

Podstawy Symfony

v3.1

Plan

- Co warto wiedzieć na początku
- Podstawowe informacje o Symfony
- Instalacja Symfony
- Jak działa Symfony?
- Podstawy pracy z Symfony
- Bundle



**Co warto
wiedzieć na
początku?**

Wzorzec projektowy (design pattern)

W inżynierii oprogramowania uniwersalne, sprawdzone w praktyce rozwiązanie często pojawiających się, powtarzalnych problemów projektowych.

Pokazuje powiązania i zależności między klasami oraz obiektami. Ułatwia tworzenie, modyfikację, oraz pielęgnację kodu źródłowego.

- Jest opisem rozwiązania, a nie jego implementacją.
- Wzorce projektowe stosowane są w projektach wykorzystujących programowanie obiektowe.
- Źródło:
[http://pl.wikipedia.org/wiki/Wzorzec_projektowy_\(informatyka\)](http://pl.wikipedia.org/wiki/Wzorzec_projektowy_(informatyka))

Model-View-Controller (MVC)

Czym jest MVC

Jest to wzorzec architektoniczny służący do organizowania struktury aplikacji posiadających graficzne interfejsy użytkownika. Zakłada podział aplikacji na trzy połączone z sobą warstwy: **model**, **widok i kontroler**.

Model

Jest reprezentacją pewnego problemu, jej struktury danych. Model jest samodzielny.

Model zajmuje się dostarczaniem danych do wykorzystania w aplikacji. Do poprawnej pracy nie wymaga obecności dwóch pozostałych części MVC. Komunikacja z nimi zachodzi w sposób niejawny.

MVC

Widok

Jest odpowiedzialny za prezentację danych w obrębie graficznego interfejsu użytkownika.

Może składać się z podwidoków zarządzających mniejszymi elementami składowymi.

Widoki mają bezpośrednie referencje do modeli, z których pobierają dane, gdy otrzymują od kontrolera żądanie ich wyświetlenia.

Możliwe są różne widoki tych samych danych.

Kontroler

Jego zadaniem jest odbiór, przetworzenie oraz analiza danych wejściowych od użytkownika.

Po przetworzeniu odebranych danych kontroler może wykonać następujące czynności:

- zmienić stan modelu,
- odświeżyć widok,
- przełączyć sterowanie na inny kontroler.

Podstawowe informacje o Symfony

Co to jest framework?

Framework albo platforma programistyczna to szkielet do budowy aplikacji.

Definiuje on strukturę aplikacji oraz ogólny mechanizm jej działania, a także dostarcza zestaw komponentów i bibliotek ogólnego przeznaczenia do wykonywania określonych zadań.

Źródło:

<https://pl.wikipedia.org/wiki/Framework>

Programista tworzy aplikację, rozbudowując i dostosowując poszczególne komponenty do wymagań realizowanego projektu. Tworzy w ten sposób gotową aplikację.

Co to jest framework?

Frameworki bywają niekiedy błędnie zaliczane do bibliotek. Typowe cechy frameworków to:

Odwrócenie sterowania

Mechanizm sterowania jest narzucany przez framework a nie przez użytkownika.

Domyślne zachowanie

Framework ma domyślną konfigurację, która musi być użyteczna i dawać sensowny wynik.

Rozszerzalność

Poszczególne komponenty frameworka powinny być rozszerzalne przez programistę.

Zamknięta struktura wewnętrzna

Programista może rozbudowywać framework, ale nie poprzez modyfikację domyślnego kodu.

Źródło:

<http://pl.wikipedia.org/wiki/Framework>

Czym jest Symfony?

Symfony to pełnowartościowy framework służący do tworzenia dowolnej skali aplikacji webowych w PHP. Stworzony i utrzymywany przez firmę SensioLabs.

Symfony Components to zestaw bibliotek rozwiązujących typowe, powtarzalne zadania spotykane przy programowaniu aplikacji webowych. Mogą być one wykorzystywane pojedynczo i niezależnie od samego frameworku.

➤ <http://symfony.com>

Podstawowe korzyści

Szybkość

Programista skupia się na rozwiązywaniu problemów ściśle związanych z zadaniem (logiką biznesową), a nie na zadaniach okołobiznesowych, czy związanych z samą strukturą projektu.

Wzorce projektowe

Budowanie aplikacji w oparciu o dobre praktyki architektury oprogramowania.

Utrzymanie i rozwój aplikacji

Wsparcie techniczne, bogata baza wiedzy oraz aktualizacje.

Jakość

Wykorzystanie sprawdzonych rozwiązań w postaci gotowych komponentów o wysokiej jakości kodu.

Współpraca zespołowa

Ułatwiona współpraca zespołowa dzięki następującym zaletom:

- znany, dobrze udokumentowany kod,
- usystematyzowane pojęcia i narzędzia programistyczne.

Symfony

Dlaczego Symfony?

- reputacja
- licencja
- referencje
- społeczność
- innowacyjność
- elastyczność

Wersja frameworku

- Na zajęciach będziemy korzystać z Symfony w wersji 2.8. Jest to popularna wersja Symfony wspierana do listopada 2019 roku.
- Najnowszą wersją Symfony to 3.2 wydana w listopadzie 2016 roku. Nie jest to wersja LTS (Long Term Support).

Instalacja Symfony

Instalacja Symfony

➤ Osoby, które nie mają przygotowanego środowiska za pomocą naszego skryptu muszą sobie same zainstalować **Symfony**.

➤ <http://symfony.com/doc/current/book/installation.html>

Od wersji 2.7 jedynym polecanym sposobem tworzenia nowych projektów w oparciu o **Symfony** jest użycie narzędzia **Symfony Installer**.

Symfony Installer wymaga PHP 5.4 lub nowszego!

Instalacja Symfony

Systemy uniksowe

```
$ sudo curl -LsS http://symfony.com/installer -o /usr/local/bin/symfony  
$ sudo chmod a+x /usr/local/bin/symfony
```

System Windows

```
c:\> php -r "readfile\('http://symfony.com/installer'\);" > symfony  
c:\> move symfony c:\projects  
c:\projects\> php symfony
```

Instalacja Symfony

Utworzenie nowego projektu przy pomocy Symfony Installer:

Systemy uniksowe

```
$ symfony new nazwa_projektu wersja_symfony
```

System Windows

```
c:\projects\> php symfony new nazwa_projektu wersja_symfony
```


Instalacja Symfony

Utworzenie nowego projektu przy pomocy Symfony Installer:

Systemy uniksowe

```
$ symfony new nazwa_projektu wersja_symfony
```

System Windows

```
c:\projects\> php symfony new nazwa_projektu wersja_symfony
```

wersja_symfony - Jako parametr opcjonalny, należy podać wersję 2.8

Jak działa Symfony?

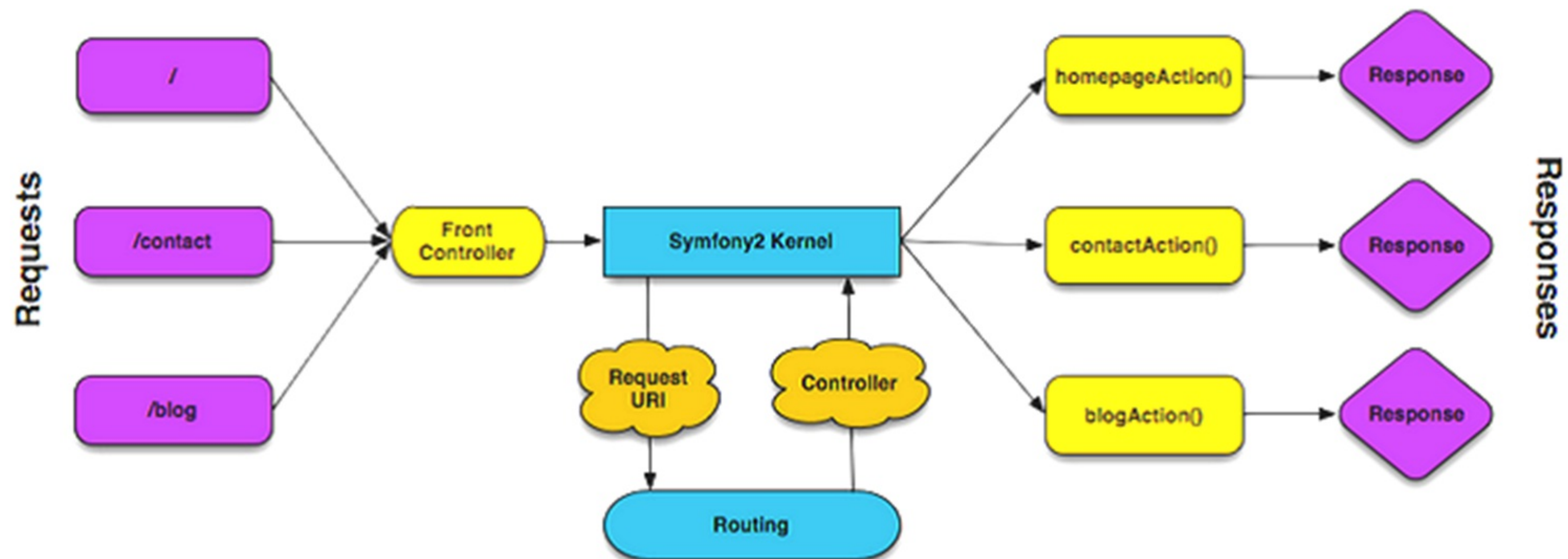
Jak działa Symfony

- Symfony zmienia podstawowy sposób działania aplikacji.
- Do tej pory różnym adresom w każdej z naszych aplikacji odpowiadały różne pliki PHP.



Jak działa Symfony?

W przypadku Symfony zasada jest trochę bardziej skomplikowana.



Co dokładnie się dzieje?

Zapytanie

Pierwsza różnica jest już w samym zapytaniu. Zamiast odwołania do konkretnego pliku - odwołujemy się do ścieżki.

STARA METODA	NOWA METODA
<code>/showUser.php</code>	<code>/user</code>
<code>/showPost.php</code>	<code>/post</code>

Adres

Nowy typ adresów wprowadzono w kilku celach:

- lepszej separacji adresu od naszego kodu,
- ukrycia nazw naszych plików na serwerze (bezpieczeństwo).

Łatwiej nam zapamiętać ten adres:

http://en.wikipedia.org/wiki/Barack_obama

niż ten:

http://en.wikipedia.org/w/index.php?title=Barack_obama

Co dokładnie się dzieje?

Adresy tworzone w ten sposób nazywamy URI (Uniform Resource Identifier). Są często stosowane w programowaniu stron internetowych.

Taki zapis możemy otrzymać również bez użycia frameworku.

Więcej na ten temat:

<http://www.addedbytes.com/articles/for-beginners/url-rewriting-for-beginners>

<http://www.smashingmagazine.com/2011/1/introduction-to-url-rewriting>

Jak działa Symfony?

Co się dzieje po otrzymaniu zapytania?

1. Zapytanie trafia do front controllera. Jest to plik PHP przejmujący wszystkie zapytania i przekazujący dalej do frameworka.
Dla Symfony jest to plik **/web/app.php** (lub **/web/app_dev.php** dla serwera deweloperskiego).
2. Front controller przesyła nasze zapytanie do silnika Symfony.
3. Silnik Symfony przesyła zapytanie do Routera, który zwraca informacje, jaki model i jaka akcja jest przypisana do tego adresu.
4. Silnik Symfony uruchamia odpowiedni kontroler i jego akcję.
5. Akcja zwraca widok, który powinien zostać wyświetlony użytkownikowi.

Podstawy pracy z Symfony

Struktura katalogów

app

Główny katalog, w którym znajduje się framework. Są tu też pliki konfiguracyjne i pliki zasobów.

bin

Katalog, w którym znajdują się dane Doctrine.

vendor

Biblioteki zainstalowane przez Composera (w tym katalogu nic nie zmieniamy).

src

Katalog, w którym znajdują się poszczególne bundle.

web

Folder „wystawiony” publicznie przez serwer HTTP, zawiera m.in.:

- front controller,
- zasoby CSS i JS,
- pozostałe zasoby tj. np. zdjęcia, pliki uploadowane przez użytkownika.

Konsola

Konsola to jeden z komponentów Symfony. Jest to zbiór narzędzi dostępnych z linii poleceń, które pozwalają zarządzać projektem oraz jego zasobami.

Aby wylistować dostępne polecenia konsoli, uruchamiamy ją bez parametrów:

```
php app/console
```

Narzędzia te pozwalają m.in.:

- zarządzać assetami oraz cachem aplikacji,
- uruchamiać serwer deweloperski,
- komunikować się z bazą danych,
- generować elementy aplikacji,
- debugować aplikację.

Zawiera także warstwę abstrakcji służącą do budowania własnych poleceń i narzędzi linii komend.

Uruchamianie serwera deweloperskiego

Symfony jest dystrybuowane z wbudowanym serwerem deweloperskim.

Aby go uruchomić, wystarczy przejść do katalogu głównego projektu i wpisać poniższą komendę:

```
php app/console server:start
```

Aby zatrzymać serwer wystarczy wpisać:

```
php app/console server:stop
```

Uruchamianie serwera deweloperskiego

System przygotowany przez Coders Lab

System przygotowany przez CL jest tak skonfigurowany, aby współpracować z wbudowanym serwerem Symfony.

Musi on być włączony na porcie **8080**:

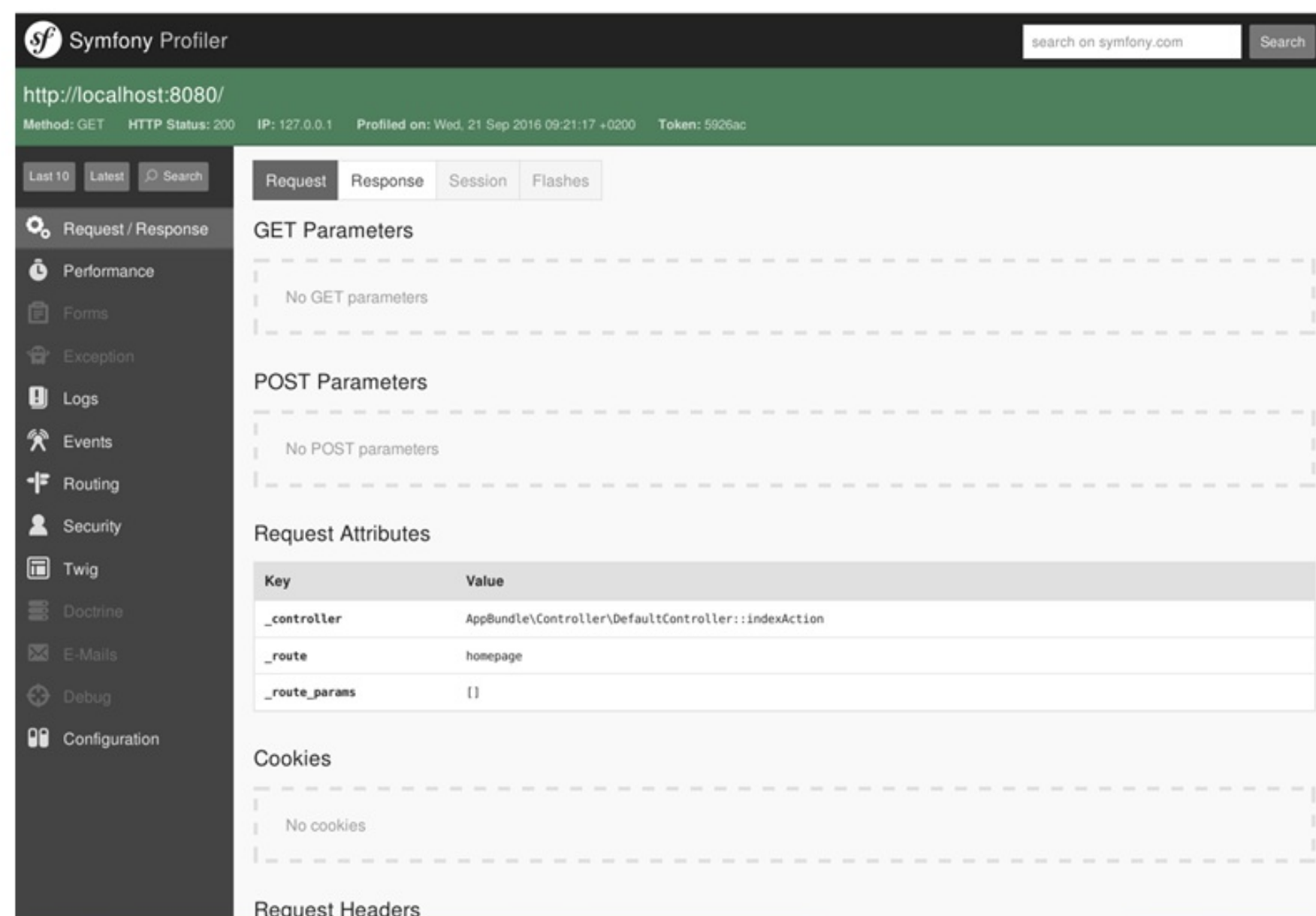
```
php app/console server:start 0.0.0.0:8080
```

Aby zatrzymać serwer wystarczy wpisać:

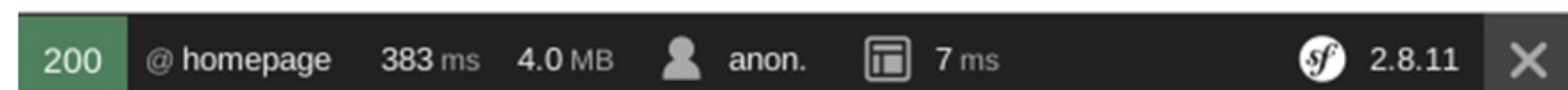
```
php app/console server:stop 0.0.0.0:8080
```

Symfony profiler

Serwer deweloperski uruchamia dodatkowo profiler, który wykorzystujemy do podglądu przydatnych informacji jak wydajność czy wartość zmiennych zapisanych w sesji.



The screenshot shows the Symfony Profiler interface. At the top, it displays the URL `http://localhost:8080/`, the method `GET`, HTTP status `200`, IP `127.0.0.1`, and the time profiled on `Wed, 21 Sep 2016 09:21:17 +0200`. The left sidebar contains a menu with options like Request / Response, Performance, Forms, Exception, Logs, Events, Routing, Security, Twig, Doctrine, E-Mails, Debug, and Configuration. The main content area is divided into tabs: Request, Response, Session, and Flashes. The 'Request' tab is active, showing 'GET Parameters' (No GET parameters), 'POST Parameters' (No POST parameters), 'Request Attributes' (a table with keys like `_controller`, `_route`, and `_route_params`), 'Cookies' (No cookies), and 'Request Headers'.



The bottom bar displays the following information: status code `200`, route `@ homepage`, execution time `383 ms`, memory usage `4.0 MB`, user `anon.`, a cache icon, and a time `7 ms`. On the right, there is a Symfony logo, the version `2.8.11`, and a close button.

Sposoby konfiguracji w Symfony

Na zajęciach będziemy korzystać z adnotacji (annotations).

Jeżeli generator będzie was pytał o sposób konfiguracji, należy wybrać tę metodę!

Adnotacje to specjalne komentarze, które wyglądają następująco:

```
/**  
*  
*/
```

Sposoby konfiguracji w Symfony

Na zajęciach będziemy korzystać z adnotacji (annotations).

Jeżeli generator będzie was pytał o sposób konfiguracji, należy wybrać tę metodę!

Adnotacje to specjalne komentarze, które wyglądają następująco:

```
/**  
*  
*/
```

→ Tu są dwie gwiazdki. Dzięki temu jest to adnotacja, a nie zwykły komentarz.

Kopiowanie zasobów

Po każdorazowym dodaniu zasobów (kodu JavaScript, pliku stylu, obrazków) do naszej aplikacji, musimy wywołać komendę, która skopiuje je do katalogu **/web**.

```
php app/console assets:install
```

Komendę tę zastosuj także zaraz po utworzeniu naszego projektu.



Bundle

Bundle

Co to jest?

- Symfony zbudowane jest na podstawie Bundli.
- Są to osobne części działające całkowicie niezależnie i implementujące pewną funkcjonalność.
- Bundle to w uproszczeniu plugin.
- W Symfony wszystko (łącznie z podstawową funkcjonalnością) jest Bundlem.

Do czego używamy?

- W naszym przypadku kod dopisanej przez nas funkcjonalności powinien znajdować się w osobnym Bundlu.
- Dzięki temu nasza funkcjonalność będzie łatwo przenoszalna między projektami.

Jak wygląda katalog Bundla?

Katalog każdego Bundla ma następujące podkatalogi:

/Controller

Tutaj znajdują się wszystkie kontrolery danego Bundla.

/DependencyInjection

Tutaj znajdują się klasy odpowiedzialne za Dependency Injection. Ten katalog nie zawsze istnieje.

/Resources/config

Konfiguracja danego Bundla.

/Resources/views

Widoki trzymane dla każdego kontrolera osobno (w katalogu z jego nazwą).

/Resources/public

Wszystkie zasoby potrzebne do działania Bundla (kopiowane do **/web** przy użyciu komendy **assets:install**).

Tests

Tutaj znajdują się testy dla danego Bundla.

Tworzenie nowego Bundla

Nowego Bundla tworzymy następującą komendą:

```
php app/console generate:bundle
```

Następnie generator przeprowadza nas przez proces tworzenia nowego Bundla. Musimy podać następujące dane:

- Namespace i nazwę (najlepiej, gdy są takie same).
- Sposób konfiguracji (annotacje).
- Ścieżkę do Bundla (zostawiamy podstawową).
- Czy ma wygenerować nam strukturę katalogu? (Tak).
- Czy ma automatycznie dopisać nasz Bundle do routingu i kernela? (2x tak).

Nazywanie naszych Bundli

Nazwa Bundla powinna spełniać następujące warunki:

- Być pisana za pomocą **camelCase**.
- Nie zawierać znaków specjalnych i spacji.
- Nie być dłuższa niż dwa słowa.
- Kończyć się słowem **Bundle**.

Usuwanie Bundla 1/3

Jeżeli chcemy usunąć Bundla, to powinniśmy wykonać następujące kroki:

1. Usunąć wpis znajdujący się w pliku **/app/config/routing.yml** zaczynający się od nazwy bundla.

Bundle o nazwie **Coderslab** wygląda następująco:

```
coderslab:  
  resource: "@CoderslabBundle/Controller/"  
  type:     annotation  
  prefix:   /
```

Usuwanie Bundla 2/3

2. Usunąć kod tworzący nowy obiekt Bundla. Znajduje się on w pliku **/app/AppKernel.php** w tablicy **\$bundles**.

```
$bundles = [  
    new Symfony\Bundle\FrameworkBundle\FrameworkBundle(),  
    new Symfony\Bundle\SecurityBundle\SecurityBundle(),  
    new Symfony\Bundle\TwigBundle\TwigBundle(),  
    new Symfony\Bundle\MonologBundle\MonologBundle(),  
    new Symfony\Bundle\SwiftmailerBundle\SwiftmailerBundle(),  
    new Symfony\Bundle\AsseticBundle\AsseticBundle(),  
    new Doctrine\Bundle\DoctrineBundle\DoctrineBundle(),  
    new Sensio\Bundle\FrameworkExtraBundle\SensioFrameworkExtraBundle(),  
    new AppBundle\AppBundle(),  
    new CoderslabBundle\CoderslabBundle(),  
];
```

Usuwanie Bundla 2/3

2. Usunąć kod tworzący nowy obiekt Bundla. Znajduje się on w pliku **/app/AppKernel.php** w tablicy **\$bundles**.

```
$bundles = [  
    new Symfony\Bundle\FrameworkBundle\FrameworkBundle(),  
    new Symfony\Bundle\SecurityBundle\SecurityBundle(),  
    new Symfony\Bundle\TwigBundle\TwigBundle(),  
    new Symfony\Bundle\MonologBundle\MonologBundle(),  
    new Symfony\Bundle\SwiftmailerBundle\SwiftmailerBundle(),  
    new Symfony\Bundle\AsseticBundle\AsseticBundle(),  
    new Doctrine\Bundle\DoctrineBundle\DoctrineBundle(),  
    new Sensio\Bundle\FrameworkExtraBundle\SensioFrameworkExtraBundle(),  
    new AppBundle\AppBundle(),  
    new CoderslabBundle\CoderslabBundle(),  
];
```

Usuwamy wpis tworzący Bundla

Usuwanie Bundli 3/3

3. Usuwamy katalog, w którym znajduje się Bundle.
Jest on zlokalizowany w `/src/nazwaBundla`

Zadania

Czas na zadania

Tydzień 1 - Dzień 1
Wstęp do Symfony