

10 Years Of Zend Framework

Original Architecture

- Components
- Bound together by an MVC

Evolution Of V1

- Forms
- Layouts
- Zend_Application
- Dojo, jQuery integration

Revolution: ZF2

- Event-driven architecture
- Dependency injection
- First class modules

... At A Cost

- Complete break in compatibility.
- Targeting expert developers == less inviting
- Components depend on integration provided by MVC layer

Ecosystem Changes

Composer



FIG Framework Interop Group

PSR-7

Every framework abstracts HTTP differently. Let's standardize that.

Frameworks Must Change

The job of a framework is to provide plumbing, and get out of your way.

ZF3 Goals

- Componentization and re-use
- Performance
- Usability
- Focus on PSR-7 and middleware

What We've Accomplished

Components

All components are now versioned separately. (Full story at bit.ly/zf2-split)

Standardized Package Structure

- PSR-4 structure for source and tests.
- Documentation per package (in progress)
- Standard QA toolchain; per-package continuous integration.

ZF2 Package

The zendframework package now simply depends on components.

```
"require": {
    "zendframework/zend-authentication": "^2.5",
    "zendframework/zend-cache": "^2.5",
    "zendframework/zend-captcha": "^2.5",
    "etc": "*"
}
```

Back To Basics

ZF3 will reduce dependencies to only what's needed for the MVC. Use *Composer* to add what you need.

Change Is Inevitable

Service Manager

- container-interop
- Consistent interfaces.
- Immutable.
- Performant (4X faster)!
- Mostly backwards compatible!

New method: build

public function build(\$name, array \$options = null)

For when you need a factory; think *plugins*

FactoryInterface

```
public function __invoke(
    ContainerInterface $container,
    $requestedName,
    array $options = null
)
```

EventManager

- 4X–15x performance based on use case!
- BC breaks:
 - Removed argument overloading for trigger().
 - Aggregate attachment is moved to aggregate implementations.

Triggers

```
trigger($eventName, $target = null, $argv = []);
triggerUntil(callable $callback, $eventName, $target = null, $ar
triggerEvent(EventInterface $event);
triggerEventUntil(callable $callback, EventInterface $event);
```

Aggregates

Before:

```
$events->attach($aggregate);
$events->attachAggregate($aggregate);
$aggregate->attach($events);
```

After:

```
$aggregate->attach($events);
```

MVC

- Updated to changes in zend-servicemanager.
- Updated to changes in zend-eventmanager.
- Essentially stays the same.
- But adds a MiddlewareListener.

Routing To Middleware

MVC Availability

4-6 weeks!

Enough With The Boring Stuff

Time To Reflect

Where is the PHP community headed?

Composer

FIG

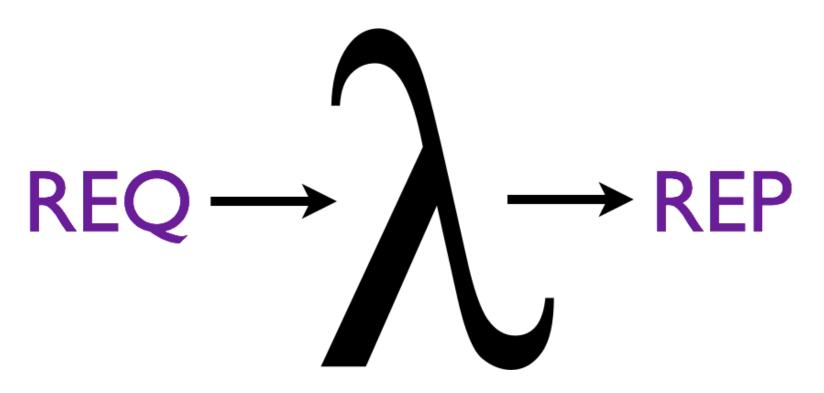
Creating shared interfaces.

Users can target *abstractions*, not *implementations*.

PSR-7

HTTP Message Interfaces
Providing a common HTTP abstraction to program against.

Middleware



Middleware Signatures

```
function (ServerRequestInterface $request) : ResponseInterface

function (
    ServerRequestInterface $request,
    ResponseInterface $response
) : ResponseInterface

function (
    ServerRequestInterface $request,
    ResponseInterface $response,
    callable $next
) : ResponseInterface
```

Container-Interop

```
interface ContainerInterface
{
    public function has($serviceName);
    public function get($serviceName);
}
```

Takeaways

Frameworks Should Be An Implementation Detail

Frameworks Should Get Out Of The Way Of Your Code

Expressive

PSR-7 middleware microframework

- Provides and consumes a routing interface.
- Pulls matched middleware from a ContainerInterface.
- Provides a templating interface, if you need it.
- Provides error handling, and a way to hook into it.

In Action:

Expressive

Wiring together commodity components.

Own Your Codebase

ZF3 Is A Movement

An end to framework silos.

DANKE SCHÖN!

Jan Burkl

http://framework.zend.com https://apigility.org jan@zend.com @janatzend