本章节介绍微信小程序已支持的多媒体资源，分别为 image（图片）、camera（照相机）、audio（音频）、video（视频）等媒体组件。

图片

用法

本小节介绍 image 组件的使用方法。

在这六种多媒体组件中，开发者最为熟悉的应该就是 image 组件了。几乎任意一个可提供服务的小程序中都会用到图片资源。它的使用方式也很简单：

引入单个图片资源，仅需一行代码：

```
<image style="{{style}}" mode="{{mode}}" src="{{src}}"></image>
```

循环展示一个数组中的图片数据：

```
<view class="section section_gap" wx:for="{{array}}" wx:for-item="item">
  <view>{{item.text}}</view>
  <view>
    <image style="width: 200px; height: 200px; background-color: #eeeeee;" mode="{{item.mode}}" src="{{item.src}}"></image>
  </view>
</view>
```

属性

本小节介绍 image 组件的属性。

image 组件共 5 个属性，最常使用的为 src 和 mode。

1. src - 图片资源地址字符串，String 类型，无默认值；

2. mode - 图片裁剪、缩放的模式，String 类型，默认值为 'scaleToFill' ；

3. lazy-load - 图片懒加载（只针对 page 与 scroll-view 下的 image 有效），Boolean 类型，默认值为 false ；

4. binderror - 当错误发生时，发布到 AppService 的事件名，事件对象 event.detail = {errMsg: 'something wrong'}， HandleEvent 类型，无默认值；

5. bindload - 当图片载入完毕时，发布到 AppService 的事件名，事件对象 event.detail = {height:'图片高度 px', width:'图片宽度 px'}，HandleEvent 类型，无默认值。

有了这 5 种属性，开发者在提供了有效的资源地址后，可以决定是否让图片进行懒加载、如何裁剪和缩放图片、捕获图片加载过程出错事件和图片加载完毕事件。加载出错时可以获知出错原因，加载完毕时可以获取图片原始宽高。

相机

用法

使用系统相机功能时，需要用户授权 scope.camera 。用户同意授权后，可以操作系统相册或者使用拍照功能。

下面是一段调用系统相机的示例代码：点击拍照按钮，调用相机拍照，展示已拍摄图片的预览图。

```
<!-- camera.wxml -->
<camera device-position="back" flash="off" binderror="error" style="width: 100%; height: 300px;"></camera>
<button type="primary" bindtap="takePhoto">拍照</button>
<view>预览</view>
<image mode="widthFix" src="{{src}}"></image>
```

```
// camera.js

Page({
  takePhoto() {
    const ctx = wx.createCameraContext()
    ctx.takePhoto({
      quality: 'high',
      success: (res) => {
```

```
        this.setData({
          src:  res.tempImagePath
        })
      }
    })
  },
  error(e)  {
    console.log(e.detail)
  }
})
```

属性

本小节介绍 camera 组件的 4 个属性。

1. device-position - 前置或后置摄像头， String 类型， 取值为 'front' 和 'back'，默认值为 'back' ；

2. flash - 闪光灯，String 类型，取值为 'auto'、 'on'、 'off'，默认值为 'auto' ；

3. bindstop - 摄像头在非正常终止时触发，如退出后台等情况，EventHandle 类型，无默认值；

4. binderror - 用户不允许使用摄像头时触发，EventHandle 类型，无默认值。

根据这 4 种属性，开发者可以在调用系统相机拍照时控制相机的摄像头为前置或后置；是否启用闪光灯；捕获摄像头异常终止事件；捕获用户拒绝授权使用摄像头请求事件。

音频

用法

在页面中插入一段音频资源。示例代码如下：

```
<!--  audio.wxml  -->

<audio  poster="{{poster}}"  name="{{name}}"  author="{{author}}"  src="{{src}}"  id="myAudio"  controls  loop></audio>
```

```
<button  type="primary"  bindtap="audioPlay">播放</button>
<button  type="primary"  bindtap="audioPause">暂停</button>
<button  type="primary"  bindtap="audio14">设置当前播放时间为  14  秒
</button>
<button  type="primary"  bindtap="audioStart">回到开头</button>
//  audio.js

Page({

  onReady:  function  (e)  {

    //  使用  wx.createAudioContext  获取  audio  上下文  context

    this.audioCtx  =  wx.createInnerAudioContext('myAudio')

  },

  data:  {

    poster:  'http://y.gtimg.cn/music/photo_new/T002R300x300M000003rs
KF44GyaSk.jpg?max_age=2592000',

    name:  '此时此刻',

    author:  '许巍',

    src:  'http://ws.stream.qqmusic.qq.com/M500001VfvsJ21xFqb.mp3?gui
d=ffffffff82def4af4b12b3cd9337d5e7&uin=346897220&vkey=6292F51E1E384E0
6DCBDC9AB7C49FD713D632D313AC4858BACB8DDD29067D3C601481D36E62053BF8DFE
AF74C0A5CCFADD6471160CAF3E6A&fromtag=46',

  },

  audioPlay:  function  ()  {

    this.audioCtx.play()

  },

  audioPause:  function  ()  {

    this.audioCtx.pause()

  },
```

```
  audio14: function () {
    this.audioCtx.seek(14)
  },
  audioStart: function () {
    this.audioCtx.seek(0)
  }
})
```

示例代码实现了将音频插入到当前页面，操作音频播放、暂停、设置当前音频播放时长以及控制音频播放完完毕后回到音频开头。

不难看出，仅使用一行代码即可实现在页面中插入音频播放器：

```
<audio poster="{{poster}}" name="{{name}}" author="{{author}}" src="{{src}}" id="myAudio" controls loop></audio>
```

属性

本小节介绍 audio 组件的 12 个属性。

1. id - audio 组件的唯一标识， String 类型，无默认值；

2. src - 音频资源地址，String 类型，无默认值；

3. loop - 是否循环播放，Bollean 类型，默认值为 false ；

4. controls - 是否显示默认控件，Boolean 类型，默认值为 false ；

5. poster - 默认控件上的音频封面的图片资源地址，如果 controls 属性值为 false 则设置 poster 无效，无默认值；

6. name - 默认控件上的音频名字，如果 controls 属性值为 false 则设置 name 无效，无默认值；

7. author - 默认控件上的作者名字，如果 controls 属性值为 false 则设置 author 无效，无默认值；

8. binderror - 当发生错误时触发 error 事件，detail = {errMsg: MediaError.code}

9. bindplay - 当开始 / 继续播放时触发 play 事件；

10. bindpause - 当暂停播放时触发 pause 事件；

11. bindtimeupdate - 当播放进度改变时触发 timeupdate 事件，detail = {currentTime, duration}

12. bindended - 当播放到末尾时触发 ended 事件。

音频组件允许开发设置与音频资源相关的属性，允许捕获音频播放过程中的事件：播放、暂停、播放进度改变、播放结束时和播放过程中出错的事件。

播放过程出错时，开发者可以通过 MediaError.code 来查看错误码，对照一下错误码表即可知道错误原因：

```
MediaError.code
```

返回错误码	描述
1	获取资源被用户禁止
2	网络错误
3	解码错误
4	不合适资源

视频

用法

在页面中插入一段视频片段的示例代码如下：

```
<view class="section tc">
  <video src="{{src}}" controls ></video>
  <view>
    <button bindtap="bindButtonTap">获取视频</button>
  </view>
</view>

<view class="section tc">
  <video id="myVideo" src="http://wxsnsdy.tc.qq.com/105/20210/snsdyvideodownload?filekey=30280201010421301f0201690402534804102ca905ce620b1241b726bc41dcff44e00204012882540400&bizid=1023&hy=SH&fileparam=302c020101042530230204136ffd93020457e3c4ff02024ef202031e8d7f02030f42400204045a320a0201000400" danmu-list="{{danmuList}}" enable-danmu danmu
```

```
-btn  controls></video>
  <view>
    <button  bindtap="bindButtonTap">获取视频</button>
    <input  bindblur="bindInputBlur"/>
    <button  bindtap="bindSendDanmu">发送弹幕</button>
  </view>
</view>
function  getRandomColor  ()  {
  let  rgb  =  []
  for  (let  i  =  0  ;  i  <  3;  ++i){
    let  color  =  Math.floor(Math.random()  *  256).toString(16)
    color  =  color.length  ==  1  ?  '0'  +  color  :  color
    rgb.push(color)
  }
  return  '#'  +  rgb.join('')
}

Page({
  onReady:  function  (res)  {
    this.videoContext  =  wx.createVideoContext('myVideo')
  },

  inputValue:  '',
    data:  {
        src:  '',
    danmuList:  [
      {
        text:  '第  1s  出现的弹幕',
        color:  '#ff0000',
        time:  1
      },
      {
        text:  '第  3s  出现的弹幕',
        color:  '#ff00ff',
        time:  3
    }]
    },

  bindInputBlur:  function(e)  {
    this.inputValue  =  e.detail.value
  },

  bindButtonTap:  function()  {
    var  that  =  this
```

```
    wx.chooseVideo({
      sourceType: ['album', 'camera'],
      maxDuration: 60,
      camera: ['front','back'],
      success: function(res) {
        that.setData({
          src: res.tempFilePath
        })
      }
    })
  },

  bindSendDanmu: function () {
    this.videoContext.sendDanmu({
      text: this.inputValue,
      color: getRandomColor()
    })
  }
})
```