事件处理

讲师：王红元
微博：coderwhy

更多学习资源

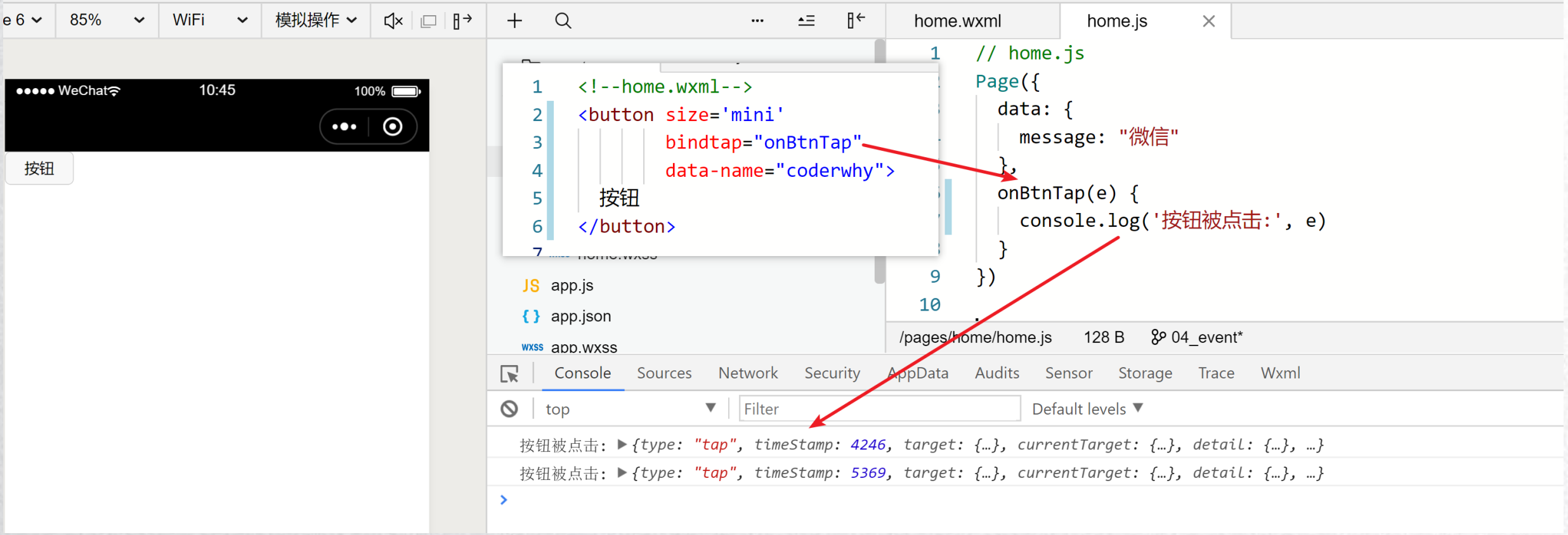
老师答疑交流

事件的介绍

- **什么时候会产生事件呢?**
  - 小程序需要经常和用户进行某种交互，比如点击界面上的某个按钮或者区域，比如滑动了某个区域;
  - 这些交互都会产生各种各样的事件;
- **事件时如何处理呢?**
  - - 事件是通过bind/catch这个属性绑定在组件上的（和普通的属性写法很相似, 以key=“value”形式）;
  - - key以bind或catch开发, 从1.5.0版本开始, 可以在bind和catch后加上一个冒号;
  - - 同时在当前页面的Page构造器中定义对应的事件处理函数tapName, 如果没有对应的函数, 触发事件时会报错
  - - 当用户点击该button区域时，达到触发条件生成事件tap，该事件处理函数tapName会被执行，同时还会收到一个事件对象event。

事件的简单演练

■ 事件的简单演练：



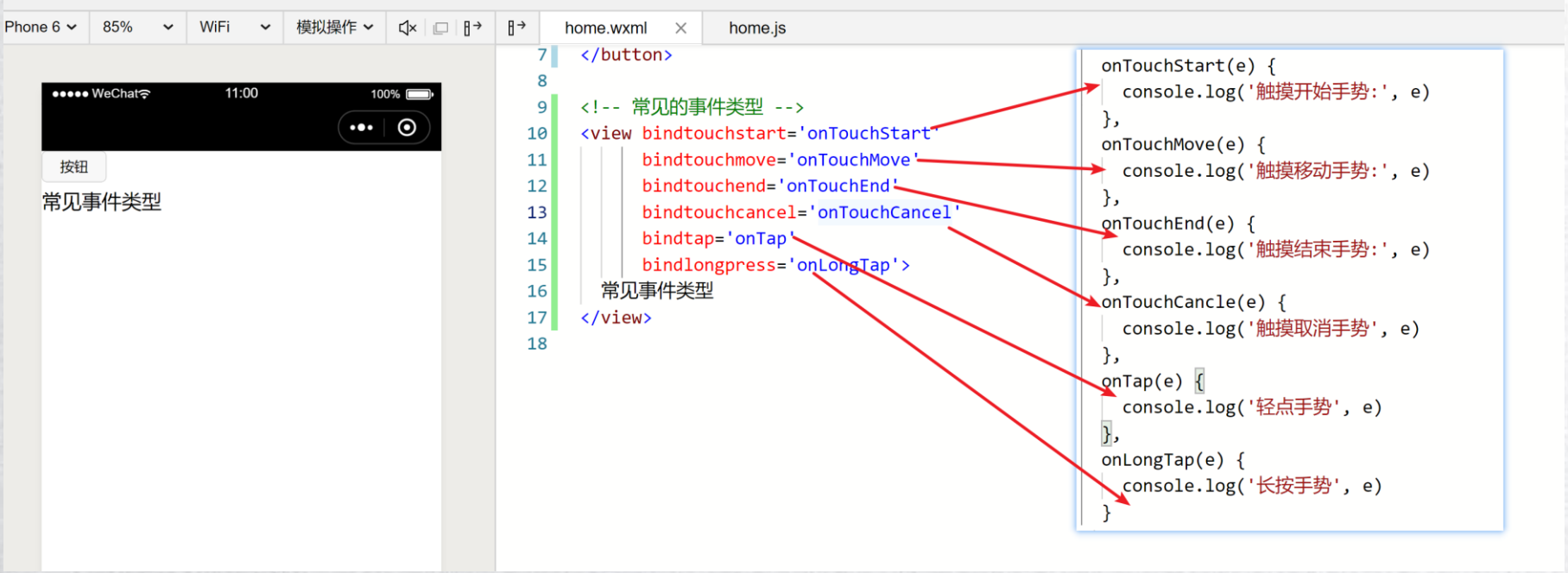
常见事件类型

- **某些组件**会有自己特性的**事件类型**，大家可以在使用组件时具体查看对应的文档
  - 比如input有bindinput/bindblur/bindfocus等
  - 比如scroll-view有bindscrolltowpper/bindscrolltolower等
- 这里我们讨论几个组件都有的, 并且也比较常见的事件类型:

类型	触发条件
touchstart	手指触摸动作开始
touchmove	手指触摸后移动
touchcancel	手指触摸动作被打断，如来电提醒，弹窗
touchend	手指触摸动作结束
tap	手指触摸后马上离开
longpress	手指触摸后，超过350ms再离开，如果指定了事件回调函数并触发了这个事件，tap事件将不被触发
longtap	手指触摸后，超过350ms再离开（推荐使用longpress事件代替）

事件类型演练

- 事件类型的演练



- **有两个注意点:**
  - Touchcancle: 在某些特定场景下才会触发（比如来电打断等）
  - tap事件和longpress事件通常只会触发其中一个

事件对象介绍

- 当某个事件触发时, 会产生一个事件对象, 并且这个对象被传入到回调函数中, 事件对象有哪些常见的属性呢?

属性	类型	说明
type	String	事件类型
timeStamp	Integer	页面打开到触发事件所经过的毫秒数
target	Object	触发事件的组件的一些属性值集合
currentTarget	Object	当前组件的一些属性值集合
detail	Object	额外的信息
touches	Array	触摸事件，当前停留在屏幕中的触摸点信息的数组
changedTouches	Array	触摸事件，当前变化的触摸点信息的数组

touches和changedTouches的区别

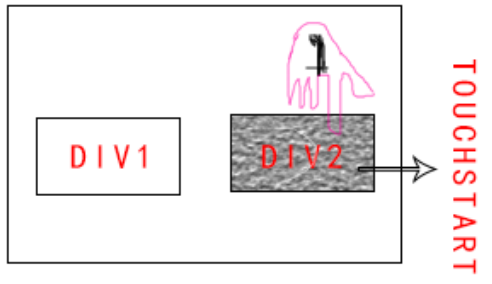
- 1.在touchend中不同
- 2.多手指触摸时不同

touchstart

```
▼ Object  type: "touchstart",  timeStamp: 2352
  ▼ changedTouches: Array[1]
    ▼ 0: Object
        clientX: 181
        clientY: 40
        identifier: 0
        pageX: 181
        pageY: 40
      ▶ __proto__: Object
      length: 1
    ▶ __proto__: Array[0]
  ▶ currentTarget: Object
  ▶ target: Object
    timeStamp: 2352
  ▼ touches: Array[1]
    ▼ 0: Object
        clientX: 181
        clientY: 40
        identifier: 0
        pageX: 181
        pageY: 40
      ▶ __proto__: Object
      length: 1
    ▶ __proto__: Array[0]
    type: "touchstart"
  ▶ __proto__: Object
```

touchend

```
▼ Object  type: "touchend"  timeStamp: 5097
  ▼ changedTouches: Array[1]
    ▼ 0: Object
        clientX: 187
        clientY: 37
        identifier: 0
        pageX: 187
        pageY: 37
      ▶ __proto__: Object
      length: 1
    ▶ __proto__: Array[0]
  ▶ currentTarget: Object
  ▶ target: Object
    timeStamp: 5097
  ▼ touches: Array[0]
      length: 0
    ▶ __proto__: Array[0]
    type: "touchend"
  ▶ __proto__: Object
```



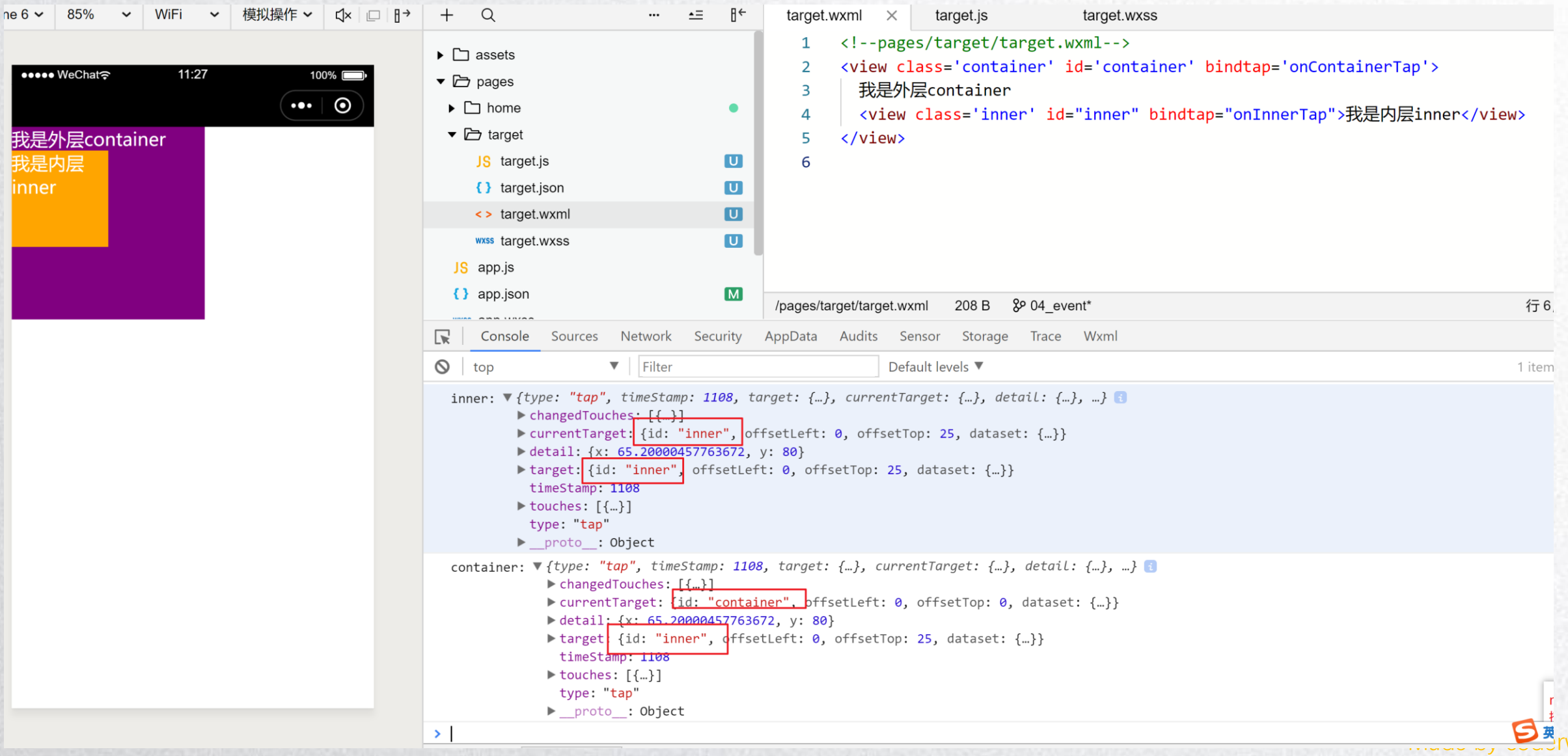
touches： ①　　touches： ①②③

changedTouches： ①　　changedTouches： ②③

touches： 当前屏幕上所有触摸点的列表

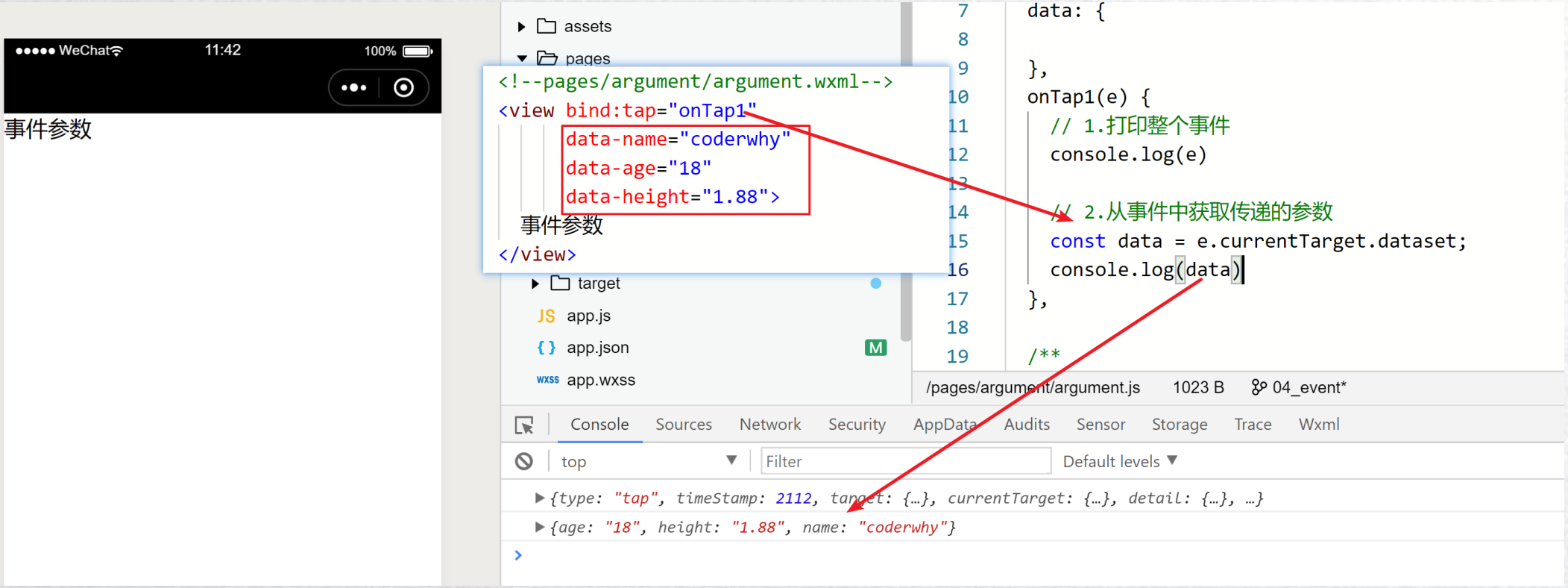
changedTouches： 触发事件时改变的触摸点的集合

currentTarget和target的区别

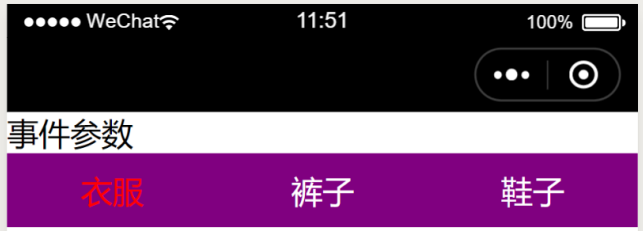


事件参数的传递

- 当视图层发生事件时，某些情况需要事件**携带一些参数到执行的函数**中，这个时候就可以通过**data-属性**来完成：
  - **格式：**data-属性的名称
  - **获取：**e.currentTarget.dataset.属性的名称



参数传递的练习



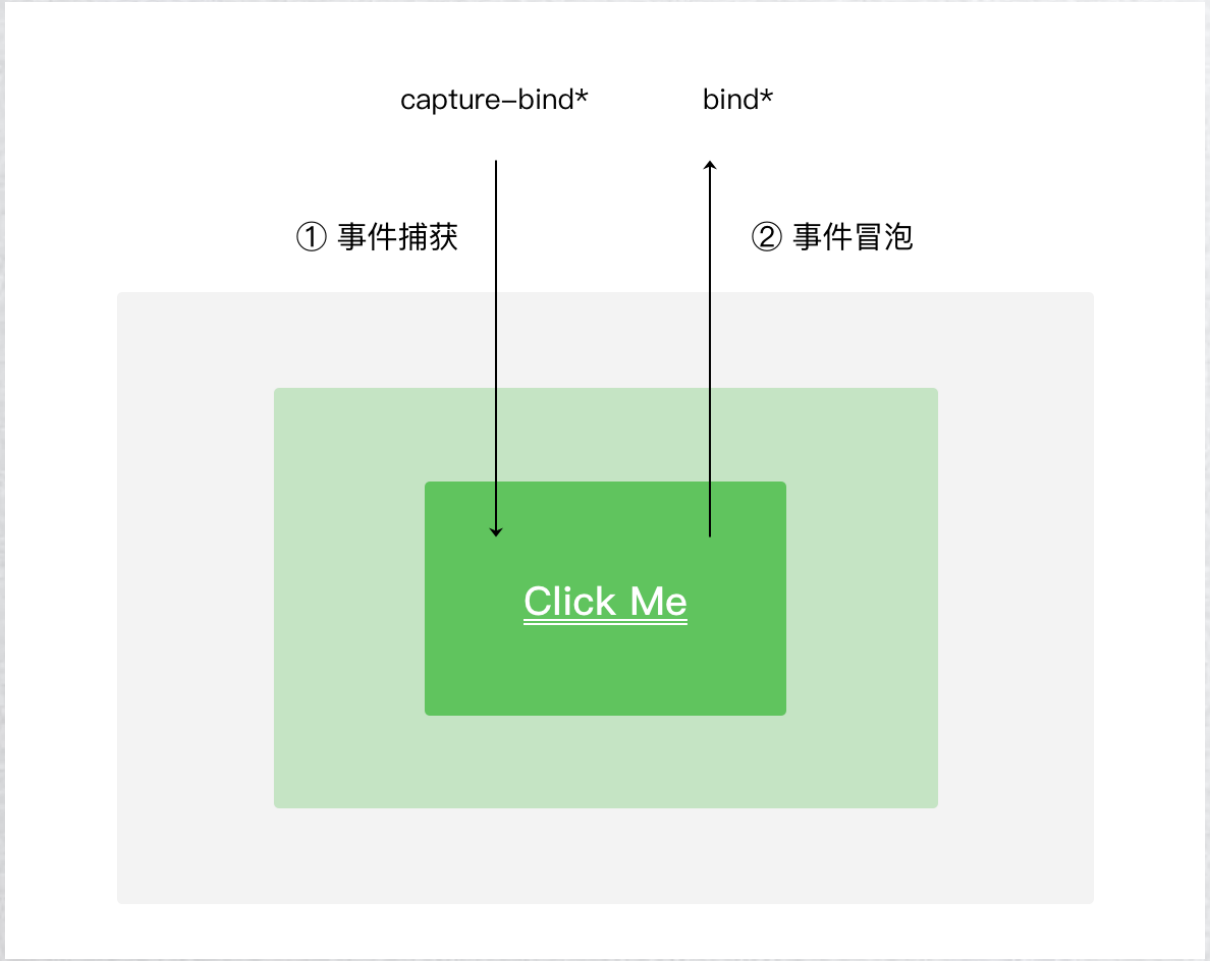

```wxss
/* pages/argument/argument.wxss */
.title {
  display: flex;
  background: purple;
}

.item {
  flex: 1;
  color: #fff;
  height: 88rpx;
  line-height: 88rpx;
  text-align: center;
}

.active {
  color: red;
}
```

```wxml
<view class='title'>
  <block wx:for="{{titles}}" wx:key="*this">
    <view class='item {{currentIndex === index ? "active": ""}}'
          bind:tap="onItemTap"
          data-index="{{index}}">
      {{item}}
    </view>
  </block>
</view>
```

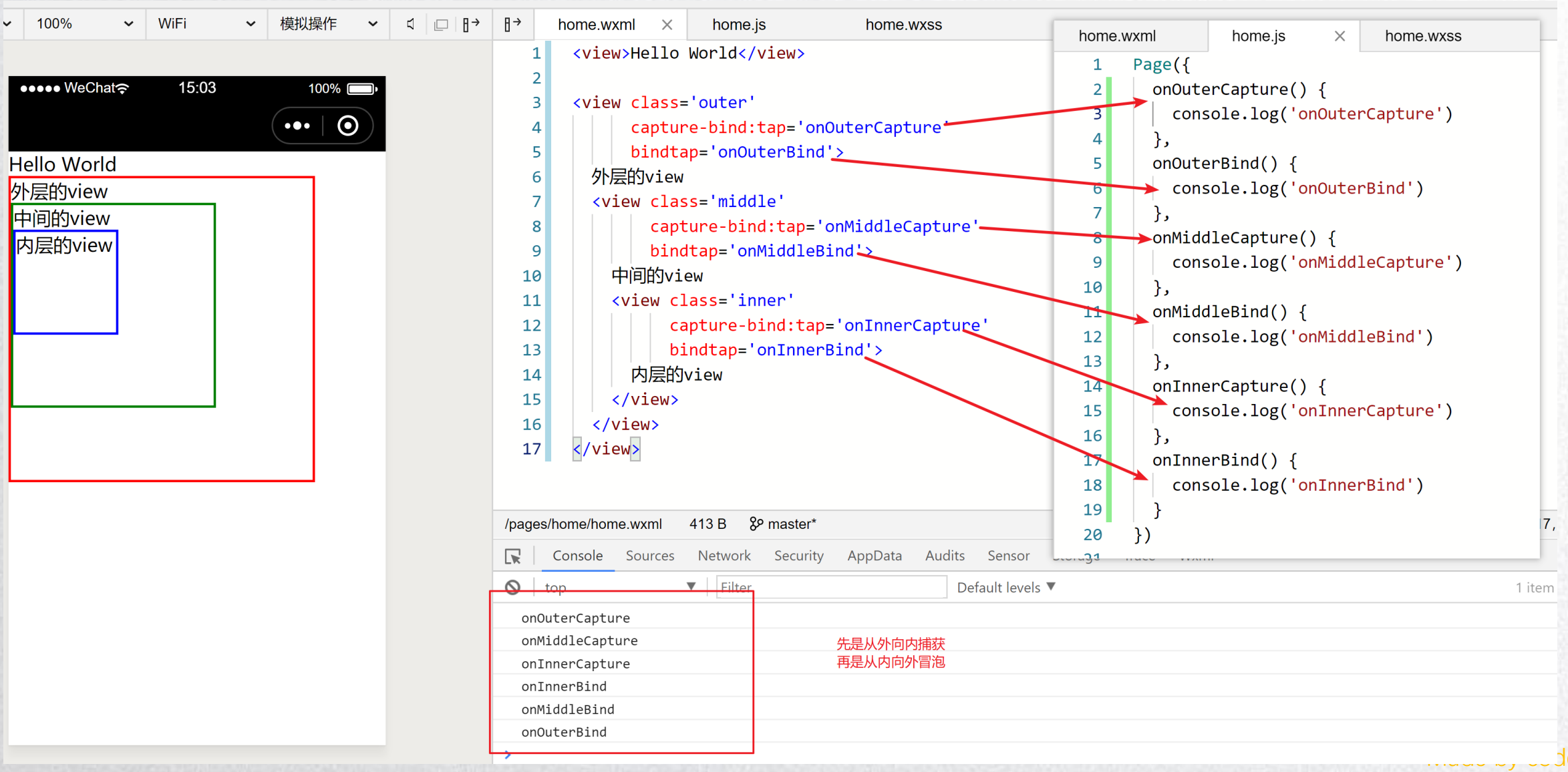
```js
onItemTap(e) {
  // 1.获取当前的index
  const index = e.currentTarget.dataset.index;

  // 2.修改currentIndex
  this.setData({
    currentIndex: index
  })
}
```

事件冒泡和事件捕获

- 当界面产生一个事件时，事件分为了捕获阶段和冒泡阶段。



代码演练


100%
WiFi
模拟操作
WeChat 15:03 100%
Hello World
外层的view
中间的view
内层的view
home.wxml
home.js
home.wxss
<view>Hello World</view>
<view class='outer'
capture-bind:tap='onOuterCapture'
bindtap='onOuterBind'>
外层的view
<view class='middle'
capture-bind:tap='onMiddleCapture'
bindtap='onMiddleBind'>
中间的view
<view class='inner'
capture-bind:tap='onInnerCapture'
bindtap='onInnerBind'>
内层的view
</view>
</view>
</view>
Page({
onOuterCapture() {
console.log('onOuterCapture')
},
onOuterBind() {
console.log('onOuterBind')
},
onMiddleCapture() {
console.log('onMiddleCapture')
},
onMiddleBind() {
console.log('onMiddleBind')
},
onInnerCapture() {
console.log('onInnerCapture')
},
onInnerBind() {
console.log('onInnerBind')
}
})
/pages/home/home.wxml 413 B master*
Console Sources Network Security AppData Audits Sensor
top Filter Default levels 1 item
onOuterCapture
onMiddleCapture
onInnerCapture
onInnerBind
onMiddleBind
onOuterBind
先是从外向内捕获
再是从内向外冒泡