# Phalcon routage

Le composant de routage Phalcon Phalcon\Mvc\Router permet de définir des routes mappées vers les contrôleurs ou vers des handlers qui reçoivent la requête. Le routage peut fonctionner en mode MVC ou en mode match-only.

## -- Définition de routes

En mode MVC, il est possible de définir des routes et de les mapper vers les actions des contrôleurs.

### -- Routes simples

Quelques exemples :

```
<?php

// Instanciation du routeur
$router = new \Phalcon\Mvc\Router();

//Création d'une route
$router->add(
    "/admin/users/my-profile",
    array(
        "controller" => "users",
        "action"     => "profile",
    )
);

//Ajout d'une autre route
$router->add(
    "/admin/users/change-password",
    array(
        "controller" => "users",
        "action"     => "changePassword",
    )
);

$router->handle();
```

Utilisation de la syntaxe courte pour définir les routes :

```
<?php

//Création d'une route
$router->add("/admin/users/my-profile","users::profile";

//Ajout d'une autre route
```

```
$router->add("/admin/users/change-password","users::changePassword");
```

## -- Routes génériques

Il est possible de définir des routes plus génériques en utilisant les expressions régulières (syntaxe PRCE)

Exemple :

```
<?php

// Create the router
$router = new \Phalcon\Mvc\Router();

//Define a route
$router->add(
    "/admin/:controller/a/:action/:params",
    array(
        "controller" => 1,
        "action"     => 2,
        "params"     => 3,
    )
);
```

Les alias **:controller**, **:action** et **:params** simplifient l'expression et évitent l'usage direct d'expressions régulières.

- Le premier paramètre de add défini le pattern de l'url interceptée
- Le second défini comment les parties de l'url se répartissent en controller, action et parameters

Si l'url frappée est **/admin/users/a/delete/dave/301**, elle sera interprétée en :

- contrôleur : users
- action : delete
- param : dave
- param : 301

Alias disponibles :

| **Alias** | **Expression régulière associée** | **Usage** |
|---|---|---|
| /:module | /([a-zA-Z0-9_-]+) | Correspond à un nom de module valide avec caractères alpha-numériques seulement |
| /:controller | /([a-zA-Z0-9_-]+) | Correspond à un nom de contrôleurvalide avec caractères alpha-numériques seulement |
| /:action | /([a-zA-Z0-9_]+) | Correspond à un nom d'action valide avec caractères alpha-numériques seulement |
| /:params | (/.*)* | Liste de paramètres optionnels séparés par des /. A utiliser uniquement à la fin de l'URL |
| /:namespace | /([a-zA-Z0-9_-]+) | Correspond à un namespace |
| /:int | /([0-9]+) | Correspond à un paramètre de type entier |

## -- Nommage des paramètres

```
<?php

$router->add(
    "/news/([0-9]{4})/([0-9]{2})/([0-9]{2})/:params",
    array(
        "controller" => "posts",
        "action"     => "show",
        "year"       => 1, // ([0-9]{4})
        "month"      => 2, // ([0-9]{2})
        "day"        => 3, // ([0-9]{2})
        "params"     => 4, // :params
    )
);
```

Une URL de la forme /news/2015/1/11/last/desc sera interprétée de la façon suivante :

- controller : posts
- action : show
- year : 2015
- month : 1
- day : 11
- params : last et desc

Version courte avec paramètres nommés :

```
<?php

$router->add(
    "/news/{year:([0-9]{4})}/{month:([0-9]{2})}/{day:([0-9]{2})}/:params",
    array(
        "controller" => "posts",
        "action"     => "show",
        "params"     => 4
    )
);
```

## -- Récupération des paramètres nommés dans l'action cible

```
<?php

class PostsController extends \Phalcon\Mvc\Controller{

    public function indexAction(){

    }
```

```
    public function showAction(){

        // Return "year" parameter
        $year = $this->dispatcher->getParam("year");

        // Return "month" parameter
        $month = $this->dispatcher->getParam("month");

        // Return "day" parameter
        $day = $this->dispatcher->getParam("day");

    }

}
```

### -- Restrictions sur la méthode HTTP

La correspondance des routes peut également être réalisée sur la méthode HTTP (GET, POST, HEAD, PUT, DELETE...) :

```
<?php

// This route only will be matched if the HTTP method is GET
$router->addGet("/products/edit/{id}", "Products::edit");

// This route only will be matched if the HTTP method is POST
$router->addPost("/products/save", "Products::save");

// This route will be matched if the HTTP method is POST or PUT
$router->add("/products/update")->via(array("POST", "PUT"));
```

### -- Fonctions de vérification

Les routes doivent parfois satisfaire des conditions bien spécifiques qu'il est possible d'inclure avec la méthode **beforeMatch** : si elle retourne false, la route sera considérée comme non correspondante :

```
<?php

$router->add('/login', array(
    'module' => 'admin',
    'controller' => 'session'
))->beforeMatch(function($uri, $route) {
    //Vérifie si la requête est réalisée en Ajax
    if ($_SERVER['HTTP_X_REQUESTED_WITH'] == 'xmlhttprequest') {
        return false;
    }
    return true;
});
```

### -- Groupes de routes

Si un ensemble de routes a des chemins identiques, il est possible de les regrouper pour les gérer plus facilement :

```
<?php

$router = new \Phalcon\Mvc\Router();

//Create a group with a common module and controller
$blog = new \Phalcon\Mvc\Router\Group(array(
    'module' => 'blog',
    'controller' => 'index'
));

//All the routes start with /blog
$blog->setPrefix('/blog');

//Add a route to the group
$blog->add('/save', array(
    'action' => 'save'
));

//Add another route to the group
$blog->add('/edit/{id}', array(
    'action' => 'edit'
));

//This route maps to a controller different than the default
$blog->add('/blog', array(
    'controller' => 'blog',
    'action' => 'index'
));

//Add the group to the router
$router->mount($blog);
```

## -- Configuration du routage

### -- Fonctionnement par défaut

La classe Phalcon\Mvc\Router a un fonctionnement par défaut offrant un routage simple fondé sur des URLs du type **/:controller/:action/:params**

Il est possible de désactiver ce fonctionnement par défaut :

```
<?php

// Instanciation du routeur sans les routes par défaut
$router = new \Phalcon\Mvc\Router(false);
```

**-- Route par défaut**

```
<?php

$router->add("/", array(
    'controller' => 'index',
    'action' => 'index'
));
```

**-- Chemins par défaut**

```
<?php

//Définition à partir des méthodes spécifiques
$router->setDefaultModule('backend');
$router->setDefaultNamespace('Backend\Controllers');
$router->setDefaultController('index');
$router->setDefaultAction('index');

//Définition à partir d'un tableau
$router->setDefaults(array(
    'controller' => 'index',
    'action' => 'index'
));
```

**-- Erreur 404 : not found path**

```
<?php

//Set 404 paths
$router->notFound(array(
    "controller" => "index",
    "action" => "route404"
));
```

# -- Initialisation du service routage

L'initialisation du service routage se fait par injection de dépendance dans le fichier index.php ou services.php (si le projet a été créé avec webtools).

```
<?php
```

```
/**
* Ajout de la fonctionnalité routage
*/
$di->set('router', function(){
    $router = new \Phalcon\Mvc\Router();
    require __DIR__.'/../app/config/routes.php';
    return $router;
});
```

Le fichier **app/config/routes.php** peut ensuite définir les différentes routes :

```
<?php
$router->add("/admin/users/my-profile","users::profile";

//Ajout d'une autre route
$router->add("/admin/users/change-password","users::changePassword");
```