

SYMPHONY TIPS & TRICKS

SymfonyCon
Madrid - 27 November 2014

Javier Eguiluz

About me

evangelist and trainer at

SensioLabs

Créateur de  Symfony

My Symfony work

Symfony Documentation Contributors

Most Active Contributors (last 2 months)

55 people have contributed to the Symfony documentation recently.



1. [Ryan Weaver](#)

191 commits 889 changes



2. [Wouter De Jong](#)

105 commits 3K changes



3. [Christian Flothmann](#)

19 commits 525 changes



4. [Javier Eguiluz](#)

17 commits 3K changes



5. [Nicolas Grekas](#)

15 commits 1K changes



6. [Fabien Potencier](#)

7 commits 14 changes



Symfony *Awards**

2013



2013 Best Blogger

Javier Eguiluz



A week of symfony #412

November 23, 2014



[Javier Eguiluz](#)

This week, Symfony project focused on tweaking and polishing all its code in preparation for the Symfony 2.6 launch of the next week, including [its shiny new installer](#). The week also continued for the [SymfonyCon conference](#), which will take place in March, the biggest event in Symfony's history.

Symfony2 development highlights

2.3 changelog:

- [367ed3c](#): compared version using PHP_VERSION_ID constant instead of version string function
- [e28f5b8](#): [FrameworkBundle] be smarter when guessing the doc directory from the command
- [92c8dfb](#): [SecurityBundle] authentication entry point is only registered with authentication listener, not with authentication listeners
- [76273bf](#): [FrameworkBundle] cache:clear command fills *.php.map files
- [4162713](#), [3039935](#): [HttpFoundation] made Request::get more consistent

2.5 changelog:

- [6e9642a](#): [WebProfilerBundle] fixed the profiler markup when there are no listeners

2.6 changelog:

- [8cf3d69](#): [TwigBundle/DebugBundle] moved dump extension & profiler to the TwigBundle

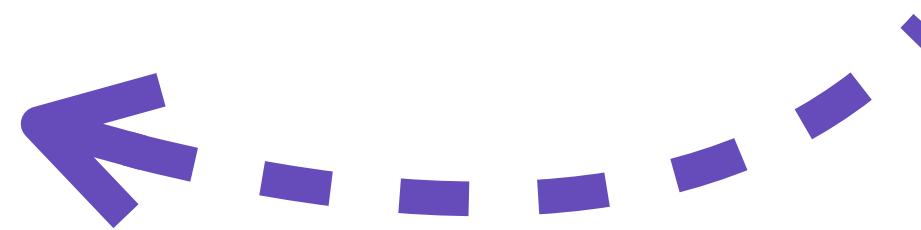
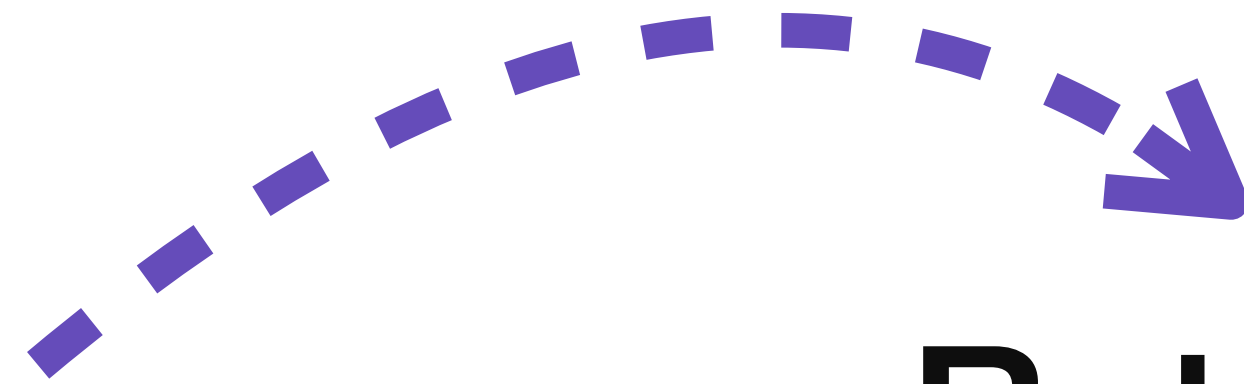
Why Symfony Tips and Tricks?

I don't work as
a developer

But I read and
review code every
single day

I also read every
single blog post
published about
Symfony

These are the tips and
tricks discovered
during this year.



**Thanks to my awesome co-workers for
sharing their knowledge!**



Grégoire



Nicolas



Hugo



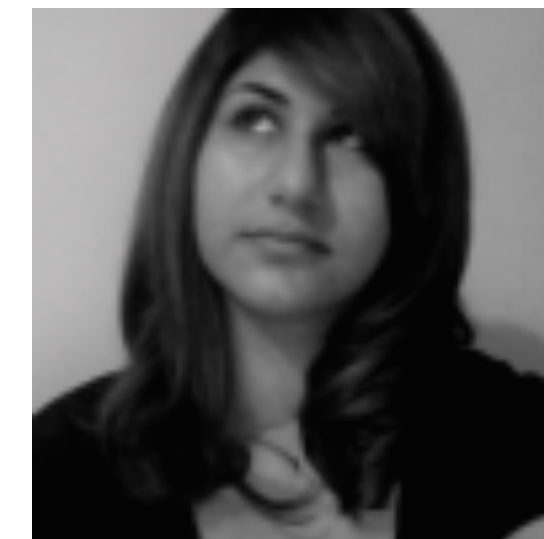
Joseph



Tugdual



Julien



Sarah



Loïc



Jérémy



Romain



FX

Agenda

Assets

Performance

Logging

Legacy

Testing

Config

Parallel

Doctrine

**Value
Objects**

Asset management

Named assets

Typical asset linking

```
{% javascripts
  '@AppBundle/Resources/public/js/jquery.js'
  '@AppBundle/Resources/public/js/jquery.ui.js'
  '@AppBundle/Resources/public/js/bootstrap.js'
  '@AppBundle/Resources/public/js/*' %}
  <script src="{{ asset_url }}"></script>
{% endjavascripts %}
```

Defining named assets

```
# app/config/config.yml
```

```
assetic:
```

```
  assets:
```

```
    # ...
```

```
    bootstrap_js:
```

```
      inputs:
```

- '@AppBundle/Resources/public/js/jquery.js'
- '@AppBundle/Resources/public/js/jquery.ui.js'
- '@AppBundle/Resources/public/js/bootstrap.js'

Using named assets

```
{% javascripts  
  '@bootstrap_js'  
  '@AppBundle/Resources/public/js/*' %}  
  <script src="{{ asset_url }}"></script>  
{% endjavascripts %}
```

Asset packages

asset() adds a leading slash

```

```


asset() adds a leading slash

```

```

A browser window mockup with a light gray header bar containing two square icons and a search bar. The main content area is white and displays the rendered HTML code.

```

```

Easiest way to define a base URL

```
# app/config/config.yml
```

```
framework:
```

```
    templating:
```

```
        assets_base_urls:
```

```
            http: ['http://static.example.com/images']
```

```
            ssl:  ['https://static.example.com/images']
```

Easiest way to define a base URL

```
# app/config/config.yml
```

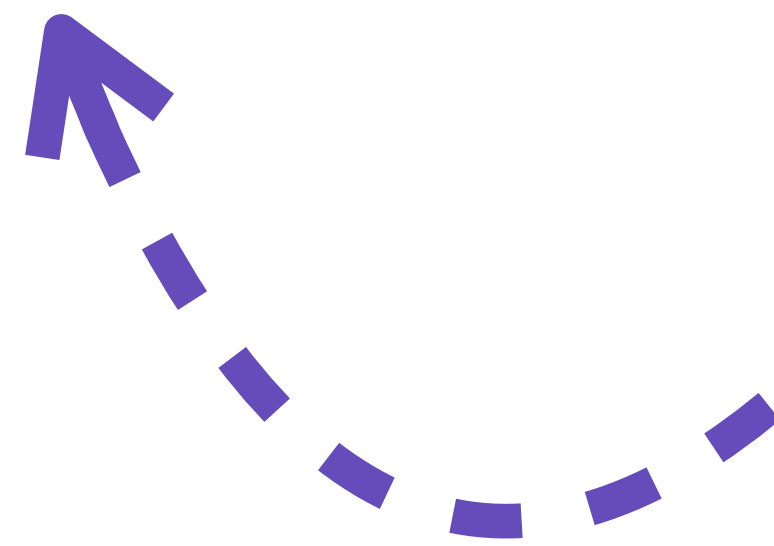
```
framework:
```

```
    templating:
```

```
        assets_base_urls:
```

```
            http: ['http://static.example.com/images']
```

```
            ssl:  ['https://static.example.com/images']
```



**if you configure several
base URLs, Symfony will
select one each time to
balance asset load**

Protocol-relative base URL

```
# app/config/config.yml
```

```
framework:
```

```
    templating:
```

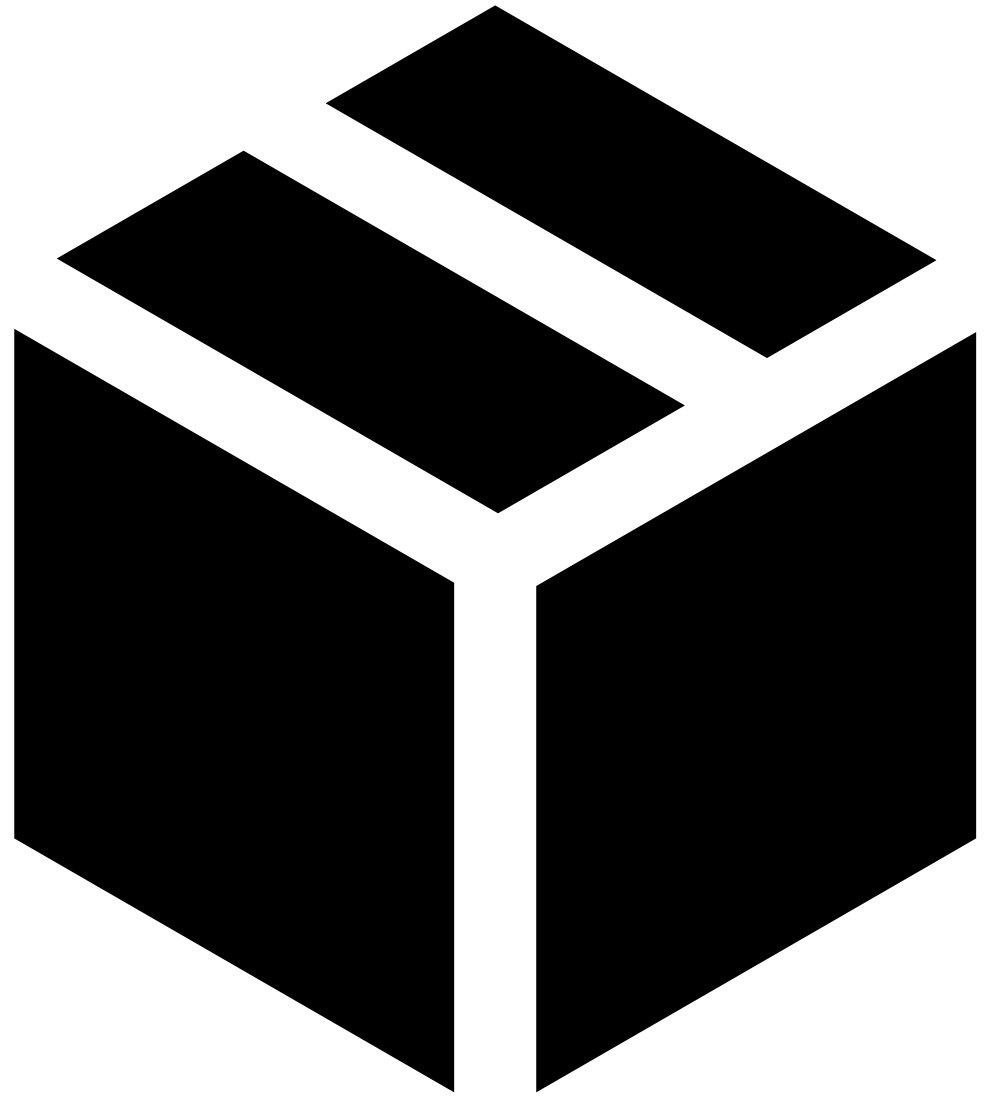
```
        assets_base_urls: '//static.example.com/images'
```

**What if we want to modify
the base URL just for a few
selected assets?**

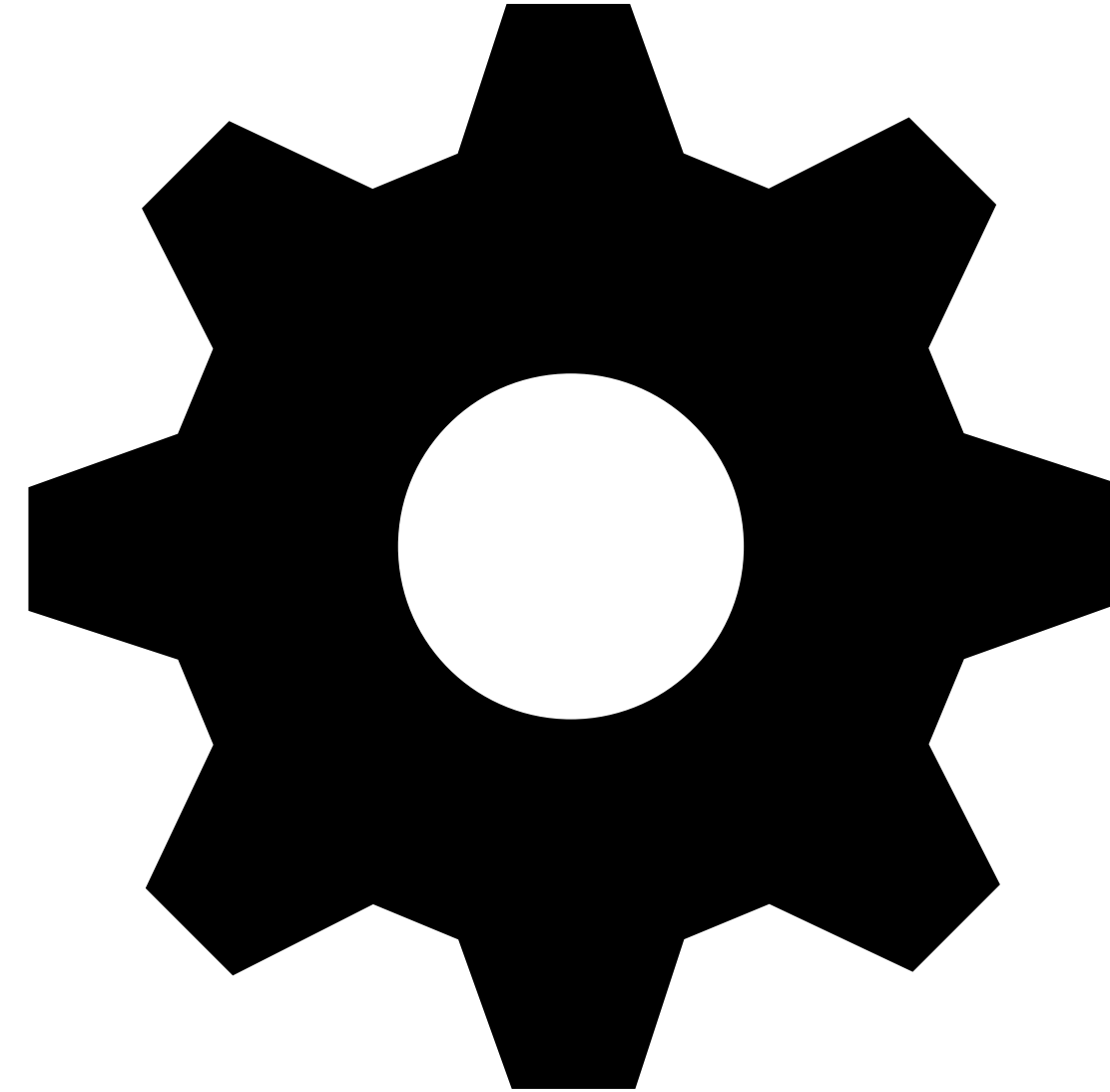
**What if we want to modify
the base URL just for a few
selected assets?**

Asset packages

Asset packages



**Group assets
logically.**



**Define config for
each package.**

Defining asset packages

```
# app/config/config.yml
```

```
framework:
```

```
    templating:
```

```
        packages:
```

```
            avatars:
```

```
                base_urls: ' //static.example.com/avatars/ '
```

Asset packages in practice

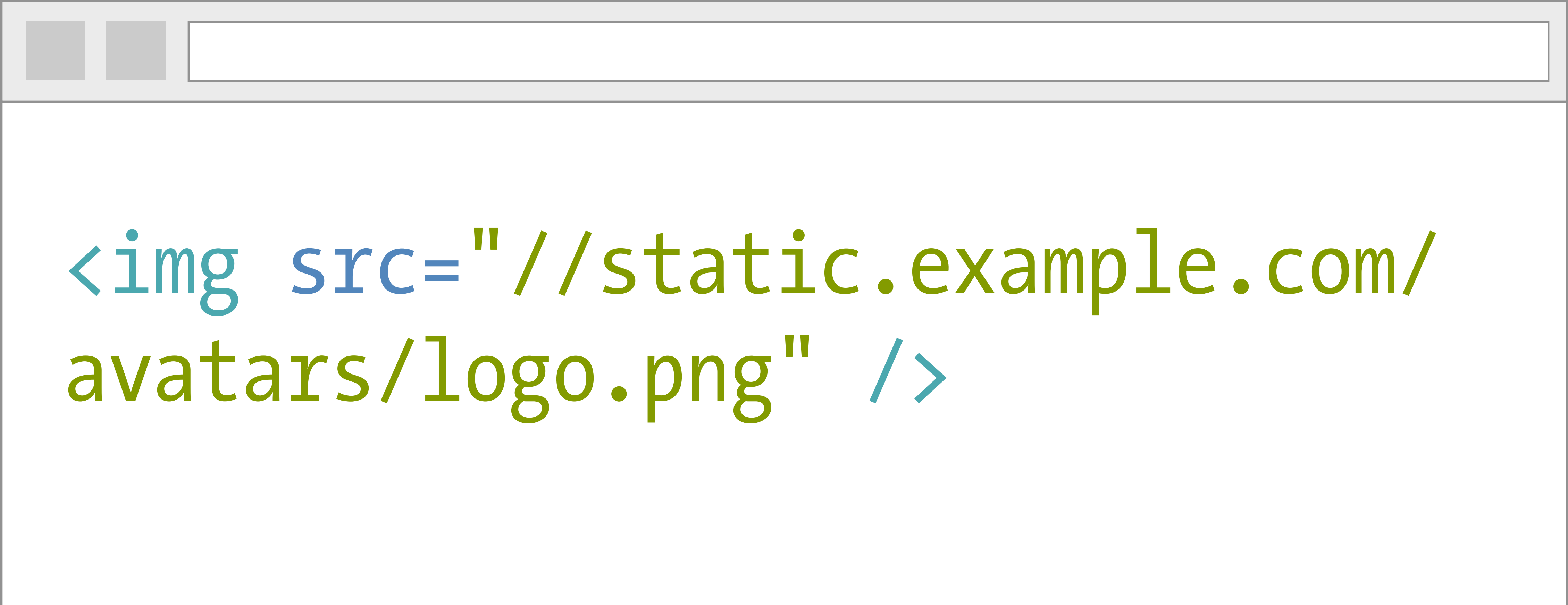
```

```

Asset packages in practice

```

```

A browser window mockup with a light gray header bar containing two small square icons and a search bar. The main content area is white and displays the rendered HTML code for the image tag.

```

```


Packages can define any asset option

```
# app/config/config.yml
```

```
framework:
```

```
    templating:
```

```
        packages:
```

```
            avatars:
```

```
                version: '20141127'
```

```
                base_urls: '//static.example.com/avatars/'
```

mini-tip

Using the Symfony console

```
$ php app/console --env=prod  
  assetic:dump
```

Using the Symfony console

```
$ php app/console --env=prod  
  assetic:dump
```

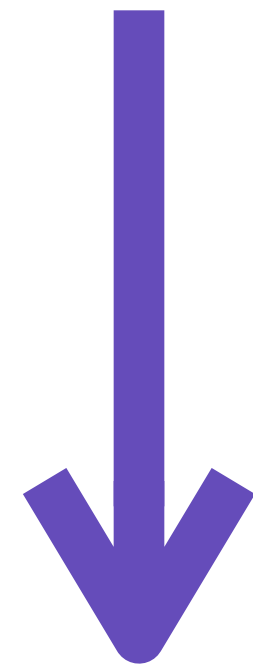


WRONG

RIGHT

Using the Symfony console

```
$ ./app/console --env=prod  
assetic:dump
```

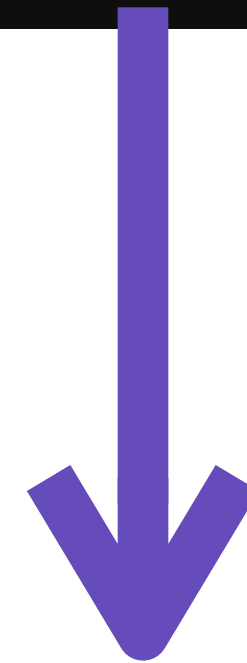


WRONG

RIGHT

Using the Symfony console

```
$ sf --env=prod assetic:dump  
alias sf = php app/console
```



WRONG

RIGHT

Using good Symfony console shortcuts

```
$ dev ...
```

```
$ prod ...
```

```
alias dev = php app/console --env=dev
```

```
alias prod = php app/console --env=prod
```

PROS

```
$ prod a:d
```

```
$ dev r:m /
```

```
$ dev cont:d
```

NEWCOMERS

```
$ php app/console --env=prod assetic:dump
```

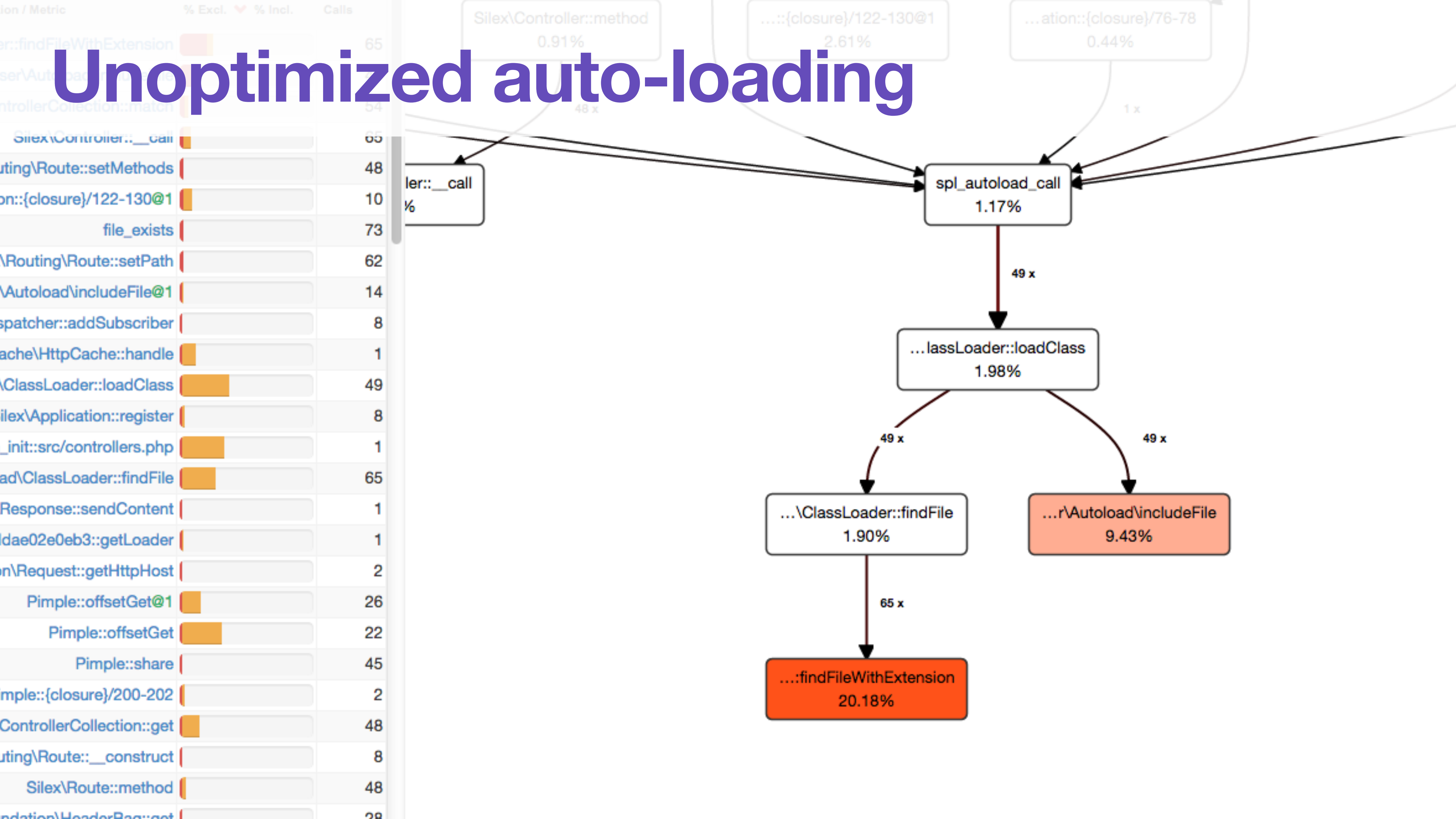
```
$ php app/console router:match /
```

```
$ php app/console container:debug
```

Performance

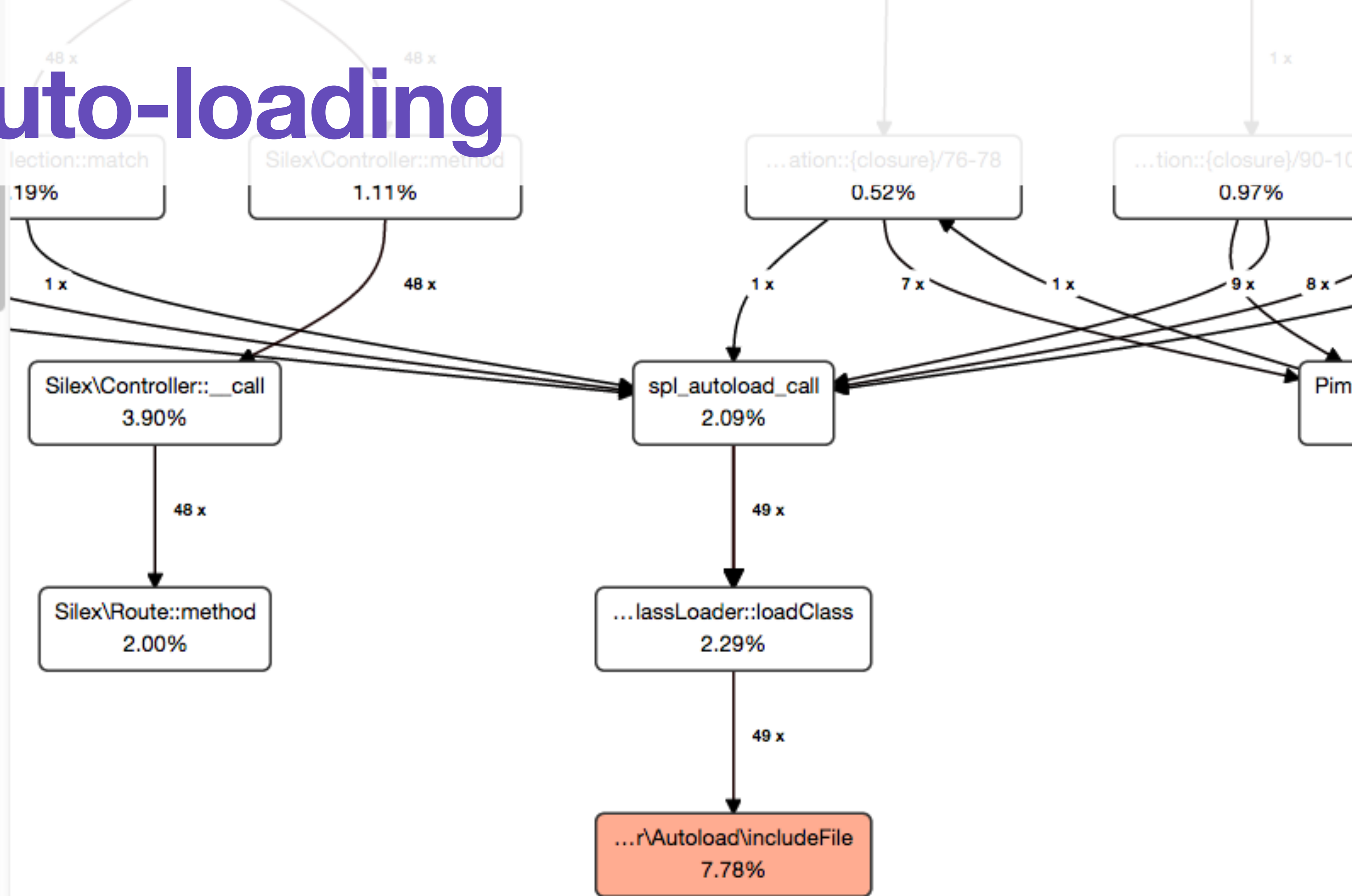
**Optimize
autoloading**

Unoptimized auto-loading



Optimized auto-loading

...poser\Autoload\includeFile	49
...Application::register	8
...ontrol\ClassLoader::loadClass	48
Silex\Controller::__call	65
...outing\Route::setMethods	48
...906c03a98e0f2::getLoader	1
...er/autoload_classmap.php	1
...nt\Routing\Route::setPath	62
...Cache\HttpCache::handle	1
...ad\ClassLoader::loadClass	49
...n\Response::sendContent	1
spl_autoload_call	50
Silex\Route::method	48
Pimple::offsetGet	22
...tion\Request::getHttpHost	2
Pimple::offsetGet@1	26
Pimple::share	45
...oundation\HeaderBag::get	28
...x\ControllerCollection::get	48
...er\Autoload\includeFile@1	14
...outing\Route::__construct	8
...oundation\HeaderBag::set	19
...Dispatcher::addSubscriber	8
run_init::src/controllers.php	1
...oundation\Request::initialize	1
...Cache\HttpCache::lookup	1



```
$ composer dump-autoload --optimize
```

Don't load test classes in production

```
{  
    "autoload": {  
        "psr-4": { "MyLibrary\\": "src/" }  
    },  
    "autoload-dev": {  
        "psr-4": { "MyLibrary\\Tests\\": "tests/" }  
    }  
}
```


**Add classes to
compile**

Three important Symfony files

- **app/bootstrap.php.cache**
Internal Symfony classes.
- **app/cache/<env>/appDevDebugProjectContainer.php**
Compiled container (services and parameters).
- **app/cache/<env>/classes.php**
Symfony bundles classes.

Adding your own classes for compiling

```
namespace AppBundle\DependencyInjection;

use Symfony\Component\Config\FileLocator;
use Symfony\Component\DependencyInjection\ContainerBuilder;
use Symfony\Component\HttpKernel\DependencyInjection\Extension;

class AppExtension extends Extension
{
    public function load(array $configs, ContainerBuilder $container)
    {
        // ...

        $this->addClassesToCompile([
            'AppBundle\\Full\\Namespace\\Class'
        ]);
    }
}
```

Adding your own classes for compiling

```
namespace AppBundle\DependencyInjection;

use Symfony\Component\Config\FileLocator;
use Symfony\Component\DependencyInjection\ContainerBuilder;
use Symfony\Component\HttpKernel\DependencyInjection\Extension;

class AppExtension extends Extension
{
    public function load(array $configs, ContainerBuilder $container)
    {
        // ...

        $this->addClassesToCompile([
            'AppBundle\\Full\\Namespace\\Class'
        ]);
    }
}
```

WARNING

It doesn't work if
the classes define
annotations

mini-tip

**How can you enable the
profiler on production for
some specific users?**

The easy way: restrict by path

```
# app/config/config.yml
```

```
framework:
```

```
    # ...
```

```
    profiler:
```

```
        matcher:
```

```
            path: ^/admin/
```

The right way: restrict by user (1/2)

```
# app/config/config.yml
framework:
    # ...
    profiler:
        matcher:
            service: app.profiler_matcher

services:
    app.profiler_matcher:
        class: AppBundle\Profiler\Matcher
        arguments: ["@security.context"]
```


The right way: restrict by user (2/2)

```
namespace AppBundle\Profiler;

use Symfony\Component\Security\Core\SecurityContext;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\RequestMatcherInterface;

class Matcher implements RequestMatcherInterface
{
    protected $securityContext;

    public function __construct(SecurityContext $securityContext)
    {
        $this->securityContext = $securityContext;
    }

    public function matches(Request $request)
    {
        return $this->securityContext->isGranted('ROLE_ADMIN');
    }
}
```

Better logs with Monolog

**De-clutter
your logs**

Symfony logs a lot of information

```
[2014-11-24 12:24:22] request.INFO: Matched route "homepage" (parameters: "_controller": "AppBundle\Controller\DefaultController::indexAction", "_route": "homepage") [] []
[2014-11-24 12:24:22] security.INFO: Populated SecurityContext with an anonymous Token [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\DebugHandlersListener::configure". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Bundle\FrameworkBundle\EventListener\SessionListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\FragmentListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\RoutingListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\LocaleListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\Security\Http\Firewall::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Bundle\AsseticBundle\EventListener\RequestListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Symfony\Bundle\FrameworkBundle\DataCollector\RoutingDataCollector::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Symfony\Component\HttpKernel\DataCollector\RequestDataCollector::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\ControllerListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\ParamConverterListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\HttpCacheListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\SecurityListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\TemplateListener::onKernelController". [] []
[2014-11-24 12:24:22] doctrine.DEBUG: SELECT p0_.id AS id0, p0_.title AS title1, p0_.slug AS slug2, p0_.summary AS summary3, p0_.content AS content4, p0_.authorEmail AS authorEmail5,
p0_.publishedAt AS publishedAt6 FROM Post p0_ WHERE p0_.publishedAt <= ? ORDER BY p0_.publishedAt DESC LIMIT 10 ["2014-11-24 12:24:22"] []
[2014-11-24 12:24:22] security.DEBUG: Write SecurityContext in the session [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\Security\Http\Firewall\ContextListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\ResponseListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\Security\Http\RememberMe\ResponseListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\HttpCacheListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Bundle\WebProfilerBundle\EventListener\WebDebugToolbarListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\StreamedResponseListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\HttpKernel\EventListener\LocaleListener::onKernelFinishRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\HttpKernel\EventListener\RoutingListener::onKernelFinishRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\Security\Http\Firewall::onKernelFinishRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.terminate" to listener "Symfony\Bundle\SwiftmailerBundle\EventListener\EmailSenderListener::onTerminate". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.terminate" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelTerminate". [] []
```

Is this useful for your application?

```
[2014-11-24 12:24:22] request.INFO: Matched route "homepage" (parameters: "_controller": "AppBundle\Controller\DefaultController::indexAction", "_route": "homepage") [] []
[2014-11-24 12:24:22] security.INFO: Populated SecurityContext with an anonymous Token [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\DebugHandlersListener::configure". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Bundle\FrameworkBundle\EventListener\SessionListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\FragmentListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\RoutingListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\HttpKernel\EventListener\LocaleListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Component\Security\Http\Firewall::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Bundle\AsseticBundle\EventListener\RequestListener::onKernelRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Symfony\Bundle\FrameworkBundle\DataCollector\RoutingDataCollector::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Symfony\Component\HttpKernel\DataCollector\RequestDataCollector::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkBundle\EventListener\ControllerListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkBundle\EventListener\ParamConverterListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkBundle\EventListener\HttpCacheListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkBundle\EventListener\SecurityListener::onKernelController". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.controller" to listener "Sensio\Bundle\FrameworkBundle\EventListener\TemplateListener::onKernelController". [] []
[2014-11-24 12:24:22] doctrine.DEBUG: SELECT p0_.id AS id0, p0_.title AS title1, p0_.slug AS slug2, p0_.summary AS summary3, p0_.content AS content4, p0_.authorEmail AS authorEmail5,
p0_.publishedAt AS publishedAt6 FROM Post p0_ WHERE p0_.publishedAt <= ? ORDER BY p0_.publishedAt DESC LIMIT 10 ["2014-11-24 12:24:22"] []
[2014-11-24 12:24:22] security.DEBUG: Write SecurityContext in the session [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\Security\Http\Firewall\ContextListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\ResponseListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\Security\Http\RememberMe\ResponseListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Sensio\Bundle\FrameworkBundle\EventListener\HttpCacheListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Bundle\WebProfilerBundle\EventListener\WebDebugToolbarListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\StreamedResponseListener::onKernelResponse". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\HttpKernel\EventListener\LocaleListener::onKernelFinishRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\HttpKernel\EventListener\RoutingListener::onKernelFinishRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\Security\Http\Firewall::onKernelFinishRequest". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.terminate" to listener "Symfony\Bundle\SwiftmailerBundle\EventListener\EmailSenderListener::onTerminate". [] []
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.terminate" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelTerminate". [] []
```


Is this useful for your application?

```
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener "Symfony\Bundle\FrameworkBundle\EventListener\SessionListener::onKernelRequest". [] []
```

```
[2014-11-24 12:24:22] doctrine.DEBUG: SELECT p0_.id AS id0, p0_.title AS title1, p0_.slug AS slug2, p0_.summary AS summary3, p0_.content AS content4, p0_.authorEmail AS authorEmail5, p0_.publishedAt AS publishedAt6 FROM Post p0_ WHERE p0_.publishedAt <= ? ORDER BY p0_.publishedAt DESC LIMIT 10 ["2014-11-24 12:24:22"] []  
[2014-11-24 12:24:22] security.DEBUG: Write SecurityContext in the session [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\Security\Http\Firewall\ContextListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\ResponseListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\Security\Http\RememberMe\ResponseListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Sensio\Bundle\FrameworkExtraBundle\EventListener\HttpCacheListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Bundle\WebProfilerBundle\EventListener\WebDebugToolbarListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.response" to listener "Symfony\Component\HttpKernel\EventListener\StreamedResponseListener::onKernelResponse". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\HttpKernel\EventListener\LocaleListener::onKernelFinishRequest". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\HttpKernel\EventListener RouterListener::onKernelFinishRequest". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.finish_request" to listener "Symfony\Component\Security\Http\Firewall::onKernelFinishRequest". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.terminate" to listener "Symfony\Bundle\SwiftmailerBundle\EventListener\EmailSenderListener::onTerminate". [] []  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.terminate" to listener "Symfony\Component\HttpKernel\EventListener\ProfilerListener::onKernelTerminate". [] []
```

Is this useful for your application?

```
[2014-11-24 12:24:22] request.INFO: Matched route "homepage" (parameters: ...  
[2014-11-24 12:24:22] security.INFO: Populated SecurityContext with an anonymous ...  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener ...  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener ...  
[2014-11-24 12:24:22] event.DEBUG: Notified event "kernel.request" to listener ...
```

Is this useful for your application?

```
[2014-11-24 12:24:22] request.INFO: Matched route "homepage" (parameters: ...  
[2014-11-24 12:24:22] security.INFO: Populated SecurityContext with an anonymous ...  
[2014-11-24 12:24:22] event DEBUG: Notified event "kernel.request" to listener ...  
[2014-11-24 12:24:22] event DEBUG: Notified event "kernel.request" to listener ...  
[2014-11-24 12:24:22] event DEBUG: Notified event "kernel.request" to listener ...
```

log channel

Hide event logs (if you don't need them)

```
# app/config/dev.yml
```

```
monolog:
```

```
  handlers:
```

```
    main:
```

```
      type:    stream
```

```
      path:    "%kernel.logs_dir%/%kernel.environment%.log"
```

```
      level:   debug
```

```
      channels: "!event"
```

De-cluttered log files

```
[2014-11-24 12:24:22] request.INFO: Matched route "homepage" (parameters: "_controller": "AppBundle\Controller\DefaultController::indexAction", "_route": "homepage")
```

```
[2014-11-24 12:24:22] security.INFO: Populated SecurityContext with an anonymous Token
```

```
[2014-11-24 12:24:22] doctrine.DEBUG: SELECT p0_.id AS id0, p0_.title AS title1, p0_.slug AS slug2, p0_.summary AS summary3, p0_.content AS content4, p0_.authorEmail AS authorEmail5, p0_.publishedAt AS publishedAt6 FROM Post p0_ WHERE p0_.publishedAt <= ? ORDER BY p0_.publishedAt DESC LIMIT 10 ["2014-11-24 12:24:22"]
```

```
[2014-11-24 12:24:22] security.DEBUG: Write SecurityContext in the session
```

**Better log
emails**

Email logs related to 5xx errors

```
# app/config/config_prod.yml
```

```
monolog:
```

```
  handlers:
```

```
    mail:
```

```
      type: fingers_crossed
```

```
      action_level: critical
```

```
      handler: buffered
```

```
  buffered:
```

```
    type: buffer
```

```
    handler: swift
```

```
  swift:
```

```
    type: swift_mailer
```

```
    from_email: error@example.com
```

```
    to_email: error@example.com
```

```
    subject: An Error Occurred!
```

```
    level: debug
```

Email logs related to 5xx errors

```
# app/config/config_prod.yml
monolog:
  handlers:
    mail:
      type: fingers_crossed
      action_level: critical
      handler: buffered
    buffered:
      type: buffer
      handler: swift
    swift:
      type: swift_mailer
      from_email: error@example.com
      to_email: error@example.com
      subject: An Error Occurred!
      level: debug
```

Email logs related to 4xx errors

```
# app/config/config_prod.yml
monolog:
  handlers:
    mail:
      type: fingers_crossed
      action_level: error
      handler: buffered
  buffered:
    # ...
  swift:
    # ...
```

Don't email 404 errors

```
# app/config/config_prod.yml
monolog:
  handlers:
    mail:
      type: fingers_crossed
      action_level: error
      excluded_404:
        - ^/
      handler: buffered
  buffered:
    # ...
  swift:
    # ...
```

Organize your logs

Use different log files for each channel

```
# app/config/config_prod.yml
```

```
monolog:
```

```
  handlers:
```

```
    main:
```

```
      type:      stream
```

```
      path:      "%kernel.logs_dir%/%kernel.environment%.log"
```

```
      level:     debug
```

```
      channels:  ["!event"]
```

```
  security:
```

```
    type:      stream
```

```
    path:      "%kernel.logs_dir%/security-%kernel.environment%.log"
```

```
    level:     debug
```

```
    channels:  "security"
```

Format your log messages

Don't do this

```
$logger->info(sprintf(  
    "Order number %s processed for %s user",  
    $number, $user  
));
```

Do this instead

```
$logger->info(  
    "Order number {num} processed for {name} user",  
    ['num' => $number, 'name' => $user]  
);
```

Do this instead

```
$logger->info(  
    "Order number {num} processed for {name} user",  
    ['num' => $number, 'name' => $user]  
);
```

WARNING

It doesn't work unless the PSR log processor is enabled.

Enable the PsrLogMessageProcessor

```
# app/config/config_prod.yml
```

```
services:
```

```
    monolog_processor:
```

```
        class: Monolog\Processor\PsrLogMessageProcessor
```

```
    tags:
```

```
        - { name: monolog.processor }
```

Monolog includes several processors

```
# app/config/config_prod.yml
```

```
services:
```

```
    monolog_processor:
```

```
        class: Monolog\Processor\IntrospectionProcessor
```

```
        tags:
```

```
            - { name: monolog.processor }
```

Create your own processor

```
$logger->info(  
    "Order processed successfully.", $order  
);
```


Create your own processor

```
$logger->info(  
    "Order processed successfully.", $order  
);
```

```
[2014-11-24 15:15:58] app.INFO: Order processed  
successfully. { "order": { "number": "023924382982",  
"user": "John Smith", "total": "34.99", "status": "PAID" }}
```

Custom monolog processor

```
namespace AppBundle\Logger\OrderProcessor;
```

```
use AppBundle\Entity\Order;
```

```
class OrderProcessor
```

```
{
```

```
    public function __invoke(array $record)
```

```
    {
```

```
        if (isset($record['context']['order']) && $record['context']['order'] instanceof Order) {
```

```
            $order = $record['context']['order'];
```

```
            $record['context']['order'] = [
```

```
                'number' => $order->getNumber(),
```

```
                'user'   => $order->get...,
```

```
                'total'  => $order->get...,
```

```
                'status' => $order->get...,
```

```
            ];
```

```
        }
```

```
        return $record;
```

```
    }
```

```
# app/config/config_prod.yml
```

```
services:
```

```
    monolog_processor:
```

```
        class: AppBundle\Logger\OrderProcessor
```

```
        tags:
```

```
            - { name: monolog.processor }
```

mini-tip

Medium-sized and Large Applications



Medium-sized and Small Applications

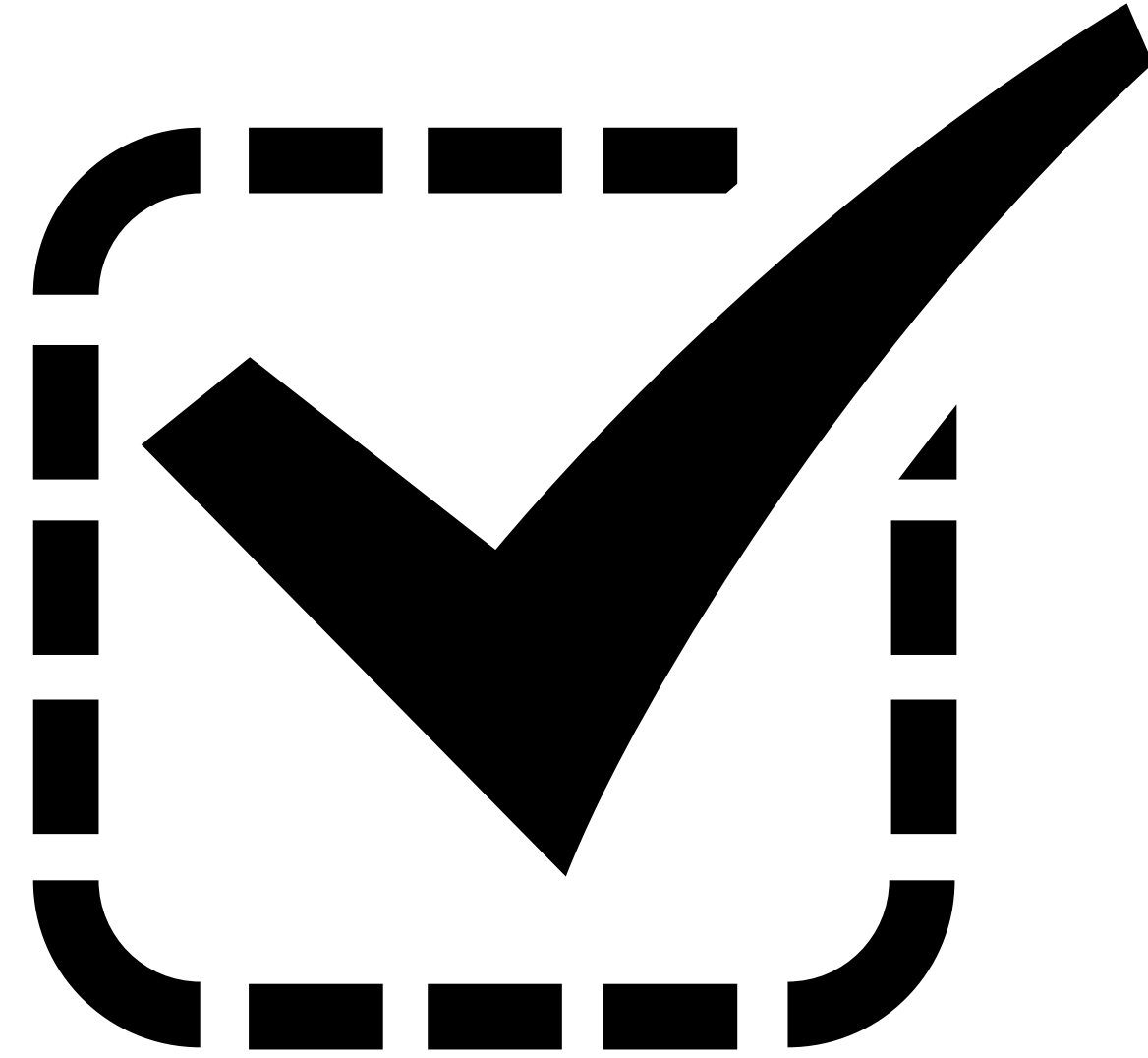
Using your own mailer is OK.

Not all emails should be treated equally



Sign Ups

Lost password



Notifications

Newsletters

Spam

Instant and delayed mailers

```
# app/config/config.yml
```

```
swiftmailer:
```

```
    default_mailer: delayed
```

```
    mailers:
```

```
        instant:
```

```
            # ...
```

```
            spool: ~
```

```
        delayed:
```

```
            # ...
```

```
            spool:
```

```
                type: file
```

```
                path: "%kernel.cache_dir%/swiftmailer/spool"
```

Using prioritized mailers

```
// high-priority emails
```

```
$container->get('swiftmailer.mailer.instant')->...
```

```
// regular emails
```

```
$container->get('swiftmailer.mailer')->...
```

```
$container->get('swiftmailer.mailer.delayed')->...
```

Legacy applications

**Pokemon™
strategy
(catch'em all)**

Controller that captures legacy URLs

```
class LegacyController extends Controller
{
    /**
     * @Route("/{filename}.php", name="_legacy")
     */
    public function legacyAction($filename)
    {
        $legacyPath = $this->container->getParameter('legacy.path');

        ob_start();
        chdir($legacyAppPath);
        require_once $legacyAppPath.$filename.'.php';
        $content = ob_get_clean();

        return new Response($content);
    }
}
```

Source: <http://cvuorinen.net/slides/symfony-legacy-integration/>

**Embedding
legacy apps**

**App
Kernel**

**Router
Listener**

LegacyKernel Listener

LegacyKernel

Legacy App

↓ Request

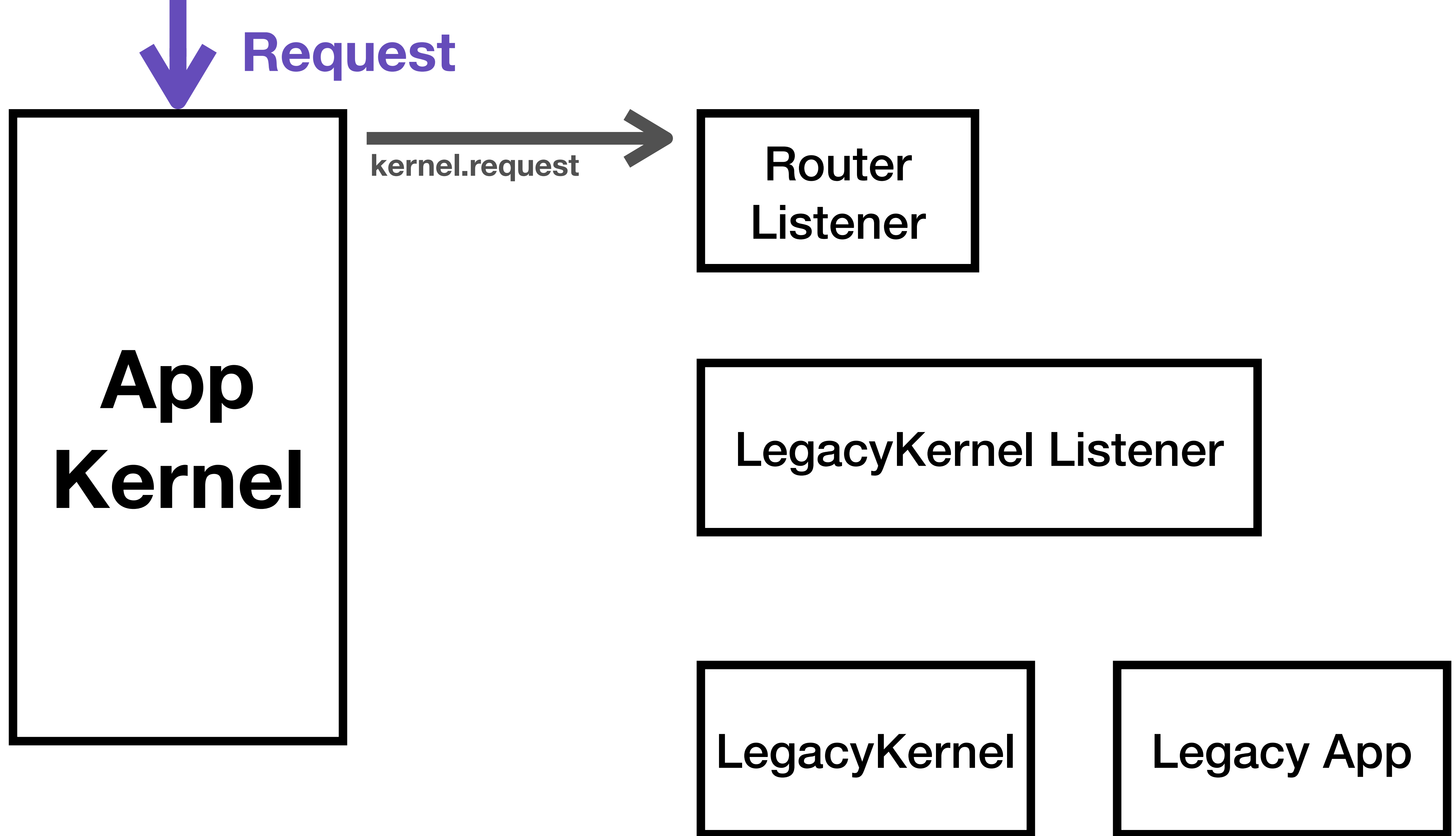
**App
Kernel**

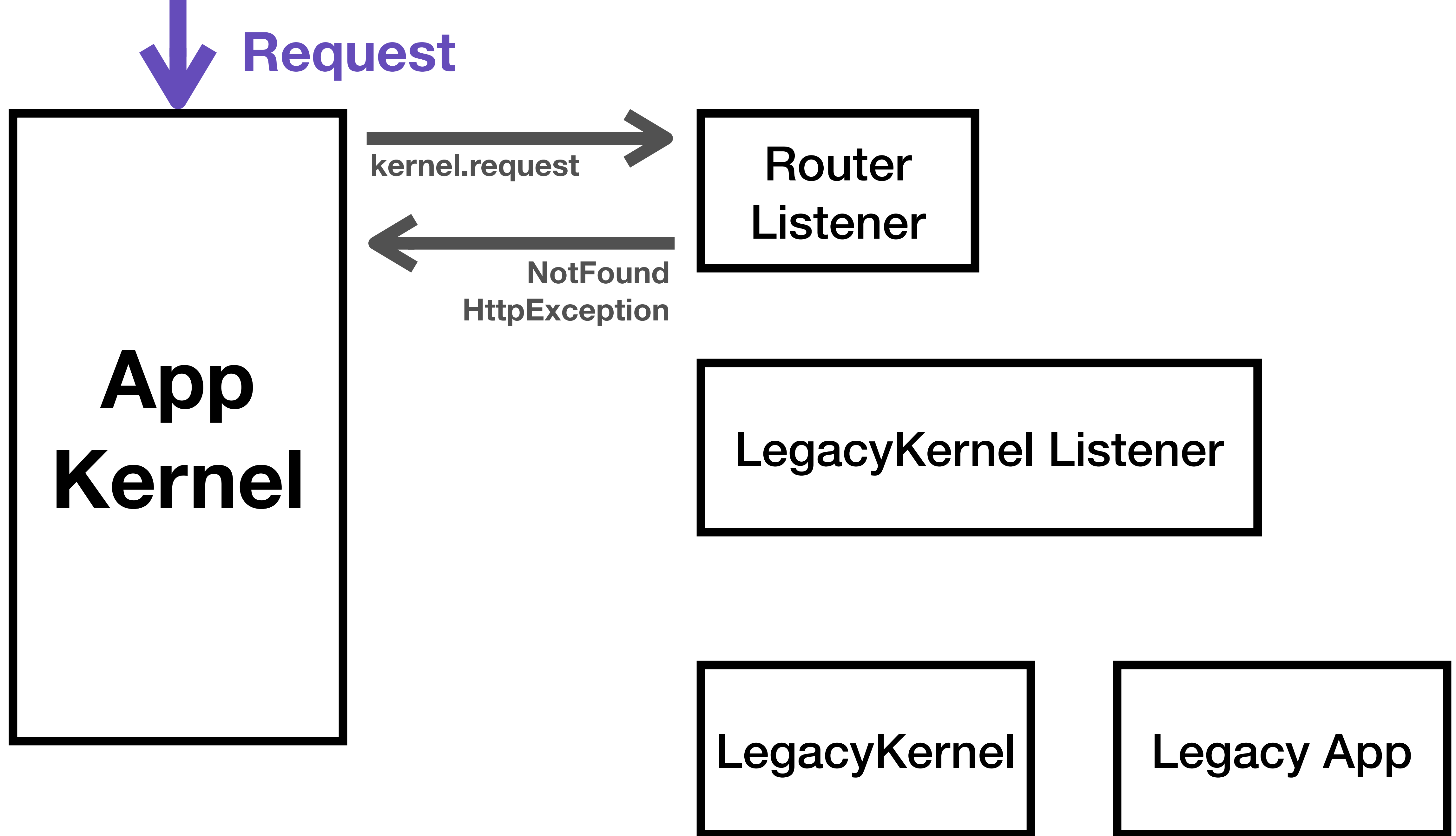
Router
Listener

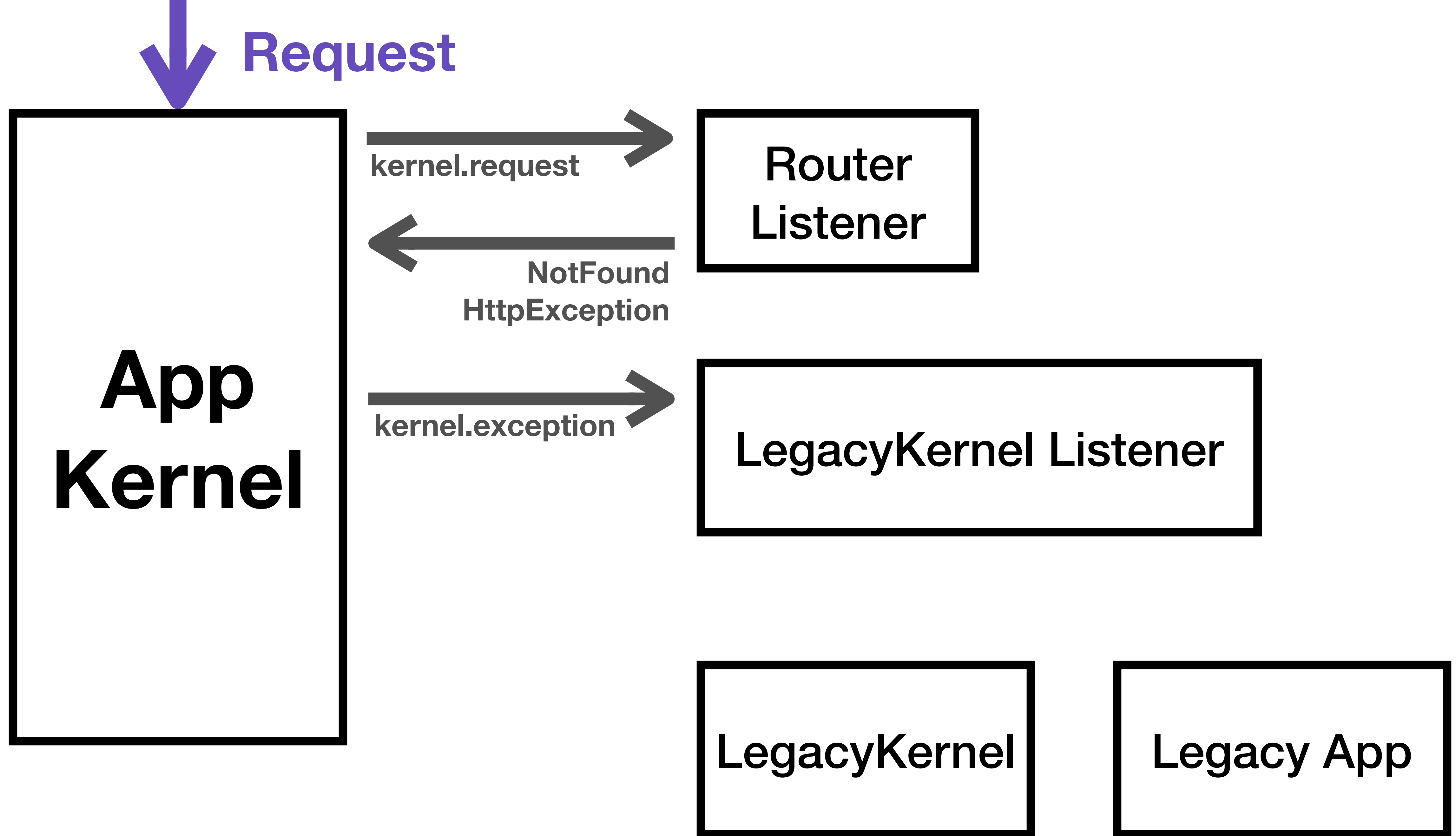
LegacyKernel Listener

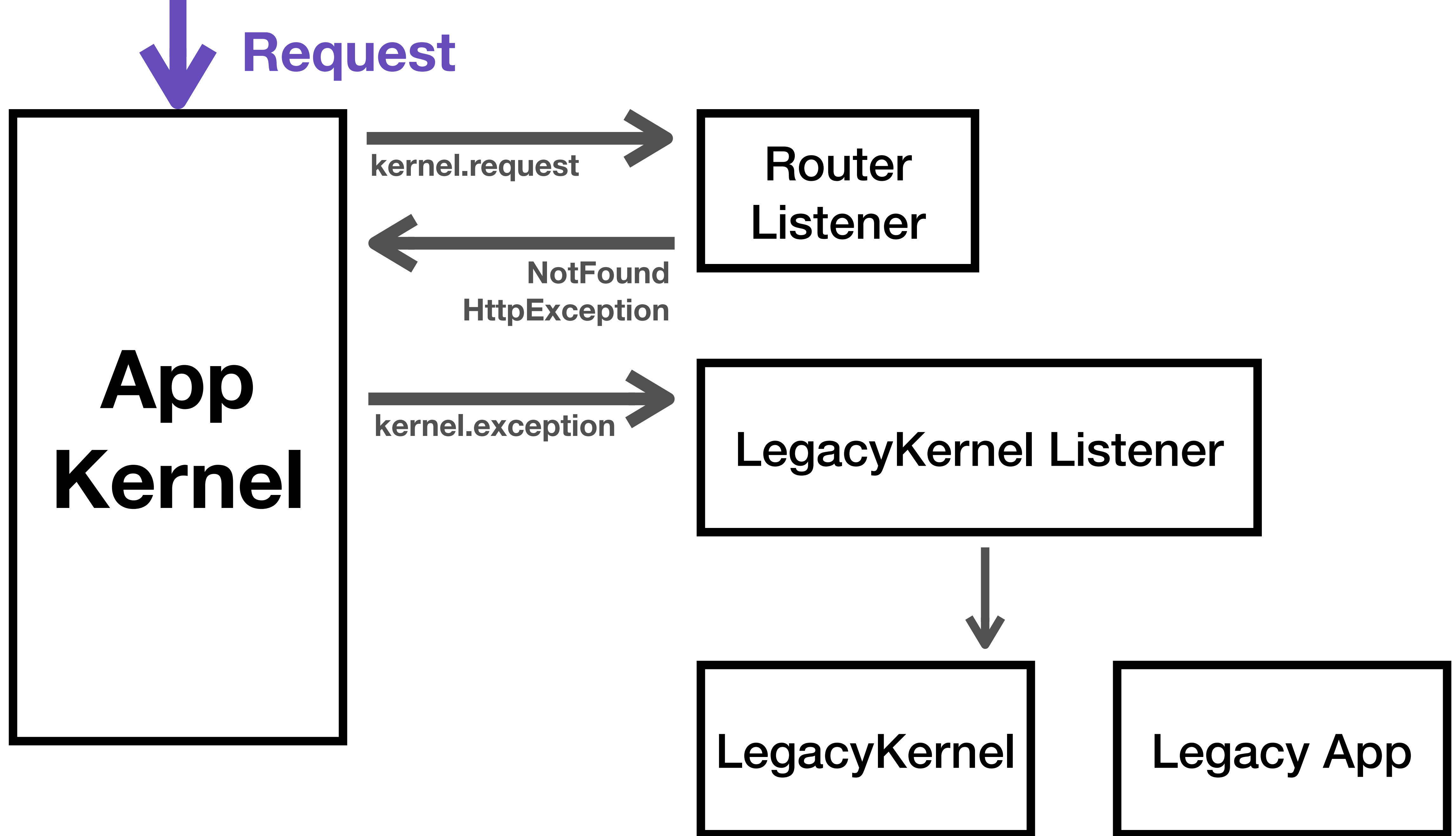
LegacyKernel

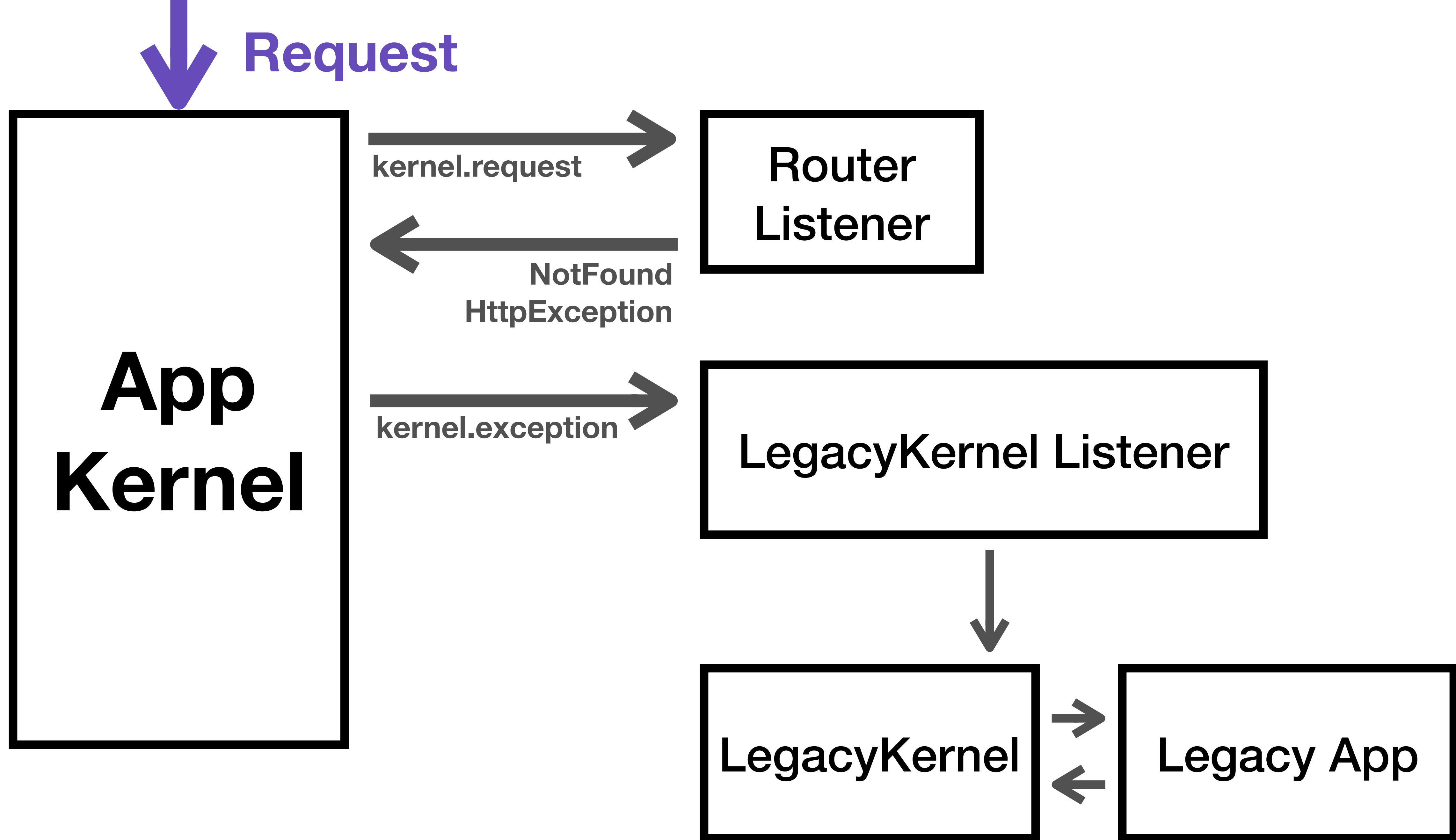
Legacy App

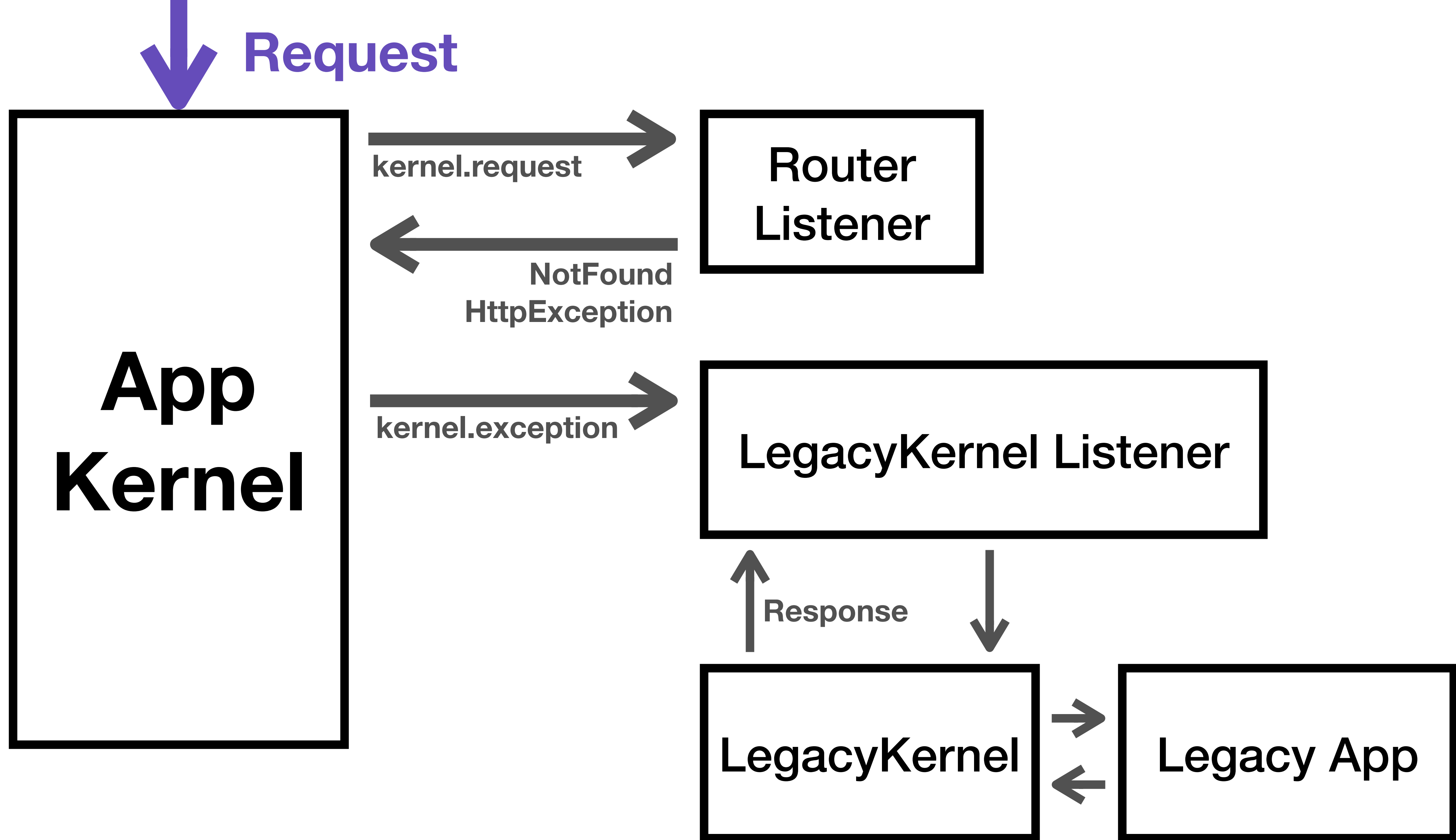


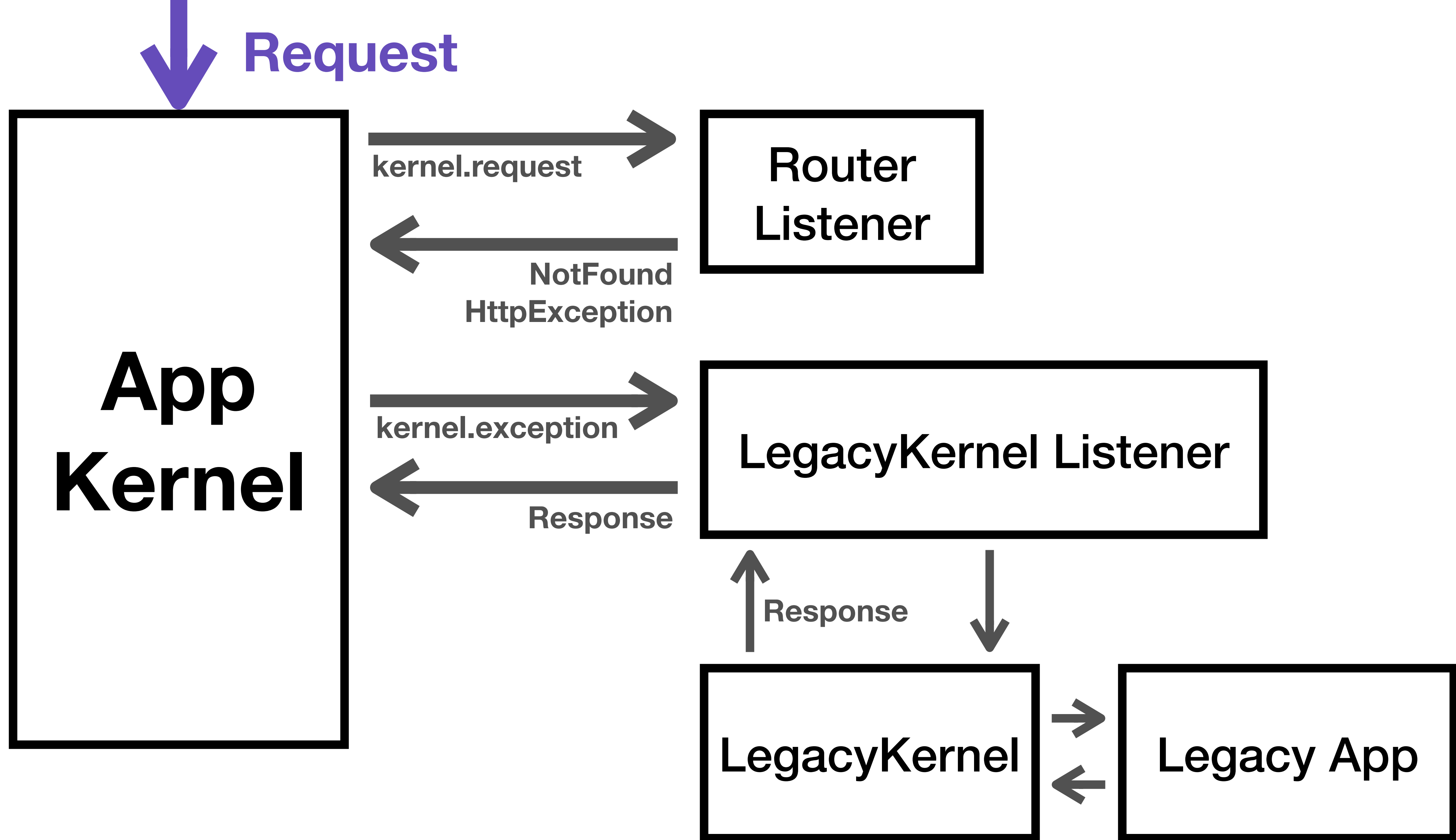


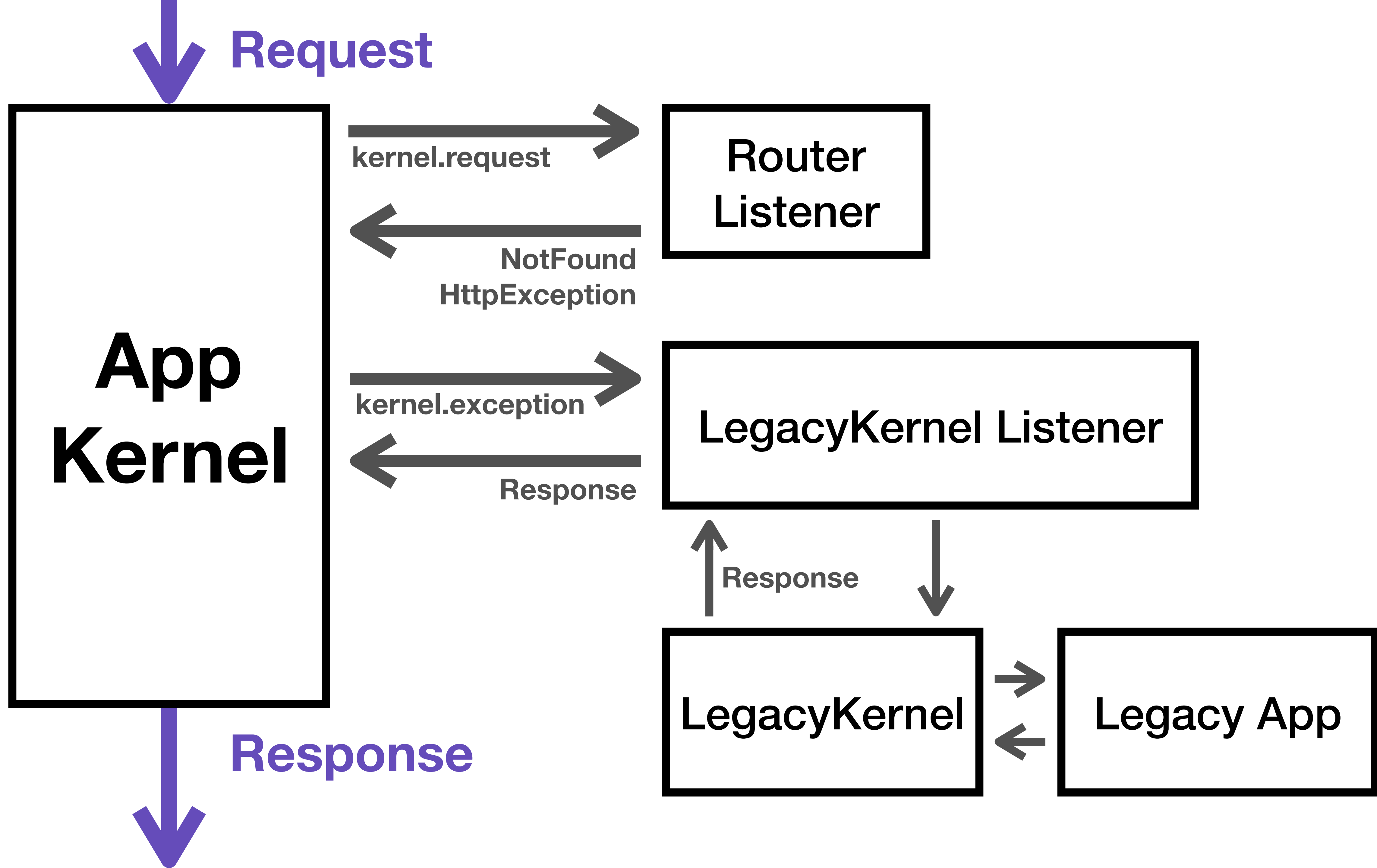












LegacyKernel to process requests

```
class LegacyKernel implements HttpKernelInterface
{
    public function handle(Request $request, ...)
    {
        ob_start();

        $legacyDir = dirname($this->legacyAppPath);
        chdir($legacyDir);
        require_once $this->legacyAppPath;

        $content = ob_get_clean();

        return new Response($content);
    }
}
```

Source: <http://cvuorinen.net/slides/symfony-legacy-integration/>

Listener to treat 404 as legacy requests

```
class LegacyKernelListener implements EventSubscriberInterface
{
    public function onKernelException($event)
    {
        $exception = $event->getException();

        if ($exception instanceof NotFoundHttpException) {
            $request = $event->getRequest();
            $response = $this->legacyKernel->handle($request);

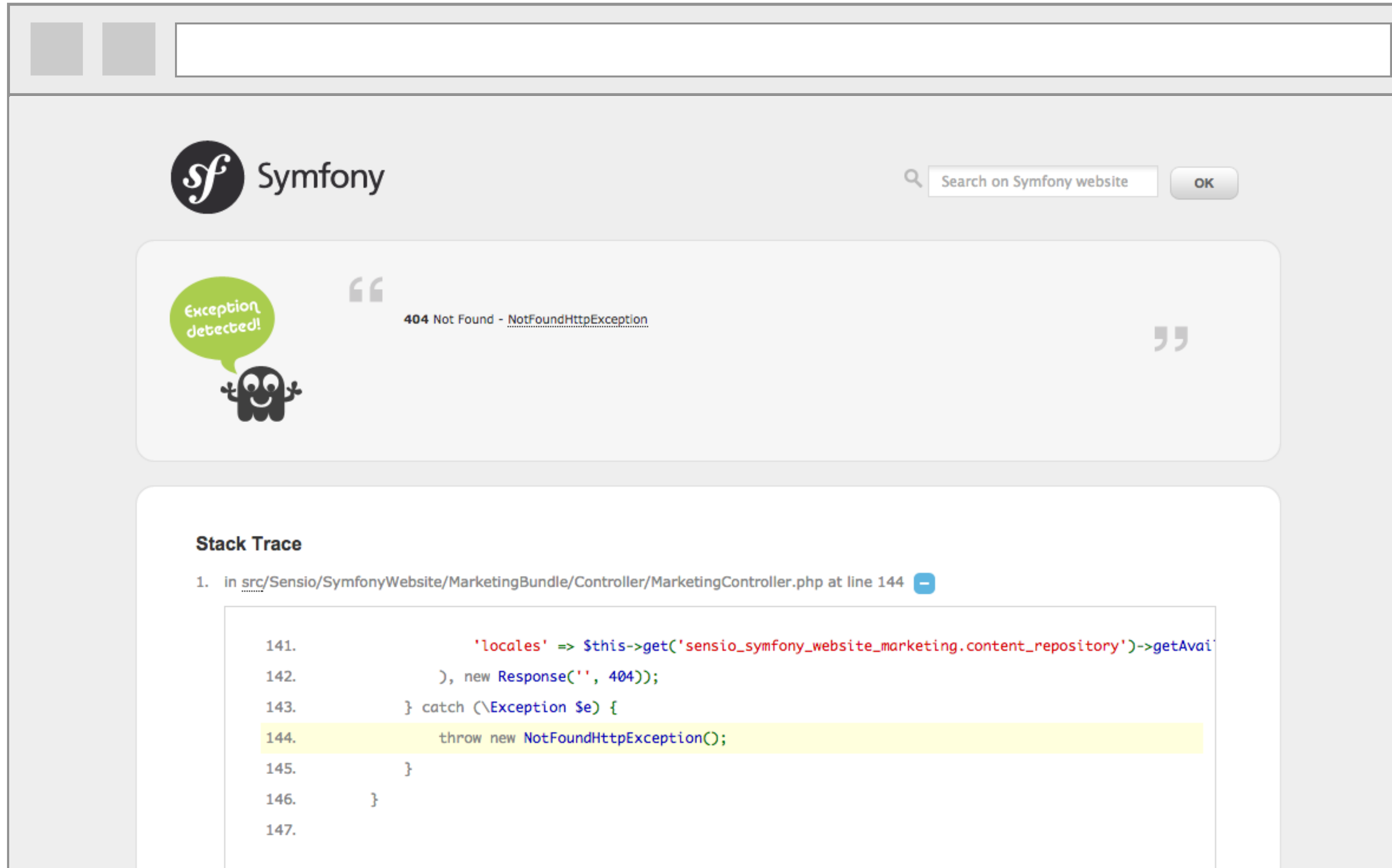
            $response->headers->set('X-Status-Code', 200);

            $event->setResponse($response);
        }
    }
}
```

Source: <http://cvuorinen.net/slides/symfony-legacy-integration/>

mini-tip

Click-to-file in Symfony Exceptions



Textmate



PHPStorm is now supported natively

```
# app/config/config.yml
```

```
framework:
```

```
    ide: "phpstorm://open?file=%%f&line=%%l"
```



PhpStorm

Testing tips

Prophecy

**Prophecy is a highly
opinionated yet very
powerful and flexible PHP
object mocking framework.**

**PHPUnit has built-in
Prophecy mocks support
since 4.5 version.**

A complete Prophecy mock

```
class SubjectTest extends PHPUnit_Framework_TestCase
{
    public function testObserversAreUpdated()
    {
        $subject = new Subject('My subject');

        $observer = $this->prophesize('Observer');
        $observer->update('something')->shouldBeCalled();

        $subject->attach($observer->reveal());

        $subject->doSomething();
    }
}
```

Prophecy vs traditional mocks (1/2)

Prophecy

```
$observer = $this->prophesize('Observer');
```

PHPUnit

```
$observer = $this->getMockBuilder('Observer')  
    ->setMethods(array('update'))  
    ->getMock();
```


Prophecy vs traditional mocks (2/2)

Prophecy

```
$observer->update('something')->shouldBeCalled();
```

PHPUnit

```
$observer->expects($this->once())  
    ->method('update')  
    ->with($this->equalTo('something'));
```

Faster smoke testing

Smoke testing is preliminary testing to reveal simple failures severe enough to reject a prospective software release.

Unnecessarily slow tests

```
namespace AppBundle\Tests\Controller;

use Symfony\Bundle\FrameworkBundle\Test\WebTestCase;

class DefaultControllerTest extends WebTestCase
{
    /** @dataProvider provideUrls */
    public function testPageIsSuccessful($url)
    {
        $client = self::createClient();
        $client->request('GET', $url);
        $this->assertTrue($client->getResponse()->isSuccessful());
    }

    public function provideUrls()
    {
        // ...
    }
}
```

Functional tests without WebTestCase

```
class SmokeTest extends \PHPUnit_Framework_TestCase
{
    private $app;

    protected function setUp()
    {
        $this->app = new \AppKernel('test', false);
        $this->app->boot();
    }

    /** @dataProvider provideUrls */
    public function testUrl($url)
    {
        $request = new Request::create($url, 'GET');
        $response = $this->app->handle($request);

        $this->assertTrue($response->isSuccessful());
    }
}
```

Source: <http://gnugat.github.io/2014/11/15/sf2-quick-functional-tests.html>

mini-tip

Impersonating users

```
# app/config/security.yml
security:
  firewalls:
    main:
      # ...
      switch_user: true
```

A browser window with a light gray border. The address bar contains the URL "http://example.com/path?_switch_user=fabien". To the left of the address bar are two small gray square icons.

http://example.com/path?_switch_user=fabien

WARNING

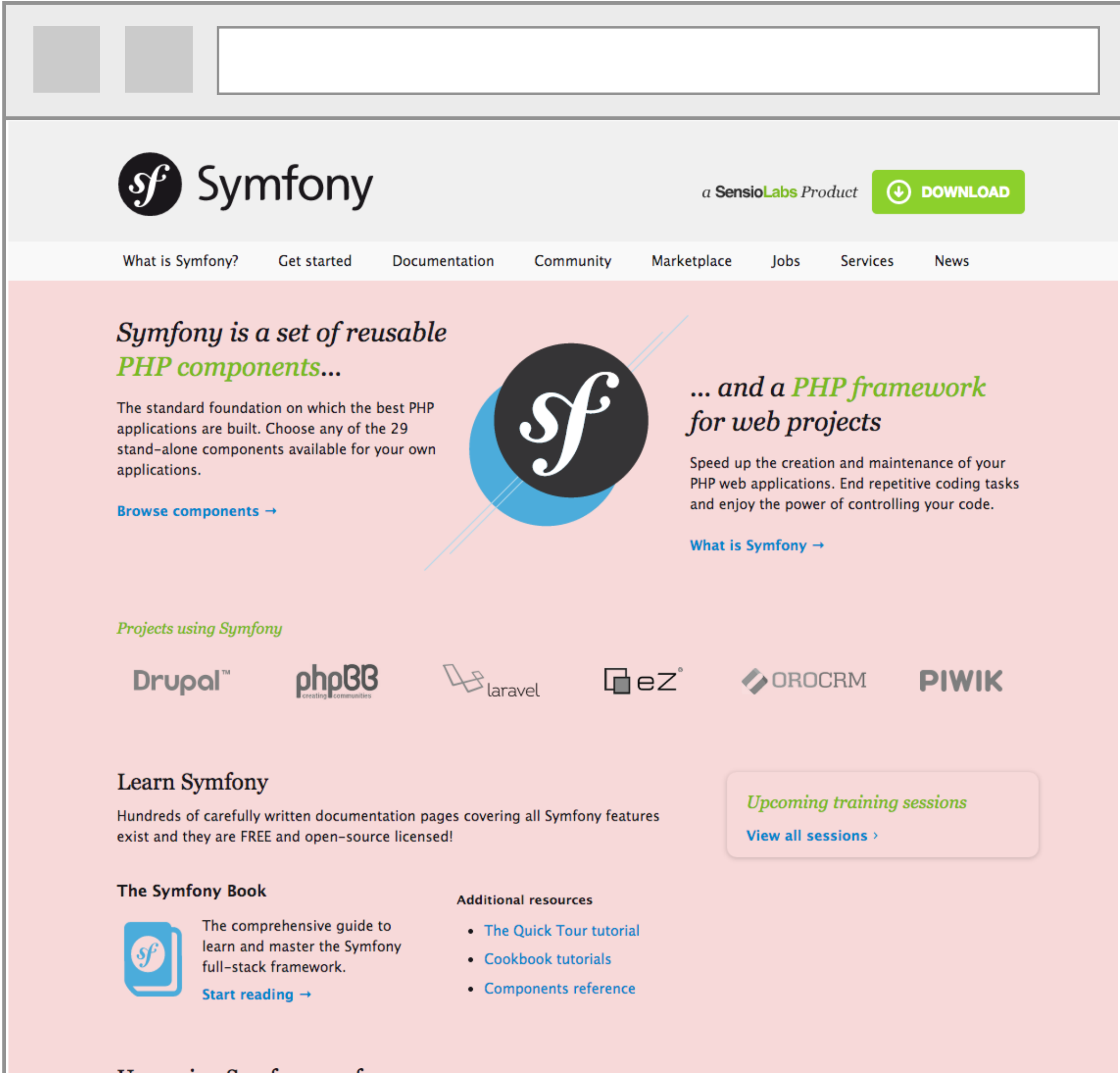
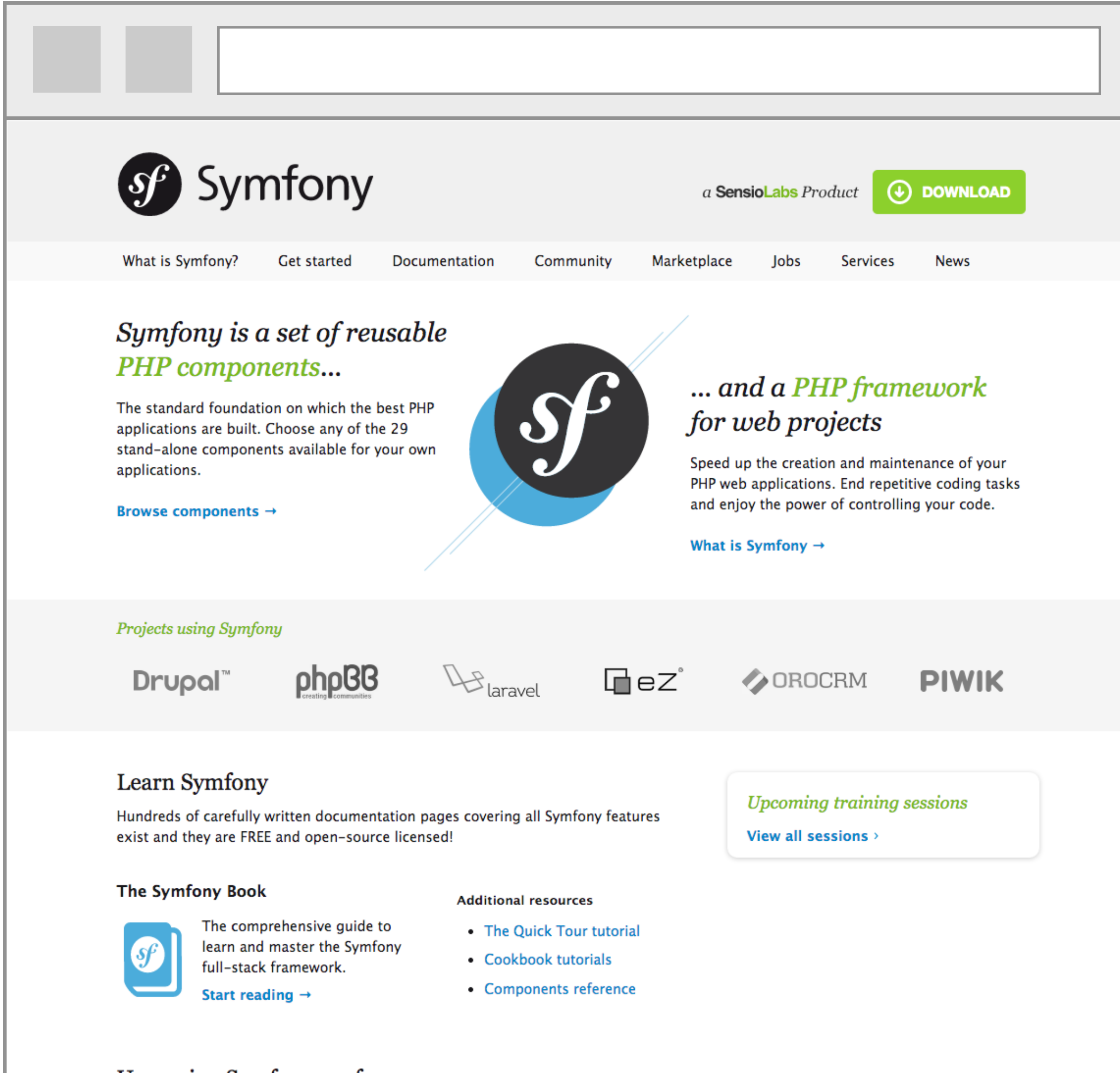
Impersonating is very dangerous and it can lead to severe errors.

Visual hints when impersonating users

```
<body class="
{% if app.user and is_granted('ROLE_PREVIOUS_ADMIN') %}
    impersonating
{% endif %}
">
```

```
.impersonating {
    background: rgba(204, 0, 0, 0.15);
}
```

Be careful when impersonating users



Configuration

Yaml casting

Implicit YAML casting

key:

value1: 5

value2: 2.7

value3: 2014-11-27

Implicit YAML casting

```
key:
  value1: 5
  value2: 2.7
  value3: 2014-11-27

array(1) {
  ["key"]=> array(3) {
    ["value1"] => int(5)
    ["value2"] => float(2.7)
    ["value3"] => int(1417042800)
  }
}
```

Explicit YAML casting

key:

value1: !str 5

value2: ! 2.7

value3: !str 2014-11-27

Explicit YAML casting

```
key:
  value1: !str 5
  value2: ! 2.7
  value3: !str 2014-11-27

array(1) {
  ["key"]=> array(3) {
    ["value1"] => string(1) "5"
    ["value2"] => int(2)
    ["value3"] => string(10) "2014-11-27"
  }
}
```


Explicit YAML casting

```
key:                                array(1) {
    value1: !str 5                  ["key"]=> array(3) {
    value2: ! 2.7                   ["value1"] => string(1) "5"
    value3: !str 2014-11-27        ["value2"] => int(2)
                                   ["value3"] => string(10) "2014-11-27"
                                   }
    }
```

! = int

!str = string

!!php/object: = serialized object

Custom Expressions

Expression Language in action

```
/**
 * @Route("/post/{id}")
 * @Cache(smaxage="15", lastModified="post.getUpdatedAt()")
 */
public function showAction(Post $post) { ... }

/**
 * @Assert\Expression("this.getFoo() == 'fo'", message="Not good!")
 */
class Obj { ... }
```

Custom md5() function

```
use Symfony\Component\ExpressionLanguage\ExpressionFunction;
use Symfony\Component\ExpressionLanguage\ExpressionFunctionProviderInterface;

class AppExpressionLanguageProvider implements ExpressionFunctionProviderInterface
{
    return array(
        new ExpressionFunction(
            'md5',
            function ($str) {
                return sprintf('(is_string(%1$s) ? md5(%1$s) : %1$s)', $str);
            },
            function ($arguments, $str) {
                return is_string($str) ? md5($str) : $str;
            }
        );
    );
}
```

Custom md5() function

```
use Symfony\Component\ExpressionLanguage\ExpressionFunction;  
use Symfony\Component\ExpressionLanguage\ExpressionFunctionProviderInterface;
```

```
class AppExpressionLanguageProvider implements ExpressionFunctionProviderInterface  
{
```

```
    return array(  
        new ExpressionFunction(  
            'md5',
```

compiled expression

```
function ($str) {  
    return sprintf('(is_string(%1$s) ? md5(%1$s) : %1$s)', $str);  
},
```

```
function ($arguments, $str) {  
    return is_string($str) ? md5($str) : $str;  
}
```

```
);
```

```
);
```

```
}
```

Custom md5() function

```
use Symfony\Component\ExpressionLanguage\ExpressionFunction;  
use Symfony\Component\ExpressionLanguage\ExpressionFunctionProviderInterface;
```

```
class AppExpressionLanguageProvider implements ExpressionFunctionProviderInterface  
{
```

```
    return array(  
        new ExpressionFunction(  
            'md5',
```

compiled expression

```
            function ($str) {  
                return sprintf('(is_string(%1$s) ? md5(%1$s) : %1$s)', $str);  
            },
```

```
            function ($arguments, $str) {  
                return is_string($str) ? md5($str) : $str;  
            }  
        )  
    );
```

executed expression

```
};  
}
```

Using custom functions in expressions

```
namespace AppBundle\Controller;
```

```
use Symfony\Bundle\FrameworkBundle\Controller\Controller;
```

```
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Security;
```

```
class BlogController extends Controller
```

```
{
```

```
    /**
```

```
     * @Security("md5(user.email) == post.author_id")
```

```
     * ...
```

```
    */
```

```
    public function editAction(...)
```

```
    {
```

```
        // ...
```

```
    }
```

```
}
```

```
# app/config/services.yml
```

```
services:
```

```
    app.expressions:
```

```
        class: AppBundle\ExpressionLanguage\Provider
```

```
        tags:
```

```
            - { name: security.expression_language_provider }
```

Expression Language service tags

```
services:
  app.expressions:
    class: ...
    tags:
      - { name: security.expression_language_provider }
```

```
services:
  app.expressions:
    class: ...
    tags:
      - { name: routing.expression_language_provider }
```


mini-tip

Have you ever used the
`{% do %}` Twig tag?

Using the {% do %} Twig tag

```
{% do form.field.setRendered %}
```

```
{{ form_start(form) }}  
    {{ form_errors(form) }}  
  
    {{ form_row(...) }}  
{{ form_end(form) }}
```

Using the {% do %} Twig tag

```
{% do form.field.setRendered %}
```

```
{{ form_start(form) }}  
    {{ form_errors(form) }}
```

```
        {{ form_row(...) }}  
{{ form_end(form) }}
```

**Makes form_rest()
and form_end() not
display the given field.**

Parallel
everything

**How many cores/threads
has your CPU?**

**How many of them are used
while developing the
application?**

Parallel tests

**« TDD test suites should run
in 10 seconds or less »»**

Source: <http://blog.ploeh.dk/2012/05/24/TDDtestsuitesshouldrunin10secondsorless/>

Parallel test execution with Fastest

```
# Install
```

```
$ composer require --dev 'liuggio/fastest' 'dev-master'
```

```
# Execute tests in 4 parallel threads
```

```
$ find src/ -name "*Test.php" | ./vendor/bin/fastest  
  "phpunit -c app {};"
```

Database isolation for parallel tests

```
getenv('ENV_TEST_CHANNEL');  
getenv('ENV_TEST_CHANNEL_READABLE');  
getenv('ENV_TEST_CHANNELS_NUMBER');  
getenv('ENV_TEST_ARGUMENT');  
getenv('ENV_TEST_INC_NUMBER');  
getenv('ENV_TEST_IS_FIRST_ON_CHANNEL');
```

Database isolation for parallel tests

```
getenv('ENV_TEST_CHANNEL');  
getenv('ENV_TEST_CHANNEL_READABLE');  
getenv('ENV_TEST_CHANNELS_NUMBER');  
getenv('ENV_TEST_ARGUMENT');  
getenv('ENV_TEST_INC_NUMBER');  
getenv('ENV_TEST_IS_FIRST_ON_CHANNEL');
```

30
minutes



7
minutes

Alternatives to Fastest

- **Paratest (Official PHPUnit plugin)**

github.com/brianium/paratest

- **Parallel PHPUnit**

github.com/socialpoint-labs/parallel-phpunit

- **Docker + PHPUnit**

marmelab.com/blog/2013/11/04/how-to-use-docker-to-run-phpunit-tests-in-parallel.html

Parallel assets

Dumping assets ... slowly

```
$ php app/console --env=prod  
  assetic:dump
```

Parallel asset dumping

```
$ php app/console --env=prod  
  assetic:dump  
  --forks=4
```


Parallel asset dumping

```
$ prod a:d --forks=4
```

Parallel asset dumping

```
$ prod a:d --forks=4
```

```
# it requires to install the Spork library  
$ composer require kriswallsmith/spork
```

Parallel HTTP requests

Parallel HTTP requests with Guzzle

```
use GuzzleHttp\Pool;
use GuzzleHttp\Client;

$client = new Client();

$requests = [
    $client->createRequest('GET', 'http://example.org'),
    $client->createRequest('DELETE', 'http://example.org/delete'),
    $client->createRequest('PUT', 'http://example.org/put', ['body' => 'test'])
];

(new Pool($client, $requests))->wait();
```

Rarely used
Doctrine features

Ordering with expressions

This does not work

```
SELECT t  
  FROM Talks t  
ORDER BY t.comments + t.likes_count
```

This does work

```
SELECT t,  
       (t.comments + t.likes_count) AS HIDDEN score  
FROM Talks t  
ORDER BY score
```


Partial queries

Selecting specific properties

```
public function findPost()  
{  
    return $this  
        ->createQueryBuilder('p')  
        ->select('p.id, p.title')  
        ->where('p.id = :id')->setParameter('id', 1)  
        ->getQuery()  
        ->getResult()  
    ;  
}
```

The executed SQL query

doctrine.DEBUG:

```
SELECT p0_.id AS id_0,  
       p0_.title AS title_1  
FROM   Post p0_  
WHERE  p0_.id = ?
```

The resulting "object"

```
array(1) {  
    [0]=> array(2) {  
        ["id"] => int(1)  
        ["title"] => string(7) "Post #1"  
    }  
}
```

Using Doctrine partials

```
public function findPost()  
{  
    return $this  
        ->createQueryBuilder('p')  
        ->select('partial p.{id, title}')  
        ->where('p.id = :id')->setParameter('id', 1)  
        ->getQuery()  
        ->getResult()  
    ;  
}
```

The executed SQL query

doctrine.DEBUG:

```
SELECT p0_.id AS id_0,  
       p0_.title AS title_1  
FROM   Post p0_  
WHERE  p0_.id = ?
```

The resulting object

```
array(1) {  
    [0]=>  
        object(stdClass)#676 (9) {  
            ["__CLASS__"] => string(21) "AppBundle\Entity\Post"  
            ["id"] => int(1)  
            ["title"] => string(7) "Post #1"  
            ["slug"] => NULL  
            ["summary"] => NULL  
            ["content"] => NULL  
            ["authorEmail"] => NULL  
            ["publishedAt"] => NULL  
            ["comments"] => string(8) "Array(5)"  
        }  
    }  
}
```

The resulting object

```
array(1) {  
  [0]=>  
    object(stdClass)#676 (9) {  
      ["__CLASS__"] => string(21) "AppBundle\Entity\Post"  
      ["id"] => int(1)  
      ["title"] => string(7) "Post #1"  
      ["slug"] => NULL  
      ["summary"] => NULL  
      ["content"] => NULL  
      ["authorEmail"] => NULL  
      ["publishedAt"] => NULL  
      ["comments"] => string(8) "Array(5)"  
    }  
  }  
}
```

The diagram illustrates the structure of the resulting object. It shows a PHP array containing a single object of type AppBundle\Entity\Post. The object has several properties: __CLASS__, id, title, slug, summary, content, authorEmail, publishedAt, and comments. Arrows point to specific fields: one points to the class name, another to the title, and others to the NULL values for slug, summary, content, authorEmail, and publishedAt. The comments field is shown as an array of 5 elements.

WARNING

**Doctrine Partials make
your code very fragile.
Use them with caution.**

Legitimate use cases for Partial

- Specific (and limited) parts of the application where performance is the top priority.
- Never when developing the application (wrong premature optimization).

**schema_filter
option**

Configuring schema_filter option

```
# app/config/config.yml
```

```
doctrine:
```

```
  dbal:
```

```
    default_connection: default
```

```
    # ...
```

```
    schema_filter: ~^(?!legacy_)
```

Filtering tables managed by other apps

```
# app/config/config.yml
```

```
doctrine:
```

```
  dbal:
```

```
    default_connection: default
```

```
    # ...
```

```
    schema_filter: ~^(?!(django|auth)_)$~
```

Using schema_filter to migrate gradually

```
# app/config/config.yml
doctrine:
    dbal:
        default_connection:    default
        connections:
            default:
                dbname:          "%database_name%"
                # ...
        legacy:
            # ...
            schema_filter: ~^(?!(prefix)_)
```

Doctrine Filters

**How do you deal with
global SQL clauses?**

Examples of global SQL clauses

CMS applications

Display only published contents.

```
WHERE status = :published
```

Examples of global SQL clauses

CMS applications

Display only published contents.

```
WHERE status = :published
```

Never display (soft) deleted items.

```
WHERE status = :not_deleted
```

Examples of global SQL clauses

CMS applications

Display only published contents.

```
WHERE status = :published
```

Never display (soft) deleted items.

```
WHERE status = :not_deleted
```

Commerce applications

Display only invoices/orders for the given user.

```
WHERE user_id = :user_id
```

**Doctrine filters allow to
add SQL to the conditional
clauses of queries,
regardless the place where
the SQL is generated.**

Creating a global Locale filter

```
namespace AppBundle\Filter\LocaleFilter;
```

```
use Doctrine\ORM\Mapping\ClassMetadata;
```

```
use Doctrine\ORM\Query\Filter\SQLFilter;
```

```
WHERE locale = 'es'
```

```
class LocaleFilter extends SQLFilter
```

```
{
```

```
    public function addFilterConstraint(ClassMetadata $targetEntity, $targetTableAlias)
    {
```

```
        return sprintf(
```

```
            '%s.%s = %s',
```

```
            $targetTableAlias, 'locale', $this->getParameter('locale')
```

```
        );
```

```
    }
```

```
}
```

Enabling the filter globally

```
# app/config/config.yml
```

```
doctrine:
```

```
    orm:
```

```
        filters:
```

```
            locale_filter:
```

```
                class: AppBundle\Filter\LocaleFilter
```

```
                enabled: true
```

Using the filter and setting its values

```
$filter = $this->em->getFilters()  
        ->enable('locale_filter');
```

```
$filter->setParameter('locale', 'es');
```

**What's the point of using
global filters if you have to
set up values manually for
each query?**

Setting filter values automatically (1/2)

```
services:
  app.locale_filter:
    class: AppBundle\Filter\LocaleFilter
    arguments: [@session]
    tags:
      - {
          name:    kernel.event_listener,
          event:   kernel.request,
          method:  onKernelRequest
        }
```

Setting filter values automatically (2/2)

```
public function __construct(SessionInterface $session)
{
    $this->session = $session;
}

// ...

public function onKernelRequest()
{
    $this->em->getConfiguration()
        ->addFilter('locale_filter', 'AppBundle\\Filter\\LocaleFilter');

    $filter = $this->em->getFilters()->enable('locale_filter');
    $filter->setParameter('locale', $this->session->get('user_locale'));
}
```

Source: <http://stackoverflow.com/a/15792119>

Constraining the scope of the filters

```
class LocaleFilter extends SQLFilter
{
    public function addFilterConstraint(ClassMetadata $targetEntity, $targetTableAlias)
    {
        if (!$targetEntity->reflClass->implementsInterface('LocaleAware')) {
            return '';
        }

        // ...
    }
}
```

Constraining the scope of the filters

```
class LocaleFilter extends SQLFilter
{
    public function addFilterConstraint(ClassMetadata $targetEntity, $targetTableAlias)
    {
        if (!$targetEntity->reflClass->implementsInterface('LocaleAware')) {
            return '';
        }

        // ...

    }
}

/** @ORM\Entity */
class Content implements LocaleAware
{
    // ...
}
```

Combining filters with annotations

```
namespace AppBundle\Entity;

use AppBundle\Annotation\UserAware;

/**
 * @UserAware(userFieldName="user_id")
 */
class Order { ... }
```

Combining filters with annotations

```
namespace AppBundle\Entity;
```

```
WHERE user_id = :id
```

```
use AppBundle\Annotation\UserAware;
```

```
/**
```

```
 * @UserAware(userFieldName="user_id")
```

```
 */
```

```
class Order { ... }
```

Combining filters with annotations

```
namespace AppBundle\Entity;
```

```
WHERE user_id = :id
```

```
use AppBundle\Annotation\UserAware;
```

```
/**
```

```
 * @UserAware(userFieldName="user_id")
```

```
 */
```

```
class Order { ... }
```

Learn how to do this at:

<http://blog.michaelperrin.fr/2014/07/25/doctrine-filters/>

Value Objects

Value Objects in theory

A value object is a small object that represents a simple entity whose equality is not based on identity.

**Two value objects are equal
when they have the same
value, not necessarily being
the same object.**

Identity is irrelevant for value objects

```
object(AppBundle\Entity\Money)#25 (3) {  
    ["id"]          => int(25)  
    ["amount"]      => int(10)  
    ["currency"]    => string(3) "EUR"  
}
```

10 EUR

```
object(AppBundle\Entity\Money)#267 (3) {  
    ["id"]          => int(267)  
    ["amount"]      => int(10)  
    ["currency"]    => string(3) "EUR"  
}
```

10 EUR

Value Objects in Doctrine

A typical Doctrine entity

```
namespace AppBundle\Entity;

/** @Entity */
class Customer
{
    /** @Id @GeneratedValue @Column(type="integer") */
    protected $id;

    /** @Column(type="string") */
    protected $name;

    /** @Column(type="string", length=120) */
    protected $email_address;

    protected $address_line_1;
    protected $address_line_2;
    protected $city;
    protected $state;
    protected $postalCode;
    protected $country;
}
```

A typical Doctrine entity

```
namespace AppBundle\Entity;

/** @Entity */
class Customer
{
    /** @Id @GeneratedValue @Column(type="integer") */
    protected $id;

    /** @Column(type="string") */
    protected $name;

    /** @Column(type="string", length=120) */
    protected $email_address;

    protected $address_line_1;
    protected $address_line_2;
    protected $city;
    protected $state;
    protected $postalCode;
    protected $country;
}
```

**Address
Value Object**

Defining a Value Object in Doctrine

```
namespace AppBundle\ValueObject;
```

```
use Doctrine\ORM\Mapping as ORM;
```

```
/** @ORM\Embeddable */
```

```
class Address
```

```
{
```

```
    /** @ORM\Column(type = "string") */  
    protected $line_1;
```

```
    /** @ORM\Column(type = "string") */  
    protected $city;
```

```
    /** @ORM\Column(type = "string") */  
    protected $postalCode;
```

```
}
```

```
    /** @ORM\Column(type = "string") */  
    protected $line_2;
```

```
    /** @ORM\Column(type = "string") */  
    protected $state;
```

```
    /** @ORM\Column(type = "string") */  
    protected $country;
```


Using a Value Object in Doctrine

```
namespace AppBundle\Entity;

use Doctrine\ORM\Mapping as ORM;

/** @ORM\Entity */
class Customer
{
    /** @ORM\Id @ORM\GeneratedValue @ORM\Column(type="integer") */
    protected $id;

    /** @ORM\Column(type="string") */
    protected $name = '';

    /** @ORM\Column(type="string", length=120) */
    protected $email_address = '';

    /** @ORM\Embedded(class="\AppBundle\ValueObject\Address") */
    protected $address;
}
```

Value Objects at SQL level

```
CREATE TABLE Post (  
  id INTEGER NOT NULL,  
  title VARCHAR(255) NOT NULL,  
  slug VARCHAR(255) NOT NULL,  
  summary VARCHAR(255) NOT NULL,  
  content CLOB NOT NULL,  
  authorEmail VARCHAR(255) NOT NULL,  
  publishedAt DATETIME NOT NULL,  
  address_line_1 VARCHAR(255) NOT NULL,  
  address_line_2 VARCHAR(255) NOT NULL,  
  address_city VARCHAR(255) NOT NULL,  
  address_state VARCHAR(255) NOT NULL,  
  address_postalCode VARCHAR(255) NOT NULL,  
  address_country VARCHAR(255) NOT NULL,  
  PRIMARY KEY(id))
```

Value Objects at query level

```
SELECT c  
  FROM Customer c  
 WHERE c.address.city = :city
```

**Embeddables are classes
which are not entities
themselves, but are embedded
in entities and can also be
queried in DQL.**

Agenda

Assets

Performance

Logging

Legacy

Testing

Config

Parallel

Doctrine

**Value
Objects**

Thank you.

Questions?

<http://joind.in/talk/view/12944>

Contact info

javier.eguiluz@sensiolabs.com

SymfonyCon
Madrid - 27 November 2014