**自定义指令中使用ng-repeat指令**

**详见：d:\ionic\test1\demo\angular\directive\compile\_link\1\1.html**

**结论一：在pre-link 和 post-link 函数中修改ng-repeat所在元素及其子元素的DOM或CSS将会无效。应当在compile函数中修改dom。**

**原因：**在compile中引用template , 如果template中的元素带有ng-repeat指令，则在pre-link 和 post –link函数被调用的时候会被注释起来，因而在pre-link 和 post –link函数中是无法获取到ng-repeat 及其子元素的对象的。

**实验：**

**1. 需求描述**

使用compile函数中引用template ， 而template中带有ng-repeat指令。

**2.代码：**

**function** directiveFun(){

**var** ret = {};

ret.restrict = "E";

ret.templateUrl = "./templates/template.html";

ret.replace = **true**;

ret.compile = compileFun;

ret.scope=**true**;

**return** ret;

**function** compileFun(element , attrs){

console.log("compileTest[compileFun]: " + element.html()); //{{item}}

**return** {

pre : preFun,

post : postFun

}

**function** preFun(scope , element ,attrs){

console.log("compileTest[preLinkFun]: " + element.html()); //undefined

}

**function** postFun(scope , element ,attrs){

console.log("compileTest[postLinkFun]: " + element.html());//undefined

}

}

**3.输出结果：**

**compileTest[compileFun]: {{item}}**

**compileTest[preLinkFun]: undefined**

**compileTest[postLinkFun]: undefined**

**如果template中不包含ng-repeat指令，则原封不动地输出：**

**compileTest[compileFun]: {{item}}**

**compileTest[preLinkFun]: {{item}}**

**compileTest[postLinkFun]: {{item}}**

**结论二：ng-repeat中需要的数据应该在link函数中注入到$scope中。其实，不管在指令的父scope中添加数据，还是在link函数中添加数据，只要子scope能访问父scope（指令中部带有 scope:{} 详见《directive 中的scope》） ,则在compile中的ng-repeat指令都可以访问。但是出于数据隔离和复用的原因，应尽量在link函数中动态添加数据。**

**原因：**访问不到数据的原因不是因为指令还没有被编译，而是因为指令没有访问父作用域的权限

**实验：**

**1. 需求描述**

使用compile函数中引用template ， 而template中带有ng-repeat指令，在link函数中绑定数据。

**2.代码：**

**function** directiveFun(){

**var** ret = {};

ret.restrict = "E";

ret.templateUrl = "./templates/template.html";

ret.replace = **true**;

ret.compile = compileFun;

//如果改为：ret.scope=｛｝，同时不在link函数中绑定数据，只在父scope中绑//定数据， 则数据不能传到ng-repeat中,因为子scope不能继承父scope

ret.scope=**true**;

**return** ret;

**function** compileFun(element , attrs){

console.log("compileTest[compileFun]: " + element.html());

**return** {

pre : preFun,

post : postFun

}

**function** preFun(scope , element ,attrs){

console.log("compileTest[preLinkFun]: " + element.html());

}

**function** postFun(scope , element ,attrs){

scope.data = [1,2,3,4];//在link函数中绑定数据

console.log("compileTest[postLinkFun]: " + element.html());

}

}

}

结果：数据能够 正常在ng-repeat中呈现。

如果使用ret.scope=｛｝，同时不在link函数中绑定数据，只在父scope中绑定数据，则ng-repeat无数据，

**结论一： 在compile函数里面，ng-repeat指令并没有把元素展开为多个，还是原始的状态。即： <span ng-repeat="item in data">{{item}}</span> 而非： <span>1</span><span>2</span>….**