

Zend Framework 2.0 is coming!

Rob Allen

PHPNW 2010, October 2010

Rob Allen?

<http://akrabat.com>

@akrabat

Zend Framework 1.0

How we got here

- Announced October 2005
- Mar 2006: v0.1 released: not a lot!
- July 2007: v 1.0: MVC + cool stuff!
- Mar 2008: v1.5: Form, Layout, Context
- Apr 2009: v1.8: Application, Tool, Nav
- Oct 2010: v1.11: To be released

Where we are today

- Zend Framework 1 core is complete
- New components, adapters and plugins are being added
- Widespread usage
- Plenty of help available

Problems with ZF1

- Learning curve
- Performance
- Too much “magic”
- Inconsistency between components

Zend Framework 2.0

“The primary thrust of ZF 2.0 is to make a more consistent, well-documented product, improving developer productivity and runtime performance.”

Matthew Weier O’Phinney

Think *evolution*,
not *revolution*

Zend Framework 2

Key goals

Key goals

- Ease learning curve
- Easier extension
- Improve performance
- Simplify

Ease learning curve

- Create better documentation
 - Cohesive tutorials
 - Reference manual
- Address API consistency
 - Options
 - Language

Easier extension

- Remove singletons
- Interfaces over abstract classes
- Remove hard-coded dependancies

Improve performance

- Aiming for 200% over ZF 1
- Deployment tools
 - Optimise for production
- Documentation

Simplify

- Focus on core components
- Make code readable
- Consistent APIS

Zend Framework 2

Key changes

Key changes

- PHP 5.3 support
- Autoloading and plugin loading
- Exception handling
- MVC
- Internationalisation

“The general theme throughout these milestones is narrowing the scope of components, and ensuring a proper separation of concerns between components.”

Matthew Weier O’Phinney

PHP 5.3 support

- Advance the use of PHP 5.3
- Namespaces
- Closures over `create_function()`

Namespaces

```
// library/Nennius/Controller/Plugin/InitAdminLayout.php
<?php
namespace Nennius\Controller\Plugin;
use \Zend\Controller\Request\AbstractRequest as Request;
use \Zend\Controller\Plugin\AbstractPlugin as Plugin;

class InitAdminLayout extends Plugin
{
    public function dispatchLoopStartup(Request $request)
    {
        $fc = \Zend\Controller\Front::getInstance();
        $bootstrap = $fc->getParam('bootstrap');
        $layout = $bootstrap->layout;
        $controllerName = $request->getControllerName();
        if (substr($controllerName, 0, 5) == 'admin') {
            $layout->setLayout('admin');
        }
    }
}
```

Autoloading

- No more `require_once`!
- Opt 1: Explicit namespace=>path pairs
 - doesn't use of include path
 - ~40% performance gain!
- Opt 2: Classmap files
 - pre-built array map (for deployment)
 - ~150% performance gain!

Plugin loading

- Alias autoloading
 - map 'alias' to class that it represents
 - Interface for consistency (clients)
- Plugin broker (central operations)
- Mapping is simpler to explain!
- Much more performant! *(up to 5x!)

Plugin loading

```
$broker = new Zend\View\HelperBroker();  
$loader = $broker->getPluginLoader();  
  
$loader->register('url', 'My\View\Helper\Url');  
  
// get a Zend\View\Helper\BaseUrl  
$baseUrlHelper = $broker->load('baseUrl');  
  
// get a My\View\Helper\Url  
$urlHelper = $broker->load('url');
```

Exception handling

- Use SPL exceptions for specific catches
- Use an interface for component catches
- Can catch `\Exception` if you don't care :)

Exceptions

```
namespace My;
interface ExceptionInterface {}
class Exception implements ExceptionInterface {}
class InvalidArgument extends InvalidArgumentException
    implements ExceptionInterface {}

try {
    $a = new A();
    $a->b($c);
}
catch (InvalidArgument $e) {
    die("Invalid arg happened\n");
}
catch (ExceptionInterface $e) {
    die("Some error happened\n");
}
```

MVC

- Refactor
 - Controller layer
 - View layer
 - Forms
 - Zend\Application
- For performance and flexibility

Internationalisation

- Improve performance
- Use DateTime
- Fix inconsistencies with PHP's APIs
- Remove statics and global states

Zend Framework 2 Development process

Development process

- git
- Unit testing
- Community Review team

git

- Smaller barrier to contribution
- Better branching and merging
- ZF's git guide: <http://bit.ly/zfgit>

Unit testing

- Write to /tmp only
- Separate out test asset classes
- Consistent usage of _files directory
- Lower memory/CPU usage

Community Review team

- Smooth the proposal process
- Assist new contributors
- “Fix” orphaned components
- Contact:
 - IRC: `#zftalk.dev` on freenode
 - Email: cr-team@zend.com

Milestones

- Autoloading & plugin loading [done]
- Exceptions [in progress]
- Testing
- MVC
- Documentation
- Internationalisation

When?

I have no idea!

Summary

- Play with the ZF2 snapshot
- Contribute
 - Report bugs!
 - Answer “calls for help”
- Need help?
 - IRC: #zftalk on freenode
 - Mailing list
 - Wiki: <http://framework.zend.com/wiki/>

Questions?

Thank you

feedback: <http://joind.in/2050>

email: rob@akrabat.com

twitter: @akrabat