

Introducción a la programación

Clase 2

Jordi Collell

jordi@tempointeractiu.com

@galigan

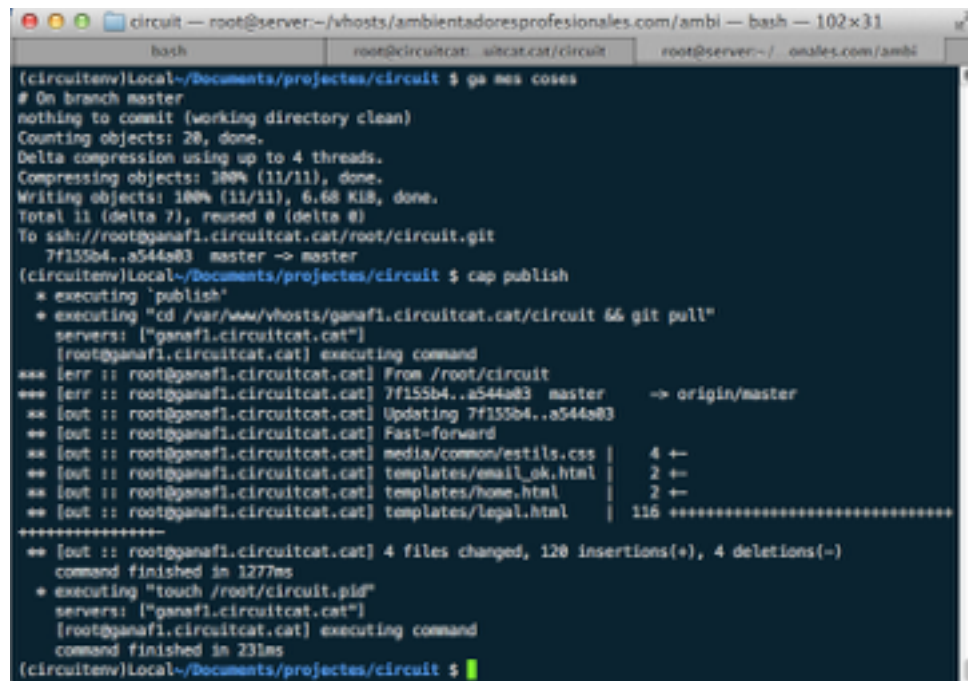
Repaso sesión anterior

- > Que es un programa
- > Una lista de ordenes
- > Compilado vs Interpretado
- > Instrucción
- > Tipo de dato
 - Número / Cadena / Boleano
- > Operador
 - + - / * % = == ===

Algunas instrucciones:
echo "hola"

- > Cliente / Servidor
- > Un servidor de web
- > Un servidor de ftp
- > Un servidor de bases de datos
- > Un servidor proxy
- > Un servidor virtual
- > Un servidor cloud

telnet y ssh



```
root@server:~/vhosts/ambientadoresprofesionales.com/ambi — bash — 102x31
bash
root@circuitcat: ~/circuitcat/circuit
root@server:~/vhosts/ambientadoresprofesionales.com/ambi
(circuitenv)Local~/Documents/proyectos/circuit $ ga mes cosas
# On branch master
nothing to commit (working directory clean)
Counting objects: 20, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 6.68 KiB, done.
Total 11 (delta 7), reused 0 (delta 0)
To ssh://root@ganafi.circuitcat.cat:/root/circuit.git
7f155b4..a544a83 master -> master
(circuitenv)Local~/Documents/proyectos/circuit $ cap publish
* executing 'publish'
* executing "cd /var/www/vhosts/ganafi.circuitcat.cat/circuit && git pull"
servers: ["ganafi.circuitcat.cat"]
[root@ganafi.circuitcat.cat] executing command
*** [err :: root@ganafi.circuitcat.cat] From /root/circuit
*** [err :: root@ganafi.circuitcat.cat] 7f155b4..a544a83 master -> origin/master
** [out :: root@ganafi.circuitcat.cat] Updating 7f155b4..a544a83
** [out :: root@ganafi.circuitcat.cat] Fast-forward
** [out :: root@ganafi.circuitcat.cat] media/common/estils.css | 4 +-
** [out :: root@ganafi.circuitcat.cat] templates/email_ok.html | 2 +-
** [out :: root@ganafi.circuitcat.cat] templates/home.html | 2 +-
** [out :: root@ganafi.circuitcat.cat] templates/legal.html | 116 +++++
+++++
** [out :: root@ganafi.circuitcat.cat] 4 files changed, 120 insertions(+), 4 deletions(-)
command finished in 1277ms
* executing "touch /root/circuit.pid"
servers: ["ganafi.circuitcat.cat"]
[root@ganafi.circuitcat.cat] executing command
command finished in 231ms
(circuitenv)Local~/Documents/proyectos/circuit $
```

Para conectar a un servidor ssh,
ejecutaremos:
ssh nombreusuario@servidor

> El terminal del ordenador es otra interfaz a nuestro sistema.

> En un ordenador *Unix*, se pueden realizar multitud de tareas desde una shell de texto

> El servicio de sesión remota está formado por un servidor y un cliente.

> Mediante este puedo acceder a un servidor remoto

> Puedo manipular archivos remotos igual, que si los editara localmente y los subiera al servidor

ejercicio 1

Abrimos la consola de Chrome:

```
date_default_timezone_set('Europe/Madrid');  
  
echo date("d-M-Y");
```

Para localizar información sobre métodos y objetos usar google. En este caso, si buscamos:

php date

nos devolverá la documentación relativa al mismo.

ejercicio 2

Abrimos Dreamweaver, creamos un site y añadimos un nuevo archivo php.

```
<html>
<head>
</head>
<body>
<?php
echo "Hola";
?>
</body>
```

Cuando cargamos el archivo (ejecutamos), en el navegador, el programa se ejecuta, y nos ofrece el resultado

Este va a ser nuestro primer programa

Para mantener de una forma organizada los distintos ejercicios, cada nuevo ejercicio, lo grabaremos en el site, con el nombre de archivo:

ejercicio_2.php

Flujo de Trabajo

1. Editaremos un Archivo

```
<html>
<head>
<script>
alert('Hola Mundo')
</script>
</head>
<body>

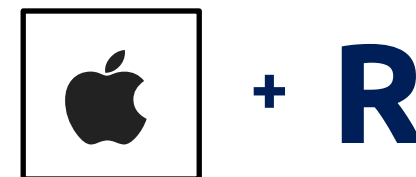
</body>
```

2. Guardaremos los cambios



Alt + Tab

3. Recargamos la página



Debemos guardar el trabajo, para que el navegador pueda cargar nuestra nueva versión del script.

Cambiamos el script y comprobamos que funciona correctamente

ejercicio 3

ejercicio_3.html

```
<html>
<head>
</head>
<body>
<?php

?>
</body>
```

Escribiremos un programa que nos devuelva el resultado de una operación matemática, en la forma:

La suma $10+17$ es = 27

Variables

```
$variable
```

```
$a = 1;  
$b = 2;  
$a+$b;
```

```
$a = "Hola"  
strtoupper($a);  
$a[0];
```

> Una variable es un espacio donde podemos almacenar información.

> En una variable podemos almacenar los tipos de datos que conocemos: *texto*, *numeros*, *booleanos*. También podemos almacenar objetos y funciones.

> Podemos definir variables con el operador **var**. En javascript no es obligatorio.

> En una variable, también podemos almacenar objetos.

> Podemos realizar operaciones con variables.

Variables

```
$uno = 5*3;  
$dos = "hola";  
echo gettype($uno);  
echo gettype($dos);  
echo $uno == $dos;  
$uno != $dos;  
$tres = "15";  
$uno === $tres;  
$uno == $tres;
```

> Con la instrucción `gettype`, podemos saber de que tipo es el contenido de una variable

> Podemos realizar comparaciones, con los operadores `==`, `===`, `!=` o `!==`

> La diferencia entre los operadores `==` y `===`, para el primero el tipo de datos no es importante. Para el segundo el valor debe de ser igual y del mismo tipo.

Comentarios

```
// esto es una  
variable  
$uno = 5  
# eso es un comm  
/*  
Esto es un  
comentario  
multilínea  
*/
```

> Dentro del código de un programa, podemos introducir instrucciones que no hacen nada. Anotaciones del programador. Estas anotaciones se llaman **comentarios**.

> En javascript podemos realizar comentarios de una línea usando la instrucción `//`.

> También podemos usar comentarios multilínea usando el bloque `/* */`

> Muchas veces podemos usar comentarios para desactivar líneas del programa.

Más operadores

```
$uno = 5;  
$uno++;  
echo $uno;  
$uno == 6;  
$uno--;  
$uno == 5;  
$uno += 5;  
$uno == 10;  
$uno -= 5;  
$uno == 5;  
$uno *= 5;  
$uno == 25;  
$uno /= 5;  
$uno == 5;
```

> El operador **++** incrementa en uno el valor de una variable numérica.

> El operador **--** decrementa en uno el valor de una variable

> El operador **+=** es equivalente a la operación:

```
$uno = $uno + 5;
```

> El operador **-=** resta.

Variables GET

```
$_GET['var1'];  
$_GET['var2'];
```

```
$_GET['v1'];  
$_GET['v2'];  
$_GET['v3'];
```

<http://localhost/script.php?var1=12>

localhost/s.php?v1=1&v2=2&v3=3

ejercicio 4

Suponer que tenemos una tienda, estamos en rebajas, y necesitamos un programa que nos calcule un descuento a los clientes el (20%)

El programa recibe el precio del producto, y nos devuelve el precio final que tiene que pagar.

Cálculo del porcentaje

Condicionales

```
$a =5;
if($a<1) {
    alert('1')
} else if ($a<2) {
    alert('2')
} else if ($a<10) {
    alert('10')
} else {
    alert('fin')
}
```

> Un condicional es una instrucción que nos permite modificar el flujo lógico de un programa en función de condiciones lógicas.

> En las condiciones

```
if($a<1) {}
```

podemos usar expresiones lógicas del tipo:

- < mas pequeño que.
- > mas grande que.
- <= mas pequeño o igual a.
- >= mas grande o igual a
- == igual a
- != distinto a
- === igual en valor y tipo de dato
- !== distinto en valor y tipo de dato

Repeticiones for

```
for($i=0; $i<50; $i++) {  
    echo $i;  
}
```

- > El bloque de control, for, sirve para repetir una operación **n** veces.
- > El fragmento de código, generará 50 mensajes en la consola de javascript.
- > Existen mas bloques de control que repetiremos

Bloques de Control

- > Tanto el **if**, como el **for**, són bloques de control que nos permiten alterar el flujo de control del programa.
- > Existen multitud más de bloques de control y normalmente los identificaremos porque son conjuntos de expresiones que van marcadas entre **{ }**
- > Otros bloques de control pueden ser: **while, switch, try....**

Partiendo del programa que genera descuentos a clientes, vamos a generar una nueva versión, que nos permita aplicar un descuento u otro, en función del importe del pago, así:

si el importe **es menor a 50€**

aplicaremos un descuento del 10%

si el importe es **menor o igual a 100€**

aplicaremos un descuento del 20%

importes **superiores**

descuento del 30%

ejercicio 6

Generaremos un programa que al ejecutarlo y en función de la hora del día, nos realizará un saludo.

Para recuperar la hora del día,
en una variable:

Saludos:

antes de las 6am

> buenas noches

antes de las 13

> buenos días

antes de las 19

> buenas tardes

despues de las 19

>buenas noches

ejercicio 7

Imaginemos que debemos programar la entrada organizada a un avión. El programa debe de recibir el número de fila y devolver, puerta delantera o trasera.

El avión tiene 30 filas. Hasta la fila 15 deben de entrar por puerta delantera. Hasta fila 30 por puerta trasera