小程序 - 颜值大师

动态设置 camera 组件的高度

1. 渲染 camera 组件

```
<camera style="height: {{wh}}px; width: 100%;" flash="off"></camera>
```

2. 在 data 中定义 `wh`

```
data: {
    // 窗口可用的高度
    wh: 0
}
```

3. 动态获取页面可用高度

```
/**
* 生命周期函数--监听页面加载
*/
onLoad: function(options) {
    const sysInfo = wx.getSystemInfoSync()
    this.setData({
        wh: sysInfo.windowHeight
    })
}
```

隐藏 navigation 导航条

在 app.json 的 window 节点中，新增如下配置：

```
{
  "pages": [
    "pages/home/home"
  ],
  "window": {
    // ... 省略其他配置
    "navigationStyle": "custom"
  },
  "sitemapLocation": "sitemap.json"
}
```

在 camera 组件之上渲染操作按钮

1. 定义如下的页面结构：

```
<camera style="height: {{wh}}px; width: 100%;" flash="off">
  <cover-view class='btn-box'>
    <!-- 切换摄像头 -->
    <cover-image src='/images/icon/reverse.png'></cover-image>
    <!-- 拍照 -->
    <cover-image src='/images/icon/camera.png'></cover-image>
    <!-- 从相册选取照片 -->
    <cover-image src='/images/icon/album.png'></cover-image>
  </cover-view>
</camera>
```

2. 美化样式：

```
.btn-box {
  display: flex;
  justify-content: space-around;
  position: absolute;
  bottom: 50px;
  width: 100%;
}

.btn-box cover-image {
  width: 50px;
  height: 50px;
  opacity: 0.7;
}
```

动态切换摄像头朝向

1. 在 data 中定义数据：

```
data: {
    // 摄像头的朝向    front    back
    position: 'front'
}
```

2. 为切换摄像头按钮绑定点击事件处理函数：

```
<!-- 切换摄像头 -->
<cover-image src='/images/icon/reverse.png' bindtap='reverseCamera'></cover-image>
```

3. 实现reverseCamera函数的功能：

```
// 点击按钮，切换摄像头
reverseCamera() {
    const newPosition = this.data.position === 'front' ? 'back' : 'front'
    this.setData({
      position: newPosition
    })
}
```

4. 为 camera 组件动态绑定 `device-position`

```
<camera style="height: {{wh}}px; width: 100%;" flash="off" device-
position='{{position}}'></camera>
```

实现拍照功能

1. 在 data 中定义数据：

```
data: {
    // 照片的路径
    src: ''
}
```

2. 为拍照按钮绑定点击事件处理函数：

```
<!-- 拍照 -->
<cover-image src='/images/icon/camera.png' bindtap='takePhoto'></cover-image>
```

3. 实现 takePhoto 函数的功能：

```
// 拍照
takePhoto() {
    // 创建相机的实例对象
    const ctx = wx.createCameraContext()
    // ctx.takePhoto 实现拍照
    ctx.takePhoto({
      quality: 'high',
      success: (res) => {
        // console.log(res.tempImagePath)
        this.setData({
          src: res.tempImagePath,
          isShowPic: true
        }, () => {
          this.getFaceInfo()
        })
      },
      fail: () => {
        console.log('拍照失败！')
        this.setData({
          src: ''
        })
      }
    })
}
```

从相册选取照片

1. 为按钮绑定事件处理函数：

```
<!-- 从相册选取照片 -->
<cover-image src='/images/icon/album.png' bindtap='choosePhoto'></cover-image>
```

2. 实现 choosePhoto 函数：

```
// 从相册选取照片
choosePhoto() {
  wx.chooseImage({
    count: 1,
    sizeType: ['original'],
    sourceType: ['album'],
    success: (res) => {
      // console.log(res)
      if (res.tempFilePaths.length > 0) {
        this.setData({
          src: res.tempFilePaths[0],
          isShowPic: true
        }, () => {
          this.getFaceInfo()
        })
      }
    },
    fail: () => {
      console.log('选择照片失败！')
      this.setData({
        src: ''
      })
    }
  })
}
```

将选择的照片渲染到屏幕上

1. 定义 UI 结构：

```
<view wx:else>
  <image src='{{src}}' style='width: 100%; height: {{wh}}px; display: block;'
mode='aspectFill'></image>
</view>
```

重选照片

1. 定义 UI 结构：

```
<button type='warn' class='reChoose' bindtap='reChoose'>重选照片</button>
```

2. 实现 reChoose 函数：

```
// 重新选择照片
  reChoose() {
    this.setData({
      isShowPic: false,
      src: ''
    })
  }
```

申请百度AI开放平台账号

1. 申请百度账号
2. 登录开放平台 `http://ai.baidu.com/`
3. 创建人脸识别的应用
4. 填写应用信息
5. 得到应用的 `API Key` 和 `Secret Key`

实现API鉴权

```
// this.globalData.access_token = 'aaa'
    wx.request({
      method: 'POST',
      url: 'https://aip.baidubce.com/oauth/2.0/token?
grant_type=client_credentials&client_id=自己的ID&client_secret=自己的KEY',
      success: (res) => {
        this.globalData.access_token = res.data.access_token
      },
      fail: () => {
        wx.showToast({
          title: '鉴权失败！',
        })
      }
    })
```

将图片转码为 base64 字符串

```
const fileManager = wx.getFileSystemManager()
const fileStr = fileManager.readFileSync(this.data.src, 'base64')
```

发起请求检测颜值数据

```
wx.request({
      method: 'POST',
      url: 'https://aip.baidubce.com/rest/2.0/face/v3/detect?access_token=' + token,
      header: {
        'Content-Type': 'application/json'
      },
      data: {
        image_type: 'BASE64',
        image: fileStr,
        // 年龄,颜值分数,表情,性别,是否戴眼镜,情绪
        face_field: 'age,beauty,expression,gender,glasses,emotion'
      },
      success: (res) => {
        console.log(res)
        if (res.data.result.face_num <= 0) {
          return wx.showToast({
            title: '未检测到人脸！',
          })
        }

        this.setData({
          faceInfo: res.data.result.face_list[0],
          isShowBox: true
        })
      },
      fail: () => {
        wx.showToast({
          title: '颜值检测失败！',
        })
      },
      complete: () => {
        wx.hideLoading()
      }
    })
```

把英文信息映射为中文信息

1. 定义映射关系：

```
data: {
    // 映射关系
    map: {
      gender: {
          male: '男', female: '女'
      },
      expression: {
        none: '不笑', smile: '微笑', laugh: '大笑'
      },
      glasses: {
        none: '无眼镜',common: '普通眼镜',sun: '墨镜'
      },
      emotion: {
        angry: '愤怒', disgust: '厌恶', fear: '恐惧', happy: '高兴',
        sad: '伤心', surprise: '惊讶', neutral: '无情绪'
      }
    }
}
```

2. 修改UI结构：

```
<view class='faceinfo_box' wx:if="{{isShowBox}}">
    <view class='face_row'>
      <text>年龄：{{faceInfo.age}}岁</text>
      <text>性别：{{map.gender[faceInfo.gender.type]}}</text>
    </view>
    <view class='face_row'>
      <text>颜值：{{faceInfo.beauty}}分</text>
      <text>表情：{{map.expression[faceInfo.expression.type]}}</text>
    </view>
    <view class='face_row'>
      <text>眼镜：{{map.glasses[faceInfo.glasses.type]}}</text>
      <text>情绪：{{map.emotion[faceInfo.emotion.type]}}</text>
    </view>
</view>
```

优化项目效果

1. 通过 `isShowBox` 控制 `颜值信息` 的显示与隐藏，防止闪烁问题
2. 添加数据加载期间的 loading 效果