用 Angular 进行开发，基本上都会遇到 Controller 之间通信的问题，本文对此进行一个总结。在 Angular 中，Controller 之间通信的方式主要有三种：

1. 作用域继承。利用子 Controller 控制父 Controller 上的数据。（父 Controller 中的数据要为引用类型，不能是基本类型，原因参见 AngularJS中的作用域 一文）
2. 注入服务。把需要共享的数据注册为一个 `service`，在需要的 Controller 中注入。
3. 基于事件。利用 Angular 的事件机制，使用 `$on`、`$emit` 和 `$boardcast`
   其中，作用域继承仅限于上下级之间的通信，注入服务和基于事件的机制可以实现任意级别的 Controller 通信。

作用域继承

原理在 作用域 一文中有讲解，这里直接上栗子。

页面：

```
<div ng-controller="parentCtrl">
  <p>data in parent controller : {{data.name}}</p>
  <div ng-controller="childCtrl">
    <input type="text" ng-model="data.name">
  </div>
</div>
```

控制器：

```
angular.module('demo', [])
.controller('parentCtrl', ['$scope', function($scope){
    $scope.data = {
        name: 'htf'
    }
}])
.controller('childCtrl', ['$scope', function($scope){

}])
```

以上是父 Controller 中的数据是引用类型的情况。如果父 Controller 中的数据是基本类型，可通过 `$scope.$parent.data` 访问。很显然，这种方式仅适用于父子级间 Controller 的通信。

注入服务

在 Angular 中，服务是一个单例，所以在服务中生成一个对象，该对象就可以利用依赖注入的方式在所有的控制器中共享。看个栗子，先定义一个 `service`：

```
angular.module('demo')
  .factory('Data', function(){
    return {
        name: 'htf'
    };
})
```

页面：

```
<div ng-controller="childCtrl1">
  <h3>data in child controller 1 : {{data.name}}</h3>
  <input class="form-control" type="text" ng-model="data.name">
</div>
<div ng-controller="childCtrl2">
  <h3>data in child controller 2 : {{data.name}}</h3>
  <input class="form-control" type="text" ng-model="data.name">
</div>
```

控制器：

```
.controller('childCtrl1', ['$scope', 'Data', function($scope, Data){
    $scope.data = Data;
}])
.controller('childCtrl2', ['$scope', 'Data', function($scope, Data){
    $scope.data = Data;
}])
```

这种方式适用于任何需要通信的 Controller 之间。

基于事件

Angular 为 `$scope` 提供了冒泡和隧道机制，`$broadcast` 会把事件广播给所有子 Controller，而 `$emit` 则会将事件冒泡传递给父 Controller，`$on` 则是 Angular 的事件监听函数，利用这三者，可以实现上下级和同级（需要构造一个共同的父级 Controller）之间的通信。

上下级之间

这种情况下比较简单。

如果是子 Controller 往父 Controller 上发送事件（从作用域往上发送事件），使用 `scope.$emit`

```
$scope.$emit("someEvent", {});
```

如果是父 Controller 往子 Controller 上发送事件（从作用域往下发送事件），使用 `scope.$broadcast`

```
$scope.$broadcast("someEvent", {});
```

无论是 `$emit` 还是 `$broadcast` 发送的事件，都用 `$scope.$on` 接收：

```
$scope.$on("someEvent", function(event, data) {
    // 这里取到发送过来的数据 data
});
```

同级之间

同级之间利用事件通信有两种方法。一种是利用上下级之间事件传播的变形，另一种是借助 `$rootScope` 。

借助父 controller

先看第一种，在子 Controller 中向父 Controller 触发一个事件，然后在父 Controller 中监听事件，再广播给子 Controller ，这样通过事件携带的参数，实现了数据经过父 Controller，在同级 Controller 之间传播。

但是要注意，通过父 Controller 作为中介进行传递的话，子 Controller 触发的事件名和父 Controller 广播用的事件名不能一样，否则会进入死循环。

看代码：

```
<div ng-controller="outerCtrl">
  <h3>data in outer controller: {{name}}</h3>
  <div ng-controller="innerCtrl1">
    <input class="form-control" type="text" ng-model="name" ng-change="change()">
  </div>
  <div ng-controller="innerCtrl2">
    <input class="form-control" type="text" ng-model="name" ng-change="change()">
  </div>
</div>
```

关键部分在控制器：

```
.controller('outerCtrl', ['$scope', function($scope){
    $scope.name = 'htf';
    $scope.$on('dataChanged', function(event, data){
        $scope.name = data;
        // 2. 父 Ctrl 监听到 dataChanged 时间后，触发 changeData 事件
        $scope.$broadcast('changeData', data);
    })
}])

.controller('innerCtrl1', ['$scope', function($scope){
    $scope.change = function(){
        // 1. 子 Ctrl1 中数据改变之后触发 dataChanged 事件
        $scope.$emit('dataChanged', $scope.name);
    }
    $scope.$on('changeData', function(event, data){
        $scope.name = data;
    })
}])

.controller('innerCtrl2', ['$scope', function($scope){
    $scope.change = function(){
```

```
        $scope.$emit('dataChanged', $scope.name);
    }
    // 3. 监听到 changeData 事件后，改变子 Ctrl2 中 数据
    $scope.$on('changeData', function(event, data){
        $scope.name = data;
    })
}])
```

借助 $rootScope

每个 Angular 应用默认有一个根作用域 `$rootScope` ， 根作用域位于最顶层，从它往下挂着各级作用域。

所以，如果子控制器直接使用 `$rootScope` 广播和接收事件，那么就可实现同级之间的通信。

看栗子：

```
<div ng-controller="innerCtrlA">
    <input class="form-control" type="text" ng-model="name" ng-change="change()">
</div>
<div ng-controller="innerCtrlB">
    <input class="form-control" type="text" ng-model="name" ng-change="change()">
</div>
```

控制器：

```
.controller('innerCtrlA', ['$scope', '$rootScope', function($scope, $rootScope){
    $scope.change = function(){
        // 广播事件
        $rootScope.$broadcast('nameChanged', $scope.name);
    }
    $rootScope.$on('nameChanged', function(event, data){
        $scope.name = data;
    })
}])
.controller('innerCtrlB', ['$scope', '$rootScope', function($scope, $rootScope){
    $scope.change = function(){
        $rootScope.$broadcast('nameChanged', $scope.name);
    }
    // 监听事件
    $rootScope.$on('nameChanged', function(event, data){
        $scope.name = data;
    })
}])
```