Maximising Performance in ZF2

Gary Hockin

Roave

Me

Roave

- Zend Framework Contributor
  - (and docs maintainer)
- Developer at Roave, Yamgo and AdSpruce
- Yolo Certified Engineer
- GeeH on IRC
- @GeeH on Twitter

(SSssshhhh... Zend Framework 2 Certified Engineer)

http://www.futurama-madhouse.net/bios/bioNibbler.jpg

This Talk Will...

- Introduce the concept of performance in PHP apps
- Introduce tools that let you record and monitor your performance
- Show you known performance hogs in ZF2, and how to counter them

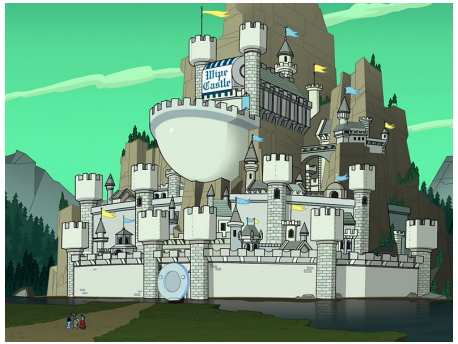
http://wiki.mafiascum.net/images/0/07/Zoidberghooray.jpg

You’re still here!

Tools

Siege (or ab)

Allows you to measure how many requests your website can handle in a given time period.



http://static2.wikia.nocookie.net/__cb20090607195642/en.futurama/images/6/67/Wipecastle.jpg

Siege

Installing:

```
yum | apt-get | brew
 install siege
```

Siege

Using:

```
siege -c 10 -t 1m -b http://mysite.dev
```

```
HTTP/1.1 200   0.45 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.44 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.46 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.48 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.42 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.45 secs:    8887 bytes ==> GET  /
```

Siege

Results:

```
Transactions:                    158 hits
Availability:                    100.00 %
Elapsed time:                    9.73 secs
Data transferred:                1.34 MB
Response time:              0.60 secs
Transaction rate:                16.24 trans/sec
Throughput:                      0.14 MB/sec
Concurrency:                     9.79
Successful transactions:     158
Failed transactions:        0
Longest transaction:        3.59
Shortest transaction:       0.39
```

Siege

Results:

```
Transactions:                  158 hits
Availability:                  100.00 %
Elapsed time:                    9.73 secs
Data transferred:                1.34 MB
Response time:                   0.60 secs
Transaction rate:               16.24 trans/sec
Throughput:                      0.14 MB/sec
Failed transactions              0
Failed transactions:             0
Longest transaction:             3.59
Shortest transaction:            0.39
```

Siege

Results:

```
Transactions:                    158 hits
Availability:                    100.00 %
Elapsed time:                    9.73 secs
Data transferred:                1.24 MB
Response time:                   0.60 secs
Transaction rate:                16.24 trans/sec
Throughput:                      0.14 MB/sec
Concurrency:                     9.79
Successful transactions:    158
Failed transactions:        0
Longest transaction:        3.59
Shortest transaction:       0.39
```

Siege

Results:

```
Transactions:                    158 hits
Availability:                    100.00 %
Elapsed time:                    9.73 secs
Data transferred:                1.24 MB
Response time:                   0.60 secs
Transaction rate:                16.24 trans/sec
Throughput:                      0.14 MB/sec
Concurrency:                     9.79
Successful transactions:         158
Failed transactions:             0
Longest transaction:             3.59
Shortest transaction:            0.39
```

Tools

Xdebug (or xhprof)

Let’s you understand which parts of your PHP application are taking so damn long.

http://upload.wikimedia.org/wikipedia/en/c/cb/FuturamaHermesConrad.png

Xdebug

Installing:

http://xdebug.org/docs/install

PECL

Brew

Yum/Apt

???

Xdebug

Using:

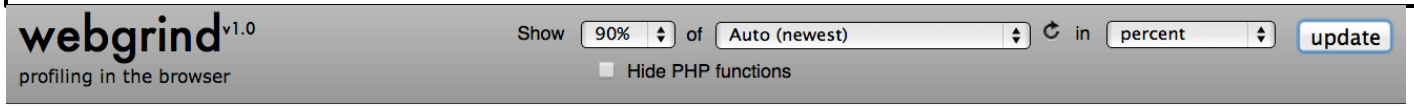
```
php.ini
[xdebug]
zend_extension="/full/path/to/xdebug.so"
xdebug.profiler_enable = 0
xdebug.profiler_enable_trigger = 1
xdebug.profiler_output_dir = "/path/to/profile/data"
```

Webgrind

https://github.com/jokkedk/webgrind

Xdebug

```
http://zf2.dev?XDEBUG_PROFILE
```



/Users/garyhockin/www/perf/public/index.php

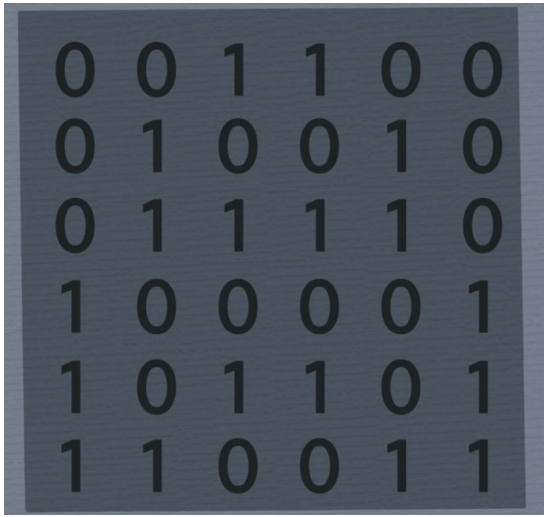
cachegrind.out.1023 @ 2014-01-13 15:28:21

1022 different functions called in **229** milliseconds (1 runs, **101** shown)

Function	Invocation Count	Total Self Cost	Total Inclusive Cost
Composer\Autoload\ClassLoader->loadClass	215	52.03	89.38
Composer\Autoload\ClassLoader->findFile	215	5.90	6.84
Zend\EventManager\EventManager->triggerListeners	16	1.59	63.73
php::call_user_func	208	1.38	132.12
Zend\ServiceManager\ServiceManager->canonicalizeName	353	1.32	2.41
Zend\ServiceManager\ServiceManager->get	100	1.11	88.99
php::strtr	658	1.11	1.11
Zend\EventManager\EventManager->getSharedListeners	32	0.93	1.38
Zend\ServiceManager\ServiceManager->doCreate	46	0.91	86.13
php::file_exists	215	0.86	0.86
Zend\ServiceManager\ServiceManager->has	191	0.82	1.36
Zend\ServiceManager\ServiceManager->setAlias	35	0.78	1.35
Zend\ServiceManager\Config->configureServiceManager	11	0.72	3.23
Zend\EventManager\EventManager->attach	56	0.70	11.06

Change

Work out what causing the slowness, and fix it.



Siege

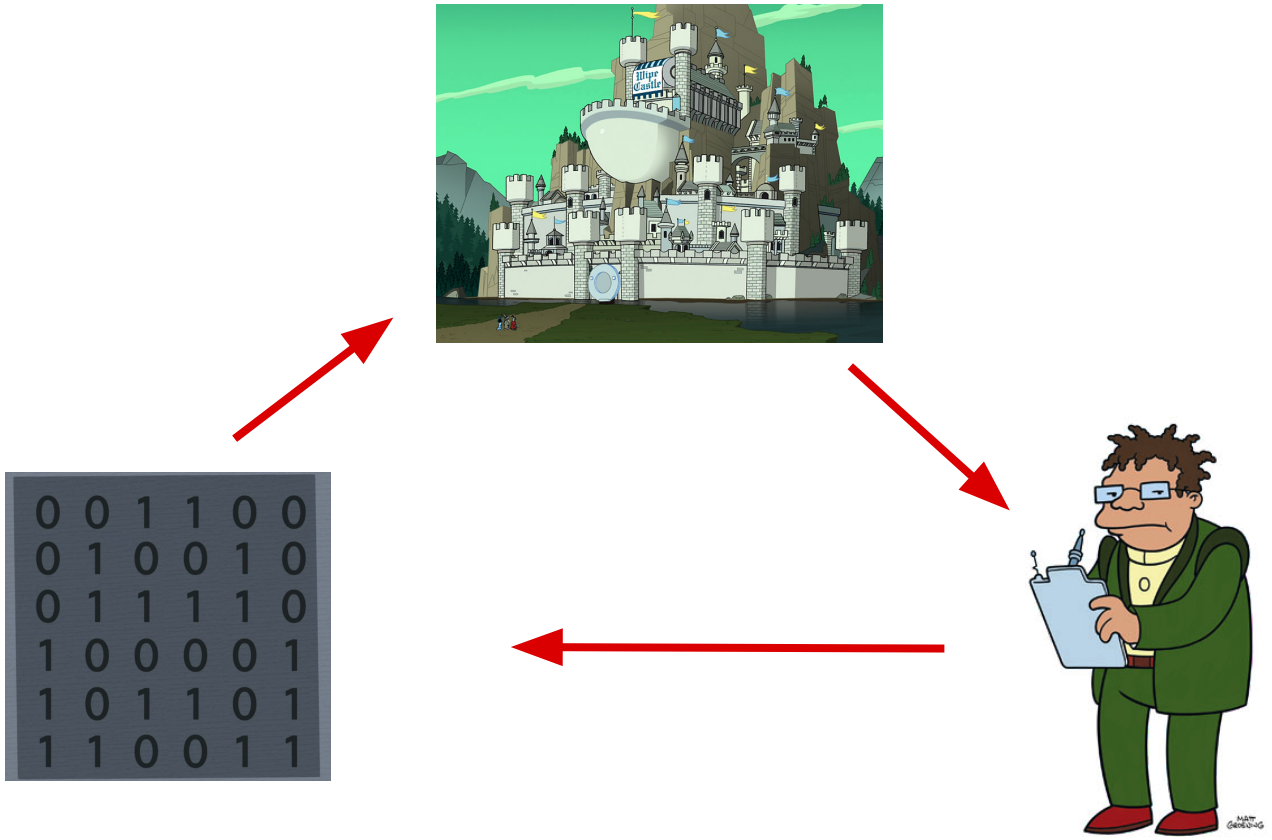
```
siege -c 10 -t 1m -b http://mysite.dev
```

```
HTTP/1.1 200   0.45 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.44 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.46 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.48 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.42 secs:    8887 bytes ==> GET  /
HTTP/1.1 200   0.45 secs:    8887 bytes ==> GET  /
```

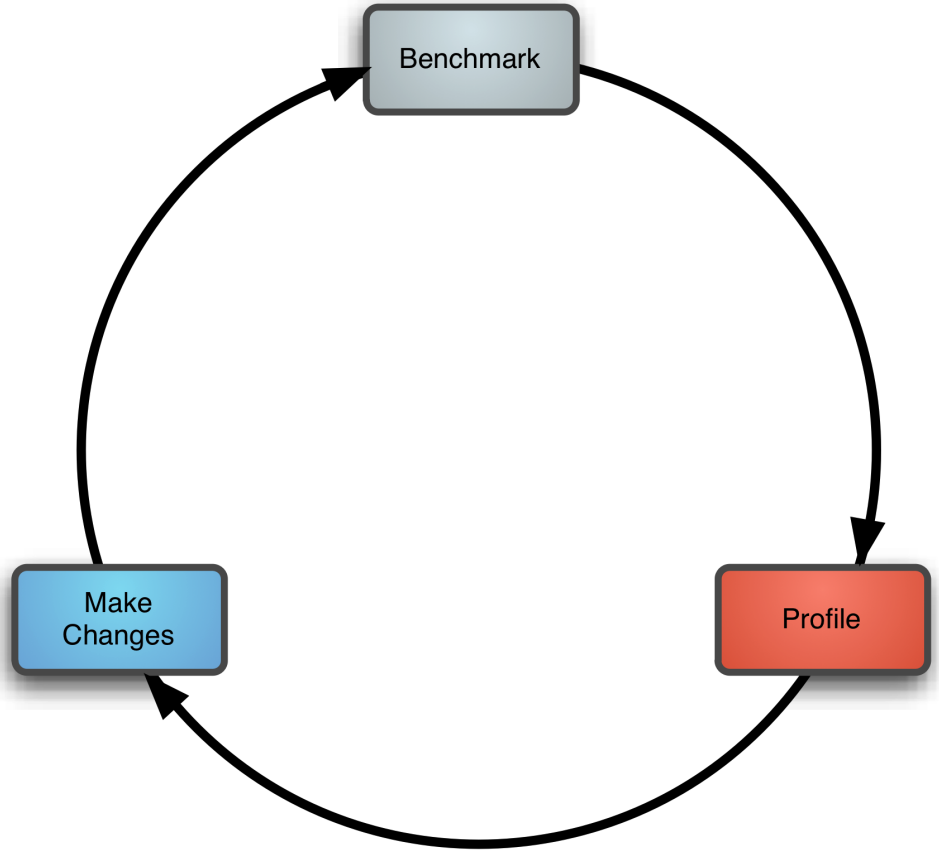
Siege

Results:

```
Transactions:                    158 hits
Availability:                    100.00 %
Elapsed time:                    9.73 secs
Data transferred:                1.34 MB
Response time:              0.60 secs
Transaction rate:                16.24 trans/sec
Throughput:                      0.14 MB/sec
Concurrency:                     9.79
Successful transactions:     158
Failed transactions:        0
Longest transaction:        3.59
Shortest transaction:       0.39
```



The Cycle of Hope™



AKA - The Performance Loop (thanks @dshafik)

Onwards to ZF2

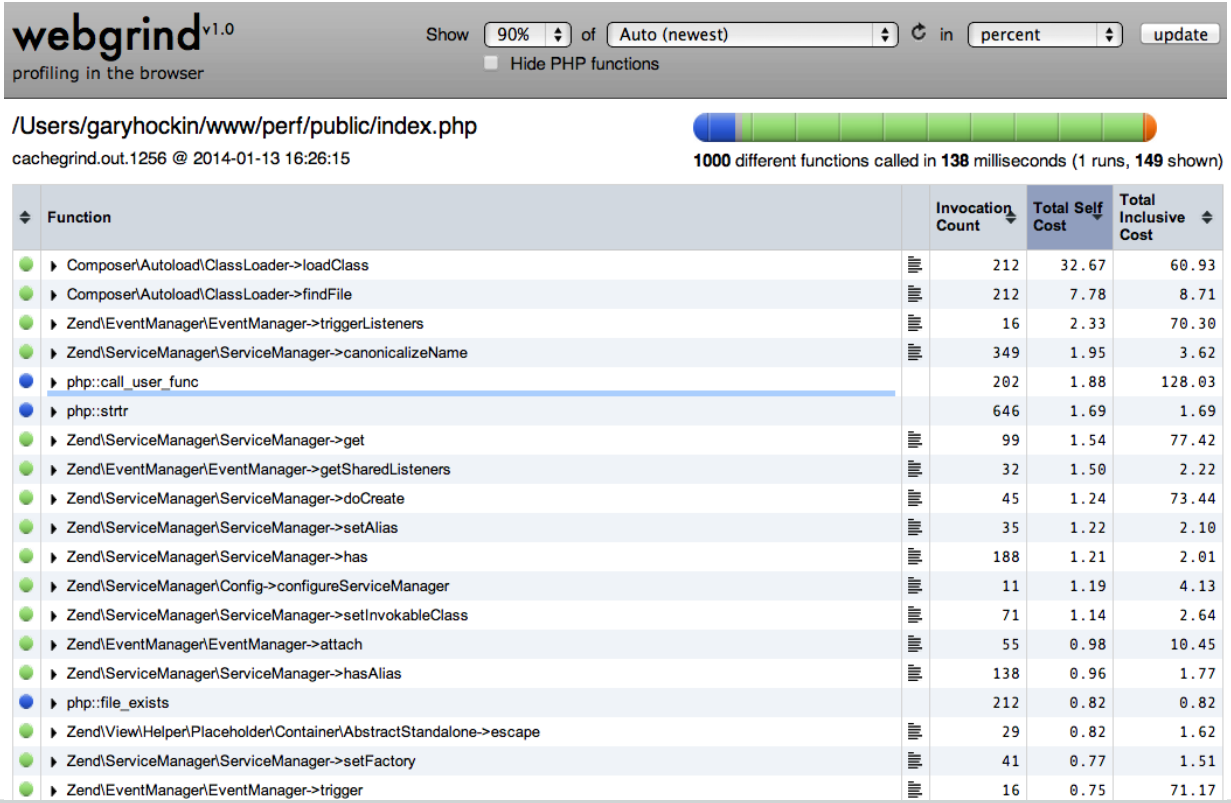
- Server Config
- Autoloading
- Config Caching
- Routing
- Template Maps
- Best Practices

ZF2 Skeleton App

Siege

```
siege -c 10 -t 1m -b http://perf.dev

Transactions:                   1971 hits
Availability:                   100.00 %
Elapsed time:                   59.30 secs
Data transferred:               10.04 MB
Response time:                  0.30 secs
Transaction rate:               33.24 trans/sec
Throughput:                     0.17 MB/sec
Concurrency:                    9.96
Successful transactions:    1971
Failed transactions:        0
Longest transaction:        0.49
Shortest transaction:       0.07
```

Profile



Function	Invocation Count	Total Self Cost	Total Inclusive Cost
Composer\Autoload\ClassLoader->loadClass	212	32.67	60.93
Composer\Autoload\ClassLoader->findFile	212	7.78	8.71
Zend\EventManager\EventManager->triggerListeners	16	2.33	70.30
Zend\ServiceManager\ServiceManager->canonicalizeName	349	1.95	3.62
php::call_user_func	202	1.88	128.03
php::strtr	646	1.69	1.69
Zend\ServiceManager\ServiceManager->get	99	1.54	77.42
Zend\EventManager\EventManager->getSharedListeners	32	1.50	2.22
Zend\ServiceManager\ServiceManager->doCreate	45	1.24	73.44
Zend\ServiceManager\ServiceManager->setAlias	35	1.22	2.10
Zend\ServiceManager\ServiceManager->has	188	1.21	2.01
Zend\ServiceManager\Config->configureServiceManager	11	1.19	4.13
Zend\ServiceManager\ServiceManager->setInvokableClass	71	1.14	2.64
Zend\EventManager\EventManager->attach	55	0.98	10.45
Zend\ServiceManager\ServiceManager->hasAlias	138	0.96	1.77
php::file_exists	212	0.82	0.82
Zend\View\Helper\Placeholder\Container\AbstractStandalone->escape	29	0.82	1.62
Zend\ServiceManager\ServiceManager->setFactory	41	0.77	1.51
Zend\EventManager\EventManager->trigger	16	0.75	71.17

Profile

`Composer\Autoload\ClassLoader->loadClass`	32.67%
`Composer\Autoload\ClassLoader->findFile`	7.78%
	40.45%

Profile

`Composer\Autoload\ClassLoader->loadClass`	32.67%
`Composer\Autoload\ClassLoader->findFile`	7.78%
	40.45%

40% autoloading!!!!...

http://2.bp.blogspot.com/_UVBIDZcvnKY/TBpGOtZT7oI/AAAAAAAAACc/YoN-hL9Rsuw/s1600/weird-futurama-brain-slug-man.png

Opcode Cache

Enable an opcode cache!

APC on < PHP 5.5

Opcache on >= PHP 5.5

Takes the “compiled” opcode and caches it so that it doesn’t need to be “compiled” per request.

Profile

Before:

```
Composer\Autoload\ClassLoader->loadClass    32.67%
Composer\Autoload\ClassLoader->findFile      7.78%

Transaction rate:         33.24 trans/sec
```

After

```
Composer\Autoload\ClassLoader->loadClass    10.88%
Composer\Autoload\ClassLoader->findFile      9.58%

Transaction rate:         89.57 trans/sec
```

Profile

Before:

```
Composer\Autoload\ClassLoader->loadClass    32.67%
Composer\Autoload\ClassLoader->findFile      7.78%

Transaction rate:         33.24 trans/sec
```

After

```
Composer\Autoload\ClassLoader->loadClass    10.88%
Composer\Autoload\ClassLoader->findFile      9.58%

Transaction rate:         89.57 trans/sec
```

Not Damn Good Enough!!!...

Introducing EdpSuperluminal

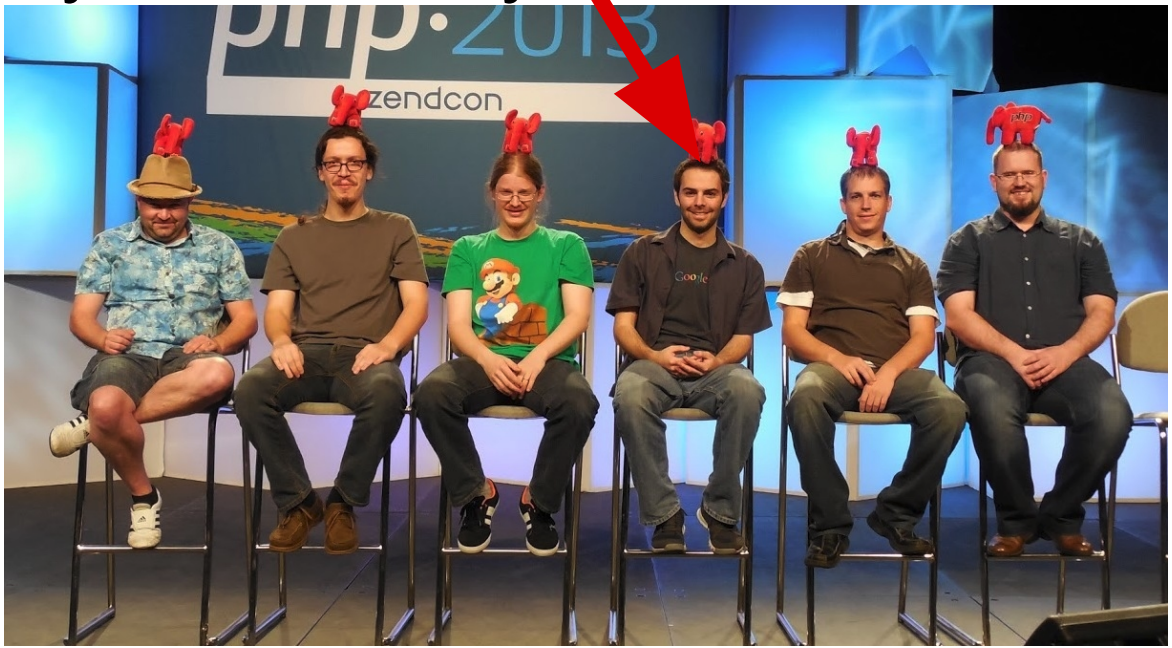
Combines all Zend component classes into one

uber-component-file

therefore eradicating autoloading

Introducing EdpSuperluminal

Written by Evan Coury



EdpSuperluminal

https://github.com/EvanDotPro/EdpSuperluminal

2 minutes to install and enable.

Use ?EDPSUPERLUMINAL_CACHE to generate the cache file (on every/heavy pages)

Don’t forget to consider generating this cache in your deployment strategy

Siege

Before:

```
Composer\Autoload\ClassLoader->loadClass       32.67%
Composer\Autoload\ClassLoader->findFile         7.78%

Transaction rate:           33.24 trans/sec
```

After OpCache:

```
Composer\Autoload\ClassLoader->loadClass       10.88%
Composer\Autoload\ClassLoader->findFile         9.58%

Transaction rate:           89.57 trans/sec
```

After EdpSuperluminal:

```
Composer\Autoload\ClassLoader->loadClass       0.29%
Composer\Autoload\ClassLoader->findFile        0.42%

Transaction rate:           117.52 trans/sec
```

YIPPEEE!

WHY DOES THIS WORK?

http://images3.wikia.nocookie.net/__cb20110916120044/en.futurama/images/f/f2/PhilipJ.Fry.png

Config Caching

Configs are merged on every request

Do configs change between requests?

Cache All The Things!

Config Caching

Before

```
Zend\Stdlib\ArrayUtils::merge    3,073µs
Transaction rate:          23.69 trans/sec
```

Config Caching

application.config.php

```
'config_cache_enabled' => true,
'config_cache_key' => 'module_config_cache',
'cache_dir' =>  './data/cache',
```

Config Caching

Before

```
Zend\Stdlib\ArrayUtils::merge    3,073µs
Transaction rate:          23.69 trans/sec
```

After

```
Zend\Stdlib\ArrayUtils::merge    3,258µs
Transaction rate:          23.87 trans/sec
```

Config Caching

What just happened?

Config Caching

module.config.php

```
‘service_manager’ => array(
    'factories' => array(
        'Application\Service\DubService' => function ($sm) {
            $tableGateway = $sm->get('DubTable');
            $ratingTable = $sm->get('RatingTable');
            return newDubService($tableGateway, $ratingTable);
        },
    ),
),
```

Config Caching

module.config.php

```
'factories' => array(
    'Application\Service\DubService' => function ($sm) {
        $tableGateway = $sm->get('DubTable');
        $ratingTable = $sm->get('RatingTable');
        return new DubService($tableGateway, $ratingTable);
    },
)
```

Closures can't be serialised when in an array!!!...

Config Caching

```
Fatal error: Call to undefined method
Closure::__set_state() in
/var/www/perf/data/module-config-
cache..php on line 70
```

Config Caching

Move to Module.php

```
public function getServiceConfig(ServiceManager $sm) {
    return array(
        'factories' => array(
           'Application\Service\DubService' => function ($sm) {
                $tableGateway = $sm->get('DubTable');
                $ratingTable = $sm->get('RatingTable');
                return new DubService(
                    $tableGateway, $ratingTable);
            },
}
```

Config Caching

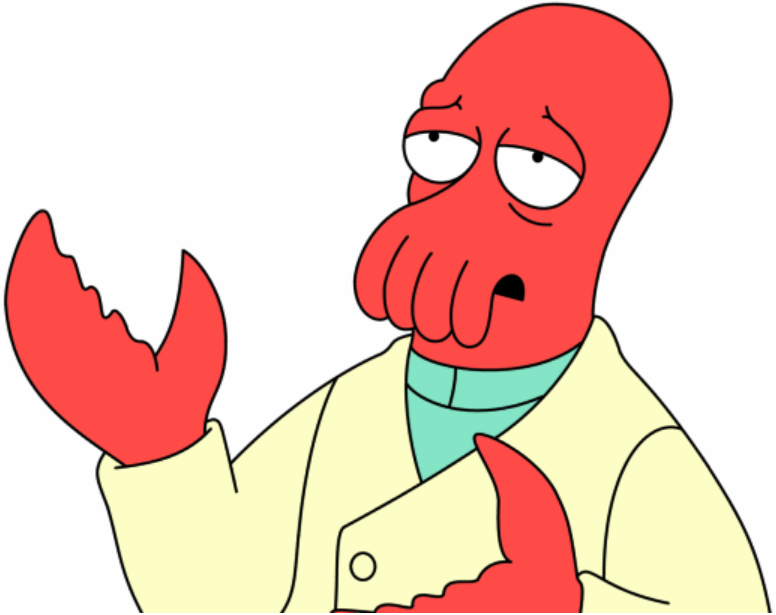
Before

```
Zend\Stdlib\ArrayUtils::merge    3,073µs
Transaction rate:          23.69 trans/sec
```

After

```
Zend\Stdlib\ArrayUtils::merge    2,394µs
Transaction rate:          25.95 trans/sec
```

Config Caching



Meh.

http://1.bp.blogspot.com/-eeocDLSYU48/T_GqlFYOR-I/AAAAAAAAC1A/EZisLdaVmlw/s1600/zoidberg.png

Routing

Good, but can be better...

The best kept secret is...

Routing

is...

Routing

... the ordering of your routes matters.

Routing

```
'routes' =>
    array(
        'home' =>
            array(
                'type' => 'Literal',
                'options' =>
                    array(
                        'route' => '/',
                        'defaults' =>
                            array(
                                'controller' => 'Application\\Controller\\Index',
                                'action' => 'index',
                            ),
                    ),
            ),
              ... 5,000 routes here ...
```

Routing

Before

```
Zend\Mvc\Router\Http\Segment->parseRouteDefinition        126,214µs

Zend\Mvc\Router\Http\TreeRouteStack->match               267,004µs

Transaction rate:              3.99 trans/sec
```

Routing

Before

```
Zend\Mvc\Router\Http\TreeRouteStack->match          267,004µs

Transaction rate:          3.99 trans/sec
```

Routing

```
'routes' =>
    array(
            ... 5,000 routes here ...
        'home' =>
            array(
                'type' => 'Literal',
                'options' =>
                    array(
                        'route' => '/',
                        'defaults' =>
                            array(
                                'controller' => 'Application\\Controller\\Index',
                                'action' => 'index',
                            ),
                    ),
            ),
```

Routing

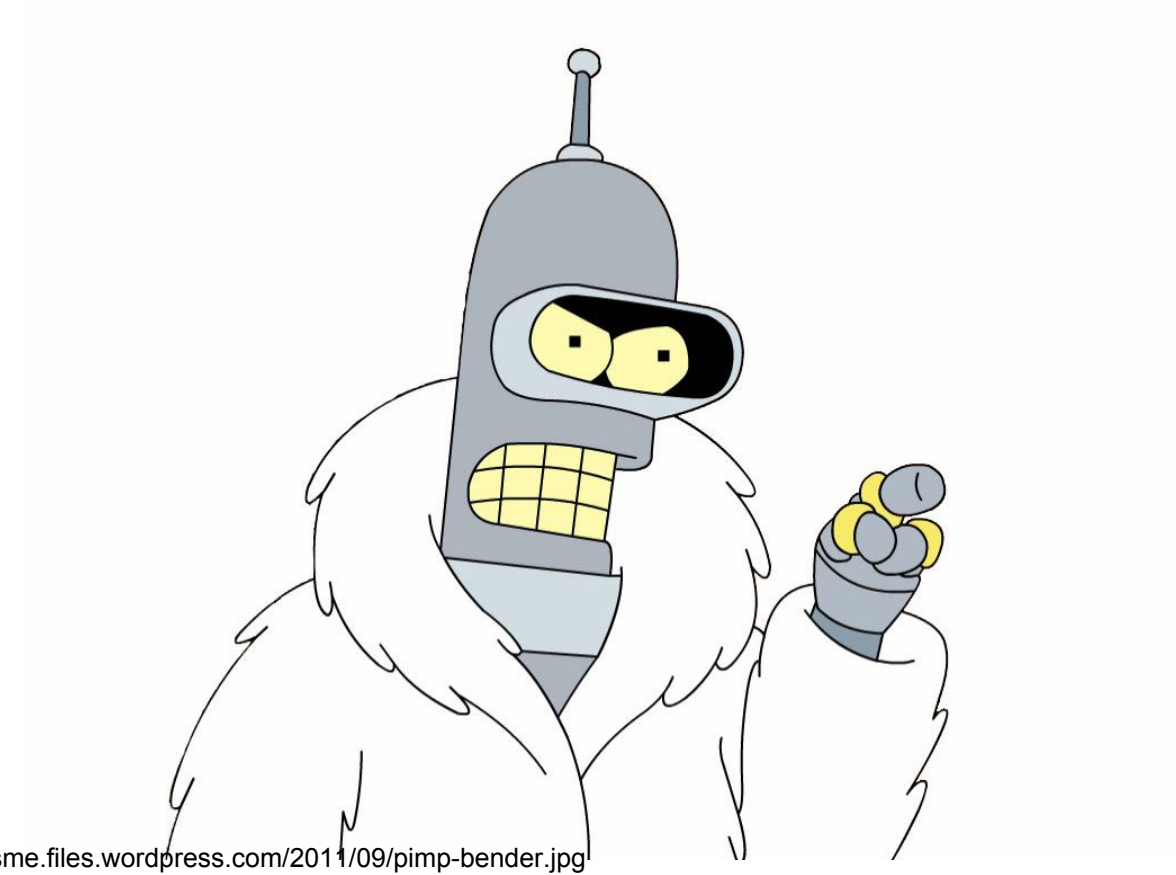
Before

```
Zend\Mvc\Router\Http\TreeRouteStack->match          267,004µs

Transaction rate:          3.99 trans/sec
```

After

```
Zend\Mvc\Router\Http\TreeRouteStack->match          135µs

Transaction rate:          4.87 trans/sec
```

We are still parsing > 5,000 routes per request!



http://hollywoodhatesme.files.wordpress.com/2011/09/pimp-bender.jpg

Module ordering counts too!

View Template Map

Tells the view renderer where to find the templates and view files it needs.

Before

```
Zend\View\Resolver\TemplatePathStack->resolve        153,445µs

Transaction rate:          13.29 trans/sec
```

View Template Map

Tells the view renderer where to find the templates and view files it needs.

module.config.php

```
'template_path_stack' => array(
        __DIR__ . '/../view',
    ),
```

View Template Map

Before

```
Zend\View\Resolver\TemplatePathStack->resolve          153,445µs

Transaction rate:              13.29 trans/sec
```

View Template Map

You can find these templates here...

```
‘view_manager’ => array(
    'template_map' => array(
            'layout/layout'           => __DIR__ . '/../view/layout/layout.phtml',
            'application/index/index' => __DIR__ . '/../view/application/index/index.phtml',
            'error/404'               => __DIR__ . '/../view/error/404.phtml',
            'error/index'             => __DIR__ . '/../view/error/index.phtml',
        ),
```

View Template Map

Before

```
Zend\View\Resolver\TemplatePathStack->resolve        153,445µs

Transaction rate:          13.29 trans/sec
```

After

```
Zend\View\Resolver\TemplatePathStack->resolve        60µs

Transaction rate:          21.07 trans/sec
```

View Template Map

“But, that sounds like a lot of work!”

View Template Map

Not to worry... in your module directory run:

```
php ../../vendor/bin/templatemap_generator.php
```

Then edit:

module.config.php

```
'template_map' => array(
    include(__DIR__ . /../template_map.php')
),
```

http://pool.theinfosphere.org/images/4/4e/Leela_promo_2.jpg

Leela Win!

Best Practices

1. Don’t use closures in module.config.php

You should know this by now.

Best Practices

2. Minimise use of the EventManager

With 100 Events:

```
Zend\EventManager\EventManager->triggerListeners       64,211µs

Transaction rate:         42.06 trans/sec
```

With 2 Events:

```
Zend\EventManager\EventManager->triggerListeners       4,943µs

Transaction rate:         82.76 trans/sec
```

http://www.unpopularscience.co.uk/wp-content/uploads/2013/05/Professor-Farnsworth.png

PERFORMANCE > MAINTENANCE?

Cache All The Things

Do you really need to be hitting that expensive storage in every request?

Cache All The Things

```
class ExpensiveStorage
{
    protected $cache;
    public function __construct(Zend\Cache\Storage\StorageInterface $cache)
    {
        $this->cache = $cache;

        public function getSomeItem($itemId)
        {
            if(!$this->cache->hasItem($itemId) {
                $this->cache->setItem($itemId, $this->getFromstorage($itemId));
            }
            return $this->cache->getItem($itemId);
        }

    }
```

Bonus

You DO have a favicon.ico right?

Bonus

You DO have a favicon.ico right?

No?

Bonus

You DO have a favicon.ico right?

No?

Really?


R | 2003

Bonus

No favicon = 2x request per single page being dispatched by the framework.

Ouch.

Think about other auto-requested files (robots.txt, apple icons etc)

Summary

1. Use Opcode Cache

Summary

1. Use Opcode Cache
2. Cache all the things
    a. Merged Config
    b. View Template Maps
    c. Expensive Storage Calls

Summary

1. Use Opcode Cache
2. Cache all the things
    a. Merged Config
    b. View Template Maps
    c. Expensive Storage Calls
3. Use EdpSuperluminal

Summary

1. Use Opcode Cache
2. Cache all the things
    a. Merged Config
    b. View Template Maps
    c. Expensive Storage Calls
3. Use EdpSuperluminal
4. Think about route ordering

Summary

1. Use Opcode Cache
2. Cache all the things
    a. Merged Config
    b. View Template Maps
    c. Expensive Storage Calls
3. Use EdpSuperluminal
4. Think about route ordering
5. Think thrice before using Event Manager

Summary

1. Use Opcode Cache
2. Cache all the things
    a. Merged Config
    b. View Template Maps
    c. Expensive Storage Calls
3. Use EdpSuperluminal
4. Think about route ordering
5. Think thrice before using Event Manager
6. Have a favicon

Summary

1. Use Opcode Cache
2. Cache all the things
    a. Merged Config
    b. View Template Maps
    c. Expensive Storage Calls
3. Use EdpSuperluminal
4. Think about route ordering
5. Think thrice before using Event Manager
6. Have a favicon
7. Use tools like Siege and Xdebug to validate performance problεµs and solutions

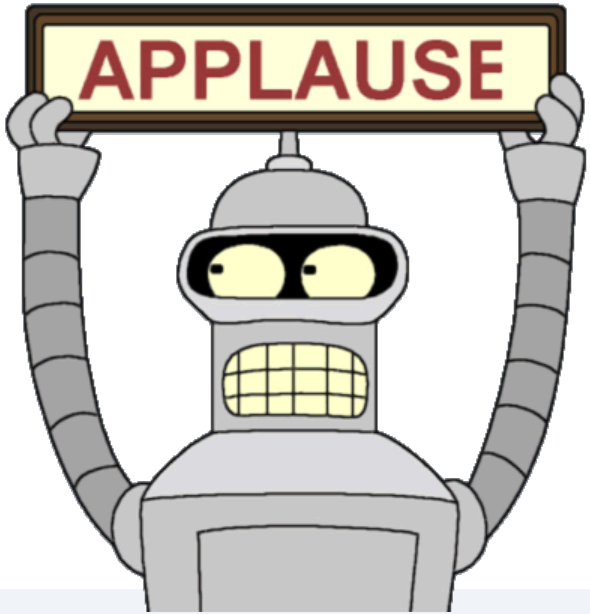
Questions?

http://theinfosphere.org/images/b/b1/Lrr.jpg

Gary Hockin
@GeeH - gary@hock.in

https://joind.in/10703

Gary Hockin
@GeeH - gary@hock.in



https://joind.in/10703