33.请求和依赖注入

学习要点：

1.Request 请求

2.依赖注入

本节课我们来开始学习 Http 的 Request 请求，然后简单了解下依赖注入。

一. Request 请求

1. 创建 UserController 控制器，在方法中注入 Request 对象；

```
namespace App\Http\Controllers;
use Illuminate\Http\Request;

class UserController
{
    //方法会自动得到 Request 对象，这个就是注入
    public function index(Request $request)
    {
        //input 的方法可以获取 http 请求的数据
        //localhost:8000/user?name=Mr.Lee
        return $request->input('name');
    }
}
```

3. 使用 all()方法，可以获取所有 url 传递过来的参数；

```
//http://localhost:8000/user?name=Mr.Lee&gender=男&age=100
return $request->all();
```

4. 路由自带参数时，可以通过第二，第三参数接受；

```
Route::get('/user/{id}/{uid}', 'UserController@index');
public function index(Request $request, $id, $uid)
```

5. 路由区域，也支持闭包注入 Request；

```
Route::get('/user', function (Request $request) {
    return $request->all();
});
```

6. 使用 path()方法，可以得到当前的 uri 路径；

```
return $request->path();
```

7. 使用 is()方法，可以判断当前的 uri 是否匹配；

```
//是否是 user/*的 uri，返回布尔值
return $request->is('user/*');
```

8. 使用 url()和 fullUrl()得到 url 地址，带参数和不带参数区别；

```
return $request->url();
return $request->fullUrl();
```

9. 使用 isMethod()来判断 HTTP 请求的方式，get、post 等等；

```
return $request->isMethod('post');
```

二. 依赖注入

1. 我们发现在控制器方法，可以直接拿到 Request 对象，而不需要 new；
2. 这种方法是系统的“服务容器”给你自动注入的；
3. 比如，我们想使用另一个控制器的方法，那原始做法如下：

```
$task = new TaskController();
return $task->read(10);
```

4. 直接使用依赖注入，也可以直接使用；

```
public function index(Request $request, TaskController $task)
{
    return $task->read(10);
}
```