# Conceptos básicos de PHP II

Ricardo Pérez López

IES Doñana, curso 2019/2020



- 1. Flujo de control
- 2. Funciones predefinidas destacadas
- 3. Arrays
- 4. Funciones definidas por el usuario
- 5. Comentarios y documentación del código
- 6. Ejercicios

## 1. Flujo de control

- 1.1 Estructuras de control
- 1.2 Inclusión de scripts

1.1. Estructuras de control

#### 1.1. Sintaxis alternativa

ricpelo's note: El do { ... } while (...); **no** tiene sintaxis alternativa.

1.2. Inclusión de scripts



#### 1.2. include, require

ricpelo's note: - El nombre del archivo debe aparecer con su extensión. No vale hacer require 'pepe';.

- Cuando un archivo es incluido, el intérprete abandona el modo PHP e ingresa al modo HTML al comienzo del archivo objetivo y se reanuda de nuevo al final.
- Si el archivo incluido tiene un return ...;, el include o el require que lo incluya devolverá el valor devuelto por el return.
- Si no se usa una ruta, se busca primero en el include\_path antes que en el directorio del propio script que hace el include. Por eso es mejor usar require
  - './pepe.php' que require 'pepe.php'.
  - Puede ser útil usar la constante \_\_DIR\_\_.

1.2. include\_once, require\_once



# 2. Funciones predefinidas destacadas

- 2.1 isset()
- 2.2 empty()
- 2.3 var\_dump()

**2.1.** isset()



#### 2.1. isset()

ricpelo's note: Cuidado si la variable contiene null. ricpelo's note: No da error ni advertencia si la variable no existe.

**2.2.** empty()



### 2.2. empty()

ricpelo's note: Para evitar el problema de empty("0") === true:

```
function is_blank($value) {
    return empty($value) && !is_numeric($value);
}
```

ricpelo's note: No da error ni advertencia si la variable no existe.

2.3. var\_dump()

## 3. Arrays

- 3.1 Operadores para arrays
- 3.2 [Funciones de manejo de arrays](http://php.net/manual/es/book.array.php)
- 3.3 foreach
- 3.4 Conversión a array
- 3.5 Ejemplo: \$argv en CLI

### 3. Arrays

ricpelo's note: Las claves pueden ser enteros o cadenas.



3.1. Operadores para arrays

#### 3.1. Operadores para arrays

ricpelo's note: **Comparaciones**: Un array con menos elementos es menor. De otra forma, compara valor por valor.

### 3.1. Acceso, modificación y agregación

3.2. [Funciones de manejo de arrays](http://php.net/manual/es/book.array.php)



### 3.2. Ordenación de arrays

3.2. print\_r()

3.2. '+' vs. array\_merge()

3.2. isset() vs. array\_key\_exists()



3.3. foreach



3.4. Conversión a array



3.5. Ejemplo: \$argv en CLI

## 4. Funciones definidas por el usuario

- 4.1 Argumentos
- 4.2 Ámbito de variables
- 4.3 Declaraciones de tipos

## 4.1. Argumentos



4.1. Paso de argumentos por valor y por referencia



#### 4.1. Argumentos por defecto

```
ricpelo's note: php function prueba($opciones = []) {
    extract($opciones); // ... }
```

4.2. Ámbito de variables



## 4.2. Ámbito simple al archivo

#### 4.2. Variables locales



#### 4.2. Uso de global

ricpelo's note: Usar global x; cuando x no existe hace que x empiece a existir y valga x



### 4.2. Variables superglobales



4.3. Declaraciones de tipos

### 4.3. Declaraciones de tipos

ricpelo's note: **NO** se hacen conversiones implícitas a array, ni en argumentos ni en devolución.



4.3. Declaraciones de tipo de argumento



4.3. Declaraciones de tipo de devolución



### 4.3. Tipos nullable (?) y void

ricpelo's note: Una función de tipo void realmente devuelve null.



#### 4.3. Tipificación estricta

ricpelo's note: El declare(strict\_types=1); se pone en el archivo que hace la llamada, no en el que define la función.



# 5. Comentarios y documentación del código

### 5. Comentarios y documentación del código

"



## 6. Ejercicios

- 6.1 ¡Hola, mundo!
- 6.2 Hamming
- 6.3 Isograma



6.1. ¡Hola, mundo!



6.2. Hamming



6.3. Isograma