# CHANGELOG

## 7.2

* Add the Requirement::UID\_RFC9562 constant to validate UUIDs in the RFC 9562 format
* Deprecate the AttributeClassLoader::$routeAnnotationClass property

## 7.1

* Add {foo:bar} syntax to define a mapping between a route parameter and its corresponding request attribute

## 7.0

* Add argument $routeParameters to UrlMatcher::handleRouteRequirements()
* Remove Doctrine annotations support in favor of native attributes
* Remove AnnotationClassLoader, use AttributeClassLoader instead
* Remove AnnotationDirectoryLoader, use AttributeDirectoryLoader instead
* Remove AnnotationFileLoader, use AttributeFileLoader instead

## 6.4

* Add FQCN and FQCN::method aliases for routes loaded from attributes/annotations when applicable
* Add native return type to AnnotationClassLoader::setResolver()
* Deprecate Doctrine annotations support in favor of native attributes
* Change the constructor signature of AnnotationClassLoader to \_\_construct(?string $env = null), passing an annotation reader as first argument is deprecated
* Deprecate AnnotationClassLoader, use AttributeClassLoader instead
* Deprecate AnnotationDirectoryLoader, use AttributeDirectoryLoader instead
* Deprecate AnnotationFileLoader, use AttributeFileLoader instead
* Add AddExpressionLanguageProvidersPass (moved from FrameworkBundle)
* Add aliases for all classes in the Annotation namespace to Attribute

## 6.2

* Add Requirement::POSITIVE\_INT for common ids and pagination

## 6.1

* Add getMissingParameters and getRouteName methods on MissingMandatoryParametersException
* Allow using UTF-8 parameter names
* Support the attribute type (alias of annotation) in annotation loaders
* Already encoded slashes are not decoded nor double-encoded anymore when generating URLs (query parameters)
* Add EnumRequirement to help generate route requirements from a \BackedEnum
* Add Requirement, a collection of universal regular-expression constants to use as route parameter requirements
* Add params variable to condition expression
* Deprecate not passing route parameters as the fourth argument to UrlMatcher::handleRouteRequirements()

## 5.3

* Already encoded slashes are not decoded nor double-encoded anymore when generating URLs
* Add support for per-env configuration in XML and Yaml loaders
* Deprecate creating instances of the Route annotation class by passing an array of parameters
* Add RoutingConfigurator::env() to get the current environment

## 5.2.0

* Added support for inline definition of requirements and defaults for host
* Added support for \A and \z as regex start and end for route requirement
* Added support for #[Route] attributes

## 5.1.0

* added the protected method PhpFileLoader::callConfigurator() as extension point to ease custom routing configuration
* deprecated RouteCollectionBuilder in favor of RoutingConfigurator.
* added "priority" option to annotated routes
* added argument $priority to RouteCollection::add()
* deprecated the RouteCompiler::REGEX\_DELIMITER constant
* added ExpressionLanguageProvider to expose extra functions to route conditions
* added support for a stateless keyword for configuring route stateless in PHP, YAML and XML configurations.
* added the "hosts" option to be able to configure the host per locale.
* added RequestContext::fromUri() to ease building the default context

## 5.0.0

* removed PhpGeneratorDumper and PhpMatcherDumper
* removed generator\_base\_class, generator\_cache\_class, matcher\_base\_class and matcher\_cache\_class router options
* Serializable implementing methods for Route and CompiledRoute are final
* removed referencing service route loaders with a single colon
* Removed ServiceRouterLoader and ObjectRouteLoader.

## 4.4.0

* Deprecated ServiceRouterLoader in favor of ContainerLoader.
* Deprecated ObjectRouteLoader in favor of ObjectLoader.
* Added a way to exclude patterns of resources from being imported by the import() method

## 4.3.0

* added CompiledUrlMatcher and CompiledUrlMatcherDumper
* added CompiledUrlGenerator and CompiledUrlGeneratorDumper
* deprecated PhpGeneratorDumper and PhpMatcherDumper
* deprecated generator\_base\_class, generator\_cache\_class, matcher\_base\_class and matcher\_cache\_class router options
* Serializable implementing methods for Route and CompiledRoute are marked as @internal and @final. Instead of overwriting them, use \_\_serialize and \_\_unserialize as extension points which are forward compatible with the new serialization methods in PHP 7.4.
* exposed utf8 Route option, defaults "locale" and "format" in configuration loaders and configurators
* added support for invokable service route loaders

## 4.2.0

* added fallback to cultureless locale for internationalized routes

## 4.0.0

* dropped support for using UTF-8 route patterns without using the utf8 option
* dropped support for using UTF-8 route requirements without using the utf8 option

## 3.4.0

* Added NoConfigurationException.
* Added the possibility to define a prefix for all routes of a controller via @Route(name="prefix\_")
* Added support for prioritized routing loaders.
* Add matched and default parameters to redirect responses
* Added support for a controller keyword for configuring route controllers in YAML and XML configurations.

## 3.3.0

* [DEPRECATION] Class parameters have been deprecated and will be removed in 4.0.
  + router.options.generator\_class
  + router.options.generator\_base\_class
  + router.options.generator\_dumper\_class
  + router.options.matcher\_class
  + router.options.matcher\_base\_class
  + router.options.matcher\_dumper\_class
  + router.options.matcher.cache\_class
  + router.options.generator.cache\_class

## 3.2.0

* Added support for bool, int, float, string, list and map defaults in XML configurations.
* Added support for UTF-8 requirements

## 2.8.0

* allowed specifying a directory to recursively load all routing configuration files it contains
* Added ObjectRouteLoader and ServiceRouteLoader that allow routes to be loaded by calling a method on an object/service.
* [DEPRECATION] Deprecated the hardcoded value for the $referenceType argument of the UrlGeneratorInterface::generate method. Use the constants defined in the UrlGeneratorInterface instead.

Before:

$router->generate('blog\_show', ['slug' => 'my-blog-post'], true);

After:

use Symfony\Component\Routing\Generator\UrlGeneratorInterface;

$router->generate('blog\_show', ['slug' => 'my-blog-post'], UrlGeneratorInterface::ABSOLUTE\_URL);

## 2.5.0

* [DEPRECATION] The ApacheMatcherDumper and ApacheUrlMatcher were deprecated and will be removed in Symfony 3.0, since the performance gains were minimal and it's hard to replicate the behavior of PHP implementation.

## 2.3.0

* added RequestContext::getQueryString()

## 2.2.0

* [DEPRECATION] Several route settings have been renamed (the old ones will be removed in 3.0):
  + The pattern setting for a route has been deprecated in favor of path
  + The \_scheme and \_method requirements have been moved to the schemes and methods settings

Before:

article\_edit:

pattern: /article/{id}

requirements: { '\_method': 'POST|PUT', '\_scheme': 'https', 'id': '\d+' }

<route id="article\_edit" pattern="/article/{id}">

<requirement key="\_method">POST|PUT</requirement>

<requirement key="\_scheme">https</requirement>

<requirement key="id">\d+</requirement>

</route>

$route = new Route();

$route->setPattern('/article/{id}');

$route->setRequirement('\_method', 'POST|PUT');

$route->setRequirement('\_scheme', 'https');

After:

article\_edit:

path: /article/{id}

methods: [POST, PUT]

schemes: https

requirements: { 'id': '\d+' }

<route id="article\_edit" pattern="/article/{id}" methods="POST PUT" schemes="https">

<requirement key="id">\d+</requirement>

</route>

$route = new Route();

$route->setPath('/article/{id}');

$route->setMethods(['POST', 'PUT']);

$route->setSchemes('https');

* [BC BREAK] RouteCollection does not behave like a tree structure anymore but as a flat array of Routes. So when using PHP to build the RouteCollection, you must make sure to add routes to the sub-collection before adding it to the parent collection (this is not relevant when using YAML or XML for Route definitions).

Before:

$rootCollection = new RouteCollection();

$subCollection = new RouteCollection();

$rootCollection->addCollection($subCollection);

$subCollection->add('foo', new Route('/foo'));

After:

$rootCollection = new RouteCollection();

$subCollection = new RouteCollection();

$subCollection->add('foo', new Route('/foo'));

$rootCollection->addCollection($subCollection);

Also one must call addCollection from the bottom to the top hierarchy. So the correct sequence is the following (and not the reverse):

$childCollection->addCollection($grandchildCollection);

$rootCollection->addCollection($childCollection);

* [DEPRECATION] The methods RouteCollection::getParent() and RouteCollection::getRoot() have been deprecated and will be removed in Symfony 2.3.
* [BC BREAK] Misusing the RouteCollection::addPrefix method to add defaults, requirements or options without adding a prefix is not supported anymore. So if you called addPrefix with an empty prefix or / only (both have no relevance), like addPrefix('', $defaultsArray, $requirementsArray, $optionsArray) you need to use the new dedicated methods addDefaults($defaultsArray), addRequirements($requirementsArray) or addOptions($optionsArray) instead.
* [DEPRECATION] The $options parameter to RouteCollection::addPrefix() has been deprecated because adding options has nothing to do with adding a path prefix. If you want to add options to all child routes of a RouteCollection, you can use addOptions().
* [DEPRECATION] The method RouteCollection::getPrefix() has been deprecated because it suggested that all routes in the collection would have this prefix, which is not necessarily true. On top of that, since there is no tree structure anymore, this method is also useless. Don't worry about performance, prefix optimization for matching is still done in the dumper, which was also improved in 2.2.0 to find even more grouping possibilities.
* [DEPRECATION] RouteCollection::addCollection(RouteCollection $collection) should now only be used with a single parameter. The other params $prefix, $default, $requirements and $options will still work, but have been deprecated. The addPrefix method should be used for this use-case instead. Before: $parentCollection->addCollection($collection, '/prefix', [...], [...]) After:

$collection->addPrefix('/prefix', [...], [...]);

$parentCollection->addCollection($collection);

* added support for the method default argument values when defining a @Route
* Adjacent placeholders without separator work now, e.g. /{x}{y}{z}.{\_format}.
* Characters that function as separator between placeholders are now whitelisted to fix routes with normal text around a variable, e.g. /prefix{var}suffix.
* [BC BREAK] The default requirement of a variable has been changed slightly. Previously it disallowed the previous and the next char around a variable. Now it disallows the slash (/) and the next char. Using the previous char added no value and was problematic because the route /index.{\_format} would be matched by /index.ht/ml.
* The default requirement now uses possessive quantifiers when possible which improves matching performance by up to 20% because it prevents backtracking when it's not needed.
* The ConfigurableRequirementsInterface can now also be used to disable the requirements check on URL generation completely by calling setStrictRequirements(null). It improves performance in production environment as you should know that params always pass the requirements (otherwise it would break your link anyway).
* There is no restriction on the route name anymore. So non-alphanumeric characters are now also allowed.
* [BC BREAK] RouteCompilerInterface::compile(Route $route) was made static (only relevant if you implemented your own RouteCompiler).
* Added possibility to generate relative paths and network paths in the UrlGenerator, e.g. "../parent-file" and "//example.com/dir/file". The third parameter in UrlGeneratorInterface::generate($name, $parameters = [], $referenceType = self::ABSOLUTE\_PATH) now accepts more values and you should use the constants defined in UrlGeneratorInterface for claritiy. The old method calls with a Boolean parameter will continue to work because they equal the signature using the constants.

## 2.1.0

* added RequestMatcherInterface
* added RequestContext::fromRequest()
* the UrlMatcher does not throw a \LogicException anymore when the required scheme is not the current one
* added TraceableUrlMatcher
* added the possibility to define options, default values and requirements for placeholders in prefix, including imported routes
* added RouterInterface::getRouteCollection
* [BC BREAK] the UrlMatcher urldecodes the route parameters only once, they were decoded twice before. Note that the urldecode() calls have been changed for a single rawurldecode() in order to support + for input paths.
* added RouteCollection::getRoot method to retrieve the root of a RouteCollection tree
* [BC BREAK] made RouteCollection::setParent private which could not have been used anyway without creating inconsistencies
* [BC BREAK] RouteCollection::remove also removes a route from parent collections (not only from its children)
* added ConfigurableRequirementsInterface that allows to disable exceptions (and generate empty URLs instead) when generating a route with an invalid parameter value