Hosting symfony applications

Welcome

Take a seat,
make yourself confortable

Hosting symfony applications

- **Who ?**
  - Grégoire HUBERT
  - Freelance symfony consultant
  - Working with gnu/linux since the 20th century
  - Has been working for a hosting company
  - Has been working for Sensio
  - Ex Sf 1.0 release manager
  - Was in charge of symfony's websites

Hosting symfony applications

- **What ?**
  - symfony 1.x

  - It's a PHP5 framework for web developpers
  - It eases developpers and maintainers tasks
  - Can be configured for performance and security
  - Can be hosted on main available OSs

Hosting symfony applications

- **How ?**
  - SF Live 2010 training

  - Creating a new hosting nest
  - Tunning Sf for your hosting plate-forme
  - Using specialized backends
  - Tunning your hosting plate-forme for Sf
  - Tunning your application for performances

Hosting symfony applications

- Before we begin, 2 softwares you have to know:
  - Bash
    - Reach and tune every part of the system
    - Because everything is File with an Unix system
    - Trying to live without the sell is harder than learning the shell
    - Think SSH, think GNU/Screen
  - Vi(m)
    - Powerful editor installed with every Unix plate-form (even OSX)
    - CLI editor able to deal with huge files
    - Even Emacs users use Vi(m)

Creating a new hosting nest

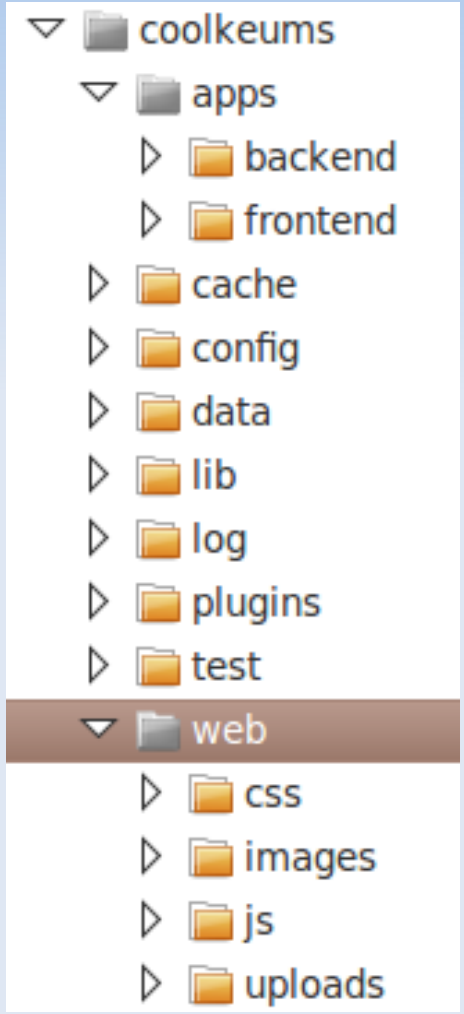
- What tools do I need ?
- What is a symfony project ?
- Configuring PHP
- Setting up symfony : the lib
- Configuring Virtualhost for Apache
- Deploying a symfony project
- Front controllers and environments
- Rsync_ignore file

Creating a new hosting nest

- What tools do I need ?
  - PHP 5.2 (see http://www.symfony-project.org/installation)
  - Apache 2.x
    - mod_php5
    - mod_rewrite (optionnal but recommended)
  - A Database (optionnal but recommended)
    - Listening on TCP/IP

What is a symfony project ?

- One or more applications
- Cache and Logs
- Librairies
- A Web directory
- A command line tool



What is a symfony project ?

- One or more applications
  - Example : frontend & backend
  - Each application has its own configuration
    - /apps/application/config
  - By default, each application uses its own session
    - /apps/application/lib/myUser.class.php
  - Differences between a project and an application ?
    - Data source model
    - Deployment & web materials
    - Libs & plugins

What is a symfony project ?

- Cache and logs
  - A project uses a cache directory to compile and store configuration, file detection mechanisms etc
  - Logs are used for developments and debugging purposes, they can grow very big if no attention is paid
  - This means write access for the apache user
  - This means no backups nor RAID controllers for theses directories (we'll see later)

What is a symfony project ?

- A web directory
  - It is intended to be the DocumentRoot of apache's virtualhost
  - ➔ No external access to libs, conf, cache etc
  - ➔ Only few .PHP files in this directory
  - Base of js, images, css etc
  - Uploads directory as well (write access)
  - .htaccess means AllowOverride or import it into apache conf file.

What is a symfony project ?

- A command line tool : «symfony» to be used from within a shell
  - php symfony (all plateforms)
  - ./symfony (*nix plateforms)
  - ./symfony –help
  - ./symfony list project
  - ./symfony --version

```
coolkeums$ ./symfony --version
symfony version 1.4.2-DEV (/var/www/dev/sf1.4/lib)
```

What is a symfony project ?

- Libraries
  - Symfony is a librairy of a symfony project
  - Is may be included with the project
  - It uses the Autoloading mechanism
    - ➔ Which stores its index in the cache
  - Think about this in your deployment process
  - config/ProjectConfiguration.class.php

What is a symfony project ?

- What about access rights ?
  - Everything is read only but
  - Cache, logs and web upload dirs
  - ./symfony project:permissions

Configuring PHP

- php.ini, beware of CLI and apache configuration
  - magic_quotes_gpc = Off (double escaping)
  - short_open_tags = Off (Do you like XML ?)
  - memory_limit = 128M (Thanks your ORMs)
  - open_basedir must include symfony, cache, logs and project directories
  - error_reporting and display_errors are set by symfony

Configuring PHP

- What modules do you need ?
  - PDO for databases
  - APC/X-CACHE/ACCELERATORS for production
  - X-DEBUG for developpment
  - MEMCACHE/MONGODB for production
  - UTF-8 support ?
- Test your PHP compilation / installation
  - ./symfony symfony:test

Configuring a VirtualHost

- For developpment, because we like .htaccess

```
1 <VirtualHost *:80>¤
2   ServerName coolkeums.localhost¤
3   DocumentRoot /var/www/dev/coolkeums/web¤
4 ¤
5   <Directory /var/www/dev/coolkeums/web>¤
6     AllowOverride All¤
7     Order Allow,Deny¤
8     Allow from All¤
9   </Directory>¤
```

Configuring a VirtualHost

- In production because we like performances

```
 5    <Directory /var/www/dev/coolkeums/web>¤
 6      AllowOverride None¤
 7      Order Allow,Deny¤
 8      Allow from All¤
 9  ¤
10      Options +FollowSymLinks +ExecCGI¤
11      <IfModule mod_rewrite.c>¤
12        RewriteEngine On¤
13  ¤
14        # we check if the .html version is here (caching)¤
15        RewriteRule ^$ index.html [QSA]¤
16        RewriteRule ^([^.]+)$ $1.html [QSA]¤
17        RewriteCond %{REQUEST_FILENAME} !-f¤
```

.htaccess

Configuring a VirtualHost

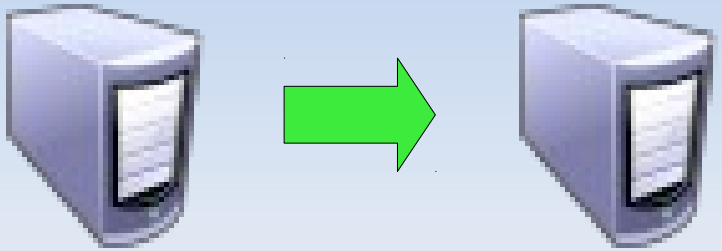
- Path to symfony and plugins assets

```
12   Alias /sf /var/www/dev/sf1.4/data/web/sf¤
13   <Directory /var/www/dev/sf1.4/data/web/sf>¤
14     Order Allow,Deny¤
15     Allow From All¤
16   </Directory>¤
17 ¤
18   Alias /sfDoctrinePlugin /var/www/dev/sf1.4/lib/plugins/sfDoctrinePlugin/web¤
19   <Directory /var/www/dev/sf1.4/lib/plugins/sfDoctrinePlugin/web>¤
20     Order Allow,Deny¤
21     Allow From All¤
22   </Directory>¤
23 </VirtualHost>¤
```

- Or create symlinks
- Or copy content

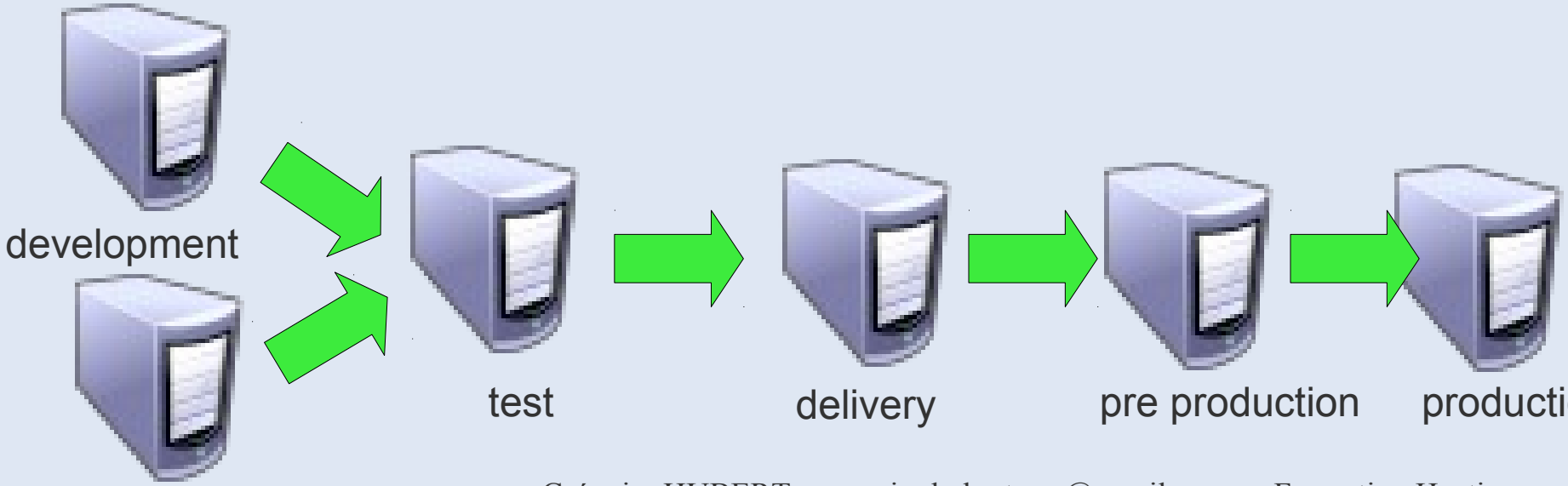
Deploying a symfony project

- It can be very different workflows



development production

Your VCS (svn, git, Hg)
Rsync over SSH



Deploying a symfony project

- Using a VCS is not an option for developpers
  - Allow collaborative developpment
  - Log activity on the project
  - Integrate with bug tracking systems
  - Allow branching and tagging
- Is the first step to create a master copy

Deploying a symfony project

- Different servers means different configurations
  - developpment:
    - Database hosted on localhost
    - No cache
    - Output error messages on screen
    - Lot of logs
  - production:
    - Remote pool of databases
    - Cache and opcode accelerators
    - No error output on screen
    - Aiming at performances

Deploying a symfony project

- Environments = multiple configurations
  - config/database.yml

```
4  all:¤
5    doctrine:¤
6      class: sfDoctrineDatabase¤
7      param:¤
8        dsn:      pgsql:host=localhost;dbname=coolkeums¤
9        username: toto¤
10       password: [illegible]
11 ¤
12 prod:¤
13   doctrine:¤
14     class: sfDoctrineDatabase¤
15     param:¤
16       dsn:      pgsql:host=[illegible];dbname=coolkeums¤
17       username: [illegible]¤
18       password: [illegible]¤
19 ¤
```

Deploying a symfony project

- Default values are overrided by
- «all» settings which are overrided by
- Your environment settings
- Environments settings control:
  - Config file refresh in the cache (admin generator)
  - Error reporting, logs & exceptions handling
  - Response compression & Etag
  - Cache ...

Deploying a symfony project

- Where is defined the current environment ?
- The front controllers
  - web/my_app.php

```
4 require_once(dirname(__FILE__).'/../config/ProjectConfiguration.class.php');¤
5 ¤
6 $configuration = ProjectConfiguration::getApplicationConfiguration('frontend', 'prod', false);¤
7 sfContext::createInstance($configuration)->dispatch();¤
```

  - web/my_app_dev.php

```
 5 if (!in_array(@$_SERVER['REMOTE_ADDR'], array('127.0.0.1', '::1')))¤
 6 {¤
 7   die('You are not allowed to access this file. Check '.basename(__FILE__).' for more information
   .');¤
 8 }¤
 9 ¤
10 require_once(dirname(__FILE__).'/../config/ProjectConfiguration.class.php');¤
11 ¤
12 $configuration = ProjectConfiguration::getApplicationConfiguration('frontend', 'dev', true);¤
13 sfContext::createInstance($configuration)->dispatch();¤
```

Deploying a symfony project

- Be carefull not to deploy developpment controllers on production servers !!

Deploying a symfony project

- Deploying to staging and production servers
- Rsync over SSH
  - Secured connections
  - Incremental transfer
  - File selection
  - Embedded in symfony CLI tool
  - ./symfony help project:deploy

Deploying a symfony project

- properties.ini
  - Config file for the *deploy* task
  - Declare servers and connection infos
  - You do NOT want to set your SSH pass here :)

```
name=myproject

[production]
  host=myapp.example.com
  port=22
  user=myuser
  dir=/home/myaccount/myproject/
```

./symfony project:deploy production --go

Deploying a symfony project

- The default options are
  - *-avz --force --delete --progress --dry-run*
  - You can specify your own options with *--rsync-options*
    - Useful if you want to ignore date changes for transmitting files (-c)
  - You have to specify the *--go* option to get ride of the --dry-run to commit your transfer

Deploying a symfony project

- Filtering files not to be transfered
  - You do not want to send frontend_dev.php to production
  - You do not want to send .svn or .git files/directories
  - You do not want to transfer editors temporary files
    - config/rsync_exclude.txt

Deploying a symfony project

- You can specify a pattern to select the files you want to synchronize
  - Set a config/rsync_include.txt
- You can even list the files you want to sync
  - Set a config/rsync.txt
  - Most of the time, this file is automatically generated by an external process.

Deploying a symfony project

- Depending on the server you sync from, theses files may not be the same
  - *--rsync-dir* sets the base directory of these files
    - config/$rsync_dir/*.txt
- For the curious:
  $sf_lib_dir/task/project/sfProjectDeployTask.class.php

Tunning symfony

- Tunning Sf for your hosting plate-forme
  - Cache
  - Logs (do you really need logs ?)
  - Webdir
- Can be placed on proper places on the disks for
  - Backup policy
  - Performances
  - Content sharing

Tunning symfony

- Why ?
  - Cache works better with no RAID
  - Cache does not need backup
  - Project dir may only needs RO access
    - Check if data/ needs to be writable
  - Only web dir is to be shared in a cluster
- The ProjectConfiguration file
  - setCacheDir()
  - setLogDir()
  - setWebDir()

Tunning symfony

- What's in the cache ?
  - Configuration files
  - Autoloading indexes
  - Use memory cache backend (APC, X-cache ...)
  - View cache
- View cache may use another backend than files
  - factories.yml
  - Drivers for databases, mongoDb as a plugin
  - Sqlite => fast for small sites but does not scale !

Tunning symfony

- I18N and translation dictionnaries
  - Xliff => file based
  - Database => good but be careful on DB accesses
  - Cache backend => need a persistent backend
- Session storage
  - Only flat data, never ever store PHP objects
  - On disk by default (hard to share in a cluster)
  - In a database, see above

Tunning symfony

- Routing cache
  - Routing file is «compiled» as a PHP file with serialized entries
  - Can be very big and slow down the application
  - Use a memory cache system
  - lookup_cache_dedicated_keys if you use a memory cache system :)

Tunning symfony

- The routing
  - Use the lazy routes deserialization
  - Is better
    - If you have hundreds of routes
    - If the most demanded routes are on top
  - Worst otherwise

Tunning symfony

- The view cache
  - Is a programmer thing but
  - We can use a different backend
  - Maybe very big (> 2Gb)
  - *settings.yml*
    - lazy_cache_key
  - *./symfony project:optimize*

Tunning your plate-form

- Apache and the run for memory
  - Using an ORM in PHP means huge memory
  - Per apache process !
  - memory_limit = 128M in php.ini
    - => only 40 processes for 4Gb ram
  - Big routing files can also lead to memory overflow
  - Apache mod_status is your friend
  - Top & htop are your friends

Tunning your plate-form

- This means finding a balance between
  - Number of apache processes running
    - StartServers
    - MinSpareServers
    - MaxSpareServers
    - MaxClients
  - Their lifetime
    - MaxRequestsPerChild

Tunning your plate-form

- Disks matter !
- Ram disks have not proven to be a good solution for performances
  - You NEED ram on your web server :)
- Ext4 is a good RO filesystem
- RO can be shared on a NAS via iSCSI
- ReiserFS4 is a good RW FS for small files
-

Tunning your plate-form

- Use (one) opcode cache system
  - APC
  - X-Cache
  - PHPAccelerator
- Works only with PHP as apache module
- Reduces the user CPU time
- Can be used as a cache backend for routing by example

Tunning your project

- Some plugins may help you to improve performances
  - SfSuperCachePlugin
  - SfDynamicsPlugin
  - SfStoragePerformancePlugin

Tunning your project

- SfSuperCachePlugin
  - Creates a HTML copy of every cached URL
  - Needs apache's rewrite engine
  - Can make your *web/* very big

    http://www.symfony-project.org/plugins/sfSuperCachePlugin
  - Makes a great job with pure Ajax websites

Tunning your project

- SfDynamicsPlugin
  - Pack all JS and CSS of your pages in one file
  - http://dynamics.dakrazy.net
  - Packed files are put in *web/dynamics*
  - Allows to store js and css in *data/*

Tunning your project

- SfPerformanceStoragePlugin
  - 2 plugins in one
  - SfMemcachedCachePlugin
    - Memory key:value database
    - Server / client architecture
  - SfMongoDBCachePlugin
    - Disk based schemaless database
    - key:value database
    - Server / client architecture

Conclusions

- Enter the project as early as you can
- Early optimizations are the root of evil
  - => let yourself the ability to optimize when needed
- If you are not sure, test it !
  - Know the limit of your architecture
  - Let you have some spare ressource in case of a failure

Thank you

Questions ?