

### References

This cheat sheet references documentation directly from the <https://laravel.com/docs/6.x/routing> from the `method="POST"` action where `profile` is a depth explanation

### Basic Route Syntax

**Route: `:verb( '/uri', UserController@method);`**

Calls index method from UserController

**Route: `:verb( '/uri', $callback);`**

Uses callback function

### Basic Route Syntax Explained

Route is a Laravel Class

verb is a static method called on the Route Class. Static methods use `::` scope resolution operator to point to the class method. Basic verbs/methods that can be called on Route: `get, post, put, patch, delete, options`

`https://laravel.com/api/6.x/Illuminate/Contracts/Routing/Registrar.html`

`$uri` points to the url

Example: `localhost:8080/user ... user is the`

### More Route Methods

**Route: `:view( '/welcome', 'welcome');`**

Shortcut if route only needs to a view and not a full controller

**Route: `:view( '/welcome', 'welcome', ['name' => 'Taylor']);`**

Optional array of data may be passed as third arg

**Route: `:match(['get', 'post', '/', function () {`**

Responds to multiple HTTP verbs

**Route: `:any('/', function () { // });`**

Responds to all HTTP verbs using the any method:

**Route: `:redirect( '/here', '/there', 301);`**

`/here` redirects to `/there`. 3rd param is optional. It overrides the default 302 status which can be verified

**Route: `:permanentRedirect( '/here', '/there');`**

Return a permanent 301 status code:

### CSRF Protection

```

<form method="POST" action="/profile"
    @csrf
    ...
</form>

```

Any HTML form pointing to POST, PUT, or DELETE routes that are defined in the web routes file should include a CSRF token field as a security layer or the request will be rejected.

### Route Parameters - Required

Used to capture a segment of URI within a route.

**Route: `:get( '/user/{id}/{name}', function ($id, $name) {`**

`return 'User #'. $id . ' is ' . $name; });` Static methods use `::` scope resolution operator to point to the class method. Route params are encased in `{}` braces. Use as many as you need. Route parameters are injected into route callbacks

`Example: localhost:8080/user/12345/Fred renders`

### Route Parameters - Optional

Placing a `?` mark after the param name makes it optional

**Route: `:get( '/user/{name}/{id?}', function ($name, $id) {`**

`return $name . 's user number is ' . $id; });`

Make sure to give the param a default value such as `0`

Example 1: `http://127.0.0.1:8000/user/Fred``

Renders: `Fred's user number is unknown`

Example 2: `http://127.0.0.1:8000/user/Fred/123``

Renders: `Fred's user number is 123`

### Constraints

overrides the default 302 status which can be verified. The where method is chained to the route and accepts

Example:

```

Route::get( 'user/{id}', function ($id) {
    //
    })->where('id', '[0-9]+');

```

### Global Constraint

To set a route parameter to always be constrained by a regular expression, use the pattern method in the boot() method in routes.php

Example:

```
public function boot() {
    // Route: :pattern('id', '[0-9]+'); parent::boot()
    // id parameter now must always consist of only numbers to execute, no matter which Route is using the id parameter
```

### Encoded Forwarded Slashes

The Laravel routing component allows all characters except /. You must explicitly allow / to be part of your routes.

```
Route::get('search/{search}', function ($search) {
    return $search;
})->where('search', '.*');
```

### Named Routes

```
Route::get('user/profile', function () {
    // ...
})->name('profile');
```



By **jlampstack**  
[cheatography.com/jlampstack/](https://cheatography.com/jlampstack/)

Not published yet.  
 Last updated 28th November, 2019.  
 Page 2 of 2.

Sponsored by **Readable.com**  
 Measure your website readability!  
<https://readable.com>