



# 微信小程序开发新手实战教程

创建快捷项目




wechatDemo


+




编辑




调试



项目




编译




关闭

▸ pages

▸ utils

 app.js

 app.json

{ } app.wxss

**app.js** 是小程序的脚本代码。我们可以在这个文件中监听并处理小程序的生命周期函数、声明全局变量。调用框架提供的丰富的 **API**，如本例的同步存储及同步读取本地数据。

**app.json** 是对整个小程序的全局配置。我们可以在这个文件中配置小程序是由哪些页面组成，配置小程序的窗口 背景色，配置导航条样式，配置默认标题。注意**该文件不可添加任何注释**。

**app.wxss** 是整个小程序的公共样式表。我们可以在页面组件的 **class** 属性上直接使用 **app.wxss** 中声明的样式规则。

```
App({
  onLaunch: function () {
    //调用 API 从本地缓存中获取数据
    var logs = wx.getStorageSync('logs') || []
    logs.unshift(Date.now())
    wx.setStorageSync('logs', logs)
  },
  getUserInfo:function(cb){
    var that = this
    if(this.globalData.userInfo){
      typeof cb == "function" && cb(this.globalData.userInfo)
    }else{
      //调用登录接口
      wx.login({
        success: function () {
          wx.getUserInfo({
            success: function (res) {
              that.globalData.userInfo = res.userInfo
              typeof cb == "function" && cb(that.globalData.userInfo)
            }
          })
        }
      })
    }
  },
  globalData:{
    userInfo:null
  }
})
```

```
    },
    onShow:function() {
      console.log("show");
    },
    onHide:function() {
      console.log("hide");
    }
  })
}
```

**注意**必须在 `app.js` 中注册 `App()`，不能注册多个。

**onLaunch:** 程序初始化执行，且只执行一次。

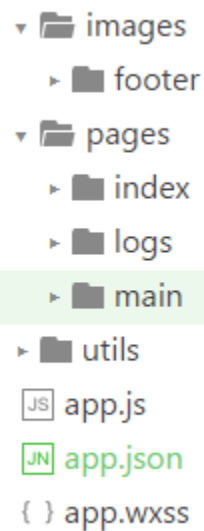
**onShow:**程序启动，或从程序后台进入前台时执行。

**onHide:**程序从前台进入后台时执行。

可以添加任意函数到 `OBJECT` 中，使用 `this` 访问。

底部导航

添加 `images` 目录，放入图片资源并添加主程序页面



```
▼ images
  ► footer
▼ pages
  ► index
  ► logs
  ► main
  ► utils
JS app.js
JN app.json
{ } app.wxss
```

添加底部导航 `tabBar`

```
{
  "pages":[
    "pages/index/index",
    "pages/logs/logs",
    "pages/main/main"
  ],
  "window":{
    "backgroundTextStyle":"light",
    "navigationBarBackgroundColor": "#dddddd",
    "navigationBarTitleText": "WeChat",
    "navigationBarTextStyle":"black"
  },
  "tabBar":{
    "color":"#000",
    "selectedColor":"#48c33c",
    "borderStyle":"block",
    "backgroundColor":"#ffffff",
    "list":[{
      "pagePath":"pages/index/index",
      "iconPath":"images/footer/index.png",
      "selectedIconPath":"images/footer/indexS.png",
      "text":"首页"
    },
    {
      "pagePath":"pages/logs/logs",
      "iconPath":"images/footer/logs.png",
      "selectedIconPath":"images/footer/logsS.png",
      "text":"日志"
    },
    {
      "pagePath":"pages/main/main",
      "iconPath":"images/footer/main.png",
      "selectedIconPath":"images/footer/mainS.png",

```

```
        "text": "主程序"
    }
]
}
```

**color:** 文字默认颜色

**selectedColor:** 文字选中颜色

**borderStyle:** 上边框颜色（只支持 black/white）

**backgroundColor:** 背景色

**list:** 菜单列表

**list 属性**

**pagePath:** 页面路径（需要在 pages 中初始化）

**iconPath:** 图片路径，大小限制 40kb

**selectedIconPath:** 选中样式图片路径，大小限制 40kb

**text:** 按钮文字

WeChat



Song

Hello World



首页



日志



主程序

数据绑定

main.js

```
Page({
  data:{
    text:"这是一个页面"
  },
  onLoad:function(options){
    // 页面初始化 options 为页面跳转所带来的参数
  },
  onReady:function(){
    // 页面渲染完成
  },
  onShow:function(){
    // 页面显示
  },
  onHide:function(){
    // 页面隐藏
  },
  onUnload:function(){
    // 页面关闭
  }
})
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7

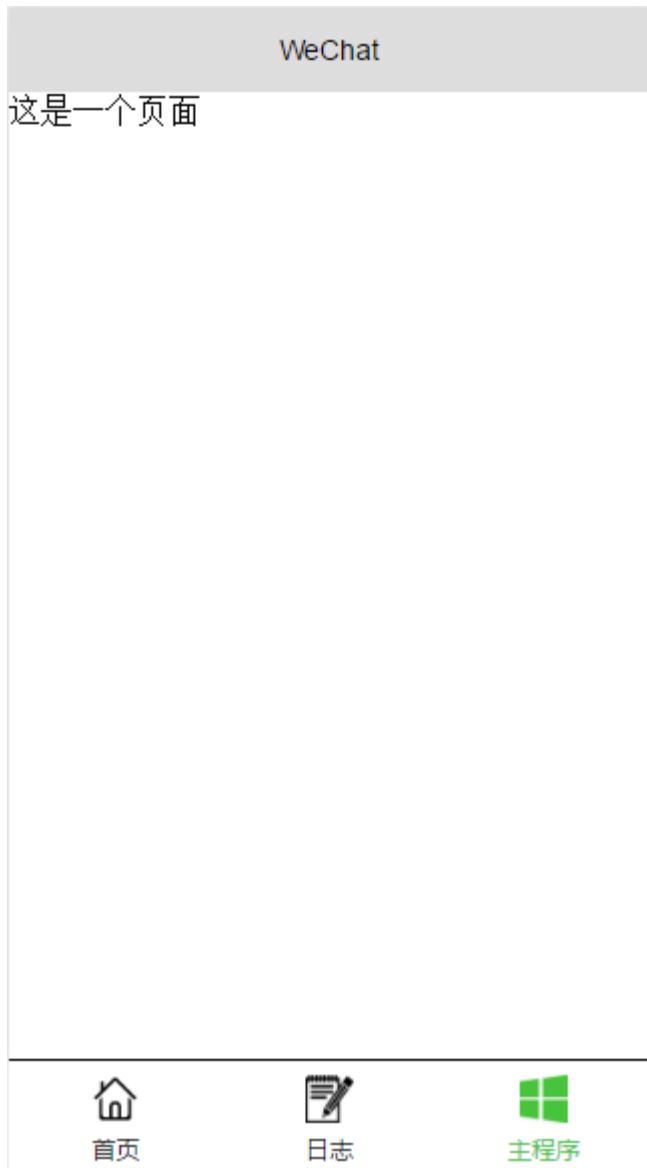
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20



main.wxml

```
<view>
  <text>{{text}}</text>
</view>
```

- 1
- 2
- 3
- 1
- 2
- 3



view 组件

**flex-direction:** row: 横向排列; column: 纵向排列

**justify-content:** flex-start: 左对齐; flex-end: 右对齐; center: 居中; space-between: 两端分散对齐; space-around:

居中分散对齐

**align-items:** flex-start: 垂直顶部; flex-end: 垂直底部; center: 垂直居中

轮播

```
<swiper indicator-dots="true" autoplay="true" duration="1000" bindchange=
"listenSwiper" >
  <swiper-item>
    <view style="background: red; height: 150px"></view>
  </swiper-item>
  <swiper-item>
    <view style="background: green; height: 150px"></view>
  </swiper-item>
  <swiper-item>
    <view style="background: blue; height: 150px"></view>
  </swiper-item>
</swiper>
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 1
- 2
- 3
- 4
- 5
- 6

- 7
- 8
- 9
- 10
- 11

属性名 类型 默认值 说明

indicator-dots Boolean false 是否显示面板指示点

autoplay Boolean false 是否自动切换

current Number 0 当前所在页面的 index

interval Number 5000 自动切换时间间隔

duration Number 1000 滑动动画时长

bindchange EventHandle current 改变时会触发 change 事件，event.detail = {current: current}

注意：其中只可放置组件，其他节点会被自动删除。

仅可放置在组件中，宽高自动设置为 100%。

获取轮播改变事件

```
listenSwiper:function(e) {  
  console.log(e)  
},
```

- 1
- 2
- 3
- 1
- 2
- 3

值

```
Object {target: Object, currentTarget: Object, type: "change", timeStamp: 353  
45, detail: Object}  
currentTarget  
:
```

```
Object
detail
:
Object
target
:
Object
timeStamp
:
35345
type
:
"change"
__proto__
:
Object
```

## 图标 icon

type 有效值: success, success\_no\_circle, info, warn, waiting, cancel, download, search, clear

size 默认 23px

color 同 css 的 color

<!--成功图标-->

```
<icon type="success" size="40"/>
```

<!--安全成功标志图标-->

```
<icon type="safe_success" size="40"/>
```

<!--提示信息图标-->

```
<icon type="info" size="40"/>
```

<!--带圆的信息提示图标-->

```
<icon type="info_circle" size="40"/>
```

<!--不带圆的成功图标-->

```
<icon type="success_no_circle" size="40"/>
```

<!--带圆的成功图标-->

```

<icon type="success_circle" size="40"/>
<!--警告图标-->
<icon type="warn" size="40"/>
<!--带圆的等待图标-->
<icon type="waiting_circle" size="40"/>
<!--等待图标-->
<icon type="waiting" size="40"/>
<!--下载图标-->
<icon type="download" size="40"/>
<!--取消图标-->
<icon type="cancel" size="40"/>
<!--清除图标-->
<icon type="clear" size="40"/>
<!--改变颜色的 success-->
<icon type="success" size="40" color="red"/>

```

## 进度条

percent Float 无 百分比 0~100

show-info Boolean false 在进度条右侧显示百分比

stroke-width Number 6 进度条线的宽度，单位 px

color Color #09BB07 进度条颜色

active Boolean false 进度条从左往右的动画

```
<progress percent="80" show-info="true" stroke-width="5" color="red" active="true"/>
```

## 按钮 button

```

<button type="defaule" bindtap="clickButton">Defalut</button>
<!--原始颜色，不可点击状态， 正在加载状态-->
<button type="primary" disabled="true" loading="true">Primary</button>
<button type="warn">warn</button>

```

**注： button-hover** 默认为{background-color: rgba(0, 0, 0, 0.1); opacity: 0.7;}

## 按钮点击事件

```

clickButton: function(e) {
  console.log(e);
},

```

## CHECKBOX

<!--checkbox-group 是 checkbox 的组，使用 bindchange，监听数据选中和取消-->

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

```
  <label style="display: flex;" wx:for-items="{{items}}">
```

```
    <checkbox value="{{item.name}}" checked="{{item.checked}}"/>{{item.value}}
```

```
  </label>
```

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

绑定事件

绑定数据

```
items: [  
  {  
    name: 'S',  
    value: 'S',  
    checked: 'true'  
  },  
  {  
    name: 'O',  
    value: 'O'  
  },  
  {  
    name: 'N',  
    value: 'N'  
  },  
  {  
    name: 'G',  
    value: 'G'  
  },  
  {  
    name: 'SONG',  
    value: 'SONG'  
  }  
]
```

绑定监听事件

```
listenCheckboxChange:function(e) {  
    console.log(e);  
},
```

至此页面

