是全局文件，对全部页面生效，自己创建的目录下的是局部文件，只对当前页面生效。

事件绑定：binftap =” name”

<view bindtap="show">我的订单在这</view>

然后在js文件中设置函数：

show: function () {

console.log("点击");

this.setData({

say: "hi!!",

isShow: false

})

}

## 页面渲染

<view>

{{say}}

</view>

两个大括号中设置参数，然后在js文件中，page里面设置相应的值。

data: {

say: "hello world!",

isShow: true

},

### 条件渲染：if-else语句

<view wx:if ="{{isShow}}">

if判断

</view>

注意wx:if = “ {{ name}}” 有等号及两个大括号；

data: {

say: "hello world!",

isShow: true

},

设置isShow 的值为true，则显示view 的内容。

### **wx:if**

在框架中，使用 wx:if="" 来判断是否需要渲染该代码块：

<view wx:if="{{condition}}"> True </view>

也可以用 wx:elif 和 wx:else 来添加一个 else 块：

<view wx:if="{{length > 5}}"> 1 </view><view wx:elif="{{length > 2}}"> 2 </view><view wx:else> 3 </view>

### **block wx:if**

因为 wx:if 是一个控制属性，需要将它添加到一个标签上。如果要一次性判断多个组件标签，可以使用一个 <block/> 标签将多个组件包装起来，并在上边使用 wx:if 控制属性。

<block wx:if="{{true}}">

<view> view1 </view>

<view> view2 </view></block>

**注意：** <block/> 并不是一个组件，它仅仅是一个包装元素，不会在页面中做任何渲染，只接受控制属性。

#### Block组件

<block wx:if = "isShow">

<view>

相当于if语句：

if() { =>block

=>view语句

}

</view>

</block>

Block是没有意义的组件，可以用作判断语句，相当于if() {}。

##### 注释格式

<!-- 注释方式 -->

#### for循环

<text wx:for = "{{ student }}">

姓名：{{item.name}} \n

年龄：{{item.age}}

</text>

现在js文件中设置好数据内容，

student:[

{name:"小明",age:"18"},

{ name: "小花", age: "12" },

{ name: "小猪", age: "11" }

]

使用item.键名输出内容

#### 使用 wx:for-item 可以指定数组当前元素的变量名

<text wx:for = "{{ student }}" wx:for-item="Na">

姓名：{{Na.name}} \n

年龄：{{Na.age}}

</text>

wx:for 也可以嵌套，下边是一个九九乘法表

<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>

### **block wx:for**

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。例如：

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

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

<view> {{item}} </view></block>

## 循环输出中的数据筛选

<block wx:for = "{{ student }}"wx:for-item="Na">

<text wx:if = "{{ Na.age>15}}" >

姓名：{{Na.name}} \n

年龄：{{Na.age}}

</text>

</block>

在组件中加上if判断即可

#### 换行/n

要在<text> </text>组件里面才会生效,<view>默认为block块级元素（/t空格）

#### template模板使用

先声明模块，name定义属性

<!-- 声明模板 -->

<template name="msgItem">

<view>

<text> {{index}}: {{msg}} </text>

<text> Time: {{time}} </text>

</view>

</template>

然后使用时只需：

<!-- 使用模板 -->

<template is="msgItem" data="{{...message}}"/>

is声明使用模块的名字，data声明使用的数据名字 message需要在js文件中写入：

message:{

index: 0,

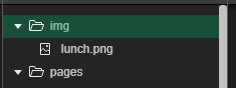
msg: 'this is a template',

time: '2020-03-07'

}

#### Image组件使用

<image src="../../img/lunch.png"></image>



在本地使用相对路径， ../上一层目录../再上一层目录

也可以从根目录开始，

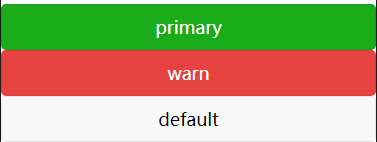
<image src="/img/lunch.png"></image>

#### button组件

<button type="primary">primary</button>

<button type="warn">warn</button>

<button type="default">default</button>



Button有很多属性：

<button type="primary" size="mini">primary</button>

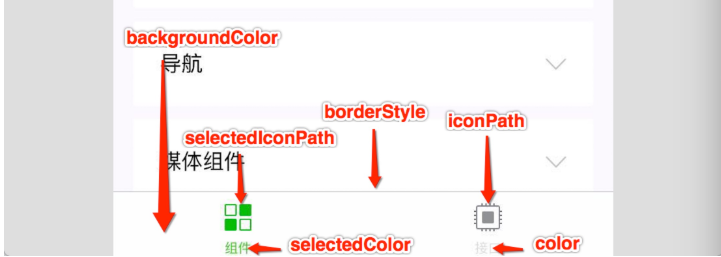
<button type="warn" plain="true">warn</button>

<button type="default" disabled="true">default</button>



#### 全局配置：

##### [tabBar](https://developers.weixin.qq.com/miniprogram/dev/reference/configuration/app.html" \l "tabBar)：底部栏设置



"list": [

{

"pagePath": "pages/index/index",

"text": "首页",

"iconPath": "img/clock.png",

"selectedIconPath": "img/refresh.png"

},

{

"pagePath": "pages/order\_meal/order",

"text": "订单",

"iconPath":"img/clock.png",

"selectedIconPath":"img/refresh.png"

}

],

其中 list 接受一个数组，**只能配置最少 2 个、最多 5 个 tab**。tab 按数组的顺序排序，每个项都是一个对象，其属性值如下：

| **属性** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| pagePath | string | 是 | 页面路径，必须在 pages 中先定义 |
| text | string | 是 | tab 上按钮文字 |
| iconPath | string | 否 | 图片路径，icon 大小限制为 40kb，建议尺寸为 81px \* 81px，不支持网络图片。 **当**position**为**top**时，不显示 icon。** |
| selectedIconPath | string | 否 | 选中时的图片路径，icon 大小限制为 40kb，建议尺寸为 81px \* 81px，不支持网络图片。 **当**position**为**top**时，不显示 icon。** |

##### 图片居中方法：

.header\_left {

width: 30%;

background-color: #999ca8;

height: 240rpx;

display: flex;

align-items: center; //水平居中

justify-content: center; //垂直居中

}



## js文件中page：

如果在函数中要用this，要先设置this的指向=>page，



# 获取用户信息

getUserInfo: function (e) {

console.log(e)

app.globalData.userInfo = e.detail.userInfo

this.setData({

userInfo: e.detail.userInfo,

hasUserInfo: true

})

}

Page({

data: {

motto: 'Hello World',

userInfo: {}, //用户信息，包含userInfo.avatarUrl头像图片，userInfo.nickName用户昵称

hasUserInfo: false,

canIUse: wx.canIUse('button.open-type.getUserInfo')

},

# 小程序数据类型转换

**整数int型转换为字符串：**

var num = 1;

this.data.stringNum = String(num);

console.log(this.data.stringNum);

**字符串转为整数int型**

var intNum  = '1';

int\_num = parseInt(intNum);

console.log(int\_num);

## 字符串转为布尔值

letparam=JSON.parse("false")

## [弹窗：](https://developers.weixin.qq.com/miniprogram/dev/api/ui/interaction/wx.showToast.html" \l "%E6%B3%A8%E6%84%8F)**[wx.showToast](https://developers.weixin.qq.com/miniprogram/dev/api/ui/interaction/wx.showToast.html" \l "%E6%B3%A8%E6%84%8F)**

## **示例代码**

wx.showToast({

title: '成功',

icon: 'success',

duration: 2000})

## wx.showToast接口只提供了两种icon【success和loading】展示形式，但是在实际开发中并不满足的。这里可以通过加image:'图片路径'

wx.showToast({

title:"成功",

icon: 'loading...',//图标，支持"success"、"loading"

image: '/images/tan.png',//自定义图标的本地路径，image 的优先级高于 icon

duration: 2000,//提示的延迟时间，单位毫秒，默认：1500

mask: false,//是否显示透明蒙层，防止触摸穿透，默认：false

success:function(){},

fail:function(){},

complete:function(){}

})

},

## 表单：form

<form bindsubmit="login">

bindsumit =“点击提交执行的函数名”，不要在button设置bindtab，

input标签里需要设置name，才能获取到数据

<input type="text" confirm-type="next" focus="ture" placeholder="请输入账号" name="user"></input>

switchTab 和 navigateTo 和navigator

1、navigator 标签使用

<navigator url="../sign/sign">未有账号？点击注册 -></navigator>

1. navigateTo

保留当前页面，跳转到应用内的某个页面。但是**不能跳到 tabbar 页面**。使用 [wx.navigateBack](https://developers.weixin.qq.com/miniprogram/dev/api/route/wx.navigateBack.html) 可以返回到原页面，如果要跳转到tabbar界面，要用switchTab

wx.navigateTo({

url: '../login/login',

})

3、switchTab 跳转到tabbar页面

wx.switchTab({

url: '/pages/index/index',

})

数据库查询：

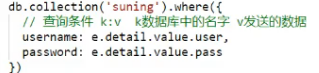
db\_data.where({

// 数据库中的用户名：表单提交的用户名

user\_name: user\_name,

user\_pwd: user\_pwd

})

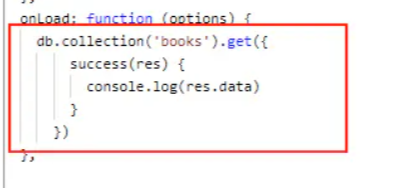


const db = wx.cloud.database()

db.collection('todos').where({

\_openid: 'xxx' // 填入当前用户 openid}).get().then(res => {

console.log(res.data)})



## 轮播图swiper

# **swiper**

基础库 1.0.0 开始支持，低版本需做[兼容处理](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html)。

滑块视图容器。其中只可放置[swiper-item](https://developers.weixin.qq.com/miniprogram/dev/component/swiper-item.html)组件，否则会导致未定义的行为。

| **属性** | **类型** | **默认值** | **必填** | **说明** | **最低版本** |
| --- | --- | --- | --- | --- | --- |
| indicator-dots | boolean | false | 否 | 是否显示面板指示点 | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| indicator-color | color | rgba(0, 0, 0, .3) | 否 | 指示点颜色 | [1.1.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| indicator-active-color | color | #000000 | 否 | 当前选中的指示点颜色 | [1.1.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| autoplay | boolean | false | 否 | 是否自动切换 | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| current | number | 0 | 否 | 当前所在滑块的 index | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| interval | number | 5000 | 否 | 自动切换时间间隔 | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| duration | number | 500 | 否 | 滑动动画时长 | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| circular | boolean | false | 否 | 是否采用衔接滑动 | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| vertical | boolean | false | 否 | 滑动方向是否为纵向 | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| previous-margin | string | "0px" | 否 | 前边距，可用于露出前一项的一小部分，接受 px 和 rpx 值 | [1.9.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| next-margin | string | "0px" | 否 | 后边距，可用于露出后一项的一小部分，接受 px 和 rpx 值 | [1.9.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| display-multiple-items | number | 1 | 否 | 同时显示的滑块数量 | [1.9.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| skip-hidden-item-layout | boolean | false | 否 | 是否跳过未显示的滑块布局，设为 true 可优化复杂情况下的滑动性能，但会丢失隐藏状态滑块的布局信息 | [1.9.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| easing-function | string | "default" | 否 | 指定 swiper 切换缓动动画类型 | [2.6.5](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| bindchange | eventhandle |  | 否 | current 改变时会触发 change 事件，event.detail = {current, source} | [1.0.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| bindtransition | eventhandle |  | 否 | swiper-item 的位置发生改变时会触发 transition 事件，event.detail = {dx: dx, dy: dy} | [2.4.3](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |
| bindanimationfinish | eventhandle |  | 否 | 动画结束时会触发 animationfinish 事件，event.detail 同上 | [1.9.0](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) |

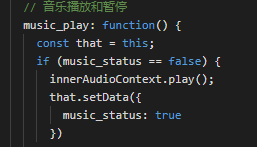
**easing-function 的合法值**

| **值** | **说明** | **最低版本** |
| --- | --- | --- |
| default | 默认缓动函数 |  |
| linear | 线性动画 |  |
| easeInCubic | 缓入动画 |  |
| easeOutCubic | 缓出动画 |  |
| easeInOutCubic | 缓入缓出动画 |  |

## **change事件 source 返回值**

## 常见错误：不能直接用data里的名字，

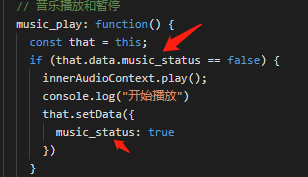
只能在setDate里使用



这样会报错：



### 正确用法：



设置动画：

.music\_img\_move {

/\* 设置动画 animation: 动画名 完成时间 持续时间 动画缓动\*/

animation: move 13s infinite linear;

}

/\* 搭配animation使用 \*/

@keyframes move {

0% {

transform: rotate(0deg)

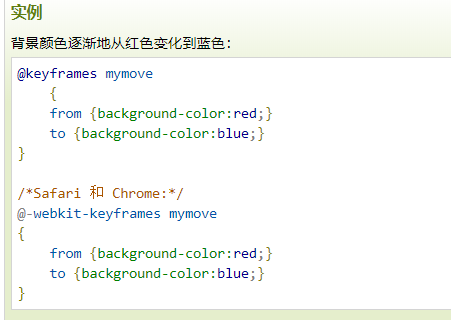
}

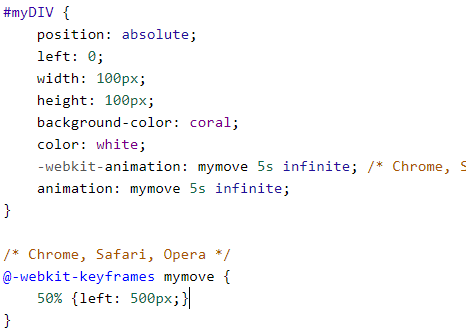
100% {

transform: rotate(360deg)

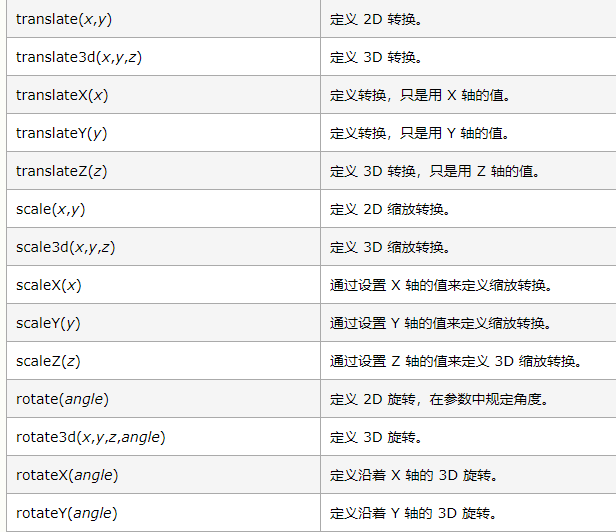
}

}

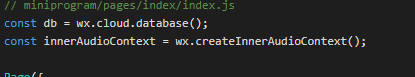








## 音乐播放：



**先在js文件里page外面声明实例**

# **InnerAudioContext**

InnerAudioContext 实例，可通过 [wx.createInnerAudioContext](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/wx.createInnerAudioContext.html) 接口获取实例。

常用属性：

**属性**

## **string src**

音频资源的地址，用于直接播放。[2.2.3](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) 开始支持云文件ID

### **number startTime**

开始播放的位置（单位：s），默认为 0

### **boolean autoplay**

是否自动开始播放，默认为 false

### **boolean loop**

常用方法：

## **方法**

### **[InnerAudioContext.play()](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.play.html)**

播放

### **[InnerAudioContext.pause()](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.pause.html)**

暂停。暂停后的音频再播放会从暂停处开始播放

### **[InnerAudioContext.stop()](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.stop.html)**

停止。停止后的音频再播放会从头开始播放。

### **[InnerAudioContext.onPlay(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.onPlay.html)**

监听音频播放事件

### **[InnerAudioContext.offPlay(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.offPlay.html)**

取消监听音频播放事件

### **[InnerAudioContext.onPause(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.onPause.html)**

监听音频暂停事件

### **[InnerAudioContext.offPause(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.offPause.html)**

取消监听音频暂停事件

### **[InnerAudioContext.onStop(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.onStop.html)**

监听音频停止事件

### **[InnerAudioContext.offStop(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.offStop.html)**

取消监听音频停止事件

### **[InnerAudioContext.onEnded(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.onEnded.html)**

监听音频自然播放至结束的事件

### **[InnerAudioContext.offEnded(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/audio/InnerAudioContext.offEnded.html)**

取消监听音频自然播放至结束的事件

## **示例代码**

const innerAudioContext = wx.createInnerAudioContext()

innerAudioContext.autoplay = true

innerAudioContext.src = 'http://ws.stream.qqmusic.qq.com/'

innerAudioContext.onPlay(() => {

console.log('开始播放')})

innerAudioContext.onError((res) => {

console.log(res.errMsg)

console.log(res.errCode)})

## 实现歌曲上一首播放：

// 上一首

last\_music: function() {

const that = this;

// innerAudioContext.pause();

console.log("上一首")

console.log(that.data.music\_id)

// 获取数据长度

const music\_length = that.data.music\_data.length;

// 歌曲ID加1

const music\_ID = that.data.music\_id+1

that.setData({

music\_id: music\_ID

})

console.log(that.data.music\_data[music\_ID].mp3)

// 设置新的音乐链接，这里注意，要配合innerAudioContext.autoplay才能实现下一首播放

innerAudioContext.src = that.data.music\_data[music\_ID].mp3;

// 开启自动播放

innerAudioContext.play();

innerAudioContext.autoplay = true // 开启自动播放

console.log("播放成功")

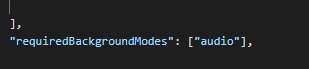
console.log(that.data.music\_id)

},

注意：改变了innerAudioContext.src后，要加上innerAudioContext.autoplay = true

## 开启背景音乐播放

**首先在页的JSON配置项添加  
“requiredBackgroundModes” : [“autio”]**



# **[BackgroundAudioManager](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.html)**

BackgroundAudioManager 实例，可通过 [wx.getBackgroundAudioManager](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/wx.getBackgroundAudioManager.html) 获取。

## **属性**

### **string src**

音频的数据源（[2.2.3](https://developers.weixin.qq.com/miniprogram/dev/framework/compatibility.html) 开始支持云文件ID）。默认为空字符串，**当设置了新的 src 时，会自动开始播放**，目前支持的格式有 m4a, aac, mp3, wav。

## **方法**

### **[BackgroundAudioManager.play()](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.play.html)**

播放音乐

### **[BackgroundAudioManager.pause()](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.pause.html)**

暂停音乐

### **[BackgroundAudioManager.seek(number currentTime)](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.seek.html)**

跳转到指定位置

### **[BackgroundAudioManager.stop()](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.stop.html)**

停止音乐

### **[BackgroundAudioManager.onCanplay(function callback)](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.onCanplay.html)**

### 解决首次播放时，设置**[BackgroundAudioManager.s](https://developers.weixin.qq.com/miniprogram/dev/api/media/background-audio/BackgroundAudioManager.stop.html)rc**

会重新播放的BUG

只需要设置一个状态，然后判断是否第一次播放

if (that.data.music\_status == false) {

// 判断是否第一次播放

if (that.data.music\_start==false) {

backgroundAudioManager.src = that.data.music\_data[that.data.music\_id].mp3;

backgroundAudioManager.title = that.data.music\_data[that.data.music\_id].name; // 必须设置title

console.log("首次播放")

that.setData({

music\_start:true,

music\_status: true

})

}

else {

backgroundAudioManager.play();

console.log("不是首次播放")

that.setData({

music\_status: true

})

}

}

## **JSON**

### **方法**

* stringify(object): 将 object 对象转换为 JSON 字符串，并返回该字符串。
* parse(string): 将 JSON 字符串转化成对象，并返回该对象。

## Math方法：

[https://developers.weixin.qq.com/miniprogram/dev/reference/wxs/07basiclibrary.html#Math](https://developers.weixin.qq.com/miniprogram/dev/reference/wxs/07basiclibrary.html" \l "Math)

## Math 对象方法

|  |  |
| --- | --- |
| **方法** | **描述** |
| [abs(x)](https://www.runoob.com/jsref/jsref-abs.html) | 返回 x 的绝对值。 |
| [acos(x)](https://www.runoob.com/jsref/jsref-acos.html) | 返回 x 的反余弦值。 |
| [asin(x)](https://www.runoob.com/jsref/jsref-asin.html) | 返回 x 的反正弦值。 |
| [atan(x)](https://www.runoob.com/jsref/jsref-atan.html) | 以介于 -PI/2 与 PI/2 弧度之间的数值来返回 x 的反正切值。 |
| [atan2(y,x)](https://www.runoob.com/jsref/jsref-atan2.html) | 返回从 x 轴到点 (x,y) 的角度（介于 -PI/2 与 PI/2 弧度之间）。 |
| [ceil(x)](https://www.runoob.com/jsref/jsref-ceil.html) | 对数进行上舍入。 |
| [cos(x)](https://www.runoob.com/jsref/jsref-cos.html) | 返回数的余弦。 |
| [exp(x)](https://www.runoob.com/jsref/jsref-exp.html) | 返回 Ex 的指数。 |
| [floor(x)](https://www.runoob.com/jsref/jsref-floor.html) | 对 x 进行下舍入。 |
| [log(x)](https://www.runoob.com/jsref/jsref-log.html) | 返回数的自然对数（底为e）。 |
| [max(x,y,z,...,n)](https://www.runoob.com/jsref/jsref-max.html) | 返回 x,y,z,...,n 中的最高值。 |
| [min(x,y,z,...,n)](https://www.runoob.com/jsref/jsref-min.html) | 返回 x,y,z,...,n中的最低值。 |
| [pow(x,y)](https://www.runoob.com/jsref/jsref-pow.html) | 返回 x 的 y 次幂。 |
| [random()](https://www.runoob.com/jsref/jsref-random.html) | 返回 0 ~ 1 之间的随机数。 |
| [round(x)](https://www.runoob.com/jsref/jsref-round.html) | 四舍五入。 |
| [sin(x)](https://www.runoob.com/jsref/jsref-sin.html) | 返回数的正弦。 |
| [sqrt(x)](https://www.runoob.com/jsref/jsref-sqrt.html) | 返回数的平方根。 |
| [tan(x)](https://www.runoob.com/jsref/jsref-tan.html) | 返回角的正切。 |

取随机数：

Math.floor(X) =X的整数位  
例如  
Math.floor(6.999)  ===  6  
Math.floor(39.001)   ===  39  
Math.floor(8)   ===  8

## **const 和let 区别使用：**

const用于永远不会变的值，会变化的值要用let或var

const abs\_num = Math.abs(ram\_num - music\_num) //取一个绝对值

let good\_num = ''; //此处不能用const，因为下面会导致good\_num变化

if(abs\_num>music\_num-1) {

console.log("大于情况"+abs\_num)

good\_num = abs\_num-1

}

else {

good\_num = abs\_num

}

文本超出范围处理：

/\* 文本控制一行显示 \*/

white-space: nowrap;

/\* 文本超出隐藏 \*/

overflow: hidden;

/\* 超出部分的显示方式 \*/

text-overflow: ellipsis;

## 动画

<https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.html>

[https://developers.weixin.qq.com/miniprogram/dev/framework/view/animation.html#%E6%BB%9A%E5%8A%A8%E9%A9%B1%E5%8A%A8%E7%9A%84%E5%8A%A8%E7%94%BB](https://developers.weixin.qq.com/miniprogram/dev/framework/view/animation.html" \l "%E6%BB%9A%E5%8A%A8%E9%A9%B1%E5%8A%A8%E7%9A%84%E5%8A%A8%E7%94%BB)

<https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.html>

onShow: function(){

var animation = wx.createAnimation({

duration: 1000,

timingFunction: 'ease',

})

# **Animation**

动画对象

## **方法**

### **[Array.<Object> Animation.export()](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.export.html)**

导出动画队列。**export 方法每次调用后会清掉之前的动画操作。**

### **[Animation Animation.step(Object object)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.step.html)**

表示一组动画完成。可以在一组动画中调用任意多个动画方法，一组动画中的所有动画会同时开始，一组动画完成后才会进行下一组动画。

### **[Animation Animation.matrix()](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.matrix.html)**

同 [transform-function matrix](https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/matrix" \t "https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/_blank)

### **[Animation Animation.matrix3d()](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.matrix3d.html)**

同 [transform-function matrix3d](https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/matrix3d" \t "https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/_blank)

### **[Animation Animation.rotate(number angle)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.rotate.html)**

从原点顺时针旋转一个角度

### **[Animation Animation.rotate3d(number x, number y, number z, number angle)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.rotate3d.html)**

从 固定 轴顺时针旋转一个角度

### **[Animation Animation.rotateX(number angle)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.rotateX.html)**

从 X 轴顺时针旋转一个角度

### **[Animation Animation.rotateY(number angle)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.rotateY.html)**

从 Y 轴顺时针旋转一个角度

### **[Animation Animation.rotateZ(number angle)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.rotateZ.html)**

从 Z 轴顺时针旋转一个角度

### **[Animation Animation.scale(number sx, number sy)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.scale.html)**

缩放

### **[Animation Animation.scale3d(number sx, number sy, number sz)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.scale3d.html)**

缩放

### **[Animation Animation.scaleX(number scale)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.scaleX.html)**

缩放 X 轴

### **[Animation Animation.scaleY(number scale)](https://developers.weixin.qq.com/miniprogram/dev/api/ui/animation/Animation.scaleY.html)**

## setTimeout 定时器

# **number setTimeout(function callback, number delay, any rest)**

设定一个定时器。在定时到期以后执行注册的回调函数

## **参数**

### **function callback**

回调函数

### **number delay**

延迟的时间，函数的调用会在该延迟之后发生，单位 ms。

### **any rest**

param1, param2, ..., paramN 等附加参数，它们会作为参数传递给回调函数。

## **返回值**

### **number**

定时器的编号。这个值可以传递给 [clearTimeout](https://developers.weixin.qq.com/miniprogram/dev/api/base/timer/clearTimeout.html) 来取消该定时。

fight:**function**(e){

   that = **this**;

   that.setData({

      buttondisabled: **true**,

      loading: **true**

   });     //<--这里是分号不是逗号

   setTimeout(**function**(){

      that.setData({

         buttondisplay: 'none',

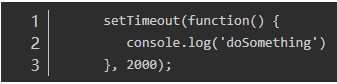
         playerdisplay: 'block'

      })        //这里的错误不太好描述，自己看了

   }, 9000);

}

# **setTimeout**

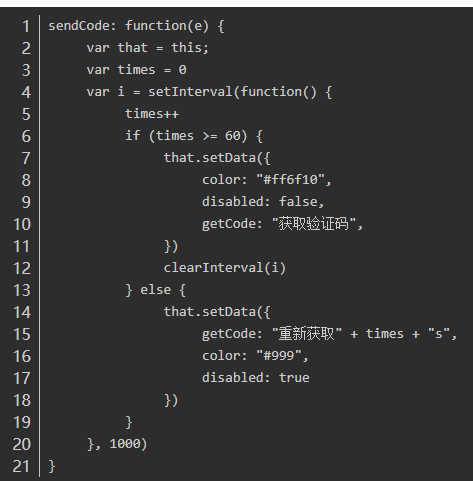


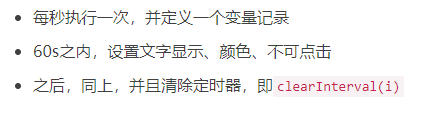
* 上面就是一个2s的定时器，最基础的简单用法。
* 这个方法也可以返回一个id，即定时器id，用来清除定时，比如：



# **setInterval**

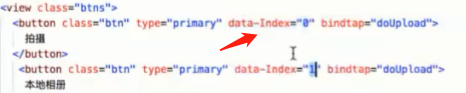
setInterval与setTimeout的使用差别不大，参数都是一样的，区别就在于setTimeout是到时执行一次，setInterval是根据设置的时间来回调的，比如每秒回调一次。

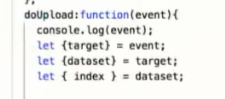




## 解构赋值

在给button添加data-index属性后，点击事件就会获取到，





点击事件传递的值，data-数据名：可以获取到值

1、选择文字有对应的效果（颜色，线条）

