[$location &window.location.url 2](#_Toc19245)

[$stateProvider & $stateParams & $urlRouterProvider 2](#_Toc19305)

[$stateProvider 2](#_Toc21419)

[urlRouterProvider 3](#_Toc21364)

[Decorator: 3](#_Toc12320)

[类的装饰器 3](#_Toc21751)

[Function decorator 3](#_Toc2321)

# $location &window.location.url

参考：<https://docs.angularjs.org/api/ng/service/$location>

$location.url vs $windows.location.href

前者依据window.location而来，且为双向绑定。Location.replace不会添加到history中。

后者添加items到history中，window.location是

# $stateProvider & $stateParams & $urlRouterProvider

angular的服务依赖： $stateParam

## $stateProvider

.state('contacts', {

url: "/contacts",

templateUrl: 'contacts.html',

Controller:’’

})

.state('contacts', {

url: "/contacts",

templateUrl: 'contacts.html',

Controller:’’

})

.state('contacts', {

url: "^/contacts",

templateUrl: 'contacts.html',

Controller:’’

})

-----^ 放置与url前，表示绝对path，不包括host

-----对于想要让某个变量值在所有的url的controllers中使用，使用resolve，参考：

https://github.com/angular-ui/ui-router/wiki/URL-Routing

创建了一个名为contacts的state对象，用于监听，当链接的path符合url时，将template渲染到界面。

使用transitionTo(“contacts”)：那么链接地址将会被更新到/contacts

Angular的地址请求中可以使用**{}**

在{}中可以应用正则表达式。如/user/{id: [0-9]{1,9}?}，注意不能使用捕获组，但是可以使用非捕获组?:

地址中的信息可以使用angular的模块$stateParams进行获取，如

<http://localhost:4002/#/invoice/invoice-detail?detailId=a163bfd0-fdac-11e6-8f9d-db8ca1d9fa95&method=add>

对于处理该网页的控制器，使用$stateParams可以获得$stateParams.detailId和$stateParams.method

url: '/users/:id/details/{type}/{repeat:[0-9]+}?from&to'

那么$stateParams的值为{ id:'123', type:'', repeat:'0' }

## urlRouterProvider

负责监听$location

$urlRouterProvider.when(state.url, ['$match', '$stateParams', function ($match, $stateParams) {

if ($state.$current.navigable != state || !equalForKeys($match, $stateParams)) {

$state.transitionTo(state, $match, false);

}

}]);

# $localStorage & $cookieStorage & $sessionStorage

提供键值形式数据，存、取时自动调用了angular的toJson和fromJson

ngStorage包括了两个服务：$localStorage & $sessionStorage

$localStorage客户端存到本地文件中，最大5M

# 微信接口

# Decorator:

固定格式：

Return function(){}

## 类的装饰器

比如：可以利用该装饰器，创建angular的service服务。单击此处输入文字。

Function decorator\_name{

Return function(constructor: Function){}

}

## Function decorator

调用该函数前，执行decorator，decorator可以重新定义被装饰方法的实现，

Function(){

Return function(target:any, propertyKey:any, desc?:PropertyDescriptor ){}

}

# Getter/setter

typescript中的getter/setter方法，在调用时，无需添加get字段

class foo {

private \_bar:boolean = false;

get bar():boolean {

return this.\_bar;

}

set bar(theBar:boolean) {

this.\_bar = theBar;

}

}

在调用时，只需要myfoo=new foo(); console.log(myfoo.bar) 即可获取到值

==============================================================================================

正则：

非捕获组：

？：

Gulp