Moving PHP forward through collaboration and standards.

PHP Standards Recommendations
PSR

Julien vinber : **@julienVinber**

Rendez-Vous PHP · Retours d'expérience - Kaliop 16 juin 2017

1.
Un peu d'histoire

Il était une fois

-

1994

Rasmus Lerdorf crés une bibliothèque logicielle pour conserver une trace des visiteurs qui venaient consulter son CV

1995 - sortie du Personal Home Page Tools/Form Interpreter

2016

-

82,6% des serveurs

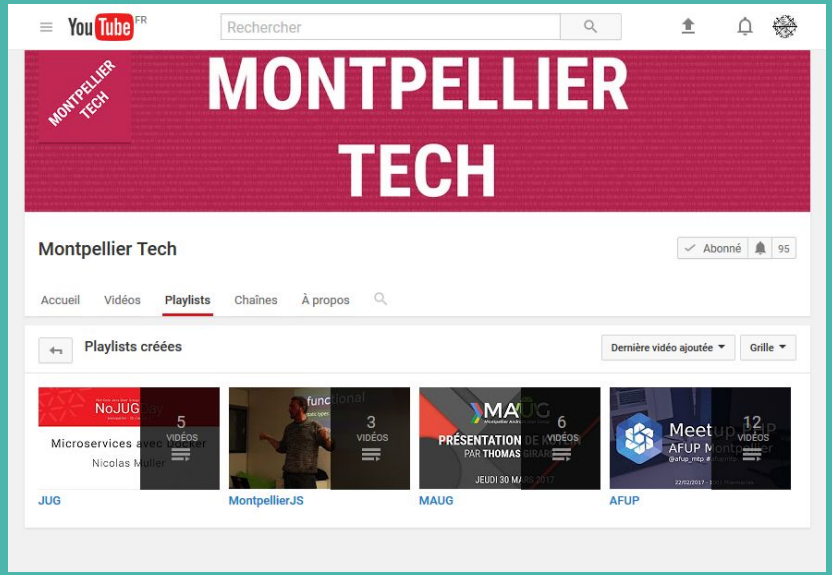
https://w3techs.com/

Pour en arriver là :

PHP s’est “professionnalisé”

Pour plus d'information

Revoir la
vidéo de Pascal MARTIN



https://www.youtube.com/channel/UC1ZadBAsgOgD0eo2R3JTgNA
https://blog.pascal-martin.fr/public/slides-meetup-php7-marseille-aix-montpellier-fevrier-2017/#/

2.
Et l'écosystème ?

LE PHP-FIG

PHP-FIG

PHP Framework Interop Group

Mais c'est qui?

https://www.sitepoint.com/the-past-present-and-future-of-the-php-fig/

http://www.php-fig.org/members/

PHP Framework Interop Group

But

Les PSR

PHP Standards Recommendations

Autoloading

Styles de codage

Interfaces

PHP-FIG

Mais encore?

POO

3.
Les PSR

Autoloading

~~PSR 0~~ - PSR 4 : Autoloader

But :

- Ne plus se battre avec des include.
- Éviter les conflits de nom
- Ne plus penser en termes de fichier, mais de classe
- Faciliter l'utilisation de composant tiers

http://www.php-fig.org/psr/psr-4/

~~PSR 0~~ - PSR 4 : Autoloader

Principe :

- Utilisation des Namespace
  - \<NamespaceName>(\<SubNamespaceNames>)*\<ClassName>
- La pseudo arborescence. du Namespace = l'arborescence sur la machine
- On peut indiquer un chemin de base pour un Namespace

http://www.php-fig.org/psr/psr-4/

Fully Qualified Class Name	Namespace Prefix	Base Directory	Resulting File Path
\Acme\Log\Writer\File_Writer	Acme\Log\Writer	./acme-log-writer/lib/	./acme-log-writer/lib/File_Writer.php
\Aura\Web\Response\Status	Aura\Web	/path/to/aura-web/src/	/path/to/aura-web/src/Response/Status.php
\Symfony\Core\Request	Symfony\Core	./vendor/Symfony/Core/	./vendor/Symfony/Core/Request.php
\Zend\Acl	Zend	/usr/includes/Zend/	/usr/includes/Zend/Acl.php

```php
<?php

namespace Afup;

$loader = require __DIR__ . '/../../vendor/autoload.php';

use Symfony\Component\BrowserKit\Client  AS BrowserKitClient;
use Symfony\Component\HttpKernel\Client;

class Meetup160617
{
    public function doIt($client)
    {
        if ($client instanceof BrowserKitClient) {
            return 0;
        } elseif ($client instanceof Client) {
            return 1;
        } elseif ($client instanceof \Monolog\Handler\MockRavenClient) {
            return 21;
        } else {
            return -1;
        }
    }
}
```

http://www.php-fig.org/psr/psr-4/

3.
Les PSR

Styles de codage

PSR 1 - PSR 2 : Styles de codage

Il n’y en a pas de meilleur, il faut juste respecter celle utilisée dans le groupe.

EN : http://www.php-fig.org/psr/psr-1/ - http://www.php-fig.org/psr/psr-2/
FR : https://gist.github.com/npotier/d5a13245ad9cd2e92fa9dec19baf0e9a - https://gist.github.com/npotier/593b645025173ef8bbb5c59d3fd455fa

PSR 1 - PSR 2 : Styles de codage

Oui, mais le groupe c’est le monde.

PSR 1 - PSR 2 : Styles de codage

- Encodage UTF-8 sans BOM
- Les noms des classes DOIVENT être déclarés en StudlyCaps.
- Les Méthodes DOIVENT être déclarées en camelCase().
- Le Code écrit pour PHP 5.3 et suivant DOIT utiliser les namespaces.

- le code DOIT utiliser une indentation de 4 espaces
- Les mots clés PHP DOIVENT être en minuscule.
- L'accolade d'ouverture doit être à la ligne pour les class et function
- L'accolade d'ouverture doit être sur la même ligne pour les if, for...

```php
<?php
namespace Vendor\Package;

use FooInterface;
use BarClass as Bar;
use OtherVendor\OtherPackage\BazClass;

class Foo extends Bar implements FooInterface
{
    public function sampleMethod($a, $b = null)
    {
        if ($a === $b) {
            bar();
        } elseif ($a > $b) {
            $foo->bar($arg1);
        } else {
            BazClass::bar($arg2, $arg3);
        }
    }

    final public static function bar()
    {
        // method body
    }
}
```

3.
Les PSR

Interfaces

Les Interfaces

Une interface et un “Contrat” que l’on passe.

```php
interface Crayon
{
    public function getNom();
    public function ecrire();
}
```

```php
class CrayonGris implements Crayon
{
    public function getNom()
    {
        return 'Crayon gris';
    }

    public function tailler()
    {
        //todo
    }

    public function ecrire()
    {
        //todo
    }
}
```

```php
class Bic implements Crayon
{
    public function getNom()
    {
        return 'Bic';
    }

    public function machouiller()
    {
        //todo
    }

    public function ecrire()
    {
        //todo
    }
}
```

PSR-3: Logger Interface

```php
<?php

namespace Psr\Log;

interface LoggerInterface
{
    public function emergency($message, array $context = array());
    public function alert($message, array $context = array());
    public function critical($message, array $context = array());
    public function error($message, array $context = array());
    public function warning($message, array $context = array());
    public function notice($message, array $context = array());
    public function info($message, array $context = array());
    public function debug($message, array $context = array());

    public function log($level, $message, array $context = array());
}
```

http://www.php-fig.org/psr/psr-3/

PSR-16: Common Interface for Caching Libraries

```php
<?php

namespace Psr\SimpleCache;

interface CacheInterface
{
    public function get($key, $default = null);
    public function set($key, $value, $ttl = null);

    public function delete($key);
    public function clear();

    public function getMultiple($keys, $default = null);
    public function setMultiple($values, $ttl = null);
    public function deleteMultiple($keys);
    public function has($key);
}
```

http://www.php-fig.org/psr/psr-16/

PSR-11: Container interface (Review)

```php
<?php
namespace Psr\Container;

use Psr\Container\Exception\ContainerExceptionInterface;
use Psr\Container\Exception\NotFoundExceptionInterface;

/**
* Describes the interface of a container that exposes methods to read its entries.
*/
interface ContainerInterface
{
    /**
     * Finds an entry of the container by its identifier and returns it.
     *
     * @param string $id Identifier of the entry to look for.
     *
     * @throws NotFoundExceptionInterface  No entry was found for this identifier.
     * @throws ContainerExceptionInterface Error while retrieving the entry.
     *
     * @return mixed Entry.
     */
    public function get($id);
    public function has($id);
}
```

https://github.com/container-interop/fig-standards/blob/master/proposed/container.md

4.
Toutes les PSR

ACCEPTED

NUM	TITLE	EDITOR	COORDINATOR	SPONSOR
1	Basic Coding Standard	Paul M. Jones	*N/A*	*N/A*
2	Coding Style Guide	Paul M. Jones	*N/A*	*N/A*
3	Logger Interface	Jordi Boggiano	*N/A*	*N/A*
4	Autoloading Standard	Paul M. Jones	Phil Sturgeon	Larry Garfield
6	Caching Interface	Larry Garfield	Paul Dragoonis	Robert Hafner
7	HTTP Message Interface	Matthew Weier O'Phinney	Beau Simensen	Paul M. Jones
13	Hypermedia Links	Larry Garfield	Matthew Weier O'Phinney	Marc Alexander
16	Simple Cache	Paul Dragoonis	Jordi Boggiano	Fabien Potencier

REVIEW

NUM	TITLE	EDITOR	COORDINATOR	SPONSOR
11	Container Interface	Matthieu Napoli, David Négrier	Matthew Weier O'Phinney	Korvin Szanto

DRAFT

NUM	TITLE	EDITOR(S)	COORDINATOR	SPONSOR
5	PHPDoc Standard	Mike van Riel	Vacant	Vacant
8	Huggable Interface	Larry Garfield	Vacant	Vacant
9	Security Advisories	Michael Hess	Korvin Szanto	Larry Garfield
10	Security Reporting Process	Michael Hess	Larry Garfield	Korvin Szanto
12	Extended Coding Style Guide	Korvin Szanto	Alexander Makarov	Robert Deutz
14	Event Manager	Chuck Reeves	Brian Retterer	Roman Tsiupa
15	HTTP Middlewares	Woody Gilk	Paul M Jones	Jason Coward
17	HTTP Factories	Woody Gilk	Roman Tsiupa	Paul M Jones

DEPRECATED

NUM	TITLE	EDITOR	COORDINATOR	SPONSOR
0	Autoloading Standard	Matthew Weier O'Phinney	*N/A*	*N/A*

Participer

Les votes sont fermés.
Mais les projets sont sur github

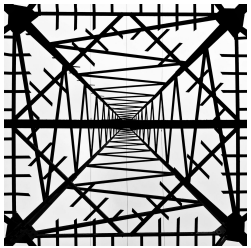
- Chat on IRC Channel
- Contribute via Github
- Join our mailing list

Merci

Des Questions ?

Me retrouver

Facebook - Twitter - LinkedIn - Meetup Afup



Julien Vinber

Retrouvez la présentation sur Slideshare

https://fr.slideshare.net/JulienVinber/meet-up-symfony-16-juin-2017-les-psr/