Symfony Cheat Sheet

A quick reference guide for Symfony, covering essential concepts, commands, and configurations.

Basic Commands

Console Commands

`php bin/console list`	Lists all available console commands.
`php bin/console help <command>`	Displays help for a specific command.
`php bin/console new:controller`	Generates a new controller.
`php bin/console make:entity`	Creates a new entity.
`php bin/console doctrine:migrations:migrate`	Executes pending database migrations.
`php bin/console cache:clear`	Clears the application cache.
`php bin/console server:run`	Starts the built-in PHP web server.
`php bin/console debug:router`	Lists all defined routes.

Environment Variables

Environment variables are defined in `.env` file. Use `$_ENV['VARIABLE_NAME']` or `getenv('VARIABLE_NAME')` to access them.

Example:

```
APP_ENV=dev
APP_SECRET=s3cretf0rt3st
```

In your code:

```
$appEnv = $_ENV['APP_ENV'];
```

Routing

Route Annotations

Example of using annotations for routing in a controller:

```
use
Symfony\Component\Routing\Annotation\Route;

class MyController extends AbstractController
{
    /**
     * @Route("/my-route",
name="my_route_name")
     */
    public function myAction()
    {
        // ...
    }
}
```

Route Parameters

Required Parameter	`@Route("/blog/{id}", name="blog_show")`
Optional Parameter	`@Route("/blog/{id?}", name="blog_show")`
Parameter with Requirements	`@Route("/blog/{id}", name="blog_show", requirements={"id"="\d+"})`

Generating URLs

Generating URLs in your code:

```
$url = $this->generateUrl(
    'route_name',
    ['id' => 123]
);
```

Controllers and Views

Controller Basics

A simple controller example:

```
use
Symfony\Bundle\FrameworkBundle\Controller\Abst
ractController;
use Symfony\Component\HttpFoundation\Response;
use
Symfony\Component\Routing\Annotation\Route;

class MyController extends AbstractController
{
    /**
     * @Route("/", name="homepage")
     */
    public function index(): Response
    {
        return $this-
>render('index.html.twig', [
            'controller_name' =>
'MyController',
        ]);
    }
}
```

Rendering Templates

Rendering a template	`$this->render('template.html.twig', ['name' => $name]);`
Rendering a template with HTTP status	`$this->render('template.html.twig', ['name' => $name], new Response(null, 201));`

Passing Data to Templates

Example of passing data to a Twig template:

```
return $this->render('profile.html.twig', [
    'name' => $user->getName(),
    'age'  => $user->getAge(),
]);
```

Services and Dependency Injection

Defining Services

Services are typically defined in `config/services.yaml`:

```
services:
    App\MyService:
        arguments: ['@logger']
```

Using Services

In a Controller:	`public function index(MyService $myService)`
	`{`
	`  $myService->doSomething();`
	`}`

Autowiring	Symfony automatically injects services based on type hints.

Service Tags

Example of tagging a service:

```
services:
    App\MyEventListener:
        tags:
            - { name: 'kernel.event_listener', event: 'kernel.request', method: 'onKernelRequest' }
```