

疯狂早茶微信小程序基础篇《五》：数据绑定（下）

教程接微信小程序开发教程(基础篇)7-数据绑定上，当需要展示一组数据时，可以使用 `wx:for`

```
//.wxml
```

```
<view wx:for="{{array}}">

  {{index}}: {{item.message}}

</view>
```

```
//.js
```

```
age({
  data: {
    array: [{
      message: 'foo',
    }, {
      message: 'bar'
    }]
  }
})
```

```
)
```

其中 `index` 是当前数据索引的默认变量名，`item` 是当前数据项的默认变量名。

也可以使用 `wx:for-item` 和 `wx:for-index` 来指定别名

```
<view wx:for="{{array}}" wx:for-index="idx" wx:for-item="itemName">
```

```
  {{idx}}: {{itemName.message}}
```

```
</view>
```

也可以嵌套使用，如下面是一个九九乘法表

```
<view wx:for="{{[1, 2, 3, 4, 5, 6, 7, 8, 9]}}" wx:for-item="i">
```

```
  <view wx:for="{{[1, 2, 3, 4, 5, 6, 7, 8, 9]}}" wx:for-item="j">
```

```
    <view wx:if="{{i <= j}}">
```

```
      {{i}} * {{j}} = {{i * j}}
```

```
    </view>
```

```
  </view>
```

```
</view>
```

类似 `block wx:if`，也可以将 `wx:for` 用在 `<block/>` 标签上，以渲染一个包含多节点的结构块

```
<block wx:for="{{[1, 2, 3]}}">
```

```
<view> {{index}}: </view>
```

```
<view> {{item}} </view>
```

```
</block>
```

wx:key

要理解为什么需要 wx:key，先来看一个例子：

```
//.wxml
```

```
<checkbox wx:for="{{objectArray}}" value="{{item.name}}" style="display: block;">
```

```
{{item.name}} </checkbox>
```

```
<button bindtap="addToFront">在上方添加一个新的 check 组件</button>
```

```
//.js
```

```
Page({
```

```
  data: {
```

```
    objectArray: [
```

```
      {id: 1, name: '我没有被选中'},
```

```
      {id: 2, name: '我没有被选中'},
```

```
    ],  
  },  
  
  addToFront: function(e) {  
    const length = this.data.objectArray.length  
  
    this.data.objectArray = [{id: length, name: '我没有被选中'}].concat(this.data.objectArray)  
  
    this.setData({  
      objectArray: this.data.objectArray  
    })  
  },  
})
```

上面的代码创建了两个 **checkbox** 组件和一个按钮，当点击按钮会在最上方新增一个 checkbox 组件。

编译代码，会显示如下界面:

- ☐ 我没有被选中
- ☐ 我没有被选中

在上方添加一个新的check组件

点击按钮，界面如下：

- ☐ 我没有被选中
- ☒ 我没有被选中
- ☐ 我没有被选中

在上方添加一个新的check组件

ok，到这里一切正常，为了更好的说明问题，加入 checkbox 选中事件的处理，当 checkbox 被选中时，将文字修改为"我被选中了"，相关代码如下：

```
//wxml
```

```
<checkbox-group bindchange="checkboxChange">
```

```
  <checkbox wx:for="{{objectArray}}" value="{{item.id}}" style="display: block;" > {{item.name}}
```

```
</checkbox>
```

```
</checkbox-group>
```

```
<button bindtap="addToFront">在上方添加一个新的 check 组件</button>
```

```
//js
```

```
Page({
```

```
  data: {
```

```
    objectArray: [
```

```
      {id: 1, name: '我没有被选中'},
```

```
      {id: 2, name: '我没有被选中'},
```

```
],  
},
```

```
addToFront: function(e) {  
    const length = this.data.objectArray.length  
    this.data.objectArray = [{id: length + 1, name: '我没有被选中'}].concat(this.data.objectArray)  
    this.setData({  
        objectArray: this.data.objectArray  
    })  
},
```

```
checkboxChange: function(e){  
    console.log('checkboxChange')  
    const length = this.data.objectArray.length  
    let checkBoxArray = this.data.objectArray  
    for (let i = 0; i < length; i++) {  
        let ischecked = false
```



```
for (let j = 0; j < e.detail.value.length; j++){  
    if (checkBoxArray[i].id == e.detail.value[j]){  
        checkBoxArray[i].name = '我被选中了'  
        ischecked = true  
    }  
}  
  
if (!ischecked){  
    checkBoxArray[i].name = '我没有被选中'  
}  
}  
  
this.setData({  
    objectArray: this.data.objectArray  
})  
}  
})
```

当选中第一个 checkbox 时，界面如下

- ☒ 我被选中了
- ☐ 我没有被选中

在上方添加一个新的check组件

这时如果点击添加组件按钮会怎样呢，期望的效果应该如下

- ☐ 我没有被选中
- ☒ 我被选中了
- ☐ 我没有被选中

在上方添加一个新的check组件

然而实际效果确是下图这样的

- ☒ 我没有被选中
- ☐ 我被选中了
- ☐ 我没有被选中

在上方添加一个新的checkbox组件

可以看到渲染引擎并没有将选中的效果和数据绑定起来，导致出现了预期之外的结果。如果想要达到预期效果，就要使用 `wx:key`

```
<checkbox-group bindchange="checkboxChange">
```

```
<checkbox wx:for="{{objectArray}}" wx:key = "id" value="{{item.id}}" style="display: block;" >
  {{item.name}}
</checkbox>
</checkbox-group>
```

<button bindtap="addToFront">在上方添加一个新的 check 组件</button>

将.wxml 文件修改为上述代码所示，就可以实现预期效果，重点就在 `wx:key = "id"` 这一句

如果列表中项目的位置会动态改变或者有新的项目添加到列表中，并且希望列表中的项目保持自己的特征和状态（如 `<input/>`

中的输入内容，`<switch/>` 的选中状态），需要使用 `wx:key` 来指定列表中项目的唯一的标识符。

`wx:key` 的值以两种形式提供

1 字符串，代表在 for 循环的 array 中 item 的某个 property，该 property 的值需要是列表中唯一的字符串或数字，且不能动态改变。

2 保留关键字 `*this` 代表在 for 循环中的 item 本身，这种表示需要 item 本身是一个唯一的字符串或者数字，

上面引用自微信官方教程。除了用于保持视图组件的状态外，使用 `wx:key` 还有助于提高渲染效率

当数据改变触发渲染层重新渲染的时候，会校正带有 key 的组件，框架会确保他们被重新排序，而不是重新创建，以确保使组件保持自身的状态，并且提高列表渲染时的效率。