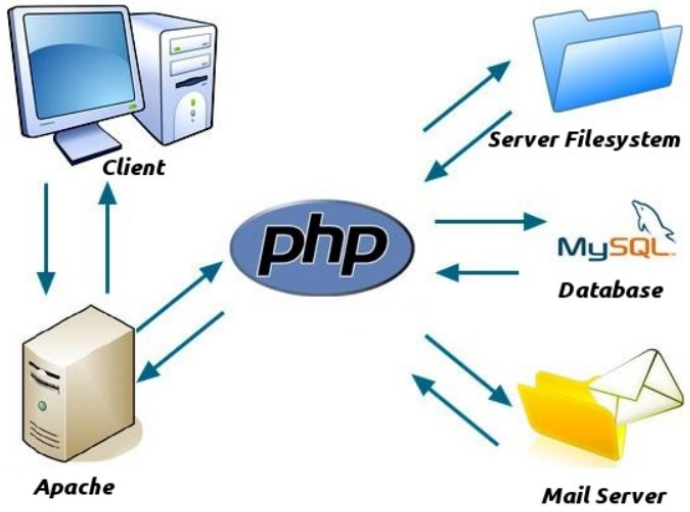
U.D.2.- Características da linguaxe PHP

Sumario

Introdución

PHP, orixinalmente o acrónimo de "*Personal Home Pages*" e posteriormente o acrónimo recursivo de "*PHP Hypertext Preprocesor*" é unha linguaxe de programación de servidor: é executado no servidor e o resultado (normalmente en formato HTML) é recibido polo navegador do cliente:



Foi creado no 1994 como un subconxunto de scripts en Perl creados por Rasmus Lerdof. Posteriormente foi modificado por outros programadores ata a versión actual, que é a 8.0 (neste 2021).

Para poder traballar con PHP temos que ter instalado un servidor de http/https co Apache. Pode ser Apache, Nginx, etc.

Na aula traballaremos en linux con Docker e contenedores, mentres que en Windows unha opción rápida pode ser empregar XAMPP.

Hai moitos sitios web nos que podes ter acceso a información de php, pero o máis usado polos programadores, e onde podemos obter toda a información que empregaremos no presente curso, pode ser o sitio oficial:

https://www.php.net

ou algo máis resumido:

https://manuais.iessanclemente.net/index.php/PHP

Inserción de código

Fíxate que o código PHP insírese dentro do código HTML, empregando as etiquetas de apertura e peche **<?php** e **?>**, respectivamente.

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>Primeiro exemplo</title>
</head>
<body>
      <h2> Primeiro exemplo en PHP </h2>
      <?php
            echo "Ola mundo<br>";
      ?>
</body>
</html>
```

Sentencias

- As sentencias teñen que rematar con ‘;’
- A última sentencia do bloque non precisa do ‘;’ pero é recomendable.
- Agruparemos varias sentencias coas chaves { …. }, por exemplo dentro dun bucle **for**

Variables

As variables comezan co símbolo de $, e non é preciso definilas antes de usalas, como nalgunhas linguaxes de programación.

Ollo coas minúsculas e maiúsculas, é distinto $apelido que $Apelido.

O nome da variable só pode comezar por unha letra ou por barra baixa: ‘_’

O resto dos caracteres pode ser calquera letra, números ou ‘_’

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>Variables</title>
</head>
<body>
      <h2> Variables </h2>
      <?php
            $a=5;
            $_b=7.35;
            $animal33=”Pantera Rosa”;

            echo $animal33, “<br>”, $a, “<br>”, $_b;
      ?>
</body>
</html>
```

Cada variable é visible no ámbito no que está definida, teremos variables locais dentro de cada función, e variables globais co seu ámbito fóra das funcións. Para que as variables globais se poidan empregar dentro da función empregaremos a palabra reservada **global.** Veremos exemplos disto polo miúdo cando vexamos as funcións.

Variables predefinidas

Existen algunhas variables predefinidas , gardando información do entorno de execución do intérprete e do propio PHP.

As máis empregadas, todos array asociativos, son:

$_GET[‘variable’]: variables recibidas por GET

$_POST[‘variable’]: variables recibidas por POST

$_SERVER[‘variable’]: todas as variables de alcance global

$_COOKIE[‘variable’]: variables pasadas a través de cookies

$_FILES[‘variable’]: ficheiros pasados a través de POST

As principais variables do servidor están nun array asociativo **$_SERVER:** *$_SERVER[‘SERVER_NAME’], _SERVER[‘SERVER_PORT’], $_SERVER[‘SERVER_SOFTWARE’], $_SERVER[‘SERVER_REMOTE_PORT], $_SERVER[‘REMOTE_ADDR’]*, etc.

Por exemplo, se temos un formulario que envía por GET, con 2 input de tipo texto, con name igual a "nome" e "apelido", un programa que captura os valores introducidos e os amosa por pantalla podería ser:

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>Paso por GET</title>
</head>
<body>
      <h2> Variables </h2>
      <?php
            $onome=$_GET['nome']; //GARDAMOS O nome NOUTRA VARIABLE

            echo $onome <br>, $_GET['apelido']; /* E O apelido DIRECTAMENTE */
      ?>
</body>
</html>
```

Operadores

Os operadores aritméticos serán os mesmos que en Javascript: + , - , * , / , %

Ademais do operador de asignación = están permitidos os operadores de asignación += , -= , *= , /=.

Os operadores de comparación son ==, != , < , > , <=, >=. Ademais, existe o operador === que compara o valor e tipo. E !== . Así:

```
12 === 12.0  // é falso porque non coincide o tipo de dato
10 === 10  //será verdadeiro
```

Estruturas de control

Alternativa simple: if, else

De forma similar a Javascript:

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>Alternativa if-else</title>
</head>
<body>
      <h2> Variables </h2>
      <?php
            $num=$_GET[‘numero1’];
            if ($num<5)
                  echo “O número é menor que 5”;
            else
                  echo “Número maior ou igual que 5”;

      ?>
</body>
</html>
```

Alternativa múltiple: if, elseif

A diferencia con outras linguaxes é que o elseif non ten espazo polo medio (~~else if~~):

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>Alternativa if-else</title>
</head>
<body>
      <h2> Variables </h2>
      <?php
            $num=$_GET[‘numero1’];
            if ($num<5)
                  echo “O número é menor que 5”;
            elseif ($num==5)
                  echo “O número é igual a 5”;
            else
                  echo “Número maior que 5”;

      ?>
</body>
</html>
```

Alternativa con switch

Cando a comparación é sempre coa mesma variable e con ==, podemos empregar a sentencia switch:

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>Alternativa switch</title>
</head>
<body>
      <h2> Variables </h2>
      <?php
            $num=$_GET[‘numero1’];
            switch ($num)
            {
            case 1: echo “Número igual a 1”;
                    break;
            case 3: echo “Número igual a 3”;
```

```
                break;
        case 5: echo “Número igual a 5”;
                break;
        default: echo “Número distinto a 1,3 e 5”;
        }
    ?>
</body>
</html>
```

Bucle 1. while

O bucle while repítese mentres a condición sexa certa, Sintaxe similar a Javascript ou C:

```
<!DOCTYPE html>
<html>
<head>
        <meta charset="UTF-8" />
        <title>While. Conta atrás </title>
</head>
<body>
    <h2> Variables </h2>
    <?php
        $num=$_GET[‘numero1’];
        while ($num > 0)
        {
            echo $num, “<br>”;
            $num--;  //dentro do bucle DEBE cambiar a variable da condición
        }
        echo “Chegamos ao 0!”;
    ?>
</body>
</html>
```

Bucle 2. do_while

Agora a condición avalíase ao final, polo que o bucle sempre se executa unha vez:

```
    <?php
        $num=$_GET[‘numero1’];
        do
        {
            echo $num, “<br>”;
            $num--;      //dentro do bucle cambia a variable da condición
        } while ($num > 0);     //Obrigatorio o ‘;’
        echo “Chegamos ao 0!”;  ?>
```

Bucle 3. for

```
<?php
            for ($i=0; $i < 20; $i++)
                  echo $i,<br>;
?>
```

Bucle4. foreach

Veremos o **foreach** na sección dos **arrays**.