



Linters





¡Hola!

Soy Jon Torrado

Trabajo en el departamento IT de @LIN3S

Soy un obsesionado del código que funcione... y
que esté bien hecho

Puedes seguirme en @jontorrado





Dónde vivo

A dark blue silhouette of a world map. A small red speech bubble with the word 'Bilbao' is positioned over the northern coast of Spain.

Bilbao





1

Introducción

Qué son los *linters*



lint was the name originally given to a particular program that flagged some suspicious and non-portable constructs (likely to be bugs) in C language source code. The term is now applied generically to tools that flag suspicious usage in software written in any computer language.



Detectar todo aquello que tiene mala pinta, bien por ser un bug, typo, mala práctica, no *clean code*...



¿Lenguajes?

¡En cualquiera! Incluso en
sistemas de plantillas como
Twig





2

Necesidades

Todo buen software soluciona un problema



Principalmente...

Gigante

Todo código se vuelve gigante en algún momento, aunque ahora mismo no lo sea.

Errores

Algo muy simple se puede volver en algo bloqueante... y durante muchos minutos, ¡incluso horas!

Compatibilidad

Todo el mundo desarrolla a toda velocidad, y JS no soporta todo



¿Dónde está el fallo?

```
var arr = [  
  1,  
  2,  
  3,  
];
```





¿Dónde está el fallo?

```
var arr = [  
  1,  
  2,  
  3, ← >= IE 9  
];
```





3

Clean code

Vamos a jugar a un juego...



HTML

``

`Elemento que creo`

`En una lista desordenada`

`En HTML transicional`

``






HTML

Elemento que creo

En una lista desordenada

En HTML transicional





CSS

```
.elemento {  
  display: inline;  
  float: left;  
  height: 100px;  
  margin: 10px auto;  
  width: 100%;  
}
```





CSS

```
.elemento {  
  display: inline;  
  float: left;  
  height: 100px;  
  margin: 10px auto;  
  width: 100%;  
}
```





SCSS

```
.elemento {  
  background-image: url("../images/bg.png");  
  border: none;  
}
```

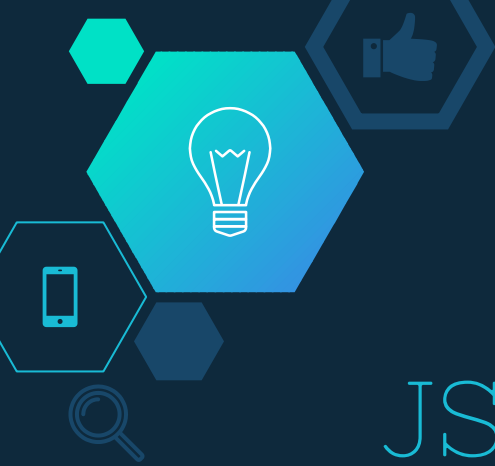




SCSS

```
.elemento {  
  _background-image: url("../images/bg.png");  
  _border: none;  
}
```

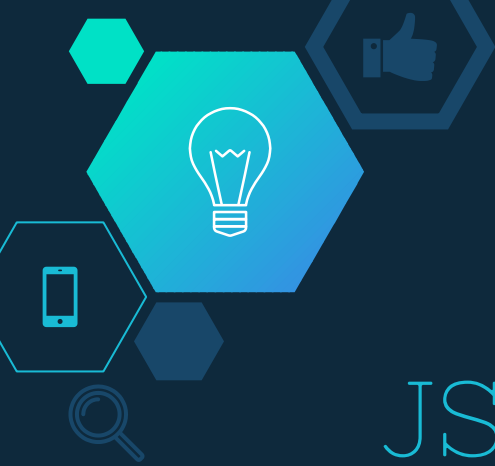




JS

```
if(elemento > 'algo'){  
  console.log('esta todo pagado')  
}
```





JS

```
if_(elemento > 'algo')_  
  console.log('esta todo pagado');  
}
```





PHP

PSR-2

◇ <http://www.php-fig.org/psr/psr-2/es/>





4

Softwares

Es hora de elegir el nuestro



HTML

- ◇ HTML hint
- ◇ html5 lint
- ◇ bootlint (bootstrap)
- ◇ htmltidy





CSS

- ◇ CSSLint
- ◇ CSSComb

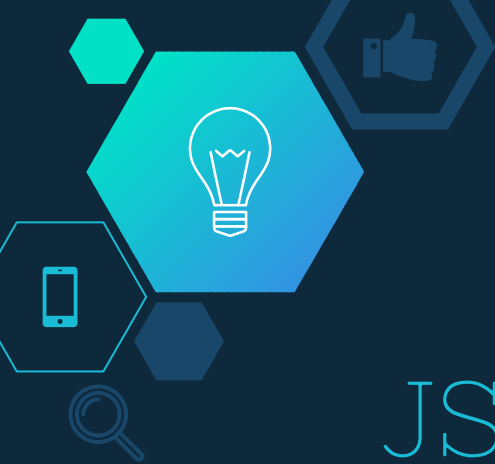




Sass

- ◇ SCSS Lint (gem = scss_lint)
- ◇ Sass Lint





JS

- ◇ JSCS
- ◇ JSHint
- ◇ JSLINT
- ◇ ESLINT





PHP

- ◇ PHPLint
- ◇ PHP CodeSniffer
- ◇ PHP CodeBeautyFixer → ¡automático!
- ◇ PHP Fixer (Fabier Potencier)





Otros

rubocop, shellcheck, pep8, oclint, golang/lint, coffeelint,
hint (haskell), elvis (erlang), ...

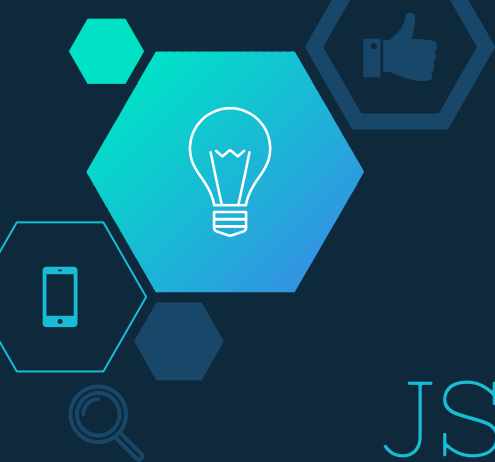




4.1

Softwares JS

Entrando un poco más en detalle



JSLint

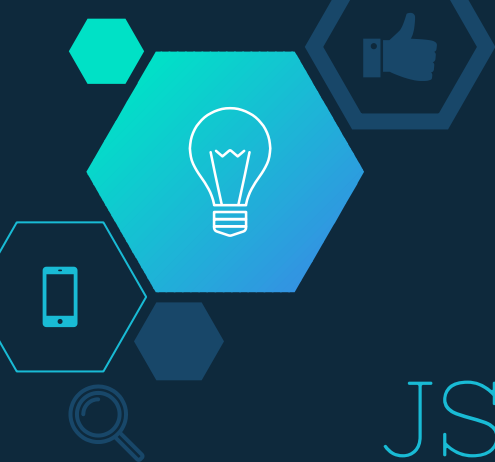
Pros

- ◆ Viene pre-configurado

Cons

- ◆ Viene pre-configurado
- ◆ Configuración extra limitada
- ◆ No puedes hacer tus reglas
- ◆ Poca documentación
- ◆ A veces difícil entender el resultado





JSHint

Pros

- ◆ Casi todo configurable
- ◆ Soporta archivo de configuración
- ◆ Gratis: jQuery, QUnit, NodeJS, Mocha, etc.
- ◆ Soporte ES6 básico





JSHint

Cons


- ◇ A veces es difícil entender el resultado
- ◇ Dos configuraciones: *enforce* y *relax* (estricto o menos estricto... o no...)
- ◇ No puedes hacer tus propias reglas





JSCS

Pros

- ◆ Soporta reporters personalizados (integración con otras herramientas)
 - ◆ Existen presets y pre-configuraciones
 - ◆ Se pueden incluir el nombre de las reglas en los reportes
 - ◆ Extensible con plugins propios
- 



JSCS

Cons

- ◇ Solo detecta violaciones de guía de estilos (no bugs como variables sin usar, globales accidentales, etc.)
- ◇ El más lento de los 4... ¿esto es un problema?





ESLint

Pros

- ◇ Flexible: todo activable y desactivable
- ◇ Muy extensible y con muchos plugins
- ◇ Fácil de entender el resultado
- ◇ Muchas reglas únicas (no las tienen otros)
- ◇ El que mejor soporta ES6; único JSX
- ◇ Soporta reporters personalizados

Cons

- ◇ Requiere configuración y es algo lento





5

Conclusiones

Resumen de lo visto hasta ahora



Linters

- ◇ Fácil de usar
- ◇ Reducen el *debugging time*
- ◇ Mejora la calidad de código
- ◇ Fuerza un una guía de estilos
- ◇ Hace el código más fácil de entender

Y tú...
¿qué opinas?





6

Linting++

Dando un paso más allá



Integración continua

Con un servidor de CI realizamos "linting" cuando se realice un cambio en el sistema de control de versiones.

¿Qué permite esto?

Ejemplos: Scrutinizr, GitLab CI, Jenkins, CodeShip, ...





Vergüenza pública

Es un gran motivador

Sistemas de medallas

◇ <https://github.com/LIN3S>

Ejemplo “pintxos”





Gulp... o cualquier otro

Es recomendable automatizarlo cuando se puede...
No siempre se puede. P.E. PHP CS Fixer

¡Lánzalo a menudo!





Extra

UnCSS

Es una herramienta que elimina el CSS no usado de las hojas de estilo. Funciona a través de múltiples archivos y soporta incluso estilos inyectados por JS

<https://github.com/giakki/uncss>





7

LIN3S CS

Llevando el linting al trabajo diario



Git hooks

Te avisan cuando algo ha pasado / va a pasar

- ◇ pre-commit
- ◇ prepare-commit-msg
- ◇ commit-msg
- ◇ post-commit
- ◇ post-checkout
- ◇ pre-rebase



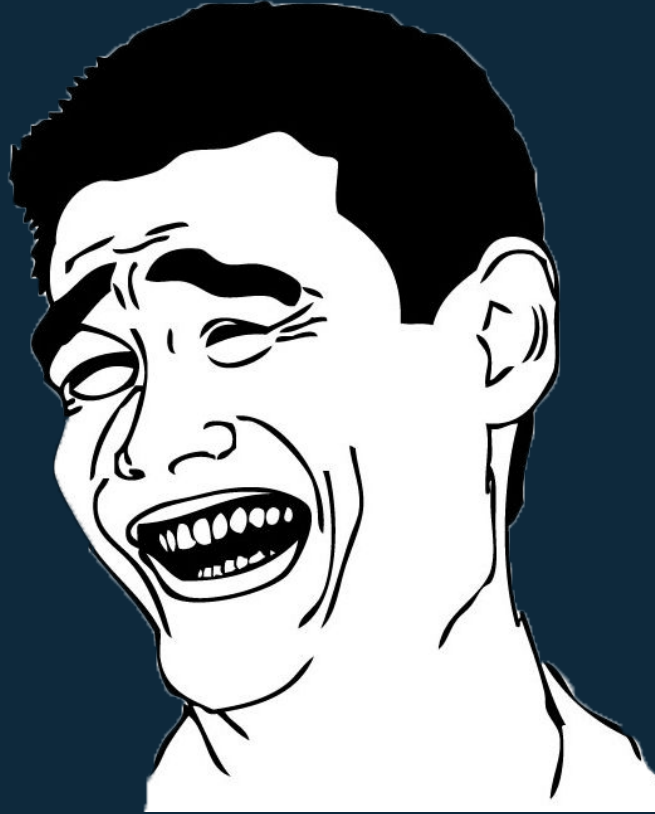


LIN3S precommit

Antes de permitirte el commit...

- ◇ Cambios en composer.json
- ◇ PHP-CS-Fixer
- ◇ PHP-Formatter
- ◇ PHPMD
- ◇ SCSS Lint
- ◇ ESLint
- ◇ Twig syntax







8

Demo

Algo sencillo en Sass



¡Gracias!

¿Preguntas?

Podéis encontrarme en:

◆ @jontorrado





Créditos

Agradecimiento especial a la gente que ha realizado y han puesto esta plantilla como recurso gratuito:

- ◇ Presentation template by [SlidesCarnival](#)
- ◇ Photographs by [Unsplash](#)





Linters

