

视图类容器组件.

① view (div).

② scroll-view (可滚动列表).

③ {swiper 轮播图容器

{swiper-item. 轮播图 item. 组件.

swiper 组件常用属性

属性	类型	默认值.	说明
indicator-dots	boolean	false	是否显示指示点
indicator-color	color	rgba.	指示点颜色
indicator-active-color	color	#000000	当前指示点颜色
autoplay	boolean	false	是否自动切换
interval	number	5000ms	切换时间间隔
circular	boolean	false.	是否衔接滑动

text 组件.

text 中的 **selectable** 属性, 实现长按 选中文本内容.

例: <view>

XXXXXX

<text selectable>133XXXX</text>.

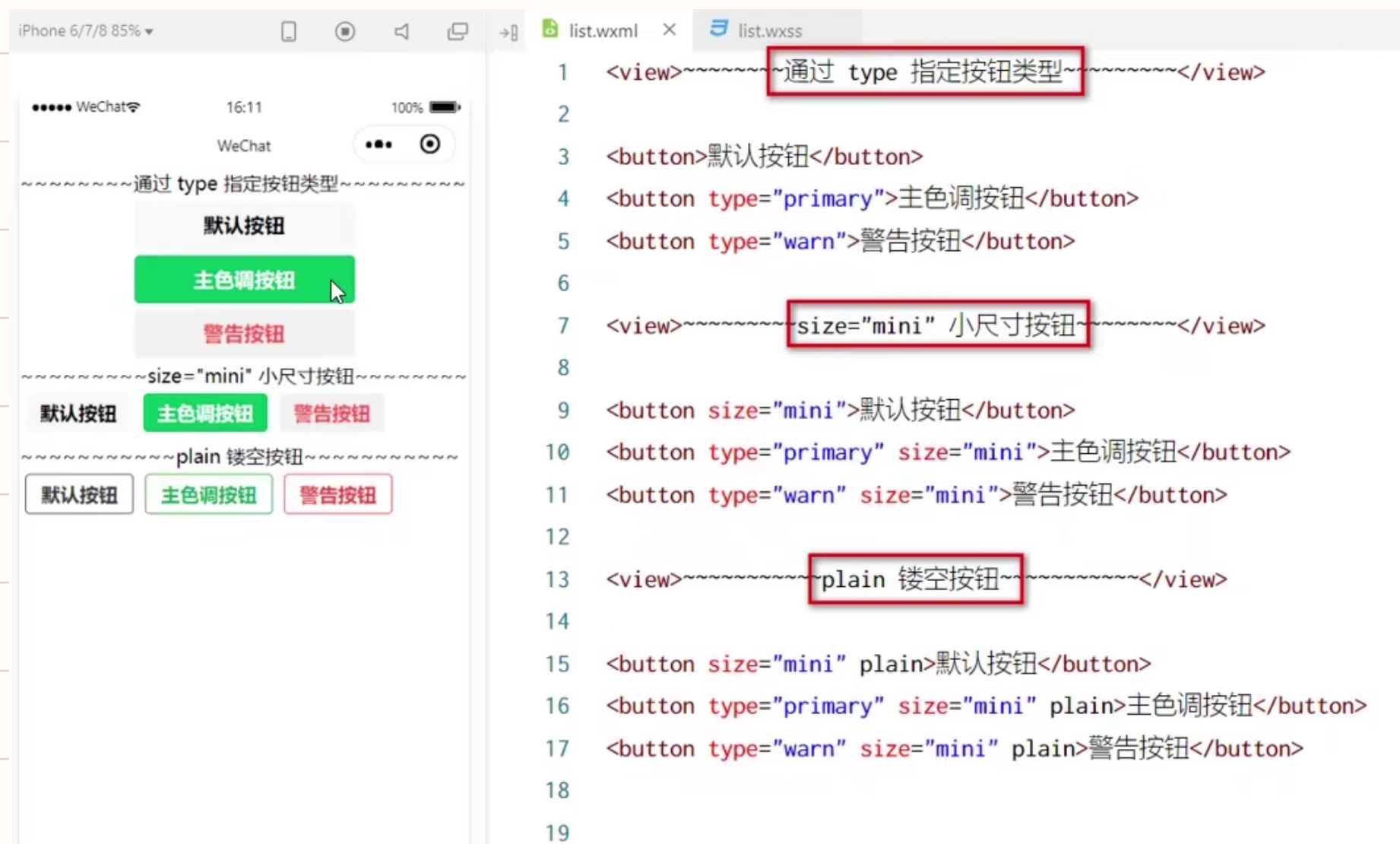
</view>.

注: 133XXXX. 支持长按选中. XXXXXX 不支持.

rich-text :

通过 rich-text 组件的 nodes 属性节点, 把 HTML 字符串渲染为对应 UI.

例: <rich-text nodes = "<h1 style = 'color:red;'>标题</h1>"></rich-text>



images 属性与 html 类似. (这有默认值约 300px, 250px).

image 中的 mode 属性如下.

mode 值	说明
scaleToFill	(默认值) 缩放模式, 不保持纵横比缩放图片, 使图片的宽高完全拉伸至填满 image 元素
aspectFit	缩放模式, 保持纵横比缩放图片, 使图片的长边能完全显示出来。也就是说, 可以完整地将图片显示出来。
aspectFill	缩放模式, 保持纵横比缩放图片, 只保证图片的短边能完全显示出来。也就是说, 图片通常只在水平或垂直方向是完整的, 另一个方向将会发生截取。
widthFix	缩放模式, 宽度不变, 高度自动变化, 保持原图宽高比不变
heightFix	缩放模式, 高度不变, 宽度自动变化, 保持原图宽高比不变

小程序官方把 API 分为了如下 3 大类：

① 事件监听 API

- 特点：以 `on` 开头，用来监听某些事件的触发
- 举例：`wx.onWindowResize(function callback)` 监听窗口尺寸变化的事件

② 同步 API

- 特点1：以 `Sync` 结尾的 API 都是同步 API
- 特点2：同步 API 的执行结果，可以通过函数返回值直接获取，如果执行出错会抛出异常
- 举例：`wx.setStorageSync('key', 'value')` 向本地存储中写入内容

③ 异步 API

- 特点：类似于 jQuery 中的 `$.ajax(options)` 函数，需要通过 `success`、`fail`、`complete` 接收调用的结果
- 举例：`wx.request()` 发起网络数据请求，通过 `success` 回调函数接收数据

数据绑定.

声明数据 =

js 文件中的. data:

例. data: { info: 'Hello World' },

调用数据 =

`<view>{{ info }}</view>`.

即可显示 Hello World .

动态绑定属性

声明:

data = {

imgSrc: 'http://www.~'

}

调用:

`<image src = "{{ imgSrc }}" > </image>`.

算术运算.

```
Page({
  data: {
```

```
    randomNum: Math.random().toFixed(2),
```

```
  }
```

```
})
```

生成一个两位小数的随机数

事件:

类型	绑定方式	事件描述
tap	bindtap 或 bind:tap	手指触摸后马上离开，类似于 HTML 中的 click 事件
input	bindinput 或 bind:input	文本框的输入事件
change	bindchange 或 bind:change	状态改变时触发

事件对象 event.

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

target 是触发该事件的源头组件，而 currentTarget 则是当前事件所绑定的组件。举例如下：

iPhone 6/7/8 85%

10:19

99%

WeChat

WeChat

按钮

```
<view class="outer-view" bindtap="outerHandler">
  <button type="primary">按钮</button>
</view>
```

点击内部的按钮时，点击事件以冒泡的方式向外扩散，也会触发外层 view 的 tap 事件处理函数。

此时，对于外层的 view 来说：

此时，对于外层的 view 来说：

- e.target 指向的是触发事件的源头组件，因此，e.target 是内部的按钮组件
- e.currentTarget 指向的是当前正在触发事件的那个组件，因此，e.currentTarget 是当前的 view 组件

事件传参：

<button type="primary" data-info="{{233}}">+2</button>

在事件处理函数中，通过 event.target.dataset.参数名 即可获取到具体参数的值，示例代码如下：

```
1 btnHandler(event) {  
2   // dataset 是一个对象，包含了所有通过 data-* 传递过来的参数项  
3   console.log(event.target.dataset)  
4   // 通过 dataset 可以访问到具体参数的值  
5   console.log(event.target.dataset.info)  
6 }
```

在小程序中，通过 input 事件来响应文本框的输入事件，语法格式如下：

① 通过 bindinput，可以为文本框绑定输入事件：

```
1 <input bindinput="inputHandler"><input>
```

② 在页面的 .js 文件中定义事件处理函数：

```
1 inputHandler(e) {  
2   // e.detail.value 是变化过后，文本框最新的值  
3   console.log(e.detail.value)  
4 }
```