Symfony □□

- □□
  - □□□□□ Symfony □□
  - □ Symfony □□□□□□□
  - □□
  - □□□

□□

□□□□□ Symfony □□

□□□□

□□□□□□□□ Symfony □□□□□□□□□□

- □□ PHP 8.2 □□□□□□□□□ PHP □□□□□□□ PHP 8 □□□□□□□□□□□□ Ctype □ iconv □ PCRE □ Session □ SimpleXML□ Tokenizer □
- □□ Composer □□□□□ PHP □□

□□□□□ Symfony CLI □□□□□□□□□□□□□□□□□□□ symfony □□□□□□□□□□□□□□□□□□□□□ Symfony □□□□□□□□□□□□

symfony □□□□□□□□□□□□□□□□□□□□

```
composer create-project symfony/skeleton:"7.1.*" my_project_directory
```

□□□□□□□□□□□□ Symfony □□□□□□□□□□□□□□□□□□ my_project_directory/ □□□□□□□□□□□□□□□□□□

□□□□□

□□ Symfony □□□□□□□□□□□□□□□□□□□□□□□□ Symfony □□□□□□□□□□□□□□□□□□□□

□□ Symfony CLI□□□□ symfony
□□□□

□□

□ Symfony □□□□□□

□□□□□□□□ HTML □□□ JSON □□□□□□□□□□

- □□□□□□□□□□□□□□□□□□□□ PHP □□□□□□□□□□□□□□□□□□□□ Symfony Response□□□□□□□□□□ HTML □□□ JSON □□□□□□□□ PDF □□□□□□□
- □□□□□□□□□□□ URL□□□ /about □□□□□□□□

□□□□□□□□□□□

□□□□□□□□□ - /lucky/number - □□□□□□□□□□□□□□□□□□□□

```
+ use Symfony\Component\Routing\Attribute\Route;

 class LuckyController
 {
+     #[Route('/lucky/number')]
     public function number(): Response
     {
         // this looks exactly the same
     }
 }
```

□□□□□□□□□□ Symfony Web □□□□□□□□ http://localhost:8000/lucky/number□□□□

□□□□□□□□□□

Web □□□□□□□□□□□□

Symfony□□□□□□□□□□ Web □□□□□□□□□□□□

```
    public function number(): Response
    {
        $number = random_int(0, 100);

        return $this->render('lucky/number.html.twig', [
            'number' => $number,
        ]);
    }
}
```

□□□□□□ templates/□□□□□□□□□□□□ Twig □□□□□□□□□□□□□ templates/lucky □□□□□□□□□ number.html.twig□□□

```
{# templates/lucky/number.html.twig #}
<h1>Your lucky number is {{ number }}</h1>
```

{{ number }}□□□□□ Twig □□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□ Web □□□□□□□□□□□□□□□□□□□□□□

□□

□□

□□□

```
// src/Controller/BlogController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogController extends AbstractController
{
    #[Route('/blog', name: 'blog_list')]
    public function list(): Response
    {
        // ...
    }
}
```

□□□□□□□□□□ blog_list □□□□□□□□□□□□ /blog URL □□□□□□□□□□□□□□□□□□ BlogController □□ list□□ □□□

□□□□ □ blog_list□ □□□□□□□□□□□□□ URL □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□ YAML□ XML □ PHP □□□□□□□

□□□□□□□ YAML□ XML □ PHP □□□□□□□□□□□□□□□□□□□□

□□ HTTP □□

□□□□□□□□□□□ HTTP □□□ GET□ POST□ PUT □□□□ methods □□□□□□□□□□□□□□□□

```
// src/Controller/BlogApiController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogApiController extends AbstractController
{
    #[Route('/api/posts/{id}', methods: ['GET', 'HEAD'])]
    public function show(int $id): Response
    {
        // ... return a JSON response with the post
    }

    #[Route('/api/posts/{id}', methods: ['PUT'])]
    public function edit(int $id): Response
    {
        // ... edit a post
    }
}
```

□□□□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□ condition □□□

```
// src/Controller/DefaultController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class DefaultController extends AbstractController
{
```

```
    #[Route(
        '/contact',
        name: 'contact',
        condition: "context.getMethod() in ['GET', 'HEAD'] and request.headers.get('User-Agent') matches
'/firefox/i'",
        // expressions can also include config parameters:
        // condition: "request.headers.get('User-Agent') matches '%app.allowed_browsers%'"
    )]
    public function contact(): Response
    {
        // ...
    }

    #[Route(
        '/posts/{id}',
        name: 'post_show',
        // expressions can retrieve route parameter values using the "params" variable
        condition: "params['id'] < 1000"
    )]
    public function showPost(int $id): Response
    {
        // ... return a JSON response with the post
    }
}
```

condition □□□□

#[AsRoutingConditionService(alias: 'route_checker')] class RouteChecker { public function check(Request $request): bool { // ... } }

□□□□□ service□□ □□□ conditions □□□□□□□ ```

```
// Controller (using an alias):
#[Route(condition: "service('route_checker').check(request)")]
// Or without alias:
#[Route(condition: "service('App\\\Service\\\RouteChecker').check(request)")]
```

□□□□□□□□□□□□□ PHP□□□□□□ condition □□□□□□□□□ PHP □□□□□□□□□□

□□□□

□□□□□□□□□□□□□□□□□□□□□□ Symfony □□□□□□□□□□□□□□□□□□□□□ debug □ router □□□□ Symfony □□□□□□□□□□□□□□□□□□

```
php bin/console debug:router

----------------  -------  -------  -----  ------------------------------------------
Name              Method   Scheme   Host   Path
----------------  -------  -------  -----  ------------------------------------------
homepage          ANY      ANY      ANY    /
contact           GET      ANY      ANY    /contact
contact_process   POST     ANY      ANY    /contact
article_show      ANY      ANY      ANY    /articles/{_locale}/{year}/{title}.{_format}
blog              ANY      ANY      ANY    /blog/{page}
blog_show         ANY      ANY      ANY    /blog/{slug}
----------------  -------  -------  -----  ------------------------------------------
```

□□□□□□□□□□□□□□□□□□□

```
| Path        | /lucky/number/{max}                                |
| ...         | ...                                                |
| Options     | compiler_class: Symfony\Component\Routing\RouteCompiler |
|             | utf8: true                                         |
+-------------+----------------------------------------------------+
```

□□□□□□□ router□ match□□□□□□□□□□□□□□ URL□□□□□ URL □□□□□□□□□□□□□□□□□□□□□

```
php bin/console router:match /lucky/number/8

  [OK] Route "app_lucky_number" matches
```

□□□□

□□□□□□□ URL □□□□□□□□□□□□ /blog □□□□□□□□□□□□□□□□□□□□

```
        // ...
    }
}
```

□□□□□□□□□□□□ {slug}□□□□□ PHP □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□ /blog/my-first-post URL□ Symfony □□□ BlogController □□□ show□□ □□□□□ $slug = 'my-first-post' □□□□□ show□□ □□□

□□□□□□□□□□□□□□□□□□□

requirements □□□□□□□□□□□□□□□ PHP □□□□□□□□□□□□□□□□□□□□□□□□ \d+ □□□□□□□□□□□□□□□□□□□□□□

```
URL□Route□□□ Parameters □□
/blog/2□blog_list□$page = 2
/blog/my-first-post□blog_show□$slug = my-first-post $slug □□□□
```

□□□□□□□□□□□□□ {parameter_name} . □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

```
// src/Controller/BlogController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogController extends AbstractController
{
    #[Route('/blog/{page<\d+>}', name: 'blog_list')]
    public function list(int $page): Response
    {
        // ...
    }
}
```

□□□□

□□□□□□□□ blog_list □ URL □ /blog/{page}□□□□□□□ /blog/1□□□□□□□□□□□□□□□ /blog□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□ {page} □□□□□□□□□□□□□□□ /blog □□□ blog_list□□□□□□□□□□□□□□□ controller □□□□□□□□□□□□□□□□□□□□□□□□□□□□ defaults □□□□□

```
// src/Controller/BlogController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
```

```
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogController extends AbstractController
{
    #[Route('/blog/{page}', name: 'blog_list', requirements: ['page' => '\d+'])]
    public function list(int $page = 1): Response
    {
        // ...
    }
}
```

□□□□□□□□ /blog □□ blog_list□□□□□□ $page □□□□□ 1□

□□□□□□□□□ URL □□□□□□□□□□□□□□□□□□□□□□□□ /blog/1 □□□ /blog □□□□□□□□□□□□ □□□□ /blog/{□ page}

□ requirements □□□□□□□□□□ {parameter_name?default_value} □□□□□□□□□□□ □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

```
// src/Controller/BlogController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogController extends AbstractController
{
    #[Route('/blog/{page<\d+>?1}', name: 'blog_list')]
    public function list(int $page): Response
    {
        // ...
    }
}
```

□□□□□

Symfony □□□□□□□□□□□□□□□□□□□□

YAML □ XML □□□□

```
// src/Controller/BlogController.php
namespace App\Controller;

use App\Entity\BlogPost;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogController extends AbstractController
{
    // ...

    #[Route('/blog/{slug}', name: 'blog_show')]
    public function show(BlogPost $post): Response
    {
        // $post is the object whose slug matches the routing parameter

        // ...
    }
}
```

□□□

```
class OrderController extends AbstractController
{
    #[Route('/orders/list/{status}', name: 'list_orders_by_status')]
    public function list(OrderStatusEnum $status = OrderStatusEnum::Paid): Response
    {
        // ...
    }
}
```

□□□□

□□□

```
    }
}
```

□□□□

□□□□ defaults □□□□□

```
class DefaultController extends AbstractController
{
    #[Route('/share/{token}', name: 'share', requirements: ['token' => '.+'])]
    public function share($token): Response
    {
        // ...
    }
}
```

□□□□

Route alias □□□□□□□□□□□□□□□

```
# config/routes.yaml
new_route_name:
    alias: original_route_name
```

□□□□□□ original_route_name □ new_route_name □□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□

□□□□□□□□□□□□□□□□□□□

```
    version: '1.2'
    message: 'The "%alias_id%" route alias is deprecated. Do not use it anymore.'
```

□□□□□□□□□ new_route_name □□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□□□□□□□□□□□□ %alias_id% □□□□□□□□□□□□□□□□□□□□□□□□□□□ %alias_id% □□□□

□□□□□□

□□□□□□□□□□□□□□□□□□□

□□□□□□ index□□ □□□□□□□□ blog_index□□ URL □□ /blog/{_locale}□ show□□ □□□□□□ blog_show□□□□ URL □□ /blog/{_locale}/posts/{slug}□□□□□□□□□□ _locale □□□□□ class □□□□□□□□□□□□□□

□□□□□□□□

Symfony □□□ Request □□□□□□□□□□□□ name □ parameters□□□□□ “request attributes” □□□□□□□ Request □□□□□□□□□□□□□□

```
// src/Controller/BlogController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class BlogController extends AbstractController
{
    #[Route('/blog', name: 'blog_list')]
    public function list(Request $request): Response
    {
        $routeName = $request->attributes->get('_route');
        $routeParameters = $request->attributes->get('_route_params');

        // use this to get all the available attributes (not only routing ones):
        $allAttributes = $request->attributes->all();

        // ...
    }
}
```

□ services □□□□□□□□□ RequestStack □□□□□□□□□□□□□□□□□ Twig □□ app □□□□□□□□□□ □ app.current_route□ □□□□ □ app.current_route_parameters□□

□□□□

Symfony
□□□

□□□□□□□□□□ URL

□□□□□□ URL □□ UNIX □□□□□□□□□□□□□□□□ https://example.com/foo/ □□□□□□□□□□□ □ https://example.com/foo□□□□□□□ URL □□□□□□□□□□□□□□□□□□□ URL □□□□ URL □□□□□□□□□□□□□□□□□

Symfony □□□□□□□□□□□□□□□□□ URL □□□□□□□□□□□ GET □ HEAD □□□□

□□□□

□□□□□□ host □□□□□□□□□□□□ HTTP □□□□□□□□□□□□□□□□□□□

```
// src/Controller/MainController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class MainController extends AbstractController
{
    #[Route(
        '/',
        name: 'mobile_homepage',
        host: '{subdomain}.example.com',
        defaults: ['subdomain' => 'm'],
        requirements: ['subdomain' => 'm|mobile'],
    )]
    public function mobileHomepage(): Response
    {
        // ...
    }

    #[Route('/', name: 'homepage')]
    public function homepage(): Response
    {
        // ...
    }
}
```

□□□□□□□□ subdomain □□□□□□□□□□□□□□□□□□□□□□□□□□□□ URL □□□□□□□□□□ subdomain □□

□□□□ □ i18n□

□□ URL □□□□□□□□□□□□□□□□□□□□□□□□

```
// src/Controller/CompanyController.php
namespace App\Controller;
```

```
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class CompanyController extends AbstractController
{
    #[Route(path: [
        'en' => '/about-us',
        'nl' => '/over-ons'
    ], name: 'about_us')]
    public function about(): Response
    {
        // ...
    }
}
```

□□□□□□□□□□□□ Symfony □□□□□□□□□□□□□□□□□ locale□

□□□□□□□□□□□□□□□□□□□□□□□□ locale □□□□□□□□□□□□ locale □□□□□□□□□□□□□□□□□□□□□□□□□ locale □□□□□□□□□□

```
# config/routes/attributes.yaml
controllers:
    resource: '../../src/Controller/'
    type: attribute
    prefix:
        en: '' # don't prefix URLs for English, the default locale
        nl: '/nl'
```

□□□□□□□□□□□□□□□□□□□

□□□□

□□□□□□□□ HTTP □□□□□□□□□

□□□ URL□□□□□□□□□□□□□□□ blog_show□□□□□□□□□□□□□□□ slug = my-blog-post □□

□□□□□□□□□□□□□□□

```
    {
        // generate a URL with no route arguments
        $signUpPage = $this->generateUrl('sign_up');

        // generate a URL with route arguments
        $userProfilePage = $this->generateUrl('user_profile', [
            'username' => $user->getUserIdentifier(),
        ]);

        // generated URLs are "absolute paths" by default. Pass a third optional
        // argument to generate different URLs (e.g. an "absolute URL")
        $signUpPage = $this->generateUrl('sign_up', [], UrlGeneratorInterface::ABSOLUTE_URL);

        // when a route is localized, Symfony uses by default the current request locale
        // pass a different '_locale' value if you want to set the locale explicitly
        $signUpPageInDutch = $this->generateUrl('sign_up', ['_locale' => 'nl']);

        // ...
    }
}
```

□□□□□□□□□□ AbstractController □□□□□□□□□□□

```
    public function someMethod(): void
    {
        // ...

        // generate a URL with no route arguments
        $signUpPage = $this->router->generate('sign_up');

        // generate a URL with route arguments
        $userProfilePage = $this->router->generate('user_profile', [
            'username' => $user->getUserIdentifier(),
        ]);

        // generated URLs are "absolute paths" by default. Pass a third optional
        // argument to generate different URLs (e.g. an "absolute URL")
        $signUpPage = $this->router->generate('sign_up', [], UrlGeneratorInterface::ABSOLUTE_URL);

        // when a route is localized, Symfony uses by default the current request locale
        // pass a different '_locale' value if you want to set the locale explicitly
        $signUpPageInDutch = $this->router->generate('sign_up', ['_locale' => 'nl']);
    }
}
```

□□□□□□ URL

□□□ Symfony □□□□□□□□□□□□□□□□□□□□□

□ JavaScript □□□ URL

□□□□ JavaScript □□□□□ Twig □□□□□□□□□ path□□ □ url□□ Twig □□□□ URL □□□□□□ JavaScript □□□□□□ escape□□ □□□□□□□□□ JavaScript □□□□□

```
<script>
    const route = "{{ path('blog_show', {slug: 'my-blog-post'})|escape('js') }}";
</script>
```

□□□□□□□□□ URL □□□□□ JavaScript □□□□□□□□□□□□□□□□□□□□□□□□□□□□□ FOSJsRoutingBundle□

□□□□□□ URL

□□□□□□ URL □□□□□□□□□□□□ URL □□□□□□□□□□□□ HTTP □□□□□□□□□□□□□□□□□ URL□□□□□ http://localhost/ □□□□□□□□□□□□□□□□□

□□□□□□□ default_uri □□□□□□□□□□ URL □□□□ “request context”□

```
# config/packages/routing.yaml
framework:
    router:
        # ...
        default_uri: 'https://example.org/my/path/'
```

□□□□□□□□□ URL □□□□□□□□□□□□

```
// src/Command/SomeCommand.php
namespace App\Command;

use Symfony\Component\Console\Command\Command;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Output\OutputInterface;
use Symfony\Component\Routing\Generator\UrlGeneratorInterface;
// ...

class SomeCommand extends Command
{
    public function __construct(private UrlGeneratorInterface $urlGenerator)
    {
        parent::__construct();
    }

    protected function execute(InputInterface $input, OutputInterface $output): int
    {
        // generate a URL with no route arguments
        $signUpPage = $this->urlGenerator->generate('sign_up');

        // generate a URL with route arguments
        $userProfilePage = $this->urlGenerator->generate('user_profile', [
            'username' => $user->getUserIdentifier(),
        ]);
```

```
        // by default, generated URLs are "absolute paths". Pass a third optional
        // argument to generate different URIs (e.g. an "absolute URL")
        $signUpPage = $this->urlGenerator->generate('sign_up', [], UrlGeneratorInterface::ABSOLUTE_URL);

        // when a route is localized, Symfony uses by default the current request locale
        // pass a different '_locale' value if you want to set the locale explicitly
        $signUpPageInDutch = $this->urlGenerator->generate('sign_up', ['_locale' => 'nl']);

        // ...
    }
}
```

□□□□□□□□

□□□□□□□

```
// src/Controller/SecurityController.php
namespace App\Controller;

use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class SecurityController extends AbstractController
{
    #[Route('/login', name: 'login', schemes: ['https'])]
    public function login(): Response
    {
        // ...
    }
}
```

□□□□□□□□ URL □□□□□ HTTPS□□□□□□□□□ path□□ Twig □□□□ URL □□□□□□□□□ HTTP □□□□□□□□□□□□□□□□□□□□□□ URL □□□□□ URL□

```
{# if the current scheme is HTTPS, generates a relative URL: /login #}
{{ path('login') }}

{# if the current scheme is HTTP, generates an absolute URL to change
  the scheme: https://example.com/login #}
{{ path('login') }}
```

□□□□□□□□□□ scheme □□□□□□□□□□ HTTP □□ /login URL□□□□□□□□□□□□ URL□□□□ HTTPS □□□

□□□□□□□□□□ HTTPS □□□□□□□□□□□□□□□□□□□□

□□ URI □□□□□□ URI □□□□□□□ URI□□□□□□□□□□□□□□□□□ URI □□□□□□□□
URI □□□□□□□□□□□□□□□□ URI □□□□□

Symfony □□□□□□ UriSigner □□□ URI □□□□□□□□□□□□□□□□□□□□□□□□□□□□

```
// src/Service/SomeService.php
namespace App\Service;

use Symfony\Component\HttpFoundation\UriSigner;

class SomeService
{
    public function __construct(
        private UriSigner $uriSigner,
    ) {
    }

    public function someMethod(): void
    {
        // ...

        // generate a URL yourself or get it somehow...
        $url = 'https://example.com/foo/bar?sort=desc';

        // sign the URL (it adds a query parameter called '_hash')
        $signedUrl = $this->uriSigner->sign($url);
        // $url = 'https://example.com/foo/bar?sort=desc&_hash=e4a21b9'

        // check the URL signature
        $uriSignatureIsValid = $this->uriSigner->check($signedUrl);
        // $uriSignatureIsValid = true

        // if you have access to the current Request object, you can use this
        // other method to pass the entire Request object instead of the URI:
        $uriSignatureIsValid = $this->uriSigner->checkRequest($request);
    }
}
```

□□□□□□□□□□□□□□ URI
□□□□□□□□□□□□□□□□□□□

```php
// src/Service/SomeService.php
namespace App\Service;

use Symfony\Component\HttpFoundation\UriSigner;

class SomeService
{
    public function __construct(
        private UriSigner $uriSigner,
    ) {
    }

    public function someMethod(): void
    {
        // ...

        // generate a URL yourself or get it somehow...
        $url = 'https://example.com/foo/bar?sort=desc';

        // sign the URL with an explicit expiration date
        $signedUrl = $this->uriSigner->sign($url, new \DateTimeImmutable('2050-01-01'));
        // $signedUrl = 'https://example.com/foo/bar?sort=desc&_expiration=2524608000&_hash=e4a21b9'

        // if you pass a \DateInterval, it will be added from now to get the expiration date
        $signedUrl = $this->uriSigner->sign($url, new \DateInterval('PT10S'));  // valid for 10 seconds from now
        // $signedUrl = 'https://example.com/foo/bar?sort=desc&_expiration=1712414278&_hash=e4a21b9'

        // you can also use a timestamp in seconds
        $signedUrl = $this->uriSigner->sign($url, 4070908800); // timestamp for the date 2099-01-01
        // $signedUrl = 'https://example.com/foo/bar?sort=desc&_expiration=4070908800&_hash=e4a21b9'
    }
}
```

```
Controller "App\\Controller\\BlogController::show()" requires that you
provide a value for the "$slug" argument.
```

□□□□□□□□□□□□□□□□ $slug□□□□□□□□□□□□□

```
public function show(string $slug): Response
{
    // ...
}
```

□□□□□□□□□□ {slug} □□□□□□□□ /blog/show□□□ {slug} □□□□□□□□ /blog/show/{slug} □□□□□□□□□□□□ $slug = null□□

```
Some mandatory parameters are missing ("slug") to generate a URL for route
"blog_show".
```

□□□□□□□□□□□□□□ blog_show □□□ URL□□□□□□□ slug □□□□□□□□□□□□□□□□□□□□□□ {slug} □□□□□□□□□□□□□□□□□□□□□□ slug □□

```
$this->generateUrl('blog_show', ['slug' => 'slug-value']);
```

or, in Twig: □□□□ Twig □□

```
{{ path('blog_show', {slug: 'slug-value'}) }}
```

□□

□□□

□□□□□□□□ PHP □□□□□ Request □□□□□□□□□□□□□ Response □□□□□□□□ HTML □□□ JSON□ XML□□□□□□□□□□ 404 □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

□□□□□

□□□□□□□□□□ PHP □□□□□□□□□□□□□□□□ Closure □□□□□□□□□□□□□□□□□□□□□

```
// src/Controller/LuckyController.php
namespace App\Controller;

use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\Routing\Attribute\Route;

class LuckyController
{
    #[Route('/lucky/number/{max}', name: 'app_lucky_number')]
    public function number(int $max): Response
    {
        $number = random_int(0, $max);

        return new Response(
            '<html><body>Lucky number: '.$number.'</body></html>'
        );
    }
}
```

□□□□ number□□ □□□□□□□□□□ LuckyController □□

□□□□□□□□□□

- □ 2 □□ Symfony □□ PHP □□□□□□□□□□□□□□□□□□□□
- □ 4 □□ Symfony □□□□□ PHP □□□□□□□□ use □□□□□□□□□□□□□□□ Response □□