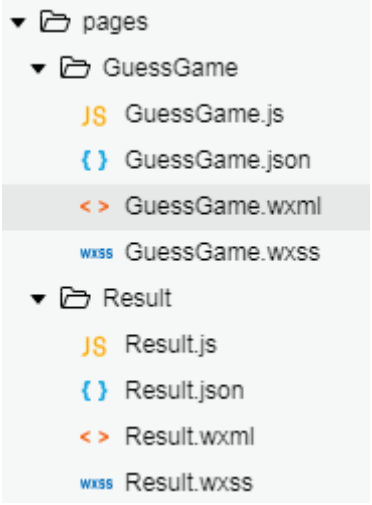
猜数字游戏制作文档

郑子豪　信息 171　学号：2017210218028　写于：2019.9.20

制作步骤展示：

1、表层 wxml 与 wxss 文件的编写，即界面制作：

首先，我认为这次的猜数字游戏可以制作成两个界面，将第一个界面称作猜测界面，第二个称作结果界面。所以在 pages 文件夹中创建了两个新的文件夹，分别命名为 GuessGae 与 Result。



此后我们先对上图中被选定的 GuessGame.wxml，以及 wxss 文件的内容做编写：根据猜数字游戏的内容，我把 0 到 9 十张图片按照一排三个的顺序平铺在界面上，并且将零放置在第四排的正中位置，并加上了该界面的标题与对游戏使用者的提示。

另外，我们还要实现在点击了对应图片之后的事件，所以预先将事件名称 click 写入

代码中。

其代码如下：

```
<!--pages/Guess/GuessGame.wxml-->
<view>
<view>
<text class="text1">Try to Guess the Number!</text>
</view>
<view>
<text class="text2">Click one number!</text>
</view>
<view>
<text class="text2">

</text>
</view>
<view>
<image id="1" bindtap="click"  src="/images/{{color}}/shuzi1.png" class="number">
</image>
<image id="2" bindtap="click"  src="/images/{{color}}/shuzi2.png" class="number">
</image>
<image id="3" bindtap="click"  src="/images/{{color}}/shuzi3.png" class="number">
</image>
<image id="4" bindtap="click"  src="/images/{{color}}/shuzi4.png" class="number">
</image>
<image id="5" bindtap="click"  src="/images/{{color}}/shuzi5.png" class="number">
</image>
<image id="6" bindtap="click"  src="/images/{{color}}/shuzi6.png" class="number">
</image>
<image id="7" bindtap="click"  src="/images/{{color}}/shuzi7.png" class="number">
</image>
<image id="8" bindtap="click"  src="/images/{{color}}/shuzi8.png" class="number">
</image>
<image id="9" bindtap="click"  src="/images/{{color}}/shuzi9.png" class="number">
</image>
<image  class="number"> </image>
<image id="0" bindtap="click"  src="/images/black/shuzi0.png" class="number"> </image>
<image  class="number"> </image>
</view>
</view>
```

```
/* pages/Guess/GuessGame.wxss */
.text1{
  color:rgb(34, 10, 252);
```

```
  font-size:20px;
  margin-left:30px
}
.text2{
  color:rgb(0, 0, 0);
  font-size:15px;
  margin-left:100px
}
.number{
  width: 220rpx;
  height: 220rpx;

  border-radius: 15%;
  margin:14rpx;
    }
```

2、 编写内部逻辑：

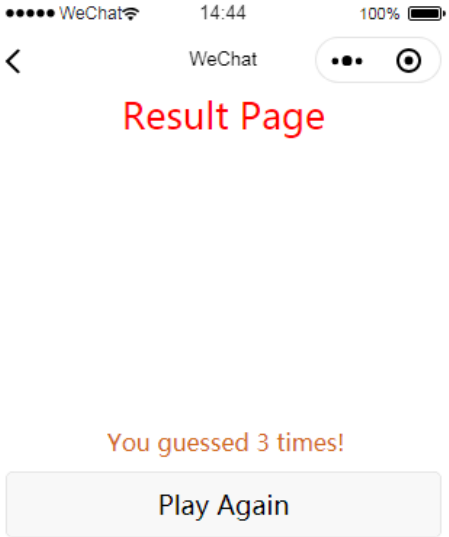
整个猜数字游戏的内部逻辑可以用如下语言表述：首先在后台生成一个随机数 N，使用者在猜测界面上选择他猜测的结果 n，与后台的随机数 N 比较是否相等，若相等则进入结果界面，若不相等则重复这个过程，且被选过的数字会变色，并记录猜测次数。

上述过程将在 js 文件中编写。

（在 3，4 完成后展示代码）

3、 新的页面：

在 Result 文件中接收来自 GuessGame 文件的选择次数数据 num，我通过查阅文献，使用了 `wx.setStorageSync` 语句来进行缓存，在结果界面展示出如下的信息。



由于考虑到游戏需要反复进行，我设计了一个按钮，使整个程序能够在按下按钮后重新开始运行。最后经过第四阶段，解决了各个问题后，成功实现了这一点。

Result 界面 wxml 与 wxss 文件代码：

```
    <!--pages/Result/Result.wxml-->
<text class="title">Result Page\n\n\n\n\n\n\n</text>
<text class="result"> You guessed {{num}} times!\n</text>
    <button bindtap="click" class="buttons"> Play Again </button>
    /* pages/Result/Result.wxss */
.title{
  color:rgb(255, 0, 0);
  font-size:25px;
  margin-left:90px
}
.result{
  color:chocolate;
  margin-top: 100px;
  margin-left:80px
}
.buttons{
  margin: 10px;
    }
```

4、 过程中出现的各类问题：

（1） 有关数据传输：

最开始我试图直接在新的界面的 data 中利用 n:wx.getStorageSync('n')这样的形式直接对 Result 界面中的数据进行操作，反复尝试发现都不行，有时甚至出现了延迟读数，数字仅对第一次有效。最后发现需要使用 setdata 来修改 data 里面的预设值，这样就能使数据完美的对应上了。

（2） 有关切换界面：

如 3 中的描述所说我希望设置一个按钮可以重新进行游戏，所以查阅文献找到了函数 wx.reLaunch 通过这个函数可以关闭所有页面重新加载特定的某个页面，符合我对程序的要求。

通过上述四个步骤我完成了猜数字游戏程序。

Js 部分代码如下：

```
    // pages/Guess/GuessGame.js
Page({

  /**
   * 页面的初始数据
   */
  data: {
```

```
    color: ['black', 'black', 'black', 'black', 'black', 'black', 'black', 'black',
'black', 'black'],
    colors: ['black', 'red'],
    num:0,
  },
  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
    var newtarget= Math.floor(Math.random() * 10)
    this.setData({
      target: newtarget
    })
  },
  click: function (e) {
    var temp = this.data.color;
    var tempnum = this.data.num;
    tempnum = tempnum + 1;
    temp[e.currentTarget.id] = this.data.colors[1];
    this.setData({
      color: temp,
      num: tempnum
    })
    console.log(e.currentTarget.num)
    if (e.currentTarget.id > this.data.target) {
      wx.showToast({
        title: 'too large',
        icon: 'none'
      })
    } else {
      if (e.currentTarget.id == this.data.target) {
        wx.setStorageSync('key', this.data.num)
        wx.showToast({
          title: 'You get it!!',
          success: function () {
            wx.navigateTo({
              url: '/pages/Result/Result'
            })
          }
        })
      }
      else {
        wx.showToast({
          title: 'too little',
```

```
        icon: 'none'
      })
    }
  }
},
/**
 * 生命周期函数--监听页面初次渲染完成
 */
onReady: function () {

},

/**
 * 生命周期函数--监听页面显示
 */
onShow: function () {
},

/**
 * 生命周期函数--监听页面隐藏
 */
onHide: function () {

},

/**
 * 生命周期函数--监听页面卸载
 */
onUnload: function () {

},

/**
 * 页面相关事件处理函数--监听用户下拉动作
 */
onPullDownRefresh: function () {

},

/**
 * 页面上拉触底事件的处理函数
 */
onReachBottom: function () {
```

```
  },

  /**
   * 用户点击右上角分享
   */
  onShareAppMessage: function () {

  }
})
```

Result 界面的 js 文件：

```
    // pages/Result/Result.js
Page({

  /**
   * 页面的初始数据
   */
  data: {
    num: 0
  },
  click: function (e) {
    wx.reLaunch({ url:'/pages/GuessGame/GuessGame'})
  },
  /**
   * 生命周期函数--监听页面加载
   */
  onLoad: function (options) {
    var n = wx.getStorageSync('key')
    this.setData({
      num:n
    })
  },

  /**
   * 生命周期函数--监听页面初次渲染完成
   */
  onReady: function () {

  },

  /**
   * 生命周期函数--监听页面显示
   */
  onShow: function () {
```

```
},

/**
 * 生命周期函数--监听页面隐藏
 */
onHide: function () {

},

/**
 * 生命周期函数--监听页面卸载
 */
onUnload: function () {

},

/**
 * 页面相关事件处理函数--监听用户下拉动作
 */
onPullDownRefresh: function () {

},

/**
 * 页面上拉触底事件的处理函数
 */
onReachBottom: function () {

},

/**
 * 用户点击右上角分享
 */
onShareAppMessage: function () {

}

  })
```