

cjw

Learnings from Real eZ Publish 5 Projects

Donat Fritschy - Webmanufaktur Ekkehard Dörre - Coolscreen



Presenters

- Donat is owner of Webmanufaktur, a full service web agency in Switzerland.
 He works as projects manager, software architect and developer and likes
 thinking outside of the box. In the last year he has been involved in major eZ 5
 projects.
- **Ekke** is a consultant with deep knowledge in eZ Publish 4 and 5, eZ Find / Apache Solr and with a faible for coming cutting edge web technologies. He is one of the organizers of the PHP Unconference since eight years.
- Members of CJW Network

cjw #ezsummer 3/64

Learnings from Real eZ Publish 5 Projects

- · Last year we have presented our Cookbook...
- · ... today you get a Ratatouille ;-)
- · Let's share the experiences you and we made in the last year!
- http://de.slideshare.net/dfritschy/how-to-start-with-ez-publish-5
- https://github.com/dfritschy/cjw-cookbook

cjw #ezsummer 4/64

Who is in the audience?

- · Who is coming from the eZ World?
- Who is coming from the Symfony World?
- · Who has realized eZ Publish 5 sites (productive)?
- · Who has realized pure eZ Publish 5 sites (no legacy)?

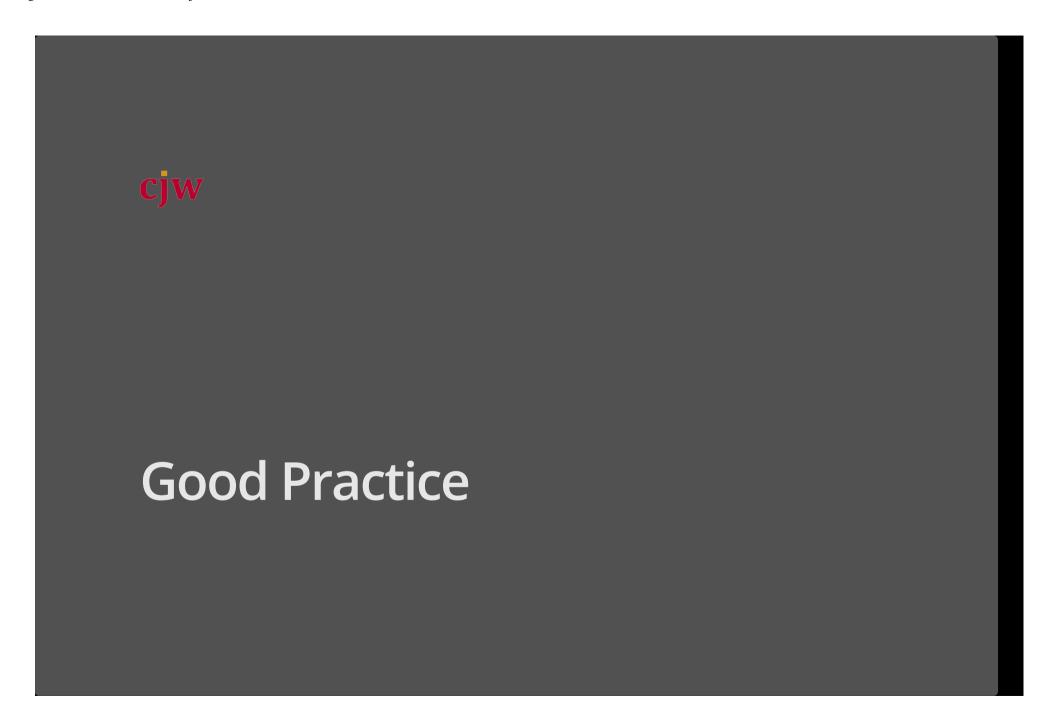
cjw #ezsummer 5/64

Agenda

Things we would like to discuss:

- Good Practice
- · ez View Cache vs. HTTP Cache
- Debugging
- Pitfalls
- MultiSite Setup

cjw #ezsummer 6/64



Team up with a Symfony Crack

- · To be honest: as eZ 4 developers, we are complete novices in eZ 5
- It's easier for a Smyfony Crack to learn eZ than other way round
- · Symfony community is hungry for a CMS, so watch out for new competition
- But @Symfony cracks: It's not easy: an eZ Publish and CMS expert will reduce your risk
- · And will make your content architecture better and more maintainable

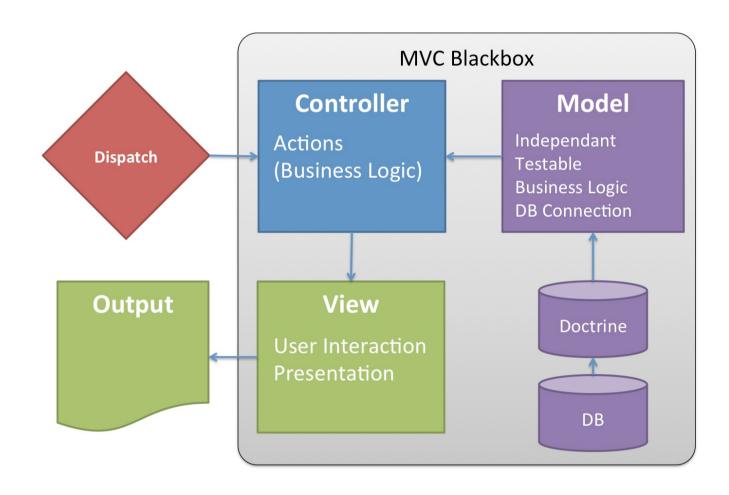
cjw #ezsummer 8/64

Think in MVC

- · A radical different thinking
- eZ 4 mangled all together in the TPL -> the view implemented the logic (fetch)
- · Symfony enforces a clean separation, from routing to the controller to the rendering of the view

cjw #ezsummer 9/64

MVC Blackbox



cjw #ezsummer 10/64

Think in Bundles

What is a bundle?

- · Use a least one bundle per site
- · Split your application in different bundles (site specific, functional, ...)
- Reuse your code: create and maintain with love a Base Bundle with general functions
- · Refactor it frequently as you learn and the product evolves
- Creating bundles is easy, don't work in the DemoBundle ;-)

\$ php ezpublish/console generate:bundle

BASH

cjw #ezsummer 11/64

Organize your config files

The standard eZ installation is a mess...

... and the DemoBundle is only slowly becoming a source of good practice

How do YOU handle this?

- · keep config in ezpublish/config as general as possible
- · it should merely describe your environment, not the application
- move all site/function specific settings to the bundle

cjw #ezsummer 12/64

Keep ezpublish.yml small (1)

ezpublish/config/ezpublish.yml

```
YML
imports:
    - {resource: "@CjwExampleBundle/Resources/config/ezpublish.yml"}
ezpublish:
    siteaccess:
        default siteaccess:%cjw.site% user
        list:
        groups:
            default group:
        match:
            \Cjw\MultiSiteBundle\Matcher\MapHost:
                www.%cjw.site%.ch: %cjw.site% user
                admin.%cjw.site%.ch: %cjw.site% admin
    repositories:
        default_repository:
            engine: legacy
            connection: default_connection
```

cjw #ezsummer 13/64

cjw #ezsummer

Keep ezpublish.yml small (2)

```
YML
    system:
        default group:
            repository: default repository
            var dir: var/%cjw.site%
            languages:
                - ger-DE
        %cjw.site% user:
            legacy mode: false
            languages:
                - ger-DE
            content:
                view cache: true
                ttl cache: true
                default ttl: 3600
        %cjw.site% admin:
            legacy mode: true
stash:
```

14 von 64 05.09.14 08:28

14/64

Keep ezpublish.yml small (3)

Can even be shorter - get inspiration from https://github.com/lolautruche /metalfrance

Extras:

- use parameters
- · standardize site access names, groups, repository names

ezpublish/config/parameters.yml

```
parameters:
...
cjw.site: frb
```

Note: prepending configuration does not work well with parameters

cjw #ezsummer 15/64

Config Files in Bundle

We keep them in a separate directory and name them as in good old eZ...

```
ExampleBundle
Resources
config
ezpublish
image.yml
override.yml
ezpublish.yml <--- includes files from ./ezpublish
parameters.yml
routing.yml
services.yml
```

cjw #ezsummer 16/64

Controllers

After several tries, we ended up with...

- · Basically one controller per full view
- · Separate controllers for navigation etc.
- Consider caching (TTL, X-Location Header)
- · Recommended: Move business logic to separate model
- Retrieve all needed data (location, children, ...)
- Prepare the data for easy processing in the templates

cjw #ezsummer 17/64

Ways to Fetch Content

- LocationService::loadLocation(\$id) --> location
- ContentService::loadContent(\$id) --> content
- SearchService::findContent(\$query) --> list of content
- SearchService::findLocations(\$query) --> list of location
- LocationService::loadLocationChildren(\$location) --> list of location
- Legacy fetch functions

cjw #ezsummer 18/64

SearchService::findContent()

The only SearchService function you will find in DemoBundle ...

- · returns full content objects with ALL attributes in ALL languages
- does not work well with multiple locations
- · no as_objects flag as in eZ 4
- scales very badly
- fetching a content tree with 116 locations took 30 seconds
- most of the time is spent in manipulating the SQL result in PHP
- Another test: 24 hits, PHP array 44'880 rows with 39 elements each, highly redundant
- http://share.ez.no/blogs/donat-fritschy/searchservice-performance

cjw #ezsummer 19/64

SearchService::findLocations()

Available from 2014.05 / 5.3 only

Roughly equivalent to good old fetch ('content', 'list')

- returns location objects with contentinfo only
- usually sufficient for building a menu
- · use ContentService::loadContent() to fetch whole object
- Performance lower than legacy, but acceptable
- fetching a content tree with 116 locations took < 1 second
- scales very well

cjw #ezsummer 20/64

LocationService::loadLocationChildren()

Think of LocationService::loadLocationChildren() as primarily intended for administration interface. Has no filtering capabilities.

Further reading:

http://www.netgenlabs.com/Blog/Fetching-content-in-eZ-Publish-5-using-Search-service

cjw #ezsummer 21/64

Legacy Fetch Functions

```
PHP
use eZFunctionHandler;
$mySearchResults = $this->getLegacyKernel()->runCallback(
    function () use ( $searchPhrase, $sort, $contentTypeIdenfiers )
       // eZFunctionHandler::execute is the equivalent for a legacy template fetch function
        // The following is the same than fetch( 'ezfind', 'search', hash(...) )
       return eZFunctionHandler::execute(
            'ezfind',
            'search',
            array(
                'query' => $searchPhrase,
                'sort by' => $sort,
                'class id' => $contentTypeIdenfiers
        );
);
```

cjw #ezsummer 22/64

Templates

How to transform a full view TPL with children to Symfony?

full/folder.tpl

cjw #ezsummer 23/64

Moving to TWIG

Resources/view/full.html.twig

```
<h1>{{ ez_render_field( content, 'title') }}</h1>
{{ ez_render_field( content, 'short_description') }}
...
{{ render( controller( "CjwBaseBundle:Default:subItems", {'locationId': location.id }) ) }}
```

Controller/DefaultController.php

cjw #ezsummer 24/64

Moving to TWIG

Resources/view/sub_items.html.twig

cjw #ezsummer 25/64

Our approach

- · Basically one template per full view
- · Render children directly in the full view template
- · Generalized full and line view templates for the easy stuff

cjw #ezsummer 26/64

How to organize Templates?

The Symfony way...

```
views
L Customer
CustomerDetail.html.twig
L Product
```

The classic eZ way...

```
views
L full
customer.html.twig
L line
```

Two approaches, both valid. Follow your taste.

cjw #ezsummer 27/64



eZ View Cache vs. HTTP Caching

eZ View Caching (Legacy)

When the pagelayout is rendered, the <code>{\$module_result.content}</code> part will be replaced with the actual output. If view caching is enabled, the entire result of the module will be cached. This means that the contents of the "module_result" variable will be put into a cache file (...)

When a new version (...) of an object is published, the system will automatically clear the view cache for the following items:

- · All published nodes of the object
- The parent nodes
- Related nodes (keywords, object relations)

https://doc.ez.no/eZ-Publish/Technical-manual/4.x/Features/View-caching

cjw #ezsummer 29/64

HTTP Expiration and Validation (Symfony)

The HTTP specification defines two caching models:

- With the expiration model, you simply specify how long a response should be considered "fresh" by including a Cache-Control and/or an Expires header.
 Caches that understand expiration will not make the same request until the cached version reaches its expiration time and becomes "stale";
- When pages are really dynamic (i.e. their representation changes often), the validation model is often necessary. With this model, the cache stores the response, but asks the server on each request whether or not the cached response is still valid. The application uses a unique response identifier (the Etag header) and/or a timestamp (the Last-Modified header) to check if the page has changed since being cached.

http://symfony.com/doc/current/book/http_cache.html

cjw #ezsummer 30/64

In Short (and much simplified...)

eZ View Cache caches content and content fragments

- · Standard TTL is 2 hours
- Is purged on content modifications (with smart cache clearing rules)

Symfony's HTTP Cache caches requests

- eZ uses Expiration model by default
- Standard TTL is 60 seconds (86400 for tree menu!)
- Symfony Cache is purged from backend

Let's dive in a bit deeper...

cjw #ezsummer 31/64

ez 4 Cache Directives

Code from ezpublish_legacy/kernel/private/classes/ezpkernelweb.php

```
foreach (
    eZHTTPHeader::headerOverrideArray( $this->uri ) +
    array(
        'Expires' => 'Mon, 26 Jul 1997 05:00:00 GMT',
        'Last-Modified' => gmdate( 'D, d M Y H:i:s' ) . ' GMT',
        'Cache-Control' => 'no-cache, must-revalidate',
        'Pragma' => 'no-cache',
        ...
    ) as $key => $value
}

header( $key . ': ' . $value );
}
```

This guarantees that every request is handled by eZ

cjw #ezsummer 32/64

eZ 5 Cache Directives

Code from vendor/ezsystems/demobundle/EzSystems/DemoBundle/Controller/DemoController.php

```
// Setting HTTP cache for the response to be public and with a TTL of 1 day.
$response = new Response;
$response->setPublic();
$response->setSharedMaxAge( 86400 );
// Menu will expire when top location cache expires.
$response->headers->set( 'X-Location-Id', $rootLocationId );
// Menu might vary depending on user permissions, so make the cache vary on the user hash.
$response->setVary( 'X-User-Hash' );
```

This effectively sets the Response free, out of the reach of eZ

cjw #ezsummer 33/64

Emulating eZ 4 Cache behaviour in eZ 5

This patch to index.php disables client and proxy caching without sacrificing the benefits of the Symfony HTTP cache. Use at own risk!

```
$response = $kernel->handle( $request );

// Emulate eZ 4 cache control
$response->headers->set( 'Cache-Control', 'no-cache, must-revalidate' );

$response->send();
$kernel->terminate( $request, $response );
```

cjw #ezsummer 34/64

Cache Recommendations

- · Read the specifications
- Use Shared Caches with caution
- Cave: setTtl() vs. setClientTtl
- Set TTL as high as possible
- Use Varnish
- http://tools.ietf.org/html/rfc2616#page-74
- https://www.mnot.net/cache_docs/

cjw #ezsummer 35/64

Cache per User - User Hash Definer

src/Cjw/SiteCustomerBundle/Identity/UserHashDefiner.php

```
PHP
namespace Cjw\SiteCustomerBundle\Identity;
use eZ\Publish\SPI\User\IdentityAware;
use eZ\Publish\SPI\User\Identity;
use eZ\Publish\API\Repository\Repository;
class UserHashDefiner implements IdentityAware
    private $repository;
    public function construct(Repository $repository)
        $this->repository = $repository;
    public function setIdentity(Identity $identity)
        $current user = $this->repository->getCurrentUser();
        $identity->setInformation('UserID', $current user->contentInfo->id);
```

cjw #ezsummer 36/64

Cache per User - yml config

src/Cjw/SiteCustomerBundle/Ressources/config/services.yml

```
parameters:
    cjw_site_customer.user_hash_definer.class: Cjw\SiteCustomerBundle\Identity\UserHashDefiner

services:
    cjw_site_customer.user_hash_definer:
        class: %cjw_site_customer.user_hash_definer.class%
        tags:
        - { name: ezpublish.identity_definer }
        arguments: [@ezpublish.api.repository]
```

cjw #ezsummer 37/64

Cache per User - Controller

src/Cjw/SiteCustomerBundle/Controller/CjwController.php

```
public function sectionInternalAction($locationId, $viewType, $layout = false, array $parar
{
    $response = new Response();
    $response->setPrivate();
    $response->headers->set('X-Location-Id', $locationId);
    $response->setVary('X-User-Hash');
    return $response;
}
```

cjw #ezsummer 38/64

38 von 64 05,09,14 08:28

Cache per User - Location View Configuration

src/Cjw/SiteCustomerBundle/Ressources/config/ezpublish
/override.yml

cjw #ezsummer 39/64



Blank screen, "503 Service not available"

- · PHP errors (Syntax error, Memory, Outdated Autoloads, ...)
- · Configuration errors (DB connection, ...)
- Switch to DEV mode for better debugging
- Check the log files

```
Apache/PHP Log
ezpublish/logs/<env>.log
ezpublish_legacy/var/log/*
ezpublish_legacy/var/<siteaccess>/log/*
```

Check write permissions on log files!

cjw #ezsummer 41/64

TwigBundle:Exception:error500.html.twig

- NEVER a Twig error!
- · Caused by response 500 "Internal Server Error" and missing error template
- · Checks as before

cjw #ezsummer 42/64

Twig Exception: Invalid variation "<variation>"

Caused by problems when accessing images

- · Check if the file exists
- Check permissions on ezpublish_legacy/var/<siteaccess>/storage
- · Check log files
- · Clear cache

cjw #ezsummer 43/64

Class 'ezxFormToken' not found

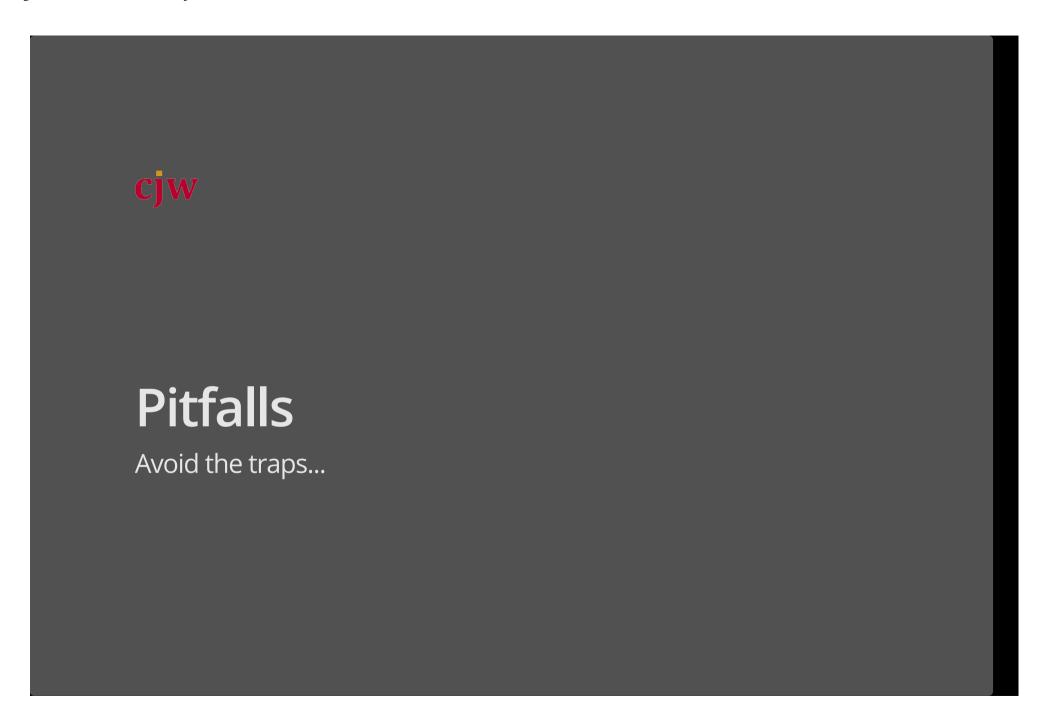
- · Usually found with fresh installations involving legacy extensions
- · Regenerate Autoloads

```
$ cd ezpublish legacy
```

BASH

\$ php bin/php/ezpgenerateautoloads.php -e -p

cjw #ezsummer 44/64



Memory limit exceeded in DEV mode

- DEV mode takes a lot of memory
- Stash Logging is the worst
- · Disable Stash Logging in ezpublish.yml

cjw #ezsummer 46/64

414 Request-URI Too Long

When doing subrequests, particularly ESI or Hinclude ones, current SiteAccess is transmitted in a serialized form, with its matcher. With a large number of configured SiteAccesses using Map\Host or Map\URI matcher (around 40, which is not uncommon for a multi-national, multi-lingual site) the URL can exceed the size of 8192 Bytes which most servers accept. As a result, the ESI/Hinclude call fails.

- Fixed in Version 5.3.3 (2014.07)
- https://jira.ez.no/browse/EZP-23168
- https://github.com/ezsystems/ezpublish-kernel/pull/949

cjw #ezsummer 47/64



Multi-Site/Multi-Repository Setup

Why a Multi-Site/Multi-Repository Setup?

- At CJW Network we have developed a multi-site/multi-repository setup for eZ Publish 4 several years ago
- · This allows us to host many individual sites on a single eZ Publish installation

Advantages:

- · Central site administration (site activation, cronjobs, ...)
- Easy deployment (update site extension with Subversion)
- Highly reduced maintenance costs (security patches, upgrades)
- Highly efficient use of hardware resources

Disadvantages:

Some Kernel patches needed

cjw #ezsummer 49/64

Multi-Site/Multi-Repository Setup in eZ 5

First Approach (proven in production)

· Use different ezpublish app directories to host the different sites

Second approach (under development)

Use CJW MultiSiteBundle

cjw #ezsummer 50/64

Directory structure - Multi-Site-Setup (old)

```
ezpublish
                      <-- not used
ezpublish legacy
extension
                     <-- each customer has its own extension and database
 ∟site customer
 ∟site customertwo
ı var
                     <-- each customer has its own var directory
 ∟site customer
 ∟site customertwo
site customer
                      <-- each customer has its own Symfony app
site customertwo
src
∟CjwNetwork
 SiteCustomerBundle <-- each customer has its own bundle
 LSiteCustomertwoBundle
```

cjw #ezsummer 51/64

Multi-Site-Setup (old) Detail ezpublish_legacy

```
ezpublish legacy
extension
                      <-- each customer has its own extension
 ∟site customer
  classes
  ∟design
  modules
  ∟settings
   ∟site.ini
    [DatabaseSettings] <-- each customer has its own database
     LDatabase=database site customer
 ∟site customertwo
  ∟[...]
  ∟settings
  ∟site.ini
    L[DatabaseSettings]
     LDatabase=database site customertwo
ı var
 ∟site customer
                <-- each customer has its own var directory
 ∟site customertwo
```

cjw #ezsummer 52/64

Detail site_customer App - Multi-Site-Setup (old)

```
site customer
∟autoload.php
_bootstrap.php.cache
cache
∟check.php
        <-- all yml files like ezpublish folder, (to improve)
∟config.yml
∟ezpublish.yml
∟ parameters.yml
console
Llogs
∟phpunit.xml.dist
Resources
∟sessions
LSiteCjwbaseCache.php
LSiteCjwbaseKernel.php
LSymfonyRequirements.php
```

cjw #ezsummer 53/64

Scripts on Shell - Multi-Site-Setup (old)

```
# Generate symlinks

php site_customer/console assets:install --symlink web

php site_customer/console ezpublish:legacy:assets_install --symlink web

# Clear Cache

php site_customer/console --env=prod cache:clear

# Dump assets

php site_customer/console assetic:dump --env=prod web

# Run cronjobs

php site_customer/console ezpublish:legacy:script runcronjobs.php --siteaccess customer_user_de
```

cjw #ezsummer 54/64

Multiple Apps: Multi-Site-Setup (old)

- · You can use one development environment with many projects
- · You can use one or more production servers or
- easily check out customer to different servers
- all customer are encapsulated apps
- solid and proven for more than 1,5 years
- Examples ...

cjw #ezsummer 55/64

Introducing CJW MultiSiteBundle

Although the first approach works fine, it has several drawbacks:

- · Application code scattered at different places (site directory, bundle, legacy extension), hard to maintain in VCS, hard to deploy
- Redundancy in config files
- No global settings
- No central site activation/administration
- Goal: keep everything in one place!

cjw #ezsummer 56/64

CJW MultiSiteBundle Features

- · Boots kernel and environment based on domain name mappings
- Handles local, staging and live domain names
- Allows for global activation of bundles
- Allows for global settings
- Provides a common console for all sites
- Caches domain name mappings
- Moves cache and log files away from the ezpublish folder
- · more to come ...

cjw #ezsummer 57/64

cjwpublish Directory

The cjwpublish application directory sits next to the ezpublish directory.

```
cjwpublish
config
config
cjwpublish.yml <-- defines active bundles
config.yml <-- allows for global settings
CjwPublishKernel.php <-- inherits from CjwMultiSiteKernel.php
CjwPublishCache.php <-- inherits from CjwMultiSiteCache.php
console</pre>
```

cjw #ezsummer 58/64

Symfony's app directory is back

Site Bundle Directory Layout

```
src
c Cjw
siteExampleBundle
app
config
cjwpublish.yml
config.yml
ezpublish.yml
...
CjwSiteExampleKernel.php
CjwSiteExampleCache.php
Controller
...
contains domain mappings
contains
```

cjw #ezsummer 59/64

Caveats

Adjustments needed in config.yml to reflect different relative location of kernel

```
assetic:

...

read_from: %kernel.root_dir%/../../../web

write_to: %kernel.root_dir%/../../../web

...

ez_publish_legacy:
...

root_dir: %kernel.root_dir%/../../../ezpublish_legacy

parameters:
ezpublish.kernel.root_dir: %kernel.root_dir%/../../../vendor/ezsystems/ezpublish-kernel
```

More problems of this kind expected!

cjw #ezsummer 60/64

Project Status

- · Currently in private Beta, not yet released
- · Ideas and Feedback welcome
- · Public Beta in October
- info@cjw-network.com
- https://github.com/cjw-network/MultiSiteBundle

cjw #ezsummer 61/64

Ressources

Slides as PDF

- See src/Cjw/SummerCampBundle/Resources/doc/learnings.pdf
- https://github.com/cjw-network/SummerCampBundle/Resources/doc/learnings.pdf

Slides (Source)

https://github.com/dfritschy/cjw-summercamp-slides

CJW MultiSiteBundle

- https://github.com/cjw-network/MultiSiteBundle
- info@cjw-network.com

cjw #ezsummer 62/64



http://vote.netgenlabs.com/

Please Vote!

<Thank You!>

Ekkehard Dörre

http://share.ez.no/community/profile/7431

@ekkeD

http://www.coolscreen.de

Donat Fritschy

https://github.com/dfritschy

http://share.ez.no/community/profile/10451

@webmanufaktur

http://www.webmanufaktur.ch

