Laravel 6.x Routing Cheat Sheet

by jlampstack via cheatography.com/102952/cs/21241/

References

This cheat sheet references documentation directly from the Laravel website where more in-depth explanations are provided.

> https://laravel.com/docs/6.x/routing

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 its Class

> Basic verbs/methods than 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 uri*

More Route Methods

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

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

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

> Optional array of data may be passed as third arg

More Route Methods (cont)

`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 using Chrome dev tools in the network tab

`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 Parameters - Required (cont)

```
Route::get('/user/{id}/{name}', function ($id,
$name) {
return 'User #'. $id . ' is ' . $name;
});
```

> Route params are encased in {} braces. Use as many as needed
> Route parameters are injected into route callbacks / controllers based on order
> Example: localhost:8080/user/12345/Fred renders "User #12345 is Fred"

Route Parameters - Optional

Placing a ? mark after the param name makes it optional

```
Route::get('/user/{name}/{id?}', function ($name,
$id = 'unknown') {
return $name . ' \'s user number is ' . $id;
});
```

> Make sure to give the param a default value such as null or Fred

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

Contraints

The where method is chained to the route and accepts the name of the parameter and a regular expression defining how the parameter should be constrained

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 of RouteServiceProvider.php file
Use the pattern method in the boot method of RouteServiceProvider.php

Example:

```
public function boot() { // Route::patte-
rn('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 name

Encoded Forwarded Slashes

The Laravel routing component allows all characters except /. You must explicitly allow / to be part of your placeholder using a where condition regular expression

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

Named Routes

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