



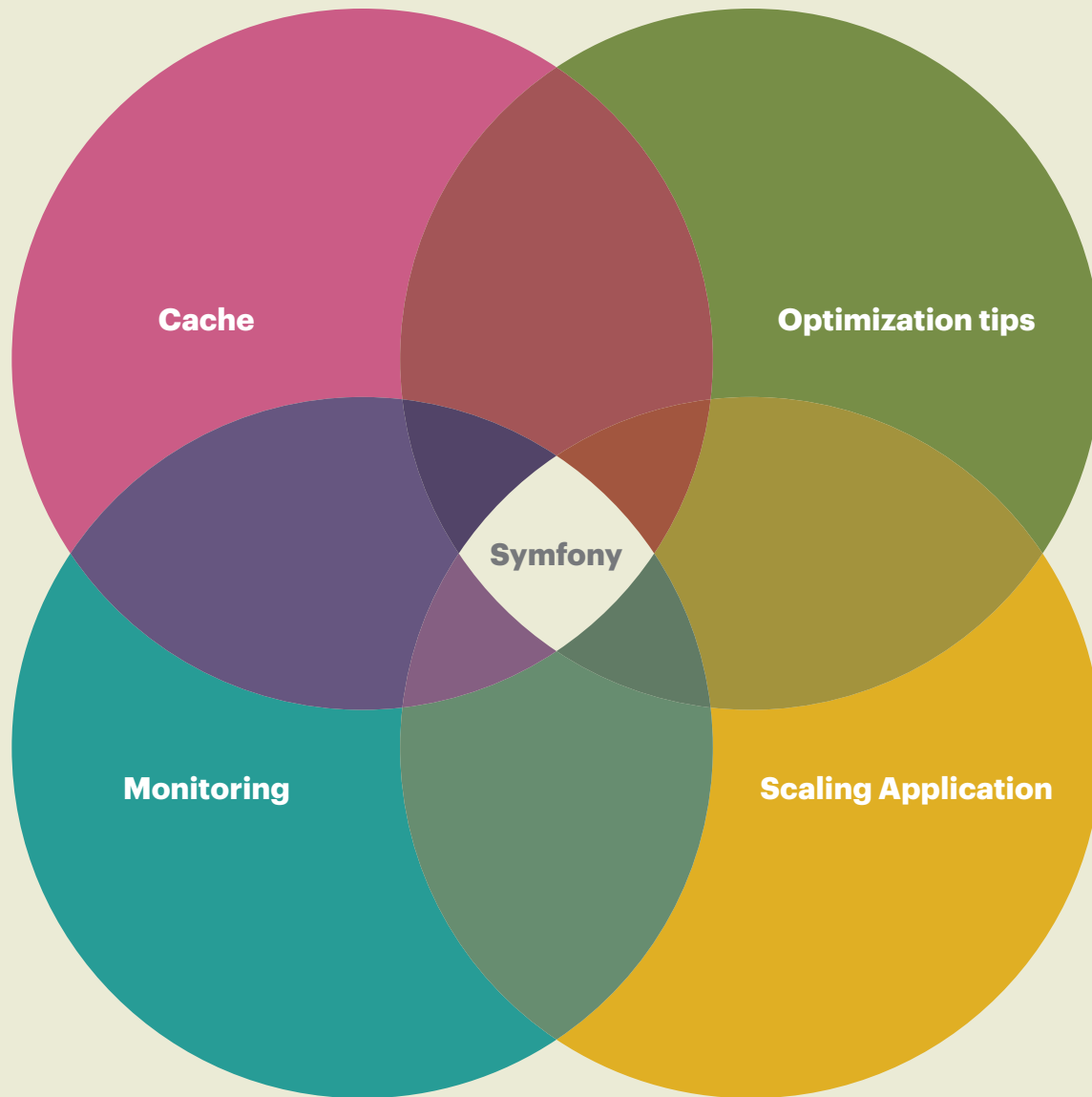
SYMFONY DAY

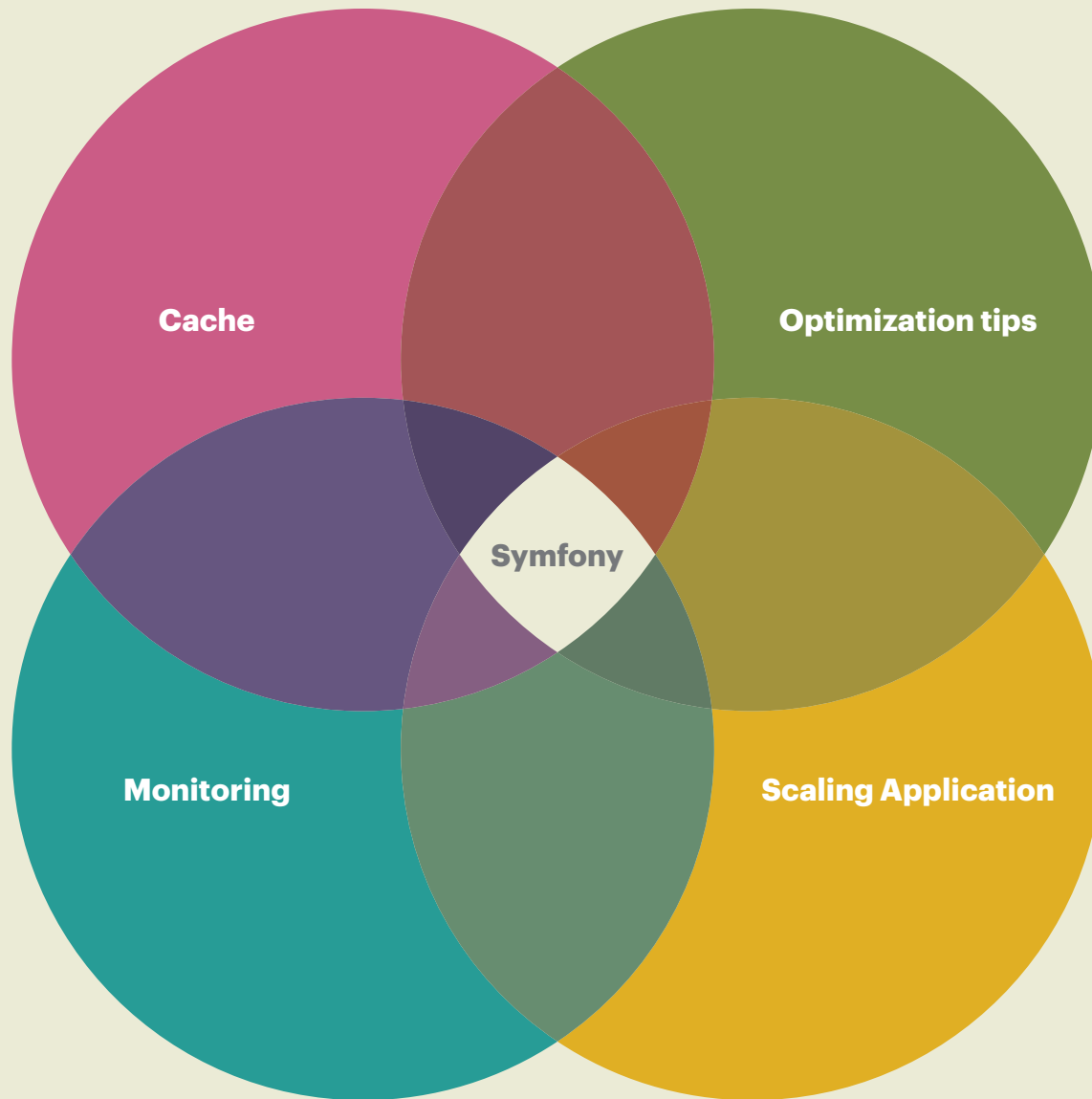
TURIN 5th OCTOBER 2012

**RATIONALLY BOOST YOUR SYMFONY2 APPLICATION
WITH CACHING, TIPS AND MONITORING**

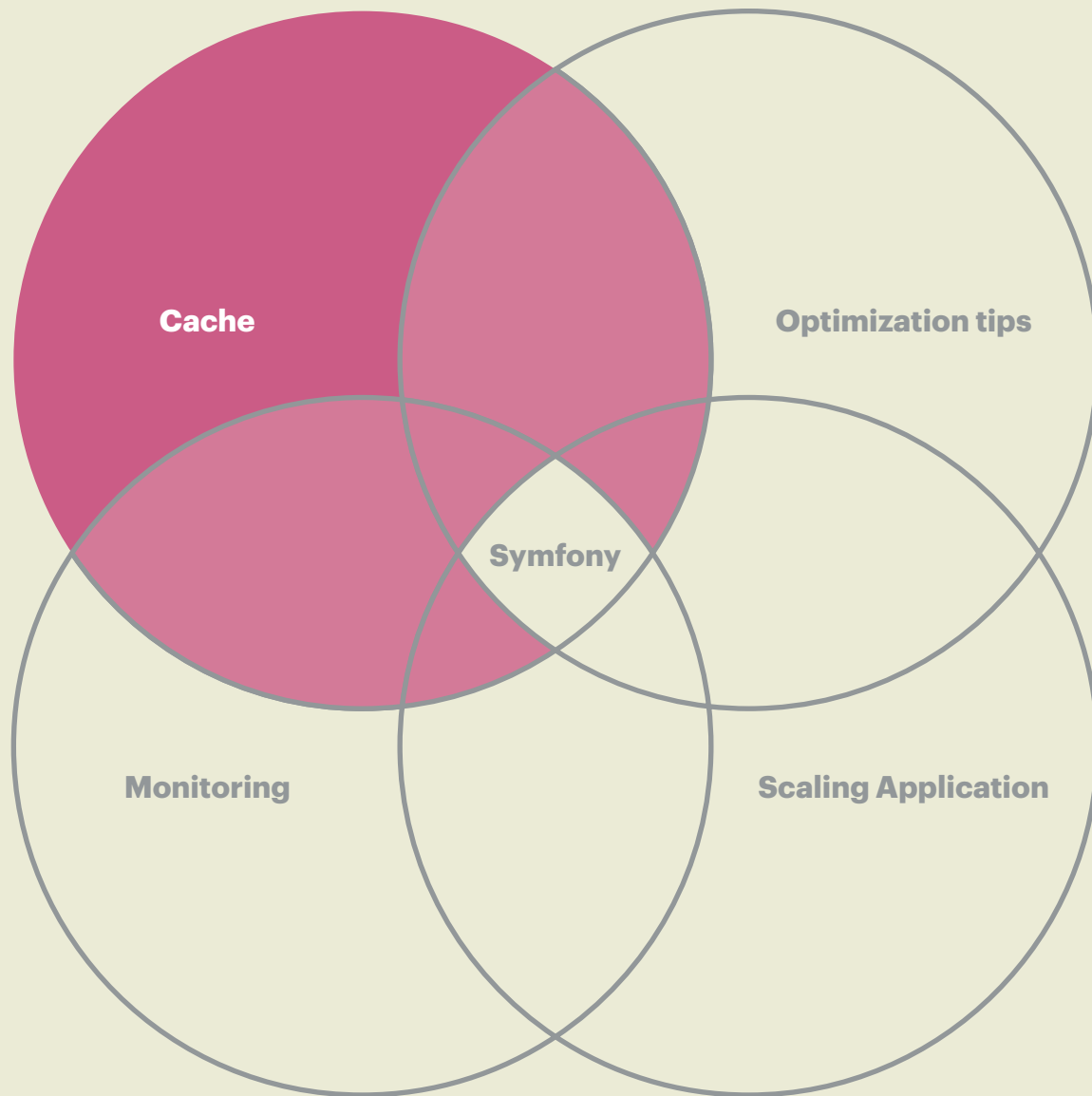
[/ @liuggio aka Giulio De Donato /]

NOT INVENTED HERE TALK
PLEASE SEE THE REFERENCES





Rationality



*“Great caching is like great sex.
It hides all the real problems”*

Vivek Haldar

Cache_(disambiguation)

Wikipedia: “a **Cache** (/ˈkæʃ/ KASH[1]) is a component that transparently stores data so that future requests for that data can be served faster.”

A **Cache Hit** happens if the Cache contains the answer.

A **Cache Miss** happens if the Cache doesn't contain the answer.

The perfect Cache - definition

A Cache is perfect when given the same request the **Cache miss** happens once. || given the same request the response is not 'processed' twice.

A Cache is perfect when the **Cache miss** happens once for the same response.

Cache

**The Cache
is a matter
of response**

Cache & Symfony 2

- Application caching: app/cache and APC
- Response caching and HTTP-CACHE
- Doctrine2 Query and Result Caching
- CDN and static files
- Other

Cache & Symfony 2 — app/cache

Terravison vendor folder is about 340 MB

The app/cache/prod folder is about 3000 files

- 1.** Annotation (Entity/Controller/...)
- 2.** Template
 - a.** name => file
 - b.** twig in php
- 3.** Assetic (definition)
- 4.** Url matcher, Url creator
- 5.** Translations array key=>value
- 6.** Doctrine Entity Proxy
- 7.** Your Bundle!

Cache & Symfony 2 — app/cache

Build your own CacheWarmer

```
namespace Symfony\Component\HttpKernel\CacheWarmer;
interface CacheWarmerInterface
    /**
     * Warms up the cache.
     * @param string $cacheDir The cache directory
     */
    public function warmUp($cacheDir);

    public function isOptional()
```

Add your class to the services with the tag *kernel.cache_warmer*

Note: the CacheWarmerInterface is not properly the same as the snippet.

Cache & Symfony2 — HTTP CACHE

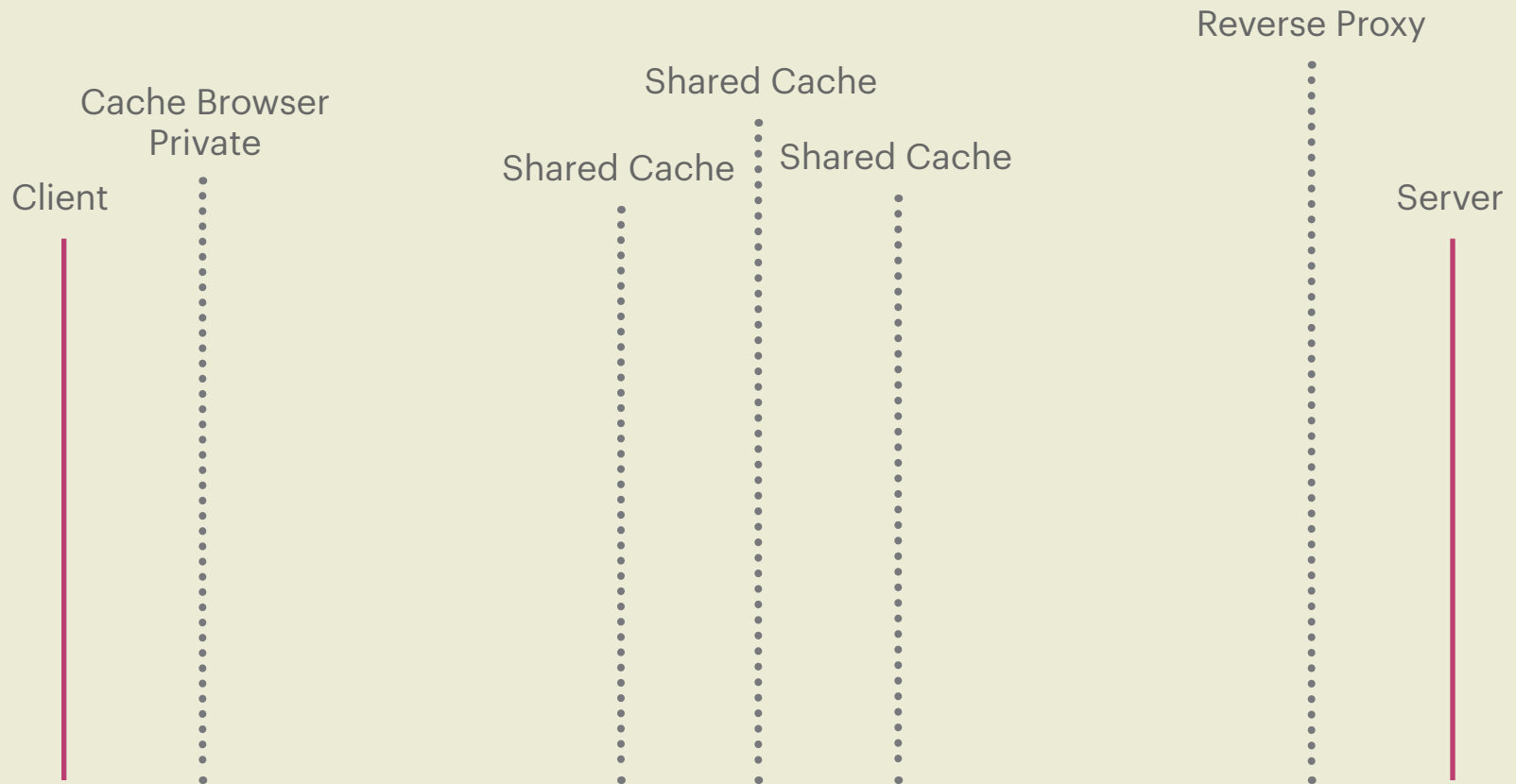
symfony.com/doc/master/book/http_cache.html

Symfony2 is a web framework built from scratch around the http specification

- a resource could be Fresh || Stale
- http-caching only on safe method
- validation || expiration

Cache & Symfony 2 — HTTP CACHE

Real world example



Cache & Symfony 2 — HTTP CACHE

symfony.com/doc/master/book/http_cache.html

```
$response->setPublic() //default is private
$response->setMaxAge(600);
// Same as above but only for shared caches
$response->setSharedMaxAge(600);

$response->setETag(md5($response->getContent()));
$response->setLastModified($datetime);
if ($response->isNotModified($this->getRequest()))
{return $response;}
else {
    // create a fresh response
}
$response->setVary('Accept-Encoding');
```

**The least
expensive
query is the
query you
never run.**

Cache & Symfony 2 — Doctrine

Doctrine\Common\Cache\Cache

```
// Fetches an entry from the cache.  
string function fetch($id);  
// Test if an entry exists in the cache.  
bool function contains($id);  
// Puts data into the cache.  
bool function save($id, $data, $lifeTime);  
// Deletes a cache entry.  
bool function delete($id);
```

Cache & Symfony 2 — Doctrine

Using Doctrine:

MetaData Cache	Annotation (APC)
Query Cache	DQL parsed into SQL (APC/REDIS/MEMCACHED)
Result Cache	The query result (REDIS/MEMCACHED/APC)

Symfony 2 real example

symfony.com/doc/current/book/doctrine.html

```
Class ProductRepository
...
    $query = $this->createQueryBuilder('p')
        ->where('p.price > :price')
        ->setParameter('price', '19.99')
        ->getQuery();

$products = $query->getResult();
```

Symfony 2 real example

symfony.com/doc/current/book/doctrine.html

```
Class ProductRepository
```

```
...
```

```
    $query = $this->createQueryBuilder('p')  
        ->where('p.price > :price')  
        ->setParameter('price', '19.99')  
        ->getQuery();
```

```
    $query->useResultCache(  
        true,  
        $lifetime,  
        __METHOD__ . serialize($query->getParameters())  
    );
```

```
    $query->useQueryCache();
```

```
    $products = $query->getResult();
```

*“There are only two hard things
in Computer Science: cache
invalidation and naming things”*

Phil Karlton

Cache Problem

Invalidation

Especially for http, but in general, you should not waste time to invalidate a value, but you should invest resources to find the right time of validity.

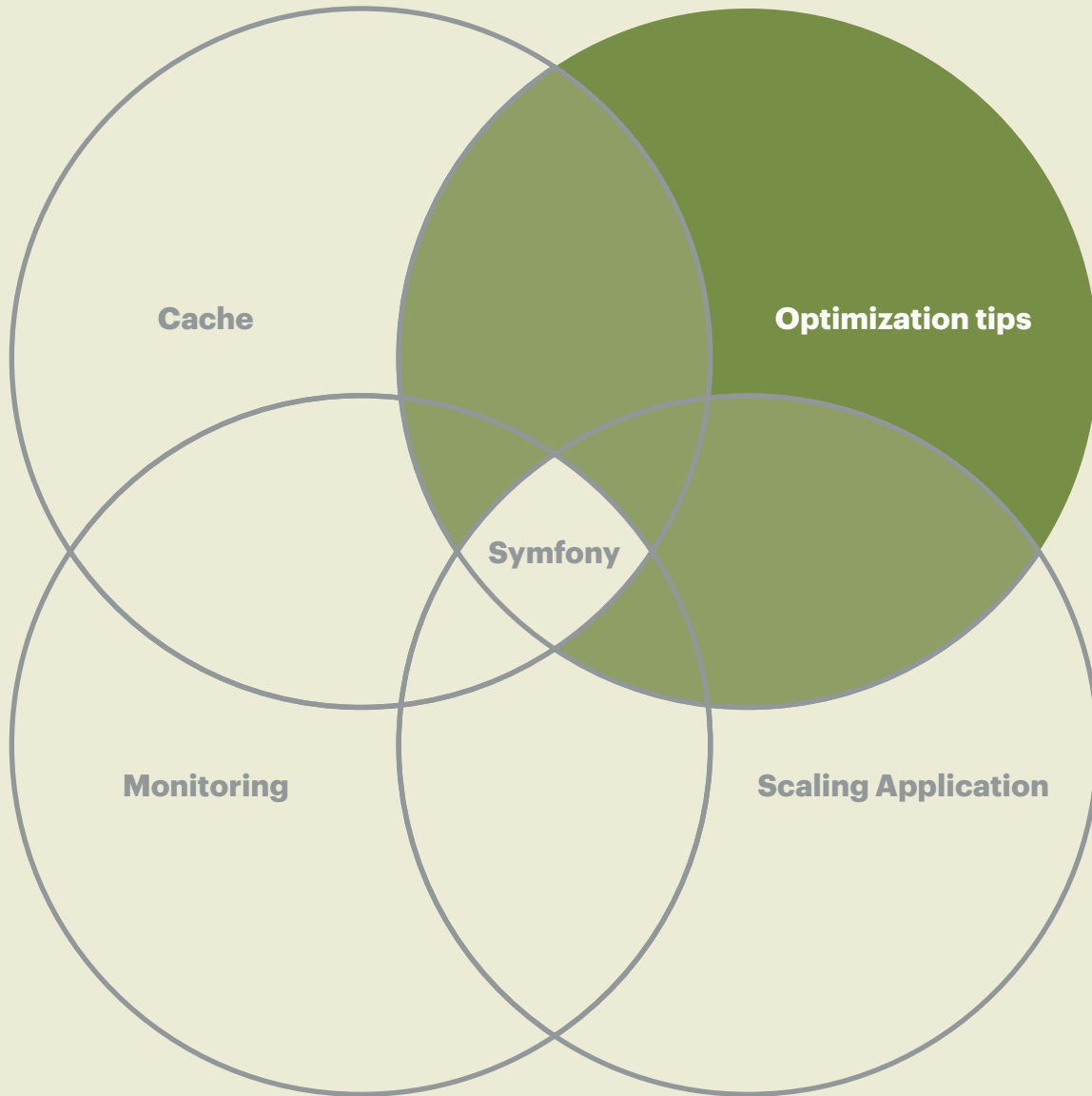
Cache miss storm

When a resource is no longer available (cache miss), and is required by many clients simultaneously (storm), a congestion happens.

**cache miss storm, cache stampede,
dog-piling, APC Cache slam**

Uniqueness of the cache key

Attention to the conflicts in the key definition, always put a prefix or namespace such as language or role.



Tips

Easy rules:

- The user should never wait,
slow operation data in a queue (Redis/*MQ)
processed by a consumer (cli + supervisors):
 - > Image Manipulation
 - > File Creation (pdf)
 - > Sending email as spool
 - > Log
- Use **Session with Redis**/Memcache
- Caution on what do you do in the loop && nested loop
- Attention on every I/O
- If there are too many “IF” the function may have an OOP problem
- Think before Flush

```
while{$sem->persist(); $sem->flush(); }
```


Tips — APC

Setting `'apc.stat=0'`

You should clear the cache after changing the code, when you deploy you should do it after the event `'symfony:cache:clear'`

the `'userx'` should restart php-fpm||apache gracefully, simply add to `/etc/sudoers`

```
userx ALL=(root) NOPASSWD: /usr/sbin/service apache2ctl graceful
```

or use the command `'php -r "apc_clear_cache();"'`

Tips — Autoloader

Improve the autoloader performance:

```
$ composer.phar dumpautoload --optimize
```

This autoloader needs maintenance, execute the command during the deploy after the event *'symfony:assets:install'*

Tips — Cache Common

Use APC in order to save your custom data

```
# config.ymlservices:
    cache:
        class: Doctrine\Common\Cache\ApcCache
```

In the Controller

```
if ($fooString = $this->get('cache')->fetch('foo')) {
    $foo = unserialize($fooString);
} else {
    // do the work
    $this->get('cache')->save('foo', serialize($foo));
}
```

Tips — Doctrine

The associations are LAZY by default

```
// Order
/**
 * @ManyToOne(targetEntity="Cart", cascade={"all"},
 *      fetch="EAGER" )
 */
private $cart;
```

If you want to change the fetch mode

```
<?php
$query = $em->createQuery("SELECT o FROM MyProject\Order o");
$query->setFetchMode("MyProject\Order", "Cart", "LAZY");
$query->execute();
```

Tips — Doctrine > 2.1

What's better than LAZY?

```
/*
 * @ORM\ManyToOne(targetEntity="Trips",
 *                  mappedBy="rides", fetch="EXTRA_LAZY")
 */
private $trips;
```

```
$turin->getTrips()->contains($turinCaselle);
$turin->getTrips()->count();
$turin->getTrips()->slice($offset, $length);
```

Tips — Doctrine > 2.1

Read Only Entity

```
/**
 * @ORM\Entity
 * @ORM\READONLY
 */
class Status
```

Tips — Doctrine

The wise use the reference

```
$post = $postRepository->find($postId);  
$comment->setPost($post);  
$comment->addPost(  
    $em->getReference( 'AcmeBundle\Entity\Post', $postId)  
);
```

Tips — Route / Virtual host

http://symfony.com/doc/2.0/cookbook/configuration/external_parameters.htm

```
$ app/console router:dump-apache --env=prod
# _welcome
RewriteCond %{REQUEST_URI} ^/$
RewriteRule .* app.php [QSA,L,E=_ROUTING__route:_welcome,
E=_ROUTING_DEFAULTS__controller:
Acme\\DemoBundle\\Controller\\WelcomeController\\:\\:
indexAction]
```

Insert the routes into the virtual host, they will become items of the global `$_SERVER` variable, and the `ApacheUrlMatcher` will read it. You could set external parameters directly into the virtual host

```
<VirtualHost *:80>
    ServerName      Symfony2
    SetEnv           SYMFONY__DATABASE__USER user
    SetEnv           SYMFONY__DATABASE__PASSWORD secret
```


Tips — Monolog

http://symfony.com/doc/2.0/reference/dic_tags.html#dic-tags-monolog

```
my_service:
  class: Fully\Qualified\Loader\Class\Name
  arguments: [@logger]
  tags:
    - { name: monolog.logger, channel: acme }
```

```
monolog:
  handlers:
    acme:
      type: stream
      path: /var/log/acme.log
      channels: acme
  doctrine:
    type: stream
    path: /var/log/doctrine.log
    channels: doctrine
```

Tips

Http-Microcaching:

- Save all the responses for 1 second
lot of examples on internet for Nginx

Query Caching :

- Don't use `'now()'` but `date('y-m-d')`
- Don't use `mysql_rand()`

Twig :

- Render with namespace
- Controller can be embedded asynchronously using the javascript library **hinclude.js**.

```
{% render '...:news' with {}, {'standalone': 'js'} %}
```

*“get your **assets** in line”*

Kriss Wallsmith

Tips — Assetic

http://symfony.com/doc/current/cookbook/assetic/asset_management.html

Asset Managment for PHP :

- Minify and combine all of your CSS and JS files
- Run all (or just some) of your CSS or JS files through some sort of compiler, such as LESS, SASS or CoffeeScript
- Run image optimizations on your images

Tips — Assetic & Cloud

```
{%- javascripts
  '@ ... /public/calendar.js' '@ ... /public/base.js'
  filter="yui_js" package='cdn'
%}<script type="text/javascript" src="{{ asset_url }}"></script>
{% endjavascripts %}
```

```
# config.yml
assetic:
  write_to: %cdn_protocol%://%cdn_container_name%
  assets:
    common:
      inputs: [public/calendar.js, public/base.js]
      package: cdn
      filters: {yui_js}
```

```
<script type="text/javascript" src="{{ assets/common.js }}"></script>
```

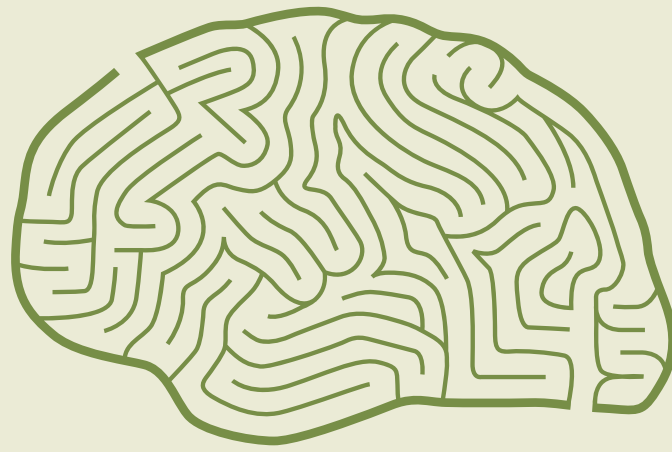
```
<script type="text/javascript"
  src="http://2dabf16.r6.cdn.eu/js/75a9295.js?20120904110916"></script>
```

Tips — Assetic && cloud

```
# config.yml
framework:
    templating:
    engines: ...
    packages:
        cdn:
            version: 20120904110916
            version_format: ~
            base_urls:
                http: [%cdn_container_url%]
                ssl: [%cdn_contailner_ssl_url%]
    assetic:
        write_to: %cdn_protocol%://%cdn_container_name%
        assets:
            common:
                inputs: [public/calendar.js, public/base.js]
                package: cdn
                filters: {yui_js}
```

Done everything?

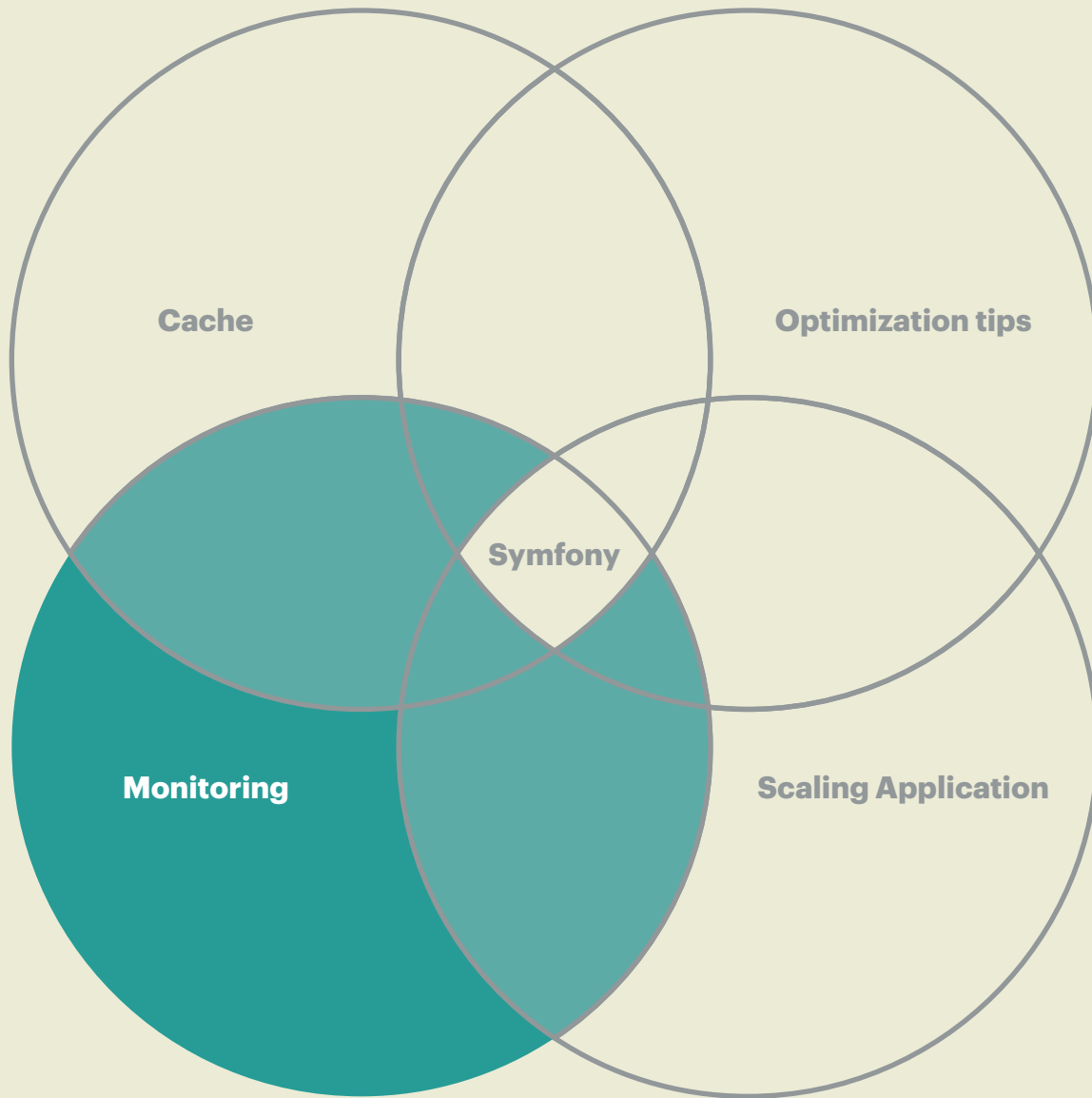
**Did you really
need it?**



Rationality

*“Knowing when optimization
is premature defines the differences
the master engineer and
the apprentice”*

Theo Schlossnagle



Profiling => Rationality

Know the problem

in dev

- xdebug
- symfony profiler

in prod

- xhprof
- monitoring and statistics
 - Server monitoring (Nagios/Cacti/NewRelic/...)
 - Application monitoring (Statsd/NewRelic/...)

Application monitoring:

The servers that you need

StatsD

- Node.JS daemon. Listens for message over UDP simple daemon for easy stats aggregation.

Graphite

- Real-time graphing system
- Components:
 - Carbon stores the data in Graphite's specialized database.
 - The data can then be visualized through graphite's web interfaces developed with django.
 - Whisper allows carbon to write multiple data points.

Application monitoring:

Installing: Statsd + Graphite + Carbon + Whisper
Symfony2 + StatsdClientBundle

Using Vagrant is **** easy:

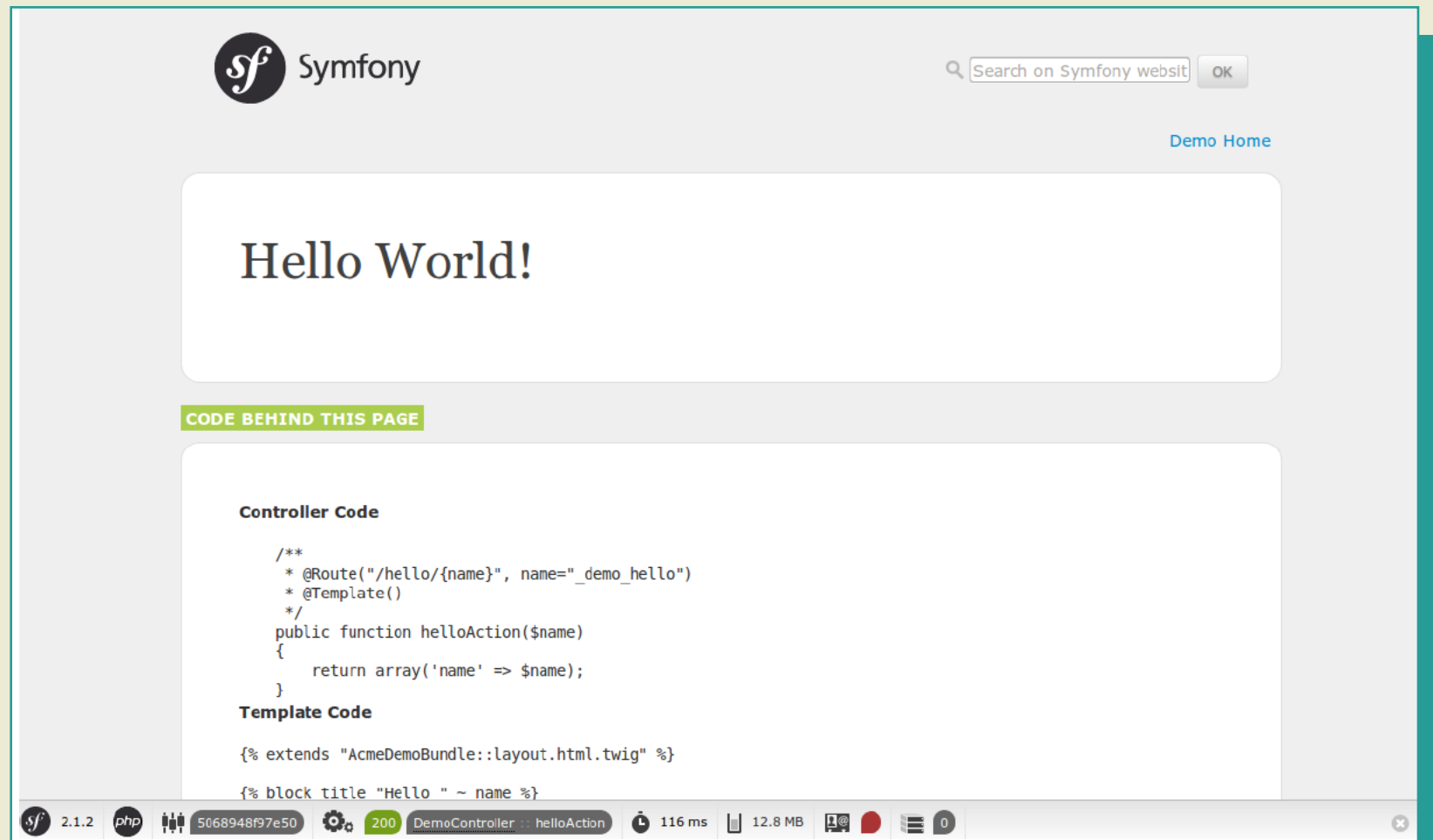
```
$ gem install vagrant
$ git clone https://github.com/liuggio/vagrant-statsd-graphite-puppet.git
$ cd vagrant-statsd-graphite-puppet.git
$ vagrant up
```

now install Symfony2 with StatsdClientBundle

```
$ php composer.phar create-project symfony/framework-standard-edition
$ composer require liuggio/statsd-client-bundle
```

add the bundle into the appKernel, copy/paste the configuration and ...

Symfony 2 && StatsDClientBundle



The screenshot shows a web browser displaying a Symfony 2 application. At the top left is the Symfony logo and the word "Symfony". At the top right is a search bar with the text "Search on Symfony websit" and an "OK" button. Below the search bar is a link "Demo Home". The main content area features a large white box with the text "Hello World!". Below this is a green button labeled "CODE BEHIND THIS PAGE". Underneath the button is a code editor showing the source code for the application. The code is divided into two sections: "Controller Code" and "Template Code". The "Controller Code" section contains a PHP class with a route and a template method. The "Template Code" section contains a Twig template that extends a layout and displays the name.

Symfony

Search on Symfony websit OK

[Demo Home](#)

Hello World!

CODE BEHIND THIS PAGE

Controller Code

```
/**
 * @Route("/hello/{name}", name="_demo_hello")
 * @Template()
 */
public function helloAction($name)
{
    return array('name' => $name);
}
```

Template Code

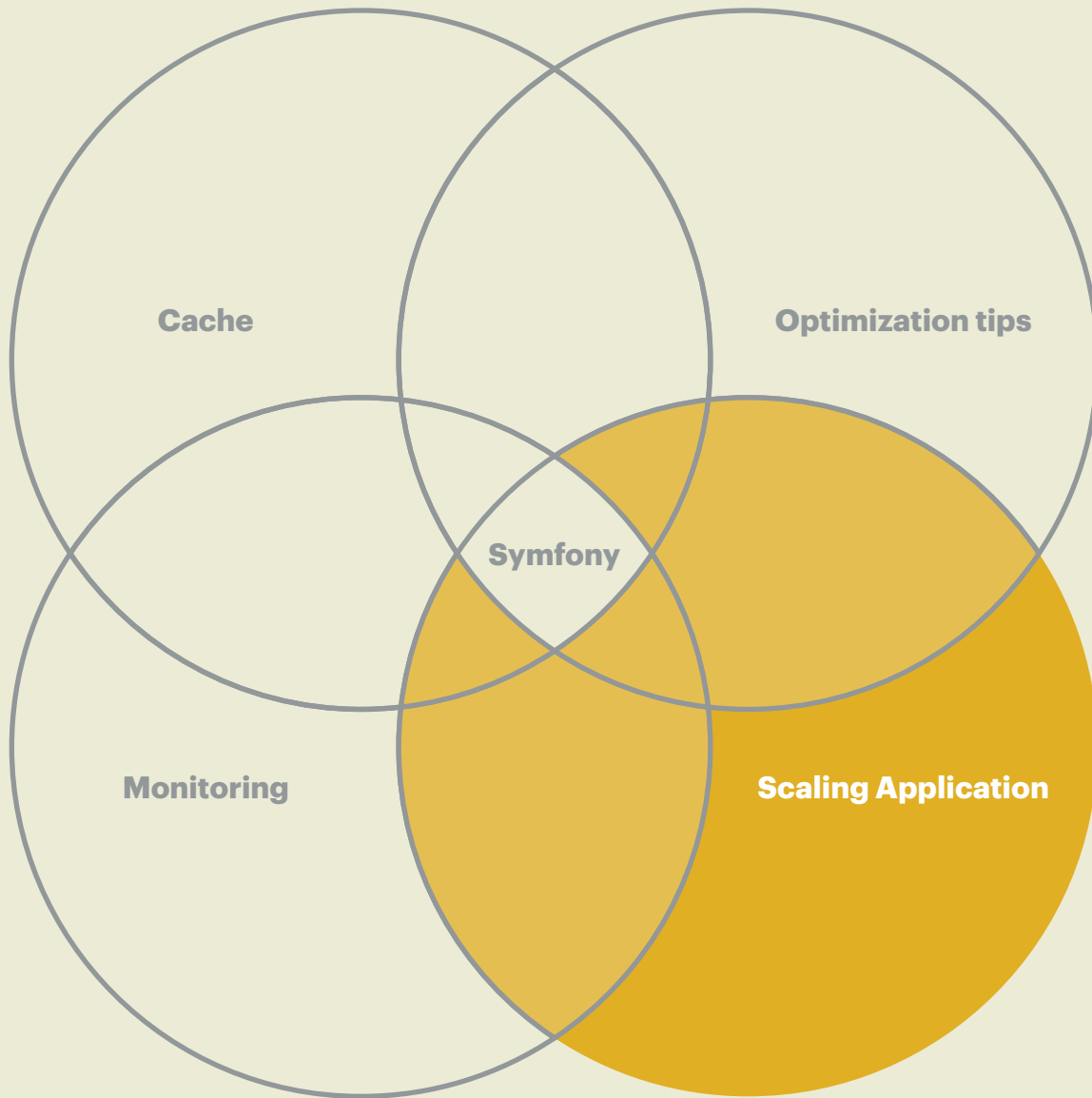
```
{% extends "AcmeDemoBundle::layout.html.twig" %}

{% block title "Hello " ~ name %}
```

sf 2.1.2 php 5068948f97e50 200 DemoController helloAction 116 ms 12.8 MB 0

Symfony 2 & StatsDClientBundle



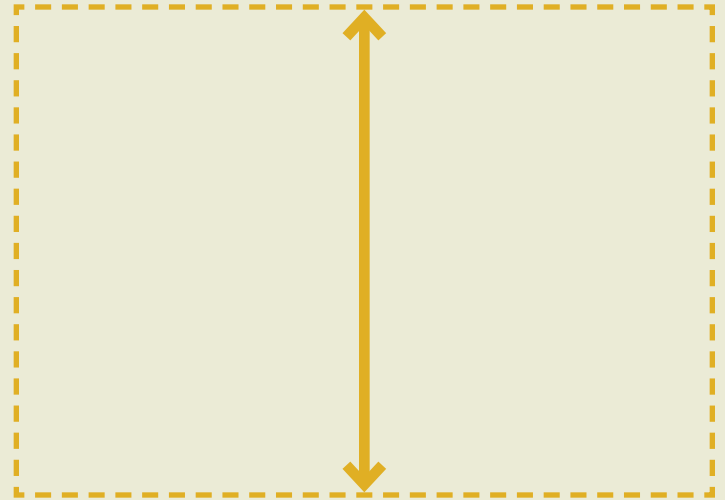


*“It won’t scale if it’s not designed
to scale”*

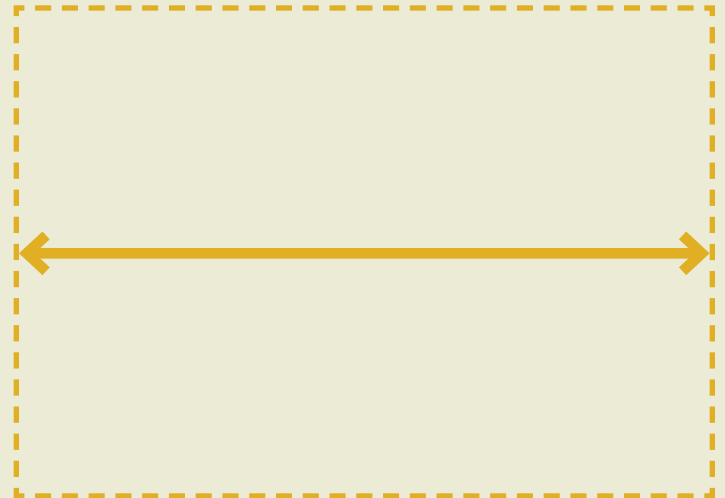
David Mitzenmacher

Scaling

Scaling vertically
get bigger



Scaling horizontally
get more



Scaling OOP Developing / SOA / EDA

“The real world can be accurately described as a collection of objects that interact”

Use the concepts from the OOP also for the services:

- Single Responsibility Principle
- Open Close Principle
- Law of Demeter
- **Don't overcomplicate**

SOA: Service Oriented Architecture

EDA: Event Driven Architecture

Conclusions

— COULD —

Use the cache

— COULD —

Optimize your application

— SHOULD —

Monitor your Servers/Applications

— MUST —

Follow the OOP/SOA

References

<http://tools.ietf.org/html/draft-ietf-httpbis-p6-cache-18>

<https://developers.google.com/speed/docs/best-practices/caching>

<http://notmysock.org/blog/php/user-cache-timebomb.html>

<http://sonata-project.org/blog/2012/5/15/assetic-package-configuration>

<http://www.slideshare.net/jonkruger/advanced-objectoriented-solid-principles>

<http://sonata-project.org/bundles/notification/2-0/doc/index.html>

<http://slides.seld.be/?file=2011-10-20+High+Performance+Websites+with+Symfony2.html>

<http://stackoverflow.com/questions/8893081/how-to-cache-in-symfony-2>

<http://www.slideshare.net/postwait/scalable-internet-architecture>

<http://techportal.inviqa.com/2009/12/01/profiling-with-xhprof/>

<http://www.mysqlperformanceblog.com/2010/09/10/cache-miss-storm>

<http://www.slideshare.net/iamcal/scalable-web-architectures-common-patterns-ap...>

<http://blog.servergrove.com/2012/04/18/how-to-create-a-cache-warmer-in-symfony2/>

<http://www.slideshare.net/jwage/doctrine-in-the-real-worldsf-live2011sanfran>

<http://www.slideshare.net/quipo/scalable-architectures-taming-the-twitter-firehose>

<https://github.com/liuggio/StatsDClientBundle>