# State

## 什么是state（状态）

在上文中讲了props（属性）,这次是学习状态的使用，props在确定好后不会在生命周期中改变了，对于那些需要动态改变的数据，就可以使用state。

典型的场景是在接收到服务器返回的新数据，或者在用户输入数据之后。你也可以使用一些“状态容器”比如[Redux](http://redux.js.org/index.html" \t "_blank)来统一管理数据流（译注：但我们不建议新手过早去学习redux）。

## 使用state

首先就是介绍定义state，主要有两种方式：

1. 第一种：

//1、第一种  
state={  
 size:80,  
}

上面一种在构造图外部定义，括号内部是定义的对象。

1. 第二种

constructor(props){  
 **super**(props);  
 //2、  
 this.state={  
 size:90,  
 }  
}

上面一种在构造图内部定义号内部是定义的对象。

举个例子：

render(){  
 **return** <View>  
 <Text style={{fontSize:**this**.state.size}}>william</Text>  
 <Text  
 style={{fontSize:20}}  
 onPress={()=>{  
 **this**.setState({  
 size:**this**.state.size-10  
 })  
 }}  
 >变小</Text>  
 <Text  
 style={{fontSize:20}}  
 onPress={()=>{  
 **this**.setState({  
 size:**this**.state.size+10  
 })  
 }}  
 >变大</Text>  
 </View>  
}

使用setSate触发生命周期的改变，控制size的大小使得第一个text变化。

# Ref

## 什么是ref

其实是组件的特殊的属性，可以理解为组件在渲染后，ref是组件的引用。

## Ref的使用

首先讲一下，ref的定义以及获取对象的几种方式。

1. 第一种

对目标组件进行定义：

<RefTest  
 ref="reftext"  
 />

对RefText组件定义为了reftext。

有两种方式可以获取该组件：

**//第一种**

**var** object=**this**.refs["reftext"];  
**//第二种**

**var** object **=this.refs.reftext；**

refs存放页面全部的ref，后面的是在定义ref的时候的名称。

1. 第二种

对目标组件进行定义：

ref={reftext => **this**.reftext = reftext}

获取该组件：

**var s=this.reftext.**

在全局中定义this.reftext赋值为该组件的引用。