Ocelote Documentation

Versión 0.9.1

Yeshua Rodas

30 de enero de 2019

Índice general

Ocelote es un paquete PHP de utiliterías, azúcar sintáctico y envolventes OO para funciones de uso común.

```
// Verificar si una variable está vacía
if(Checker::isEmpty($variable)) {
        // Hacer algo...
}

// Convertir una cadena json en un arreglo
$data = Converter::toArray($someJson);

// Explotar una cadena
$strAsArray = StringHelper::explode($someString, ",");
```

¡Y mucho más!

CAPÍTULO 1

Ocelote

Ocelote es un paquete de utilerías y azúcar sintáctica de propósito general.

El lenguaje PHP se concibió para potenciar algunos *scripts* personales y terminó convirtiéndose en el lenguaje más popular para potenciar sitios y aplicaciones Web. PHP es un lenguaje potente, flexible y dinámico, con una historia de momentos accidentados y con una cantidad ingente de información desfasada que pulula por Internet sobre el lenguaje, información que malencamina a nuevos programadore(a)s, propagando malas prácticas y código de menor calidad.

No existe una forma *canónica* para programar PHP propiamente dicho, pero la comunidad, afortunadamente, ha desarrollado recomendaciones, herramientas y mejor documentación que permiten la construcción de mejor código con PHP. Aún así, siempre quedan escollos que pulir.

Desde la versión 5.0 de PHP existe un soporte rico del paradigma de programación orientada a objetos, no obstante muchas funcionalidades permanecen en implementaciones netamente imperativas, que si bien no causan *ningún* daño, enturbian la elegancia que pretende proveer el paradigma. Por otra parte existen funcionalidades comunes, *utilerías*, que trascianden los proyectos, por ejemplo utilerías para la manipilación de cadedas, arreglos y estructuras de datos.

1.1 Arquitectura

Ocelote provee un conjunto de clases, *traits* e interfaces que resultarán útiles en cualquier proyecto independientemente de si se trata de un gran sistema o de un pequeño conjunto de *scripts* minimalistas.

1.1.1 Clases

1.1.1.1 ArrayHelper

El erreglo (`array`) de PHP es la *gran* estructura de datos del lenguaje. En realidad se trata de una suerte de *mapa odenado* que relaciona claves con valores. Es flexible, potente y pragmática; no sin sus deficiencias y compleja. PHP provee un conjunto amplio de funciones para la manipulación de `array`, y con ello algunas inconsistencias como que una gran parte de dichas funciones comiencen con «*array_*» y otras no (`count()`, `key()`, etc.). La clase estática `ArrayHelper` provee un conjunto de métodos que sirven de envolvente o adaptadores OO sobre muchas de las

funciones que provee PHP, así como también provee de funcionalidades extra para la manipulación de arreglos que no se encuentra en las funciones por defecto.

Ejpmplos:

```
// Verificar si un arreglo contiene un valor determinado con PHP plano
echo in_array($valor, $array);

// Verificar si un arreglo contiene un valor determinado con Ocelote
echo ArrayHelper::hasValue($array, $value);

// validar una clave y obtener un valor con PHP plano
$val = isset($array['key']) ? array['key'] : 'defecto';

// validar una clave y obtener un valor con Ocelote
$val = ArrayHelper::getValue($array, 'key', 'defecto');
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/ArrayHelper.php

1.1.1.2 Bag

`Bag` es una clase que provee un envolvente OO al arreglo de PHP.

Nota: No obstante este envolvente, recomienzo dar un vistazo y emplear «Estructuras de datos eficientes para PHP», una extensión que provee a PHP de ocho construcciones: *Vector, Deque, Map, Set, Stack, Queue, PriorityQueue, Pair*.

```
$bag = new Bag($array);

echo $bag->get('key');
echo $bag->has('key');
echo $bag->set('key', 'valor');
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/Bag.php

1.1.1.3 Checker

Esta clase estática provee métodos de verificación, incluyendo envolventes OO para los métodos `id_*` de PHP.

```
// Verificar si una variable está vacía con PHP plano
echo empty($variable);

// Verificar si una variable está vacía con Ocelote
echo Checker::isEmpty($variable);

// Verificar si una variable es un entero con PHP plano
echo is_int($var);

// Verificar si una variable es un entero con Ocelote
echo Checker::isInteger($var);
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/Checker.php

1.1.1.4 Converter

Provee métodos estáticos para convertir datos y tipos.

```
// Convertir un JSON en array o Bag
$unArray = Converter::toArray($unJson);
$unBag = Converter::toBag($unJson);

// O un array en un JSON
$unJson = Converter::toJson($unArray);
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/Converter.php

1.1.1.5 Inflector

Provee métodos para *flexionar* cadenas, por ejemplo tomar una cadena en minúscula cuyas palabras se *separen* por guiones bajos y convertirla en *CamelCase*.

```
$cadena = "una_cadena";

echo Inflector::camelize($cadena); // UnaCadena
echo Inflector::humanize($cadena); // una cadena
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/Inflector.php

1.1.1.6 Mime

Provee métodos para obtener el *Mimetype* de un amplio abanico de extensiones (`getTypeByExt()`) o bien, obtener la extesión a partir del *MimeType* (`getExtByType()`).

```
echo Mime::getTypeByExt('json'); // application/json
    echo Mime::getExtByType('application/json'); // json
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/Mime.php

1.1.1.7 StringHelper

Semejante a `ArrayHelper`, provee un conjunto de métodos que envuelven algunas de las funciones que PHP provee por defecto para la manipulación de cadenas, así como varias funcionalidades extra.

```
// Desacoplar una cadena por un delimitador con PHP plano
$array = explode($cadena, $delimitador);

// Desacoplar una cadena por un delimitador con Ocelote
$array = StringHelper::explode($cadena, $delimitador);
```

- Código: https://gitlab.com/xibalba/ocelote/blob/master/src/StringHelper.php