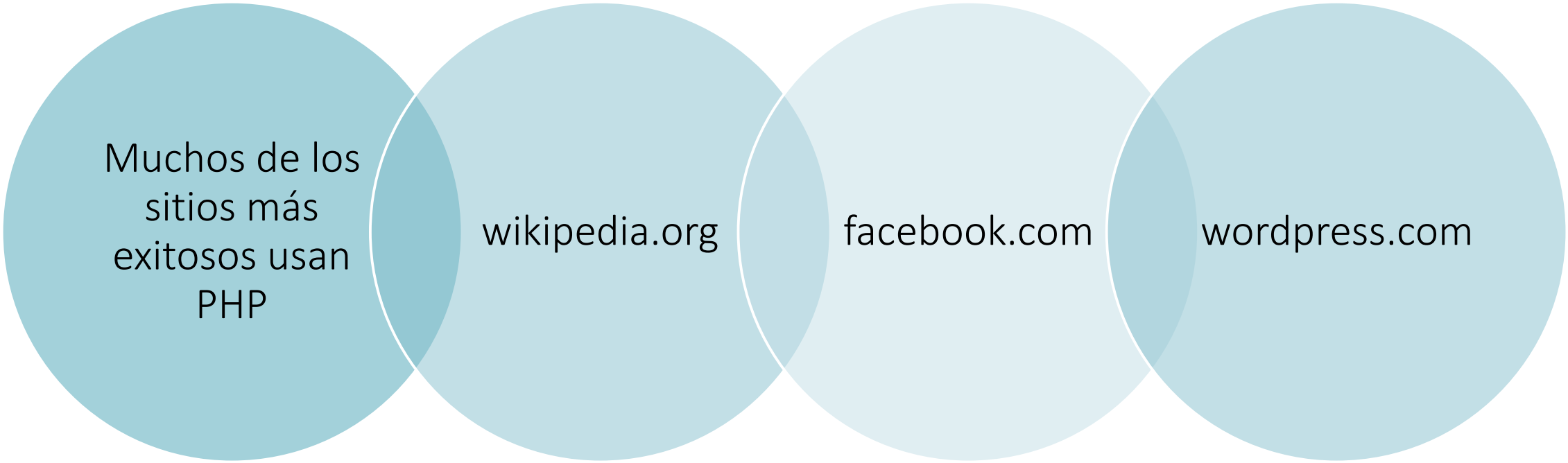
Ventajas de PHP



Desventajas de PHP



PHP en la vida



La arquitectura cliente servidor

¿Qué ocurre detrás de nuestro cable cuando nos conectamos a internet?

El usuario común abre el navegador favorito y teclea la página a la que quiere acceder

En alguna parte del mundo existen computadoras llamadas servidores

Ejercicio: Encender el equipo que tienes a tú disposición, la contraseña de acceso es labintelxx2015-2 donde xx es el número de equipo

IP

Todo equipo conectado a una red tiene asignada una dirección llamada IP

Internet es una red

Al acceder a cualquier ordenador esto se hace mediante su dirección IP

IP internas, externas y de bucle local

Las IP internas sirven para identificar a un equipo en una red local

Las IP externas sirven para identificar el equipo en una red más amplia, como lo es internet

La IP de bucle local permite conectar al equipo en que se trabaja, es decir, así mismos

Ejercicio: Accede a la página 127.0.0.1, desconecta el cable de red y vuelve a acceder a esa página, conecta el cable de red

Servidor web y cliente

Cuando un usuario (cliente) solicita una página web por medio de un programa cliente (el navegador) se conecta con una computadora que la almacena

La computadora que almacena esa página tiene funcionando un programa que busca la página para enviarla al cliente

El servidor web es aquel equipo que almacena páginas web con un programa instanciado que devuelve esa página al cliente que la solicitó

DNS

Un usuario no escribe la dirección IP de un servidor web, escribe un nombre

El Servidor de Nombre de Dominio es un servidor que relaciona direcciones IP con el nombre de un dominio

Es más fácil recordar un nombre que un número

Permite movilidad entre servidores, es decir cambiar físicamente de un servidor a otro

Resolver el nombre

Ejercicio:
Accede a la consola y escribe ping túPáginaFavorita.com, observa como se resuelve la IP, escribe ahora esa IP en el navegador
Busca el archivo de hosts de Windows y simula un DNS, experimenta accediendo a 127.0.0.1 y a localhost en el navegador

Cuando se solicita una página se conecta primero al DNS

El DNS busca en su lista la página por su nombre devolviendo la IP al navegador, al proceso de búsqueda se le llama resolver el nombre

Nota: Una vez que el navegador obtiene la IP del servidor este procede a realizar la conexión directa, si al navegador se le proporciona una IP, este no se conecta con un DNS, el navegador conoce al DNS por medio del ISP

Proxy

Dado a que en la antigüedad las redes eran lentas se creo algo que se le conoce como servidor proxy

Este servidor almacena copias de las páginas visitadas con más frecuencia

El servidor sirve a una determinada área con la finalidad de tener la información a la mano

Tiene la desventaja de que no siempre se puede tener la página más reciente, además de que con la mejoras de infraestructura esta perdiendo sentido

TCP/IP

Conectar millones de equipos no es fácil

Es necesario que se normalice la manera en que se comunican

Un protocolo de comunicación hace que se comuniquen todos de la misma forma

Todos los equipos en internet se comunican siguiendo la pila de protocolos TCP/IP

Esta pila abarca aspectos como tipos de información a manejar

HTTP, FTP, HTTPS, SMTP, POP, TELNET

Modelo OSI

La ISO es una agencia encargada de normalizar todo lo que se pueda normalizar

El modelo OSI (Interconexión Normalizada Abierta) creado por la ISO reúne todos los protocolos de los distintos servicios de una red, se divide en diversas capas que va desde nivel de hardware o físico hasta la de aplicación

Cualquier equipo, con cualquier sistema operativo debe respetar dichas normalizaciones para que sea útil al poderse conectar a una red

Paquetes

Toda la información que viaja en una red de transmite en pequeños fragmentos llamados paquetes

Los paquetes en el destino se reagrupan

Los paquetes pueden viajar por diferentes caminos, pueden viajar paquetes simultáneamente

Si un paquete se pierde o se deteriora sólo es necesario retransmitir ese paquete

Estructura de un paquete

Cabecera

- Protocolo: Según el servicio
- Tamaño y checksum: Este se usa para verificar que el paquete haya llegado correctamente, de lo contrario de pide su reenvío
- Direcciones IP: IP origen e IP destino, PHP puede leer ese paquete y determinar la IP del cliente en sistemas seguros, si existe un proxy su IP puede interponerse como IP cliente, PHP tiene un sistema para recuperar la del cliente original
- Puerto: Se verá enseguida
- Información adicional: Por ejemplo la ruta que sigue el paquete

Cuerpo

Puerto

Cada servicio implica el uso de uno o varios puertos

Un puerto es una canal numerado para transmitir distintos tipos de información

Un S.O. reconoce más de 65000 puertos

Los primeros 1024 se les conoce como puertos bien conocidos, usados para servicios concretos

Los siguientes puertos no tienen un propósito más que usarse como se desee

Una computadora se dice que **habla** cuando transmite datos por un puerto y escucha cuando los recibe o esta en espera de ellos, una comunicación implica una computadora hablando y otra **escuchando**

Cortafuegos

Muchos puertos programables implica que se puede hacer transferencias especificas implementadas por un programador

Muchos puertos implica vulnerabilidad sino están cerrados los que no se usan

Un puerto cerrado es aquel que no permite el paso de información

La realidad es que en los sistemas comerciales existen puertos abiertos aunque no se usen implicando riesgo de intrusión

El **cortafuegos** o firewall es un programa encargado de atenuar dicho riesgo

Socket

Cuando se escribe en el navegador https://sites.google.com/site/laurasandova/ se llama a la página principal alojada en el servidor web, como se esta comunicando por https (portocolo de transferencia de hipertexto seguro) se asume que el servidor escuha por el puerto 443.

Cuando el servidor escucha por otro puerto, se debe especificar, suponiendo que es el 44443 se debe escribir https://sites.google.com/site/laurasandova:44443, a **dicha notación se le llama socket** (dirección:puerto o IP:puerto)

Sitio dinámicos y estáticos

Una sitio es estático cuando

- Usa tecnologías del lado del cliente
- El servidor siempre envía la misma información
- Su contenido no cambia

Un sitio es dinámico cuando

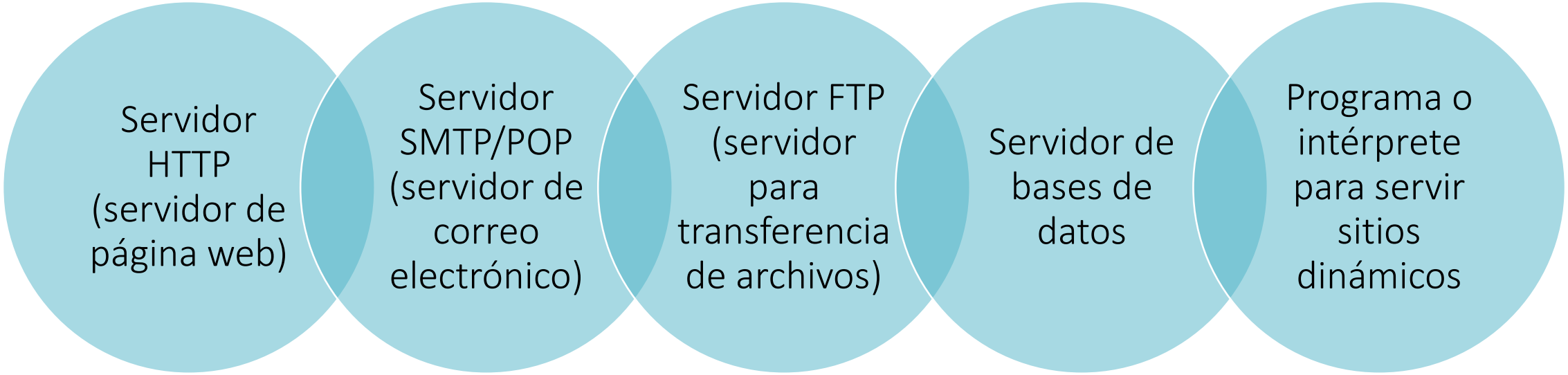
- Los contenidos son dependientes de ciertos datos
- El contenido se genera al momento de la solicitud
- Emplea lo que se conoce como programación del lado del servidor :)

Cuando se llama a una página **estática** el servidor la entrega al cliente tal cual la tiene alojada, cuando es **dinámica** se crea el contenido por medio de un proceso en el servidor entregando eso al cliente

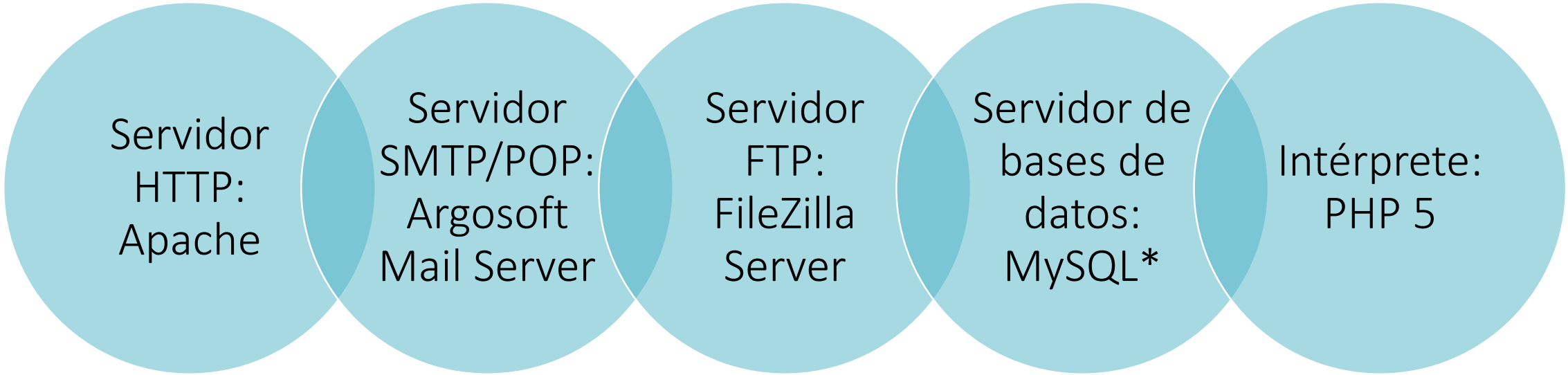
Palabras clave

- PHP (1)
- Ventajas de PHP (2)
- Dirección IP (3)
- Dirección externa, interna y de bucle local (4)
- Servidor web (5)
- DNS (6)
- Resolver el nombre (7)
- Proxy (8)
- Protocolo (9)
- Pila TCP/IP (10)
- Modelo OSI (11)
- Paquete (12)
- Puerto (13)
- Puerto bien conocido (14)
- Puerto cerrado (15)
- Cortafuego (16)
- Hablar y escuchar (17)
- Socket (18)
- Sitio dinámico (19)
- Sitio estático (20)

Componentes de un servidor web



Componentes que usaremos



Nota: los componentes mencionados tienen la característica de ser bastante robustos, seguros y de uso totalmente libre

Componentes a detalle

Apache: Permite que el equipo pueda atender peticiones como servidor web. Es el servidor más extendido, es gratuito, es excesivamente seguro ante ataques y es de fácil configuración y administración.

ArgoSoft Mail Server: Es de necesario para poder recibir y enviar correo electrónico desde el servidor creado.

FileZilla Server: Tiene la característica de ser muy fácil de configurar además de ser gratuito, es necesario para poder comunicar archivos entre equipos remotos, por ejemplo para actualizar la página web.

MySQL: Tener bases de datos en un sitio es indispensable. Es un manejador de base de datos gratuito, sus bases de datos son pequeñas y su motor es muy veloz, ofrece además integridad y seguridad.

PHP5: Cuando un cliente solicita una página desarrollada en PHP, el interprete ejecuta el código de dicha página entregando un resultado, este resultado puede ser contenido HTML de maneara clásica pero puede generar muchos más. PHP tiene la capacidad de usar todos los componentes mencionados.

En Windows

Existe un paquete que se llama AppServ que contempla:

Instalar el servidor Apache, la base de datos MySQL, el interprete de PHP y una interfaz gráfica vía web para administrar la base de datos llamada phpMyAdmin

Las páginas oficiales para descargar los componentes son:

AppServ: http://www.appservnetwork.com/

ArgoSoft Mail Server: http://argosoft.com/rootpages/mailservernet/default.aspx

FileZilla Server y FileZilla Client: https://filezilla-project.org/

Existen soluciones que ya tiene implementadas algunas versiones de Windows, no se consideran por las restricciones que tiene el fabricante en cuanto su uso

En Linux

AppServ, ArgoSoft Mail Server, Filezilla Server no existen en Linux.

La alternativa de AppServ es un paquete llamado LAMP, el cual hace uso de MySQL/MariaDB como gestor de bases de datos, Apache como servidor web y Perl, PHP o Python como lenguajes de programación.

IRedMail como servidor de mail el cual instalara Apache y MySQL.

PureFTPd como alternativa a Filezilla Server para crear un servidor FTP.

Consideraciones

Con todo lo anterior ya es posible realizar la instalación de un servidor web en tú propio equipo, se debe tener en cuenta:

- Se debe contratar un dominio para estar en la lista de DNS
- Si no se hace es posible acceder a los servicios conociendo la IP de la interfaz del enrutador que conecta la red local a internet
- Es necesario configurar el enrutador para que cuando reciba una petición de algún servicio, sepa a que equipo de la red local dirigirlo
- Contratar con el ISP una IP fija (normalmente varían conforme el transcurso del tiempo)
- Si no se realiza lo anterior se debe consultar la IP pública frecuentemente
- Todo lo anterior se puede hacer contratando un servicio de hospedaje que ofrece más servicios aún, la diferencia es que no se hizo nada para configurar al servidor ni se tiene acceso físico
- Es posible hacerse revendedor de algún servicio de hospedaje

Tema 2

Iniciando con PHP

Primer pasos

Los archivos en donde formalmente se escribe un programa de PHP son en sí archivos de texto plano

Se debe verificar la codificación con la que es guardada el archivo, se recomienda UTF8

El archivo se debe guardar en una carpeta accesible por el servidor HTTP, si se instalo AppServ esta es C:\AppServ\www

Cuando se hace una petición al servidor web, este busca un archivo llamado index.html, index.htm o index.php que es el que mostrará por defecto

Considerar que cada carpeta puede tener distintas directivas, por ejemplo restricción de acceso si no existe un archivo index en ella, todo ello mediante el servidor HTTP

Hola mundo en PHP

Ejercicio:

Abrir un editor de texto plano y escribir las siguientes líneas:

```
1.  <!doctype html>
2.  <html>
3.  <head>
4.      <meta charset="utf-8">
5.      <title>Hola mundo</title>
6.  </head>
7.  <body>
8.      <h2><?= "hola mundo" ?></h2>
9.  </body>
10. </html>
```

Guardar el archivo con el nombre index.php en la carpeta www, abrir el navegador y escribir la dirección de bucle local

CGI

Tecnología que permite que un cliente pueda solicitar los datos que proporciona un programa alojado en el propio servidor web.

Este programa es compilado y almacenado con la extensión .cgi en una carpeta preconfigurada por el servidor HTTP

Existe un protocolo para intercambiar datos con el programa y el cliente

Eran la primera forma de sitios dinámicos

CGI vs PHP

Ejercicio:

- Descarga el material de tema 1 en la página http://labintel.centrotr.com/cursos
- Descomprime la carpeta y ubica los archivos donde se te indique
- Saca tus conclusiones

Tarea:

- Instala la plataforma AppServ en tú equipo, será necesaria para que realices las tareas y el proyecto
- Escribe un programa CGI que sume dos números ya definidos
- Cuando accedes a un servidor web y no existe un archivo index, Apache crea una respuesta HTML que no es más que un navegador por los directorios de la carpeta, es una vulnerabilidad ya que cualquiera puede ver la estructura de la página, investiga cómo hacer que una carpeta sea restringida si no existe un archivo index al consultarla desde una petición HTTP (nota, investiga sobre el archivo .htaccess, no tomes la opción de crear un index vacío)

Integración con otros lenguajes

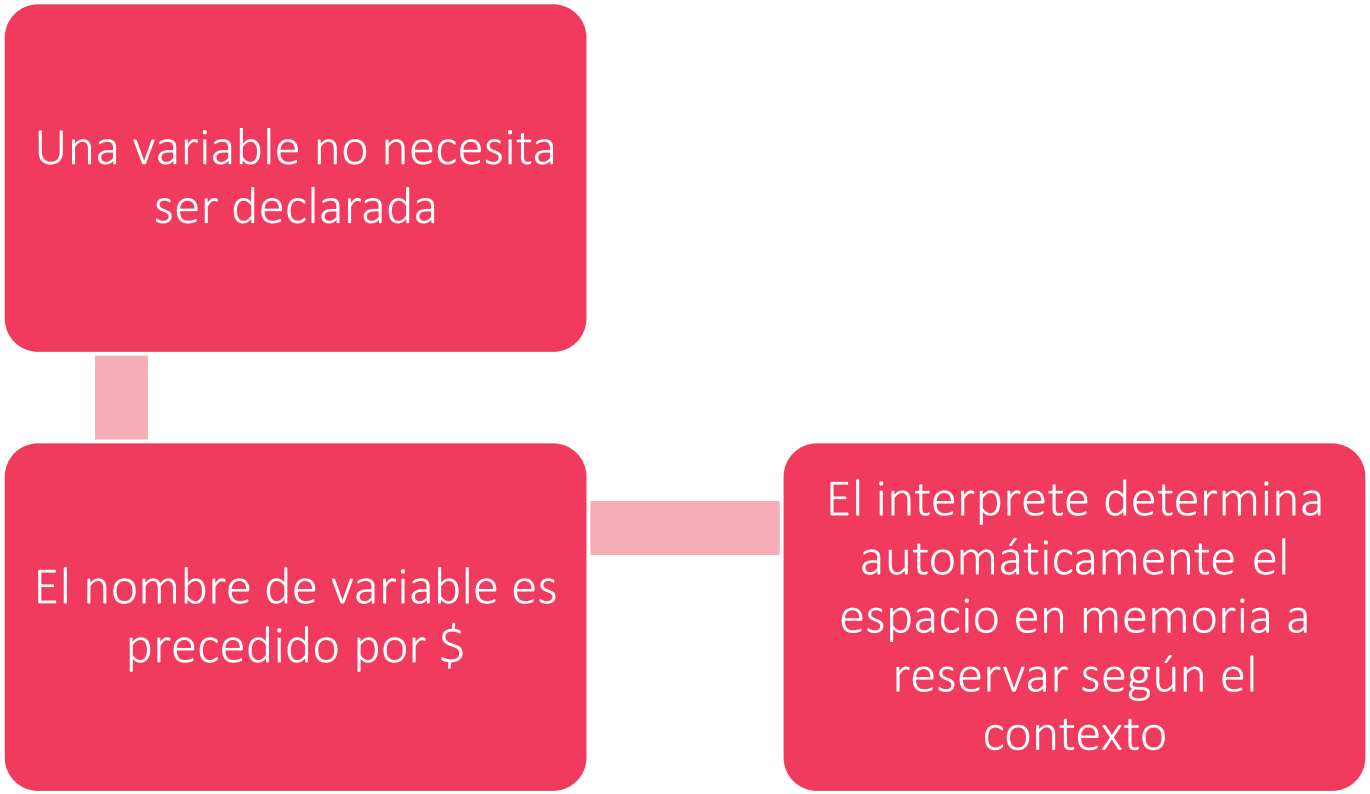
Existen lenguajes web del lado del cliente como JS y HTML, son los únicos que entiende el navegador

PHP necesita ofrecer una alta integración con dichos lenguajes (capitulo 8)

PHP permite incrustar código HTML mostrando cadenas que se enviarán al navegador

La función por excelencia para mostrar cadenas es ***echo***, en realidad es un constructor del lenguaje

Variables en PHP



Nombre de variables

El nombre de una variable es similar a otros lenguajes

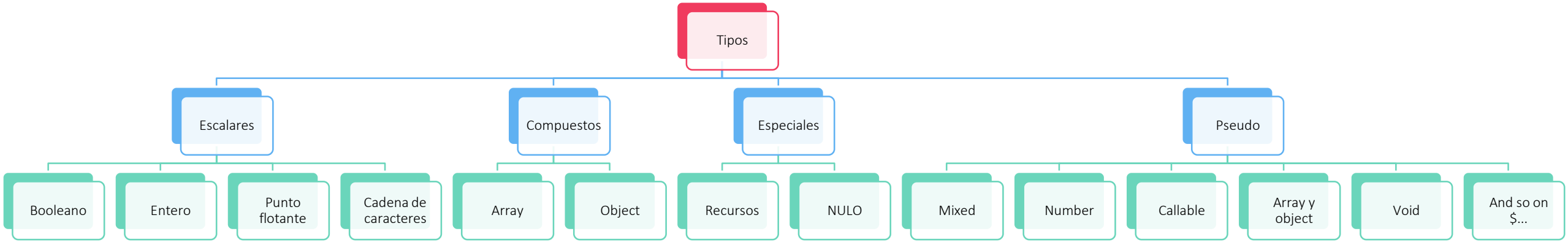
El nombre de una variable debería ser coherente a su uso

PHP es sensible a las mayúsculas

$this es una variable especial

*Nombre válido: [a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]**

Tipos de datos



Booleano

Es un valor de verdad, puede ser verdadero o falso

Se le nombra (bool) [PHP 4.2.0] o (boolean) para la conversión de dato

Tiene dos constantes TRUE o FALSE (insensibles a mayusculas)

Normalmente se obtiene de una operación lógica que es evaluada por una estructura de control

Entero

Pueden ser expresados en distintas notaciones

Pueden llevar signo + o -, PHP no soporta enteros sin signo

Tamaño de un entero: PHP_INT_SIZE

Valor máximo: PHP_INT_MAX

Linux AMD64 soporta enteros de 64 bits, Windows x86, x64 y Linux x86 soporta enteros de 32 bits

Al desbordarse un entero se interpreta como número de coma flotante

Al dividir dos números enteros se convierten en coma flotante

Se nombra (int) o (integer) para conversión explicita

La conversión normalmente es automática

Formalización de enteros

decimal : [1-9][0-9]* | 0

hexadecimal : 0[xX][0-9a-fA-F]+

octal : 0[0-7]+ (si se pone algún caracter no válido, se ignoran los consecuentes)

binario : 0b[01]+ (a partir de PHP 5.4.0)

entero :

[+-]?decimal
| [+-]?hexadecimal
| [+-]?octal
| [+-]?binary

Número de punto flotante

Llamados números flotantes, de coma flotante, reales, dobles

Pueden ser por ejemplo 1.234, 1.2e3, 1E2, 1E-2

Su precisión es de 64 bits IEEE

Su uso puede tener resultados no deseados dado a los problemas de precisión, existen funciones para controlarlo

Se nombra (double) o (float) [PHP 4.2.0] para la conversión de un dato a float

Cadena de caracteres

Un carácter es un byte

Una carácter soporta hasta 256 valores distintos

El valor máximo de una cadena es de 2147483647 bytes (2 GB)

Un string es un arreglo de bytes acompañado de un entero que dice su longitud en el buffer

Su codificación depende de la codificación en la que este el fichero, a nivel binario no almacena el como traducir su contenido

Se pueden concatenar mediante .

Se nombra como (string) para la conversión de algún otro tipo de dato

Nombramiento de cadenas

Existen 4 maneras de especificar una cadena:

'entrecomillado simple';

"entrecomillado doble";

<<<EOT heredoc
EOT;

<<<'EOD' nowdoc
EOD;
[PHP 5.3.0]

Entrecomillado simple

Es la manera más sencilla de delimitar una cadena

La sentencia es la siguiente: 'esta es una cadena';

Se puede incluir una variable mediante una concatenación:

```
'Nombre del alumno: '.$nombreAlumno;
```

Para incluir una comilla simple se necesita escapar:

```
'Y su personaje \'inmutaba\' a cualquier espectador';
```

Para incluir una diagonal invertida igualmente se escapa:

```
'La ruta del archivo es: C:\\www\\ejemplo.php';
```

Para incluir comillas dobles solamente se escriben:

```
'Y le hacían llamar "la maraca"';
```

Entrecomillado doble

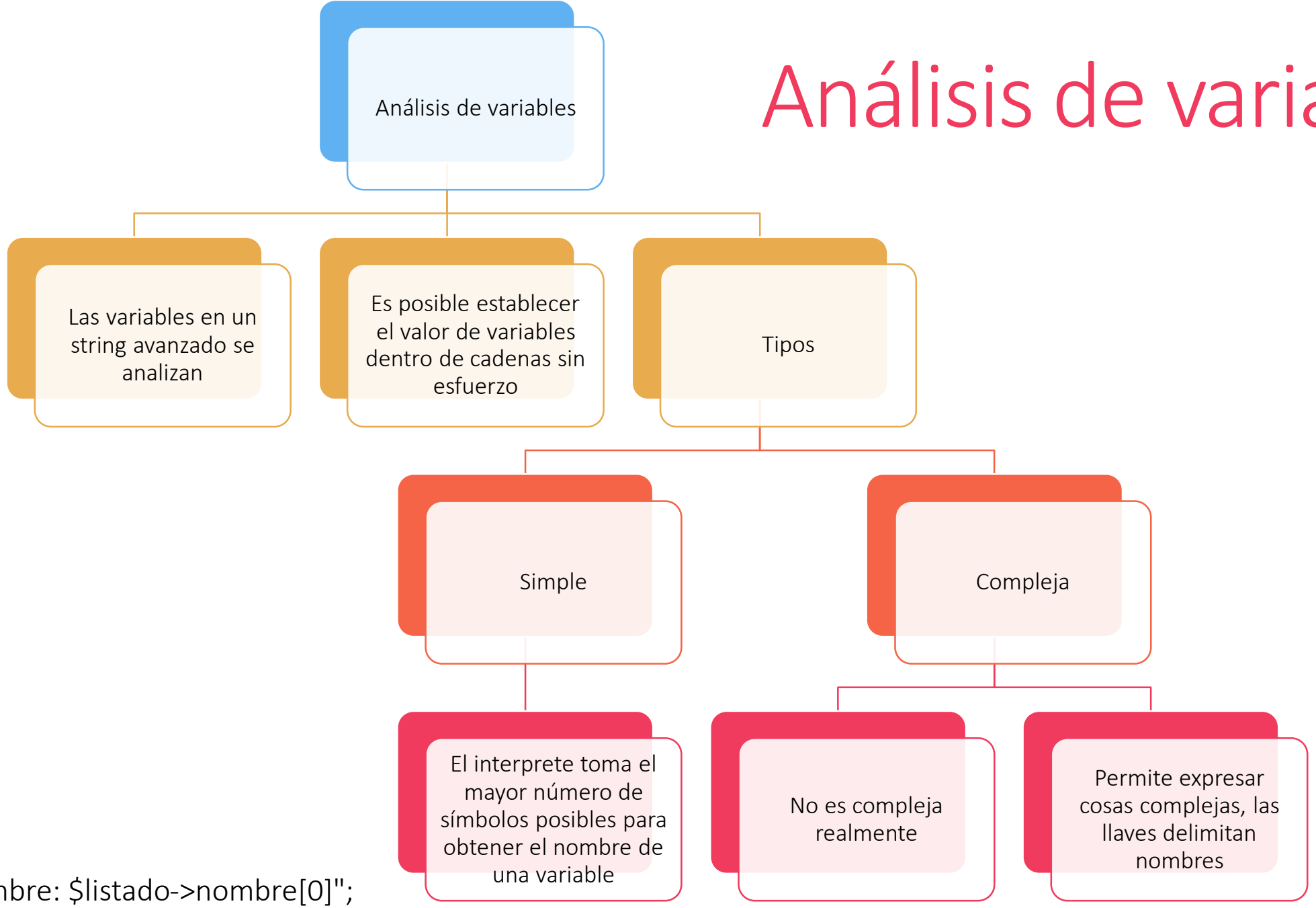
Es más robusto que el entrecomillado simple

Se define así: "esto es una cadena mejor que la otra, en realidad es igual"

Permite tener más sentencias de escape como \" \n \t etc.

Su característica más importante es...

Análisis de variables




```
"Este es mi nombre: $listado->nombre[0]";
"Esto proviene de un lado complicado: {${$lista->obtieneNombre($alumnos->nombre['completo'][0])}}";
```

Expansión de variables

Simplemente es el hecho de que cuando una variable esta en una cadena, se toma su valor

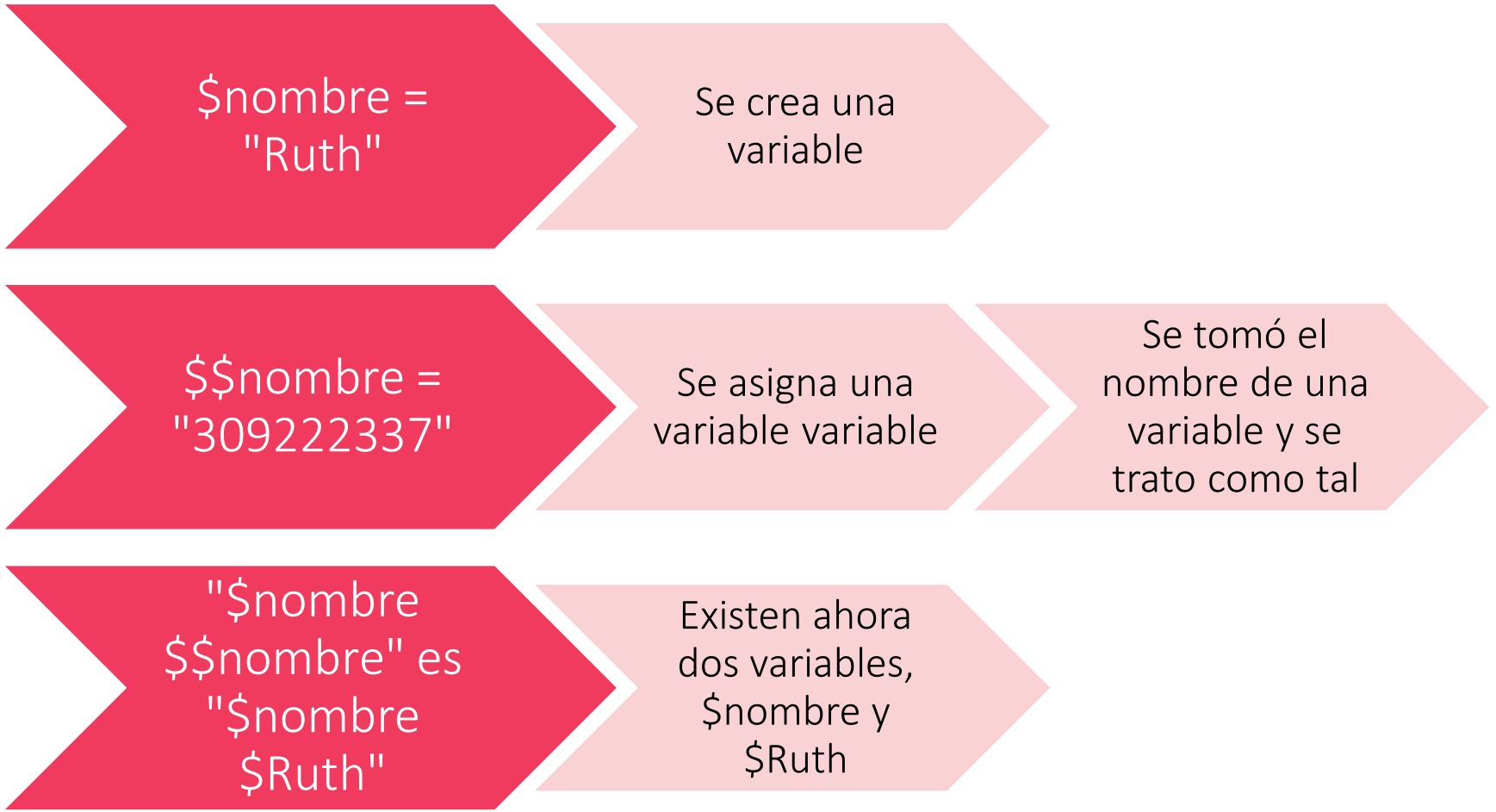
Proviene de un análisis de variables

Para omitirlo se puede escapar la variable o bien usar entrecomillado simple

Variables variables

PHP puede reconocer variables que refieren a otras variables

Son variables que se pueden usar dinámicamente



Asignación por referencia

Al asignar una variable a otra, se copia el valor, por lo que se obtienen dos variables independientes

Se puede asignar variables por referencia anteponiendo ampersand &$variable, de esta manera lo que le pase a esa variable le pasará a la otra

Asignar una variable significa que se trabaja con un alias o mejor dicho, se apunta a esa variable

Sólo las variables con un nombre pueden ser referenciadas así

Constantes

Son identificadores asociados a un valor simple

Su valor es el mismo durante todo el script

Existen contantes predefinidas y constante reservadas

PHP_EOL define por ejemplo el fin de línea correcto

define ("PI", M_PI);

Arreglos

Los arreglos en PHP son mapas ordenados

El que sea un mapa ordenado significa que se tiene una extraordinaria potencia en este tipo de datos

Un arreglo se define con el constructor array

Un arreglo puede usarse como

- Matrices
- Vectores
- Pilas
- Colas
- Árboles
- Diccionarios
- Colecciones
- Tablas
- …

Arreglos indexados

El índice es un número y se puede acceder al contenido así $arreglo[índice]

El índice puede ir explicito o implícito

- Explicito
  - $arreglo = array(1 => $nombre, 50 => 1 , 50=> "hola");
- Implícito
  - $arreglo = array($nombre, 1 , "hola");
- Mixto
  - $arreglo = array(true, false => "hola", 1 => $nombre);

Se puede agregar un nuevo elemento especificando el índice consecuente o así $arreglo = "más cosas"

Arreglos asociativos

Usan una cadena como índice o mejor dicho clave

Permite organizar mejor la información

Permite búsquedas de información rápida

```
$arreglo = array("nombre" = "Isauro", "dirección" => "Av. Universidad 2000");
```

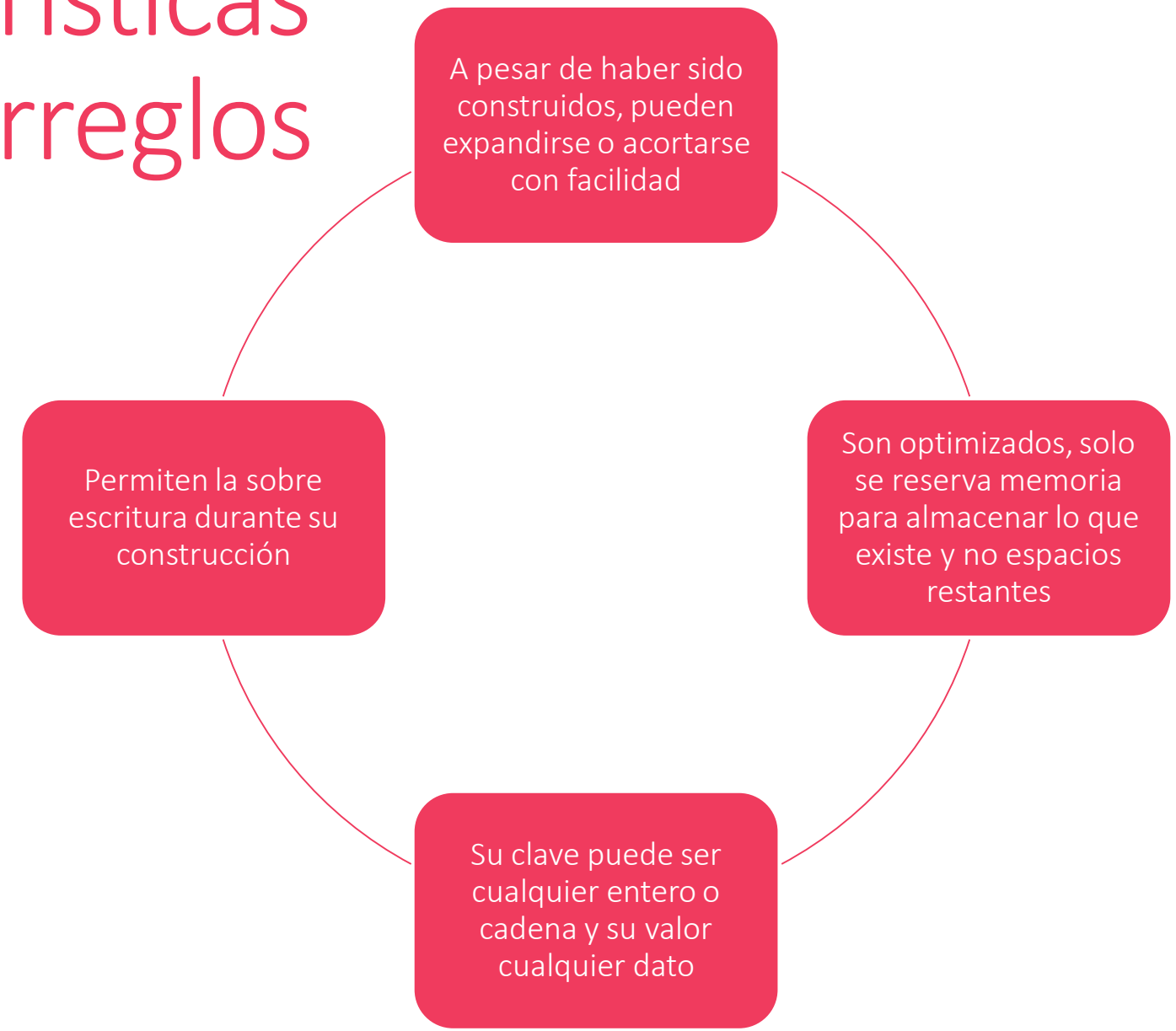
Arreglos mixtos y multidimensionales

Es posible combinar distintos tipos de claves, sea numéricas o cadenas

Es posible que un valor sea otro arreglo, por lo que se pueden tener una estructura compleja de información

```
$ejemplo = array(
    "valor0",
    50 => "valor50",
    63 => true,
    "listaNombres" => array(
        "nombre0" => array(
            "nombre" => "Edgar",
            "apellido" => "García"
        ),
        "nombre1" => "Fulanito1",
        0 => "Inscritos 60",
    ),
);
echo $ejemplo["listaNombres"]["nombre0"][nombre];
```

Características de los arreglos



Variable nula

Sólo posee un único tipo de valor, null

Es insensible a mayúsculas y minúsculas

Es una variable sin valor, existe cuando:

- Es asignada la constante NULL
- La variable se creo sin asignarle nada
- El contenido de la variable fue destruido

Su conversión se realiza mediante una función

Conversión de datos automático

PHP permite la conversión automática de datos en diferentes contextos, esto permite que no se tenga una real necesidad de convertir datos, por ejemplo si se escribe un booleano en donde debería ir un valor numérico se convierte automáticamente, otro ejemplo es un entero desbordado o al dividirse con otro se convierte en flotante, una cadena que contiene un número entero al sumarse se convierte en entero, etc.

Es posible asignar distintos tipos de datos a una misma variable por esta propiedad.

PHP no devuelve errores en la mayoría de los casos por el manejo de tipo de datos que tiene, este manejo es excesivamente robusto.

A veces puede causar conflicto en casos especiales, por ejemplo, si se desea obtener el valor ASCII de un carácter pero existen funciones para obtenerlo, lo cual hace que el lenguaje se adapte a lo común y se especialice en casos

Conversión por casteo

(int), (integer) - forzado a entero (considerar posible truncamiento)

(bool), (boolean) - forzado a booleano

(float), (double), (real) - forzado a flotante

(string) - forzado a cadena

(array) - forzado a arreglo

(object) - forzado a objeto

(unset) - forzado a NULL [PHP 5]

(binary), b - forzado a cadena binaria [PHP 5.2.1]

Conversión por función

Existe una función para la conversión de dato: bool settype (mixed `&$var` , string `$type`)

Tiene una salida verdadera en caso de éxito o falsa, como primer parámetro tiene la variable a trabajar, el segundo es el tipo:

- "boolean" (o, desde PHP 4.2.0, "bool")
- "integer" (o, desde PHP 4.2.0, "int")
- "float" (únicamente posible desde PHP 4.2.0, sino usar "double")
- "string"
- "array"
- "object"
- "null" (desde PHP 4.2.0)

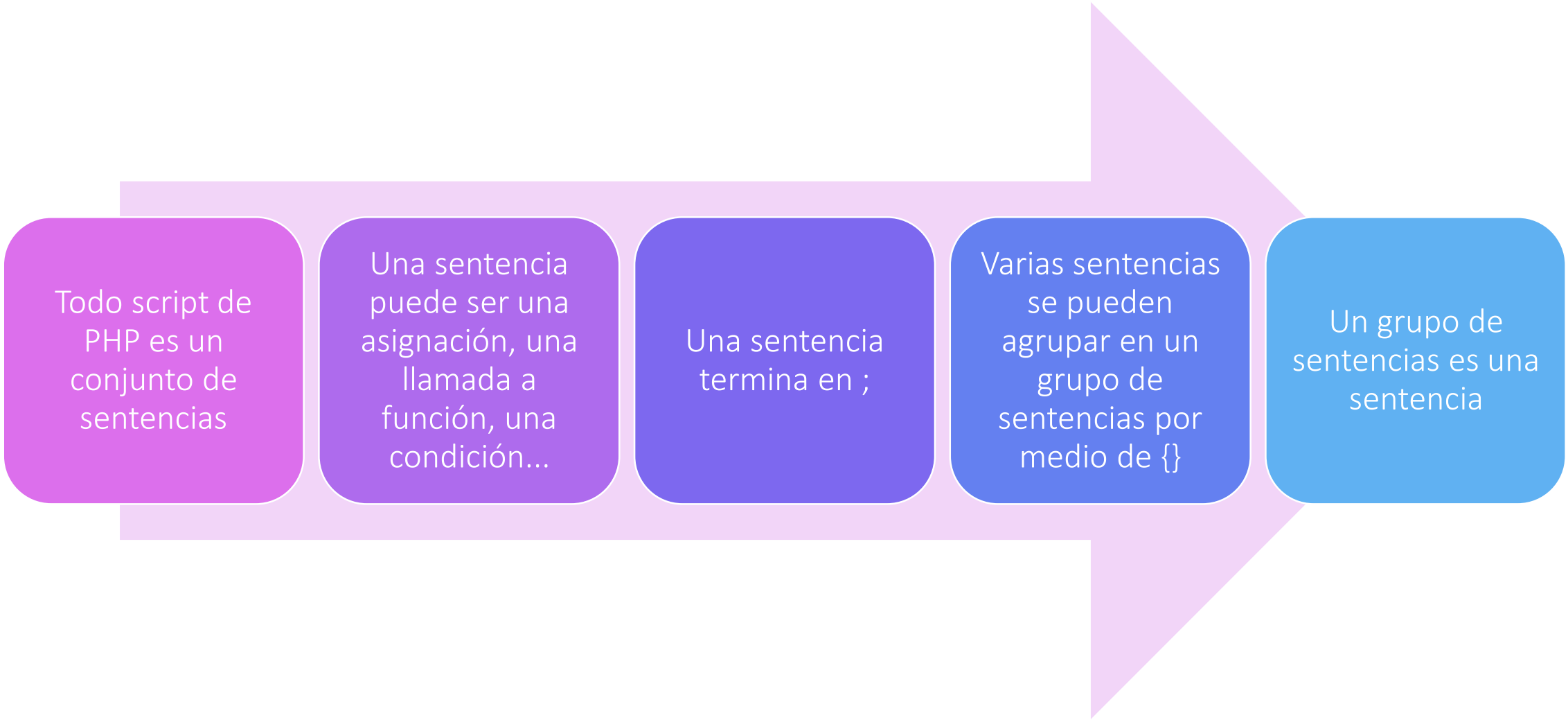
Rastreo de variables

En la depuración de un script de PHP suele ser necesario verificar el contenido de alguna o varias variables (partes de arreglos, arreglos enteros, variables sencillas, objetos, etc.).

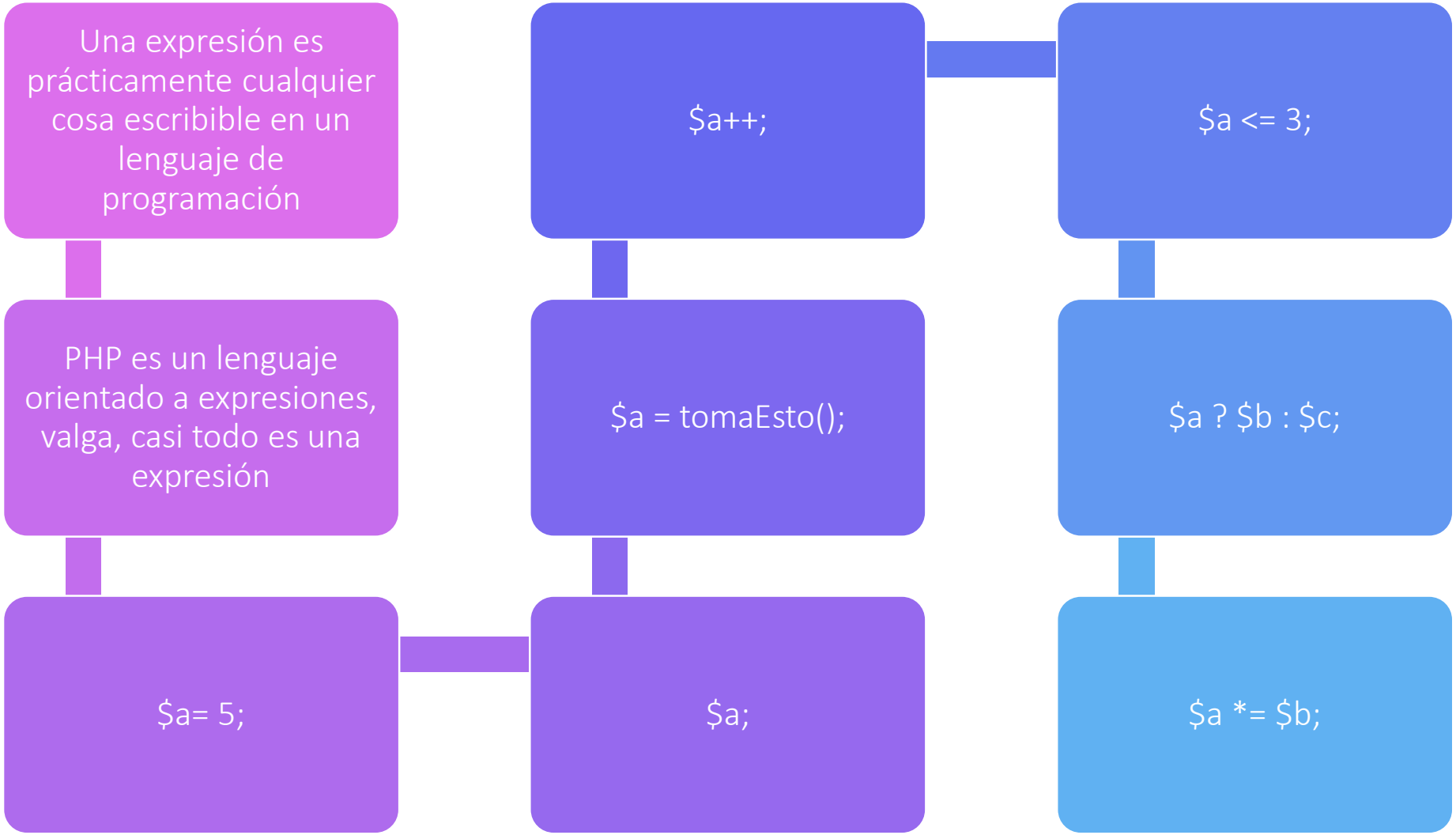
Se puede rastrear el contenido de dicho mediante la función var_dump, la cual recibe tantos elementos como se requieran verificar y devuelve una cadena con su contenido.

void var_dump (mixed **$expression** [, mixed **$...**])

Sentencias


Todo script de PHP es un conjunto de sentencias
Una sentencia puede ser una asignación, una llamada a función, una condición...
Una sentencia termina en ;
Varias sentencias se pueden agrupar en un grupo de sentencias por medio de {}
Un grupo de sentencias es una sentencia

Expresiones



Operadores

Operadores aritméticos
Operadores de asignación
Operadores bit a bit
Operadores de comparación (flexibles y estrictas)
Operadores de control de errores
Operadores de ejecución
Operadores de incremento/decremento
Operadores lógicos
Operadores para strings
Operadores para arrays
Operadores de tipo

En PHP existen diversos tipos de operadores de acuerdo al tipo de dato y función que se desee realizar.

if

Es un constructor elemental

Evalúa la expresión de manera booleana

Si la evaluación es TRUE se ejecuta la sentencia

```
if (expresión)
    sentencia
```

```
if (expresión):
    sentencia
endif;
```

else

Extiende la sentencia if

Si la expresión valuada es FALSE se ejecuta la sentencia 2

Si la evaluación es TRUE se ejecuta la sentencia

```
if (expresión)
        sentencia
else
        sentencia 2
```

elseif ó else if

Extiende la sentencia if

Si la expresión valuada es FALSE, se tratará de ejecutar el resto de las sentencias elseif hasta que alguna se value como TRUE, de lo contrario se ejecuta la sentencia else si existe

```
if (expresión)
        sentencia a
elseif
        sentencia b
elseif
        sentencia c
else
        sentencia d
```

```
if (expresión):
        sentencia a
elseif:
        sentencia b
elseif:
        sentencia c
else:
        sentencia d
endif;
```

switch

Similar a varias sentencias elseif, permite evaluar una variable que puede tomar distintos valores

Permite la sintaxis alternativa

Solo ejecuta la sentencia si el valor es flexiblemente igual a la variable

Necesita la estructura de break* al final de una sentencia

```
switch(variable) {
    case valor1:
        sentencia
        break;
    case valor2:
        sentencia
        break;
    case valor3
        sentencia
        break;
    case valor4;
    case valor5;
        sentencia
        break;
    default:
        break;
}
```

?: u operador ternario

Es una estructura condicional

Si la expresión 1 es verdadera se evalúa expresión 2

Si es falso se evalúa expresión 3

Se puede anidar expresiones

(expresión1) ? (expresión2) : (expresión3)

while

Una iteración es la ejecución de la sentencia en un bucle una vez

Se ejecutará la sentencia tantas veces como la expresión sea evaluada verdadera

En el inicio de cada iteración se evalúa la expresión

```
while (expresión)
        sentencia

while (expresión):
        sentencia
endwhile;
```

do-while

Hace lo mismo que while, sólo que la evaluación se hace al final

Garantiza que la sentencia se ejecutará por lo menos una vez

Tiene una sola sintaxis

```
do {
        sentencia
} while(expresión);
```

for

for(expresión1; expresión2; expresión3)
sentencia

Posee también la sintaxis alternativa*

Es la es bucle más complejo

La expresión 1 se evalúa al inicio del bucle

El bucle se ejecuta mientras la expresión 2 sea verdadera, esta expresión se evalúa al comienzo de cada iteración

La expresión 3 se evalúa al fina de cada iteración

Una expresión que se evalúa se tiene que ejecutar

Pueden anidarse varias expresiones separadas por ,

foreach

Es la es bucle práctico para recorrer arreglos

Sólo acepta arreglos

Recorre todo el arreglo por medio del puntero de arreglo

Al final pone el puntero al inicio del arreglo

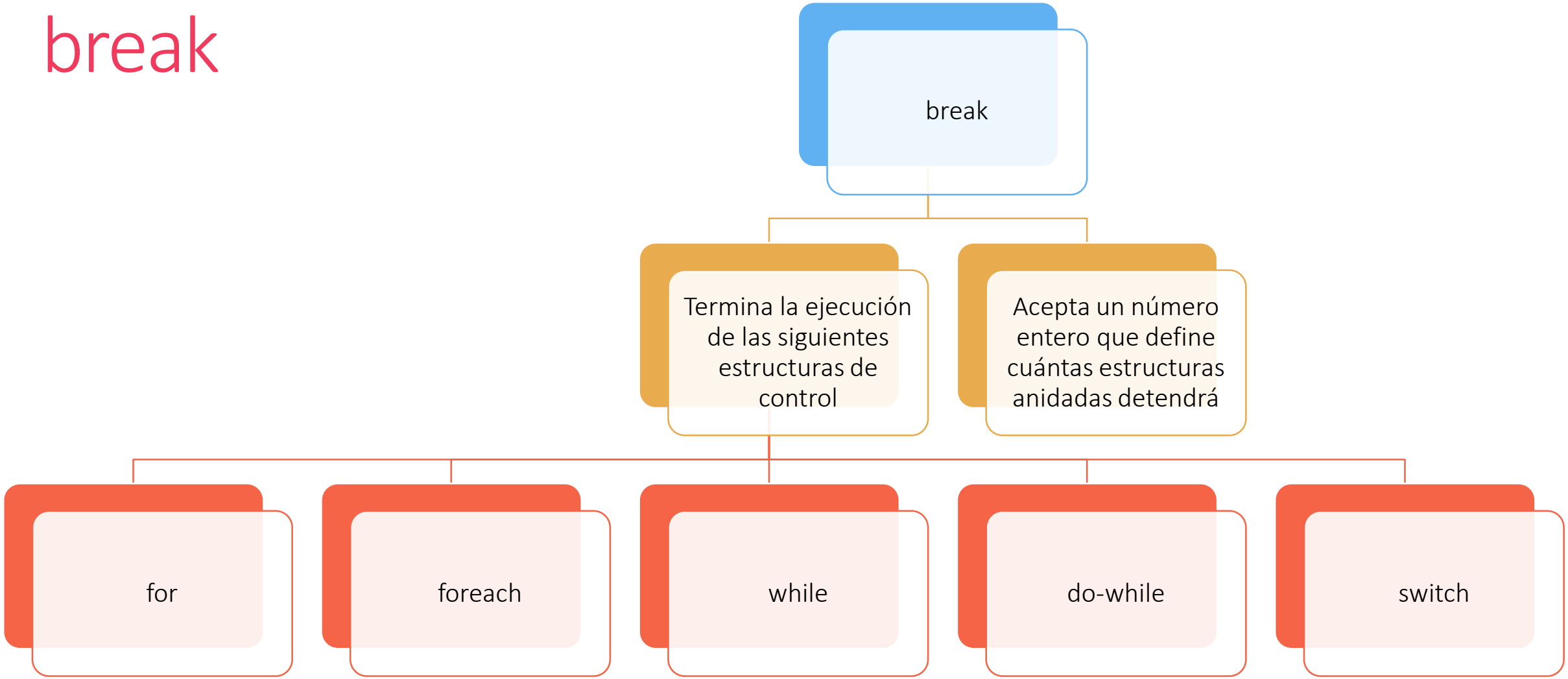
En cada iteración, la sentencia dispone de las variable valor y si se definió clave

Al inicio de cada iteración clave toma la clave del arreglo apuntada y valor el valor apuntado

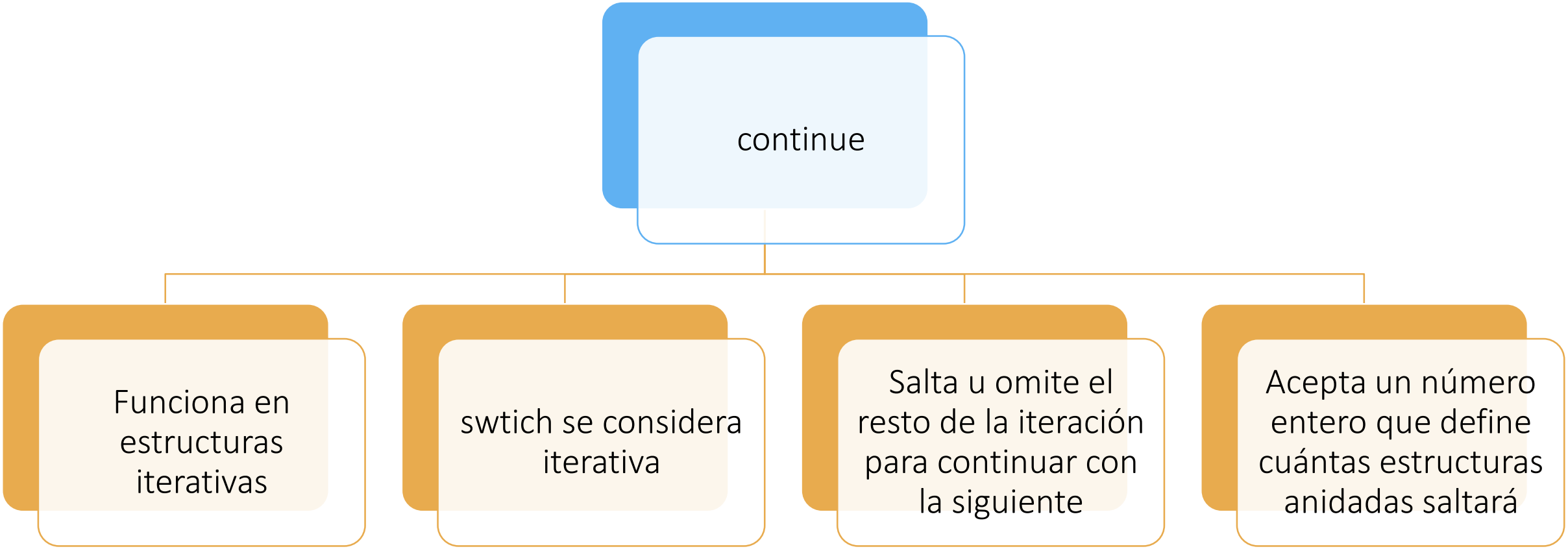
```
foreach(arreglo as $valor)
        sentencia

foreach(arreglo as $clave => valor)
        sentencia

foreach(arreglo as $clave => &$valor)
        sentencia
```

Si se desea modificar el valor real del arreglo, valor debe referenciarse

break


break
Termina la ejecución de las siguientes estructuras de control
Acepta un número entero que define cuántas estructuras anidadas detendrá
for
foreach
while
do-while
switch

continue


continue
Funciona en estructuras iterativas
swtich se considera iterativa
Salta u omite el resto de la iteración para continuar con la siguiente
Acepta un número entero que define cuántas estructuras anidadas saltará

return

Devuelve el control al módulo ejecutado

Si el módulo en el principal se termina la ejecución del script

En el caso de las funciones tiene un comportamiento especial

include y require

Dichas estructuras sirven para anexar al script principal código en otros archivos

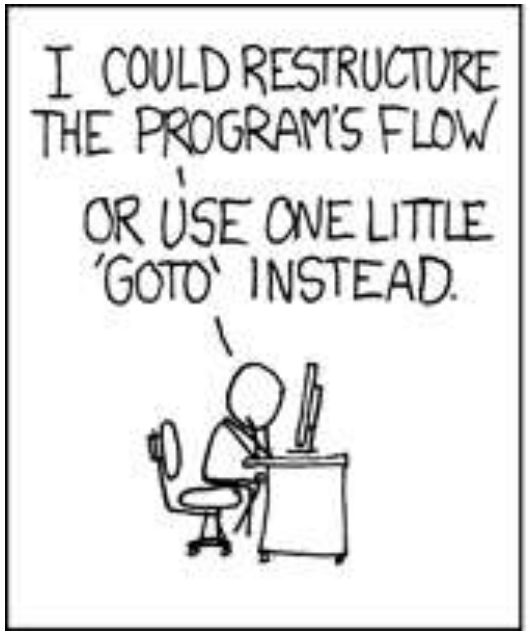
Al incluir un archivo esté se evalúa en el punto donde se hace la referencia

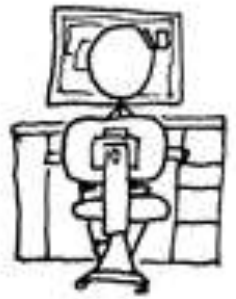
Si include no encuentra el archivo se despliega una advertencia, si require no lo encuentra se detiene el script y se presenta un error fatal

Todas las variables en el ámbito se heredan, es decir el archivo incluido las tiene, las funciones y variables de ese archivo quedan en el ámbito global***

Adicionalmente existe include_once y require_once, estos no evalúan de nuevo el archivo si ya fue incluido

goto





Se utiliza para ir a alguna parte del mismo script, en el mismo fichero a un punto indicado por una etiqueta.
Su uso no se recomienda
PHP 5.3

Funciones

A menudo se requieren fragmentos de código disponibles en todo un script e incluso en todo un proyecto que tienen un uso común

PHP ofrece un sin número de funciones (más de 5800) y su llamado no implica hacer nada, de hecho el interprete esta optimizado para cargar sólo lo necesario y no tener todo cargado

Pese al número de funciones que proporciona PHP puede requerirse funciones que se apeguen al problema que se resuelve

```
function nombreDeLaFunción(argumentos) {
        Sentencias
}
```

Características de las funciones

Una función puede colocarse dentro de una condicional y no existe hasta que no se ejecute su sentencia

Una función puede colocarse dentro de otra función y no existe hasta que no se llame dicha función

Los argumentos de una función pueden ser opcionales si se pre inicializan (argumento predeterminado) o se omiten (no son necesarios)

Una función puede llamarse así misma (recursividad) lo cual optimiza muchos algoritmos (no pasar más de 100 niveles de recursividad)

La llamada a una función es insensible a mayúsculas y minúsculas

Argumentos


Argumentos
Se pueden pasar por
Se admiten listas de argumentos variables
Valor
Referencia
Valores predeterminados
Se necesitan de funciones para definirlo, PHP 5.6 soporta tokens
Sólo se modifica el valor de la variable dentro de la función
Se modifica el valor dentro y fuera de la función
Si al llamar la función no se menciona toma su valor por defecto

Valores de retorno

La estructura return se puede usar en funciones para detener la función en un cierto punto, es opcional.

Puede tener un parámetro que será el valor de retorno

El valor de retorno, a diferencia de otros lenguajes, puede ser de cualquier tipo

Acerca de las funciones internas

El prototipo de funciones esta incluido en muchas IDE's de desarrollo para PHP y en el manual, interpretarlo es esencial

PHP tiene muchas funciones precargadas para hacer casi cualquier cosa, muchas de ellas están disponibles para su uso directo (funciones de núcleo) pero muchas otras necesitan extensiones o que PHP sea compilado para funcionar con tales, por ejemplo para el trabajo con imágenes o bases de datos

Ámbito de las variables

Funciones variables

Si al nombre de una variable le siguen paréntesis y contiene una cadena, el interprete tratará de encontrar una función con el nombre que tiene la variable almacenada

```
$variable = "retificaCorreo";
$variable();
```

Lo anterior buscará una función llamada rectificaCorreo()

Funciones para variables

Establecer el tipo de una variable: bool settype (mixed &$var , string $type)

Obtener el tipo de una variable: string gettype (mixed $var)

Determinar si una variable está vacía: bool empty (mixed $var)

Determinar si una variable está definida y no es NULL: bool isset (mixed $var [, mixed $...])

Destruir una variable especificada: void unset (mixed $var [, mixed $...])

Devolver una matriz con todas las variables definidas: array get_defined_vars (void)

Imprimir el contenido de manera legible: mixed print_r (mixed $expression [, bool $return = false])

Generar una representación apta para el almacenamiento de un valor: string serialize (mixed $value)

Crear un valor a partir de una representación almacenada: mixed unserialize (string $str)

Funciones básicas para arreglos

Crear un arreglo*: array array ([mixed $...])

Conjuntar variables y crear un arreglo con ellas: array compact (mixed $varname1 [, mixed $...])

Contar los elementos: int count (mixed $array_or_countable [, int $mode = COUNT_NORMAL])
ó sizeof

Crear variables a partir de un arreglo: int extract (array &$array [, int $flags = EXTR_OVERWRITE [, string $prefix = NULL]])

Verificar que exista un elemento: bool in_array (mixed $needle , array $haystack [, bool $strict = FALSE])

Verificar que exista una clave o índice:bool array_key_exists (mixed $key , array $array) ó key_exists

Asignar el contenido de un arreglo a una lista de variables: array list (mixed $var1 [, mixed $...])

Ordenar un arreglo de manera aleatoria: bool shuffle (array &$array)

Funciones generales para arreglos

Crear un nuevo arreglo, usando una matriz para las claves y otra para sus valores: array array_combine (array $keys , array $values)

Devolver los valores de un arreglo: array array_values (array $array)

Contar los diferentes valores de un arreglo: array array_count_values (array $array)

Rellenar un arreglo: array array_fill (int $start_index , int $num , mixed $value)

Intercambiar valores por claves: array array_flip (array $array)

Devolver claves según un criterio de búsqueda: array array_keys (array $array [, mixed $search_value [, bool $strict = false]])

Combinar arreglos:array array_merge (array $array1 [, array $...])

Rellenar un arreglo hasta cierta longitud: array array_pad (array $array , int $size , mixed $value)

Quitar el último elemento: mixed array_pop (array &$array)

Insertar elementos al final: int array_push (array &$array , mixed $value1 [, mixed $...])

Reducir un arreglo a un sólo valor: mixed array_reduce (array $array , callable $callback [, mixed $initial = NULL])

Invertir un arreglo: array array_reverse (array $array [, bool $preserve_keys = false])

Reemplazar valores: array array_splice (array &$input , int $offset [, int $length [, mixed $replacement = array()]])

Buscar un valor: mixed array_search (mixed $needle , array $haystack [, bool $strict = false])

Funciones para ordenar arreglos

Ordenar arreglo inverso manteniendo la relación clave=>índice: bool arsort (array &$array [, int $sort_flags = SORT_REGULAR])

Ordenar arreglo manteniendo la relación clave=>índice: bool asort (array &$array [, int $sort_flags = SORT_REGULAR])

Ordenar las claves de un arreglo inversamente: bool krsort (array &$array [, int $sort_flags = SORT_REGULAR])

Ordenar las claves de un arreglo: bool ksort (array &$array [, int $sort_flags = SORT_REGULAR])

Ordenar arreglo destruyendo claves inversamente: bool rsort (array &$array [, int $sort_flags = SORT_REGULAR])

Ordenar arreglo destruyendo claves: bool sort (array &$array [, int $sort_flags = SORT_REGULAR])

Ordenar arreglos de acuerdo a una función: bool uasort (array &$array , callable $value_compare_func)

Ordenar claves de un arreglo de acuerdo a una función: bool uksort (array &$array , callable $key_compare_func)

Ordenar valores de un arreglo de acuerdo a una función destruyendo claves: bool usort (array &$array , callable $value_compare_func)

Funciones para puntero de arreglos

Devolver el elemento apuntado: mixed current (array &$array) ó pos

Apuntar al último elemento: mixed end (array &$array)

Obtener la clave del elemento apuntado: mixed key (array &$array)

Apuntar al siguiente elemento: mixed next (array &$array)

Apuntar al elemento anterior: mixed prev (array &$array)

Apuntar al primer elemento: mixed reset (array &$array)

Funciones para manejo de cadenas

Obtener un carácter ASCII: string chr (int **$ascii**)

Convertir cadena a arreglo: array explode (string **$delimiter** , string **$string** [, int **$limit**])

Convertir arreglo a cadena:string implode (string **$glue** , array **$pieces**)

Dar formato a números: string number_format (float **$number** [, int **$decimals** = 0])

Obtener el valor ASCII: int ord (string **$string**)

Imprimir cadena con formato: int printf (string **$format** [, mixed **$args** [, mixed **$...**]])

Rellenar una cadena: string str_pad (string **$input** , int **$pad_length** [, string **$pad_string** = " " [, int **$pad_type** = STR_PAD_RIGHT]])

Buscar y reemplazar: mixed str_replace (mixed **$search** , mixed **$replace** , mixed **$subject** [, int **&$count**])

Longitud de la cadena: int strlen (string **$string**)

Buscar una subcadena en una cadena: mixed strpos (string **$haystack** , mixed **$needle** [, int **$offset** = 0])

Cadena a minúsculas o a mayúsculas: string strtolower (string **$string**) y string strtoupper (string **$string**)

Eliminar caracteres no deseados: tring trim (string **$str** [, string **$character_mask** = " \t\n\r\0\x0B"])

Convertir a mayúscula la primera letra de cada palabra: string ucwords (string **$str**)

Funciones básicas para matemáticas

Consultar en http://php.net/manual/es/ref.math.php

Tema 3

Formularios

Introducción a HTML

En general HTML por medio de etiquetas, determina la forma en que una página web es desplegada.

Consiste en un archivo de texto que se delimita por etiquetas (o tags).

El navegador lee esas etiquetas y crea objetos, esos objetos tienen una interpretación gráfica.

Por medio de los atributos de la etiqueta, se asignan los atributos del objeto, los que no se mencionan toman su valor por defecto.

Consideraciones

HTML		
El navegador hace todo lo posible para omitir o bien corregir errores en el código, así como interpretar lo que queríamos hacer aunque sea incorrecto (permisivos)	Es necesario seguir un orden, codificar de la mejor manera, tener un orden para que la página a diseñar sea optima	En la muchos navegadores, la tecla F12 permite mostrar un dialogo que permite detectar errores

Etiquetas sin cierre

	Permiten crear un solo objeto sin permitir tener más objetos
Son las etiquetas más simples, aunque en algunas deben asignarse atributos para que tengan funcionalidad	
	<etiqueta atributo="valor">

Etiquetas con cierre

Permiten crear un solo objeto permitiendo tener dentro la información que formatearán y más objetos que heredarán atributos del padre

Permiten la asignación de atributos para tener mayor funcionalidad, se debe definir el cierre para determinar el fin del objeto

```
<etiqueta atributo=“valor”>
        <varios objetos>
</etiqueta>
```

Etiqueta <!DOCTYPE html>

Su uso es importante, permite definir al navegador en qué modo se trabajará

Se define el trabajo con el modo estándar, asumiendo que se escribe en HTML versión 5

Es importante que no haya espacios, es una etiqueta sin cierre, de no respetarse se trabajará en modo de capricho

Etiqueta <html></html>

Crea el objeto HTML, propiamente el documento HTML

Soporta básicamente las etiquetas <head></head> y <body></body>

En las etiquetas anteriores se aglomeran objetos que definen las propiedades de la página y su contenido respectivamente

Etiqueta <head></head>

Define la cabecera del documento, es decir en ella se definen propiedades de la página

Es una etiqueta con cierre que agrupa etiquetas especiales para definir dichas características

Entre las etiquetas que puede agrupar son <title></title>, <meta>, <link>

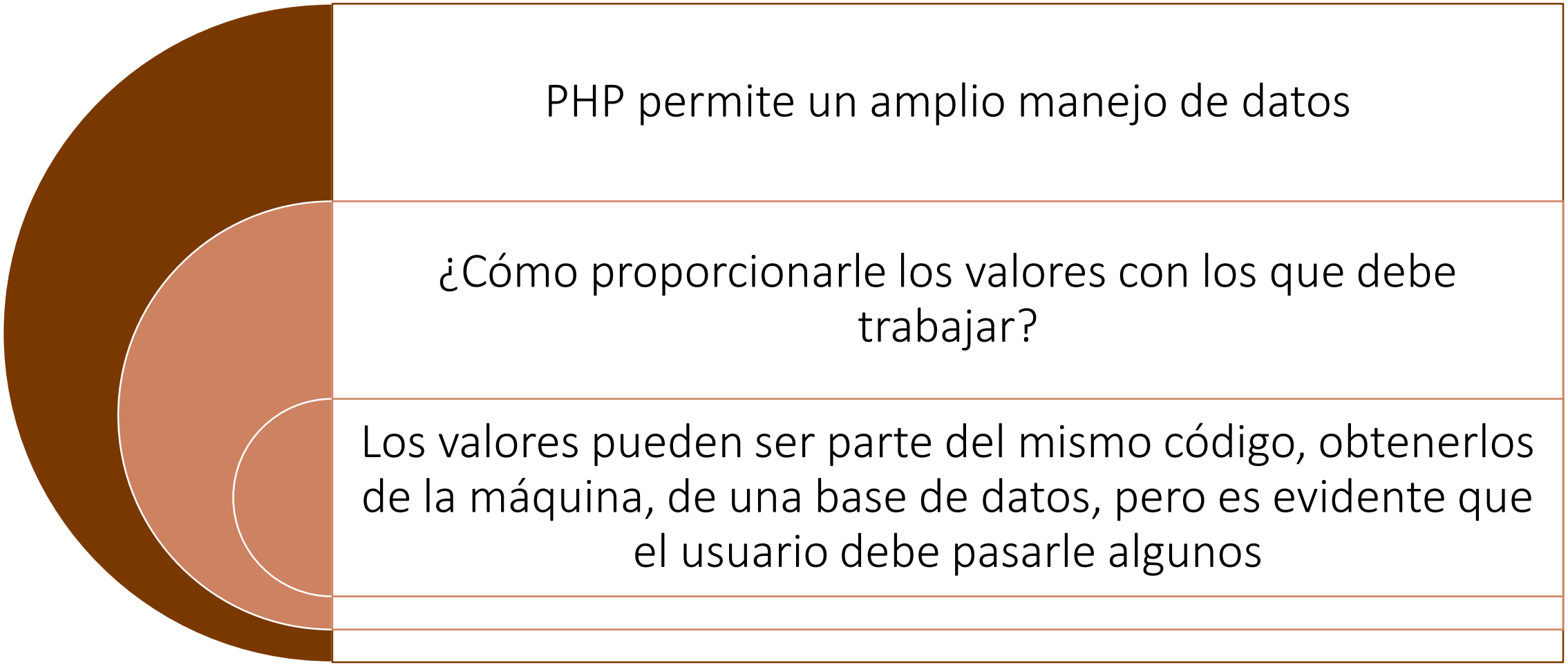
Etiqueta <body></body>

En ella se agrupan todos los objetos que propiamente generarán el contenido del documento

Existen etiquetas para definir cabezas, párrafos, ligas, separadores, saltos de línea, imágenes y tablas que son los más comunes

Las etiquetas para lo anterior son <hx>, , <a>, <hr>,
, <img>, <table></table>

Introducción a formularios



Formularios

Un formulario permite captar datos en el navegador del cliente por medio de objetos HTML, donde el usuario puede darles valor, un script e incluso algún script de PHP previo

El diseño del formulario depende totalmente de las necesidades y las habilidades del programador, siendo, no necesariamente una encuesta a llenar

<form>

Todo formulario está englobado entre las etiquetas <form> y </form>

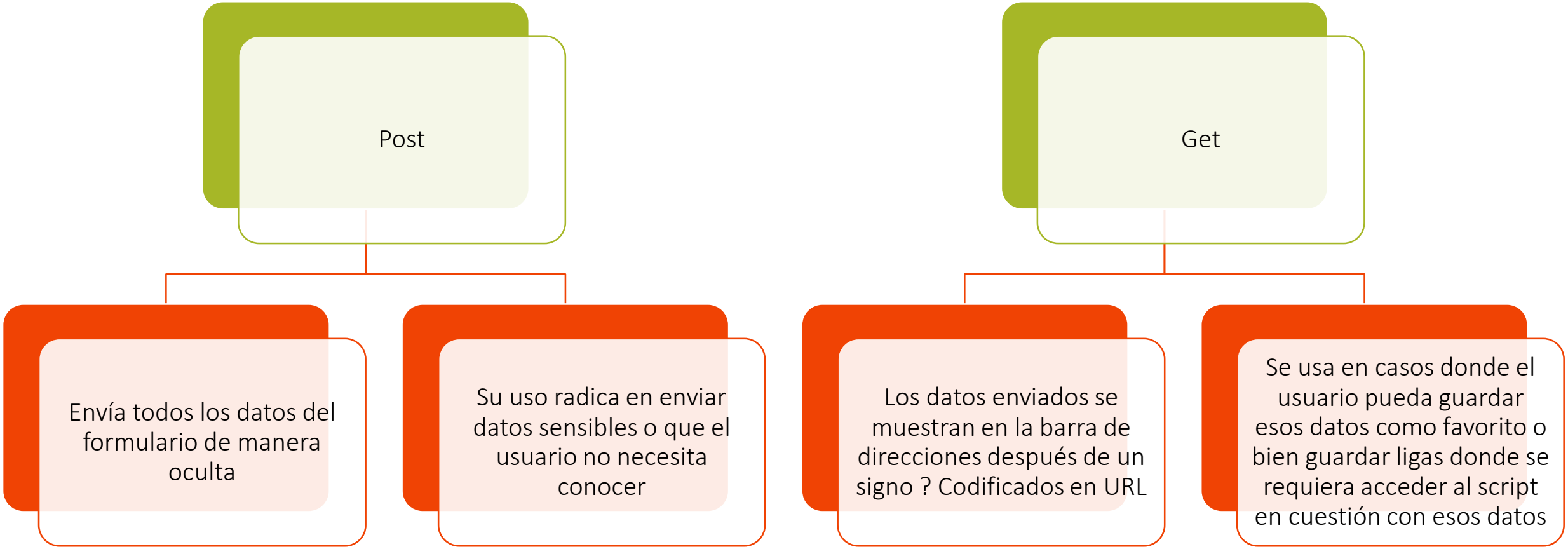
Las entradas de formulario se engloba en la etiqueta sin cierre <input> y en las etiquetas con cierre <select> y <textarea>

Los parámetros de las etiquetas anteriores se recomienda consultarlas en http://www.w3schools.com/tags

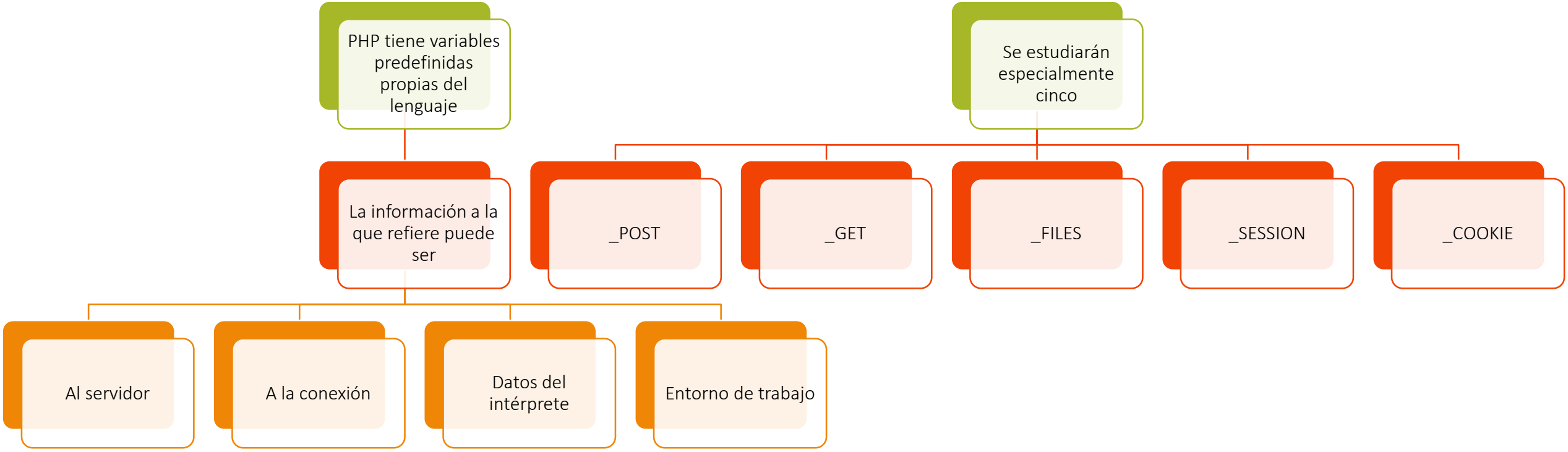
Los parámetros que tiene la etiqueta con cierre <form> son los siguientes

Atributo	Valores	Uso
action	(URL)	La dirección del script al que se enviarán los datos
autocomplete (HTML5)	on off	Precisa si los campos tendrán autocompletado
enctype	application/x-www-form-urlencoded multipart/form-data text/plain	En el método post especifica cómo serán los datos codificados para ser enviados al servidor
method	get post	Especifica cuál método se usará para enviar los datos al servidor
name	(texto)	El nombre del formulario
novalidate (HTML5)	novalidate	El formulario no será validado antes de ser enviado
target	_blank _self _parent _top (name)	Dónde se mostrará la respuesta al formulario

Métodos de envío



Variables del intérprete



_GET y _POST

_GET y _POST son arreglos muy similares a GLOBALS

_GET almacena todos las variables recogidas por un formulario enviado por el método GET

_POST almacena todos las variables recogidas por un formulario enviado por el método POST

Ambos arreglos permiten acceder a las variables después de enviar un formulario

Su alias formal son: HTTP_GET_VARS y HTTP_POST_VARS

Son arreglos asociativos, el índice es el nombre de la variable y el valor su contenido

¿Por qué se usa simplemente el nombre de las variables?

Adiós al uso de variables mágicas

Después de enviar un formulario automáticamente se crean variables con el nombre de cada campo

Esto se debe a una directiva activada: Register Globals

La directiva es una amenaza dado que, sumado a que PHP no soporta la declaración de variables, es posible incrustar variables que no son necesariamente parte de la entrada del script

Un script de PHP, no sabe con que formulario funcionar por lo que existe vulnerabilidades

Se recomienda apagar la directiva, y las variables creadas por un formulario sólo serán accesibles por el lugar correcto

Amenazas de formularios

Los formularios no están atados a qué script funcionar, como el cliente tiene el poder sobre ellos...

Se puede prestar a diversas amenazas, o incluso sólo pequeñas vulnerabilidades

Un usuario puede saltar alguna validación o función JS al desactivarlo, o por error si no lo tiene activado

Lo más robusto es hacer que PHP haga todas las validaciones necesarias usando variables de intérprete y funciones de seguridad

Control de errores

Es frecuente, que no se puedan controlar todos lo errores que PHP pueda lanzar o incluso uno nuestro (mejor dicho)

Como administradores de un sistema se debe ser cuidadosos, detallistas, exigentes a sí mismos, prever situaciones de error en la ejecución

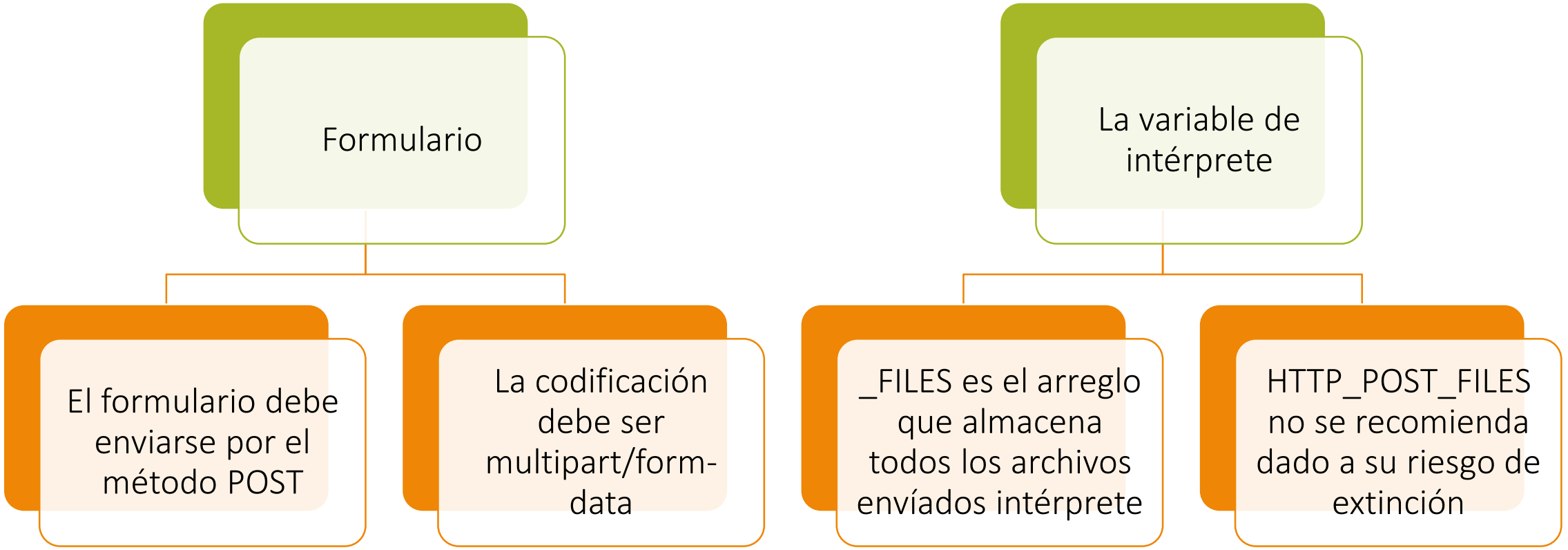
Como humanos, aprendemos por medio de errores

Los scripts de PHP pueden lanzar errores que no pensamos, o según nosotros contemplamos y atacamos, pero la complejidad inherente del lenguaje puede no permitirlo al 100%

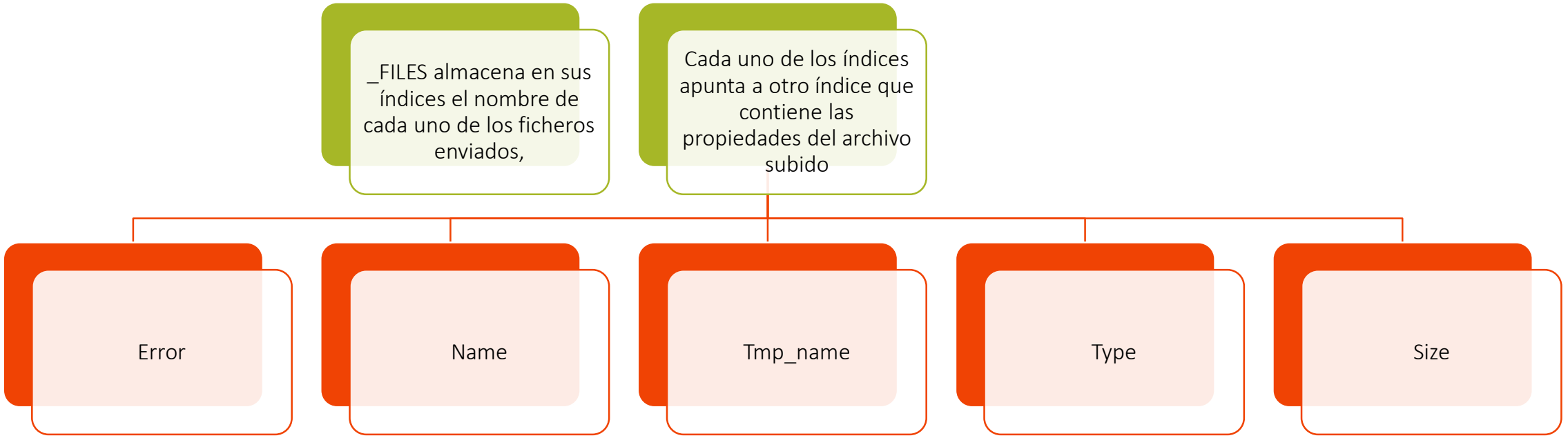
@, el operador de control de errores, se coloca al inicio de líneas criticas, este operador omitirá nuestro error en pantalla

Un error puede verse desagradable a la vista, más para un usuario, pero encima revela partes de o funcionalidades del programa

Envío de archivos



_FILES


_FILES almacena en sus índices el nombre de cada uno de los ficheros enviados,
Cada uno de los índices apunta a otro índice que contiene las propiedades del archivo subido
Error
Name
Tmp_name
Type
Size

Límite de tamaño

Establecer un límite de fichero permite dos cosas

Evitar que se sature la banda del servidor y colapse

Evitar que se llene rápidamente el almacenamiento del servidor

Simplemente debe condicionarse el valor de size en el arreglo _FILES

Dicho ese método también cabe mencionarse la restricción al tipo del archivo

Múltiples archivos

Se explotará una característica que tiene el intérprete de PHP, el uso de arreglo de entradas

Simplemente es necesario agregar corchetes a un mismo nombre de entrada

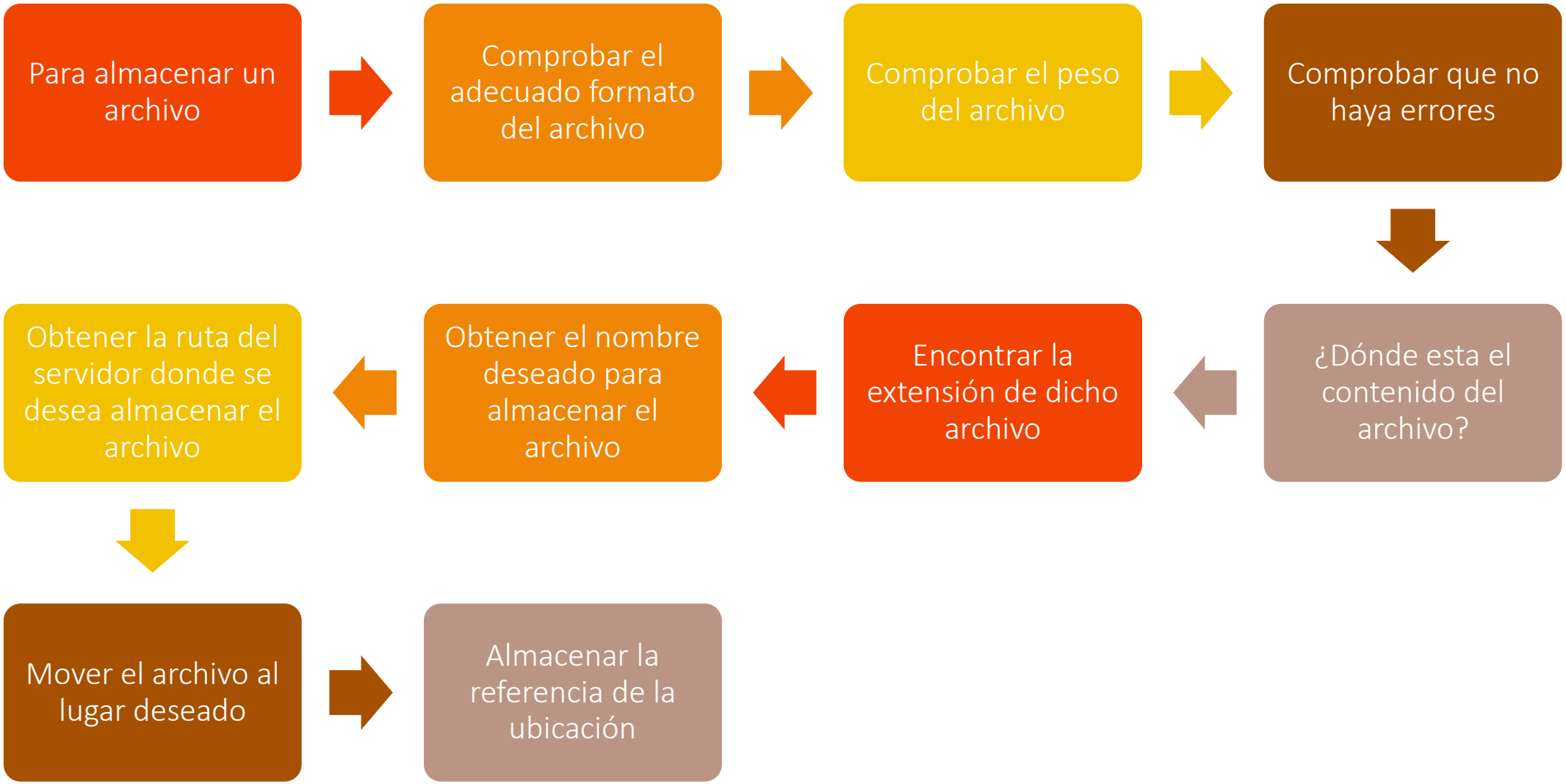
Si se especifica un nombre dentro de los corchetes se pasará de usar un arreglo indexado a uno asociativo

Contabilizar el número de elementos de _FILES

Puede que no todos tenga archivo

Sólo procesar los campos que realmente tengan algo

Procesamiento de archivos



Páginas auto procesadas

El hecho de tener el script que procesa un formulario en el mismo archivo que el formulario se conoce como páginas auto procesadas

Se tiene que hacer dos cosas, si la página es cargada por primera vez, mostrar el formulario, si ya ha sido autollamada mostrar el resultado

Simplemente se rectifica que existan las variables en el arreglo _POST

Formularios complejos implica scripts complejos, qué tan correcto es usar este método

Tema 4

Ficheros

Introducción

Los dos puntos de vista son, ver a los scripts de PHP como ficheros que, de hecho lo son, y ver cómo PHP puede gestionar ficheros

El primer aspecto abarca temas como la organización de un proyecto complejo así como la reutilización de código

El segundo aspecto abarca la gestión de ficheros para diferentes propósito como lo es almacenar ciertos tipos de datos que no conviene necesariamente

Varios scripts

Un script de PHP es un fichero

Usar varios ficheros se recomienda para

- Almacenar las definiciones de funciones o métodos* de distintas naturalezas, por ejemplo para la gestión de base de datos*
- Organizar código, se recomienda separar el código es distintos apartados, por ejemplo el que procesa las cadenas HTML, JS, CSS que se mostrarán al usuario (vista en la arquitectura MVC), y donde se almacenan las funciones lógicas que procesan la información (vista en MVC)

Es posible llamar a estos ficheros por medio de las funciones (en realidad constructores)…

Scripts muy externos

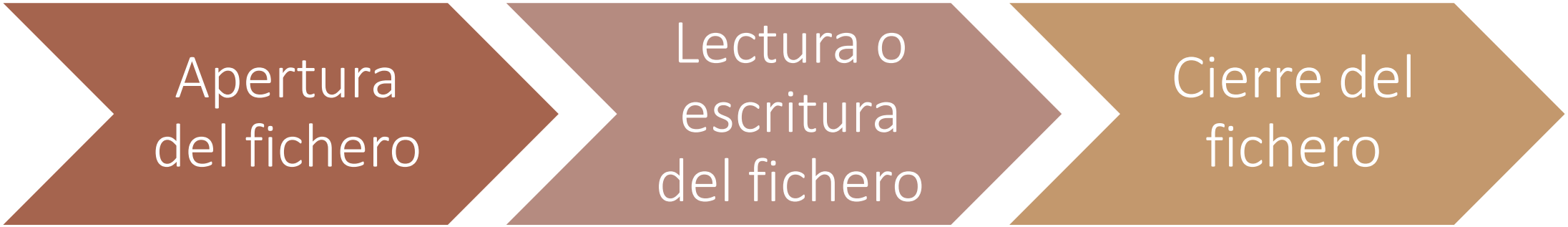
Un script puede estar alojado en la misma carpeta del script que hace el llamado

Un script puede encontrarse en otra carpeta, deberá escribirse su ruta relativa o bien la absoluta

Un script puede estar en otro servidor, debe colocarse su URL, sin embargo, la directiva allow_url_fopen debe estar activada en php.ini

Ficheros

Un fichero se usa para almacenar o recuperar información del siguiente modo



Esto es análogo al manejo de un fichero en papel

Apertura del fichero

```
resource fopen ( string $filename , string $mode
[, bool $use_include_path =
false [, resource$context ]] )
```

Modo de apertura	Descripción
a	Abre el fichero en sólo escritura para escribir al final del mismo
a+	Modo a más lectura
r	Abre el fichero para sólo lectura, si el fichero no existe se retorna una advertencia
r+	Modo r, permitiendo la escritura al inicio del archivo
w	Abre el fichero para su escritura reemplazando cualquier contenido existente
w+	Modo w más lectura
b	Abre un fichero de manera binaria*

Existen otros modos de apertura de fichero, lo más comunes son los anteriores, la función retorna un manejador o identificador que debe asignarse a una variable

Cierre de un fichero

Para cerrar el apuntador, manejador o identificador de un fichero abierto de usa la función bool fclose (resource `$handle`).

Bajo ninguna circunstancia debe dejarse un fichero abierto, al terminar de trabajar con un fichero debe cerrarse, de lo contrario lo que puede pasar con el fichero es totalmente impredecible, puede ir desde no pasar absolutamente nada y comportarse como debe ser o perder información

Lectura de ficheros

Como concepto, existen funciones que trabajan con un puntero de fichero, que indica qué debe de leer, este puntero se queda en donde haya sido dejado de usar por la función

Otro concepto es que todo fichero tiene una marca a su final llamado EOF (end of file) que indica el final del fichero y no provocar un desbordamiento de la memoria, las funciones de lectura al encontrar este marcador no hacen más

El modo b, o mejor dicho bandera b, permite usar ficheros en forma binaria, lo cual le es indiferente a Linux e Unix, en Windows es distinto un archivo de texto plano a otro (que es binario), por lo que se recomienda siempre usar la bandera

Funciones para lectura de ficheros

Leer pone en el buffer de salida (salida estándar): int fpassthru (resource $handle)

Abre y pone en el buffer de salida: int readfile (string $filename [, bool $use_include_path = false [, resource $context]])

Leer ciertos caracteres: string fread (resource $handle , int $length)

Rebobinar el puntero (no aplica en modo a y a+): bool rewind (resource $handle)

Obtener el tamaño de un fichero: int filesize (string $filename)

Obtener un solo carácter: string fgetc (resource $handle)

Obtener una línea (se asume 1024): string fgets (resource $handle [, int $length])

Obtener línea y quitar etiquetas HTML y de script: string fgetss (resource $handle [, int $length [, string $allowable_tags]])

Revisar si el puntero no esta en EOF: bool feof (resource $handle)

Obtener un arreglo en el que cada índice es una línea: array file (string $filename [, int $flags = 0 [, resource $context]])

Obtener el puntero: int ftell (resource $handle)

Mover el puntero: int fseek (resource $handle , int $offset [, int $whence = SEEK_SET])

Funciones para manejo de ficheros

Escritura de una cadena en un fichero especificando el máximo de tamaño: int fwrite (resource $handle , string $string [, int $length]) ó fputs

Eliminar un archivo del servidor (irreversible): bool unlink (string $filename [, resource $context])

Copiar un fichero: bool copy (string $source , string $dest [, resource $context])

Renombrar archivos o directorios (o moverlos):bool rename (string $oldname , string $newname [, resource $context])

Propiedades de los ficheros

Antes de llevar a cabo una maniobra con un fichero, debe verificarse que sea posible llevarla a cabo.

Verificar si un fichero existe: bool file_exists (string `$filename`)

Verificar que sea posible la lectura: bool is_readable (string `$filename`)

Verificar que sea posible la escritura: bool is_writable (string `$filename`)

Verificar si es posible la ejecución: bool is_executable (string `$filename`)

Verificar si es fichero: bool is_file (string `$filename`)

Verificar si es directorio: bool is_dir (string `$filename`)

Verificar si es enlace simbólico*: bool is_link (string `$filename`)

Permisos

Para trabajar con un archivo es necesario que se tenga permiso para hacerlo, en la siguiente tabla se muestran los 3 niveles de usuario y los 3 niveles de permisos, debe considerarse que el sistema de numeración es octal.

	Propietario	Grupo	Otros
Lectura	4	4	4
Escritura	2	2	2
Ejecución	1	1	1

Ejemplo: 0740
Describir los niveles de permiso que tiene cada usuario

Funciones para permisos

Cambiar los permisos de un fichero, recordar que mode debe ser octal: bool chmod (string `$filename` , int `$mode`)

Leer los permisos de un fichero, el valor devuelto no es octal: int fileperms (string `$filename`) Encontrar la forma de formatear el valor devuelto

Directorios

Todo fichero se organizan en estructuras compartimentadas

En Windows son las carpetas, en Unix-Linux directorios

Tienen diferencias en el sistema de archivos pero el concepto es similar

En PHP se usan directorios para diferentes propósitos, como organizar la información que obtiene el servidor así como el mismo código del sistema

Funciones para el manejo de directorios

Crear un directorio (el mode en Windows se ignora): bool mkdir (string `$pathname` [, int `$mode` = 0777 [, bool `$recursive` = false [, resource `$context`]]])

Cambiar el directorio: bool chdir (string `$directory`)

Borrar un directorio (debe estar vacío y tener permisos de escritura): bool rmdir (string `$dirname` [, resource `$context`])

Obtener el directorio actual: string getcwd (void)

Funciones para leer directorios

Abrir el directorio a trabajar: resource opendir (string **$path** [, resource **$context**])

Leer el contenido: string readdir ([resource **$dir_handle**])

Apuntar al primer elemento: void rewinddir ([resource **$dir_handle**])

Cerrar el directorio: void closedir ([resource **$dir_handle**])

Tema 5

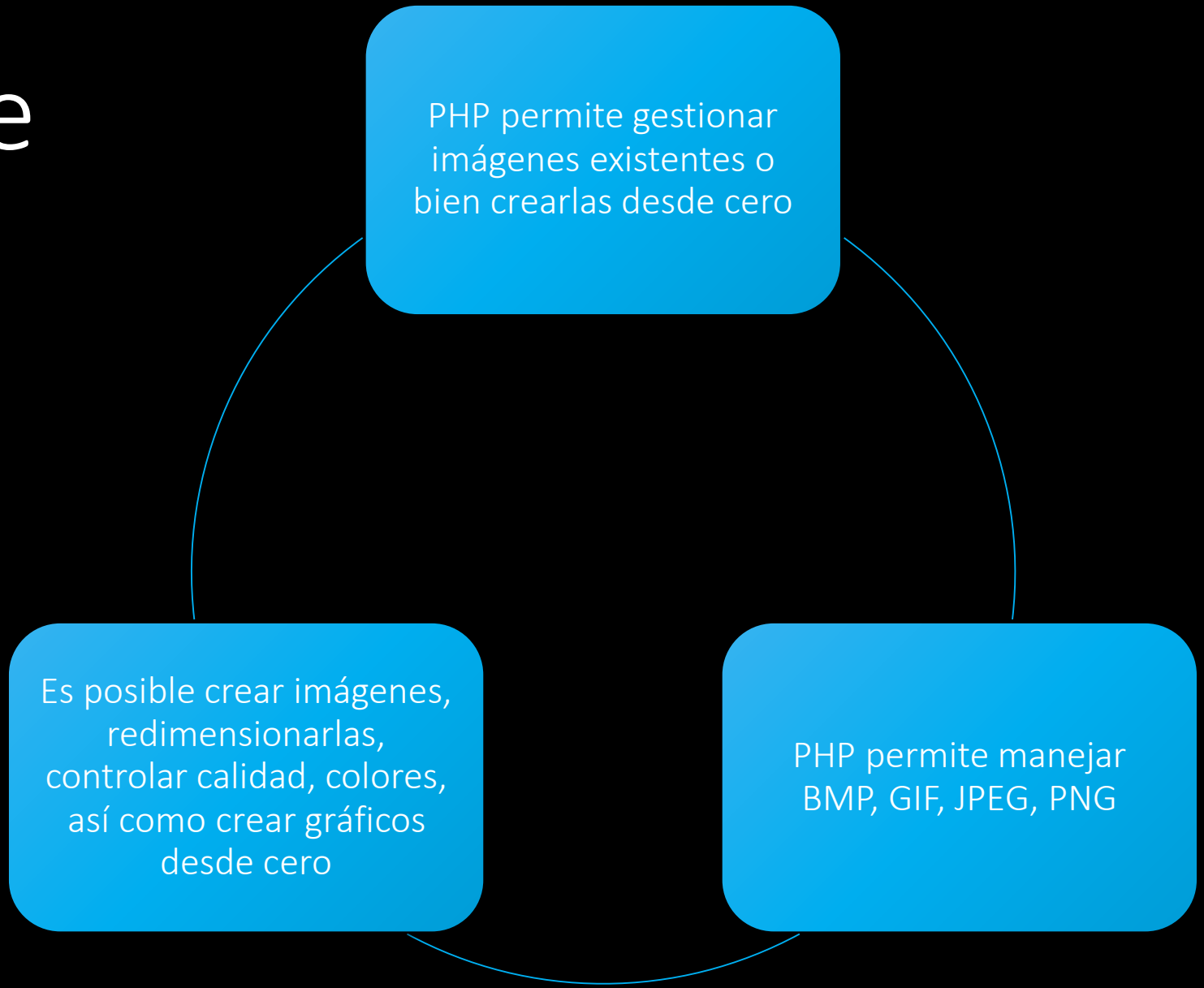
Aplicaciones

Imágenes

HTML permite insertar imágenes, pero PHP permite procesarlas antes de ser enviadas al navegador cliente

El manejo de imágenes requiere de una librería especifica llamada GD2, PHP debió haber sido compilado para tal

Alcance


PHP permite gestionar imágenes existentes o bien crearlas desde cero
PHP permite manejar BMP, GIF, JPEG, PNG
Es posible crear imágenes, redimensionarlas, controlar calidad, colores, así como crear gráficos desde cero

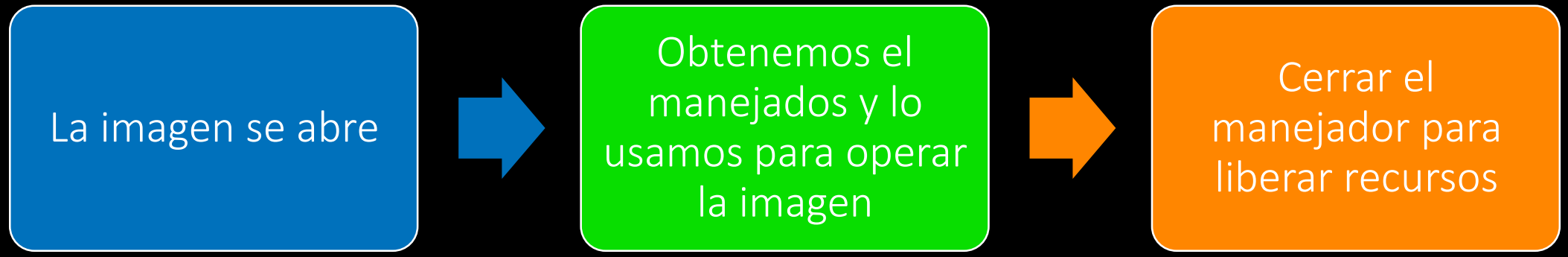
Iniciando

Encontrar las función que permite verificar que una función exista

Verificar la existencia de las funciones imagegif(), imagejpeg(), imagepng(), imagewbmp()

Header("Content-type: image/jpeg");

gif, png. wbmp son otras alternativas



Manejador de imágenes existentes

Existen las funciones

- imagecreatefromjpeg()
- imagecreatefromgif()
- imagecreatefrompng()
- imagecreatefromwbmp()
  - *Todas reciben la ruta del fichero y devuelve un manejador*
  - *Se recomienda usar @*

Para destruir el manejador

- imagedestroy()
  - *Recibe el manejador*

Mostrar la imagen

imagejpeg() y homólogos

- Reciben en el primer parámetro un manejador
- El segundo parámetro es opcional, es la ruta donde se guardará la imagen, si no se indica se envía a la salida estándar
- El tercer parámetro opcional, indica la calidad de la imagen de salida

En HTML, en lugar poner en src una imagen puede ponerse un script con esa salida

Altura, anchura y propiedades

imagesx()

Proporciona el ancho en pixeles, recibe manejador

imagesy()

Proporciona el alto en pixeles, recibe manejador

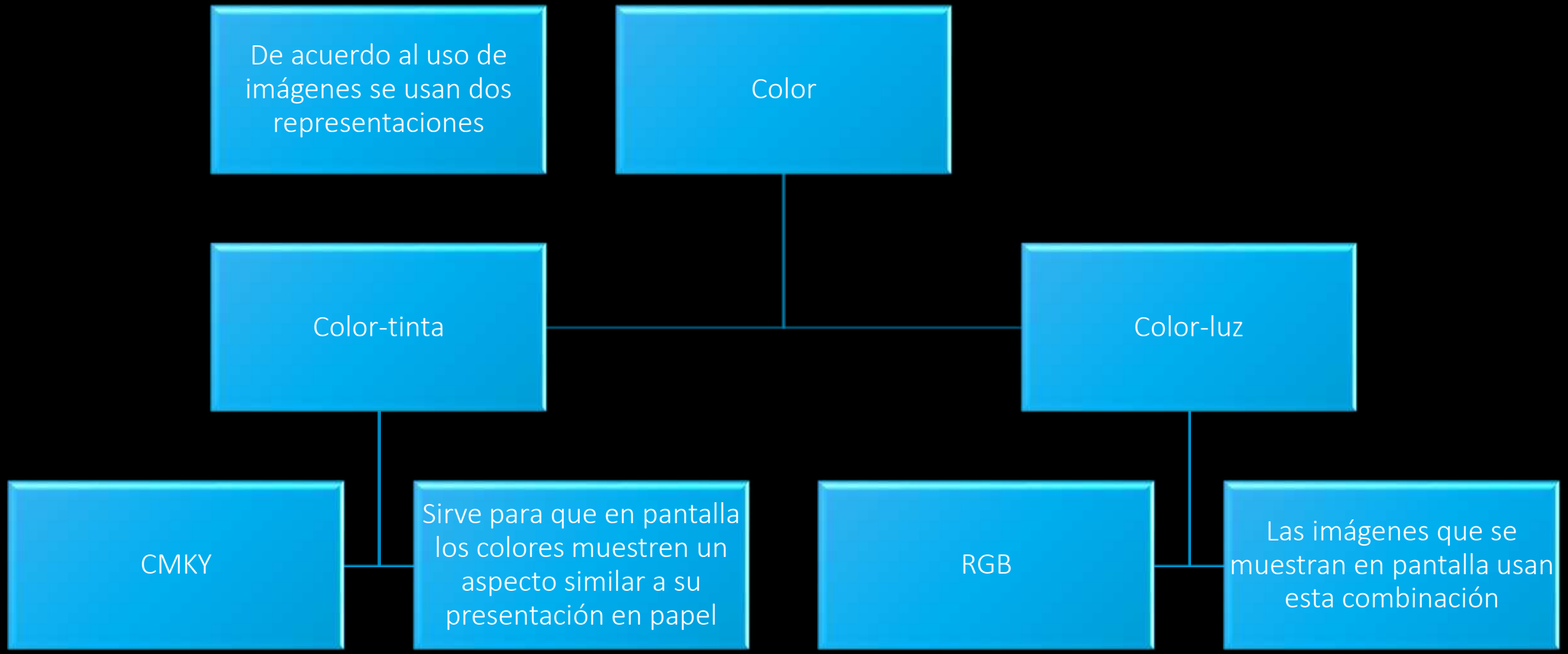
getimagesize()

Obtiene el tamaño de la imagen, recibe el nombre del archivo, y algo más

Recorre el arreglo de salida y determina qué esta devolviendo

Si se desea obtener la salida de esto, debe ser en un script de salida texto

Colores-luz y colores-tinta



RGB

Se forma por los tres colores luz básicos

- Rojo, verde y azul
- Cada color puede tomar valores de 0 a 255, hasta 16 millones de colores pueden generarse

Todas las imágenes tienen asociada una paleta de colores, son los colores que la imagen usa, esos colores están dispuestos por valores RGB

RGBA considera además un cuarto elemento que es la transparencia del color, se le conoce como facto alfa

Funciones de color

imagecolortotal()

Devuelve la cantidad de colores que forman la paleta de una imagen, su argumento es el manejador

imagecolorforindex()

Devuelve un color, recibe el manejador y algún índice que corresponda a la paleta de colores

Una imagen tiene una paleta de colores asociada, cada color de la paleta se identifica por medio de un índice con una composición RGB

imagecolorset()

Argumentos: manejador, índice a modificar, valor rojo, valor verde y valor azul

imagecolorallocate() / imagecolorallocatealpha()

Añade nuevos colores, recibe el manejador y los tres colores, devuelve el índice, o -1 si hay error

Imagecolordeallocate()

Retira un color añadido

Ejercicio: hacer un programa que muestre las propiedades de una imagen, obtenga la paleta de colores y establezca el valor opuesto con el fin de obtener la imagen en negativo

Creación de imágenes

Es posible el manejo de imágenes existentes, pero también lo es el crear una imagen

- imagecreatetruecolor() o imagecreate()
  - *Recibe la anchura y la altura de la imagen en pixeles, devuelve el manejador*
- imagecolorallocate()
  - *Crear la paleta de colores, el primer valor será el fondo*
- imagesetpixel()
  - *Establece el color de un pixel, recibe el manejador, la coordenada x e y, así como el índice de la paleta*
- imagecolorat()
  - *Devuelve el índice de paleta de un pixel, recibe manejador y coordenadas x e y*

Se debe guardar la imagen creada o mostrarla

Toda referencia es a partir de la coordenada 0, 0 localizada en la esquina superior izquierda

Copiar imágenes

Usar imagecopyresized() para realizar lo mismo pero modificando tamaño

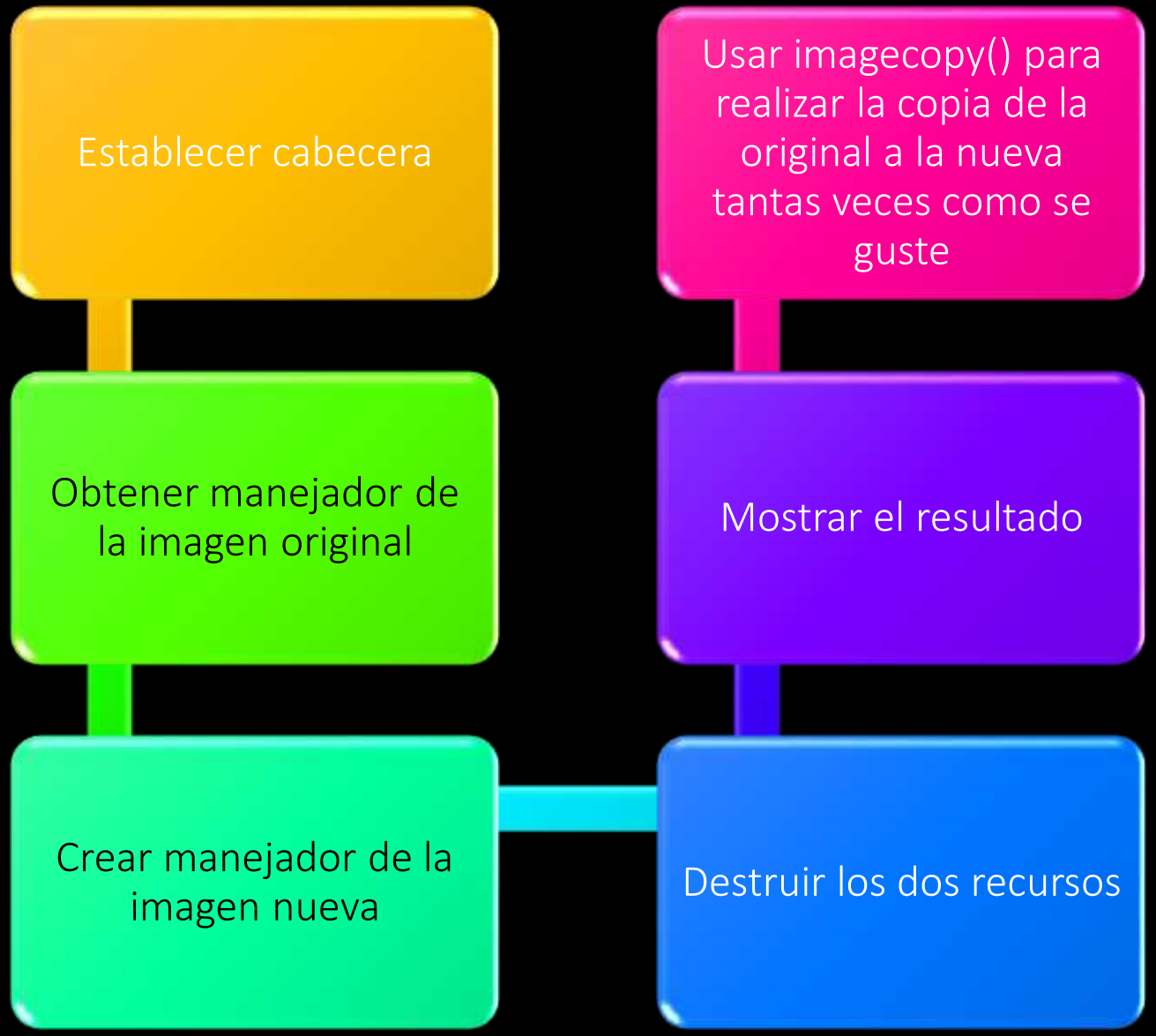


Figura predefinidas

Consulta las funciones en la documentación, las funciones para dibujar:

- Una línea recta: imageline()
- Una línea recta discontinua: imagedashedline()
- Un rectángulo: imagerectangle()
- Un arco, una elipse o una circunferencia: imagearc()
- Una curva cerrada: imageellipse()
- Un polígono: imagepolygon()

Rellenar figuras

Para rellenar figuras exite la función imagefill()

- Se asigna un punto en la imagen y la función comienza a rellenar hasta encontrar un borde delimitador, de lo contrario colorea toda la imagen

imagefilldedrectangle, imagefilledpolygon() o imagefilledellipse() crean figuras rellenas

Filtros de imagen

imagefilter() permite aplicar un filtro a la imagen, véase el prototipo en el manual, los filtros son los siguientes:

- IMG_FILTER_NEGATE: Es el negativo de la imagen
- IMG_FILTER_GRAYSCALE: Convierte la imagen en escala de grises
- IMG_FILTER_BRIGHTNESS: Cambia el brillo de la imagen. El argumento 1 es el nivel de brillo de -255 a 255.
- IMG_FILTER_CONTRAST: Cambia el contraste de la imagen. El novel es igual que en brillo
- IMG_FILTER_COLORIZE: Es un filtro de color para la imagen con 4 argumentos para cada canal
- IMG_FILTER_EDGEDETECT: Resalta los bordes de la imagen por medio de detección, un borde es la zona donde hay cambios de color
- IMG_FILTER_EMBOSS: Da un efecto de relieve
- IMG_FILTER_GAUSSIAN_BLUR: Pone borrosa la imagen usando el método Gaussiano, es un desenfoque Gaussiano
- IMG_FILTER_SELECTIVE_BLUR: Pone borrosa la imagen. La desenfoca
- IMG_FILTER_MEAN_REMOVAL: Utiliza eliminación media para lograr un efecto "superficial“, estilo boceto al eliminar el suavizado de los detalles
- IMG_FILTER_SMOOTH: Suaviza la imagen. El primer argumento es el nivel de suavidad
- IMG_FILTER_PIXELATE: Pixelea la imagen, el primer argumento es el tamaño del nuevo pixel y el segundo el modo

Texto en las imágenes

imagestring()

Permite insertar una cadena de texto, recibe el manejador, el tamaño que va de 1 a 5, las coordenadas de inicio c e y, el texto y el color de la paleta

imagestringup()

Igual que la anterior pero permite colocar el texto vertical u horizontalmente

imagettftext()

PHP permite incluir tipografías TTF (true type), recibe manejador, tamaño de la fuente en puntos, el ángulo en grados que rota contrario al reloj, coordenadas x e y de inicio, índice de color de la paleta, la ruta de la tipografía y la cadena de texto

PDF

Similar a un archivo de texto enriquecido

Permite mantener su formato en cualquier sistema

Crear un PDF normalmente es de aplicaciones costosas

PHP permite crear PDF

Necesitamos de una clase destinada a crear PDF

Herramientas de creación.

PHP no incluye el manejo de PDF de manera nativa

PDFlib es de pago

FPDF es una librería gratuita

www.fpdf.org

Se encontrará manuales y documentación

Inicio

Se requiere el fichero con la clase

require(“fpdf/fpdf.php”);

Se instancia el objeto

$pdfObj = new FPDF(); Puede recibir el formato de papel, unidad de medida y papel

Se crea una página

$pdfObj -> AddPage();

Se establece una fuente

$pdfObj -> SetFont(“Arial”, “I”, 16); Se recibe fuente, tipo y tamaño en puntos

Creamos una celda que es un contenedor, el texto se coloca en dichos

$pdfObj -> Cell(150, 20, “Un texto”); Recibe el ancho de la celda, el alto y el texto

Se obtiene la salida

$pdfObj -> Output();

Dar Formato

Es necesario que un objeto de la clase FPDF este instanciado.

Los métodos modificadores siempre se ejecutan antes.

Se crean celdas para estructurar el PDF

Cell(), MultiCell(), Text(), Image()

SetFont(), SetDrawColor(), SetLineWidth(), SetFillColor(), SetTextColor()

Consideraciones

Se analizará un script que permite mostrar un PDF más complejo

PDF permite generar facturas, informes de estadística, financieros, etc.

Existen librerías adicionales, por ejemplo para poner una imagen de fondo al archivo PDF creado

Correo Electrónico

PHP permite el envío de correos electrónicos.

PHP cuenta con la función mail()

Es necesario que exista un servidor de correo electrónico configurado correctamente.

Correo Electrónico Complejo

Uso de la clase phpMailer

Compuesta por dos scripts class.phpmailer.php y class.smtp.php

phpMailer es código libre.

El cuerpo del mensaje se trabaja con HTML

Tema 6

Cookies y sesiones

Introducción

http:// y https:// son protocolos sin estado.

Son formas de almacenar provisionalmente datos.

Sesiones ->servidor

Cookies -> cliente

Cookies.

- Archivo de texto plano que se almacena en el cliente.
- Para hacer su manejo en PHP usamos $_COOKIE[]
- setcookie(nombre, valor, fecha, dominio, ruta de acceso, pagina segura)

Sesiones.

Sistema para almacenar pares nombre-valoren el servidor.

Las variables de sesión están disponibles en $_SESSION[] o en $HTTP_SESSION_VARS[]

Flujo.


Iniciar sesión.
session_start()
Registrar sesión.
session_register(var1,var2,…)
Verificar sesión.
session_name()
session_id()
Cerrar sesión.
session_unregister(varSesion)
session_destroy()
session_unset()


php