

移动项目开发实验报告一

一、实验目的与要求

1. 运用Wechat Devtools制作一个“猜数字”的简单微信小程序
2. 用户通过选择数字的形式进行猜数字
3. 最后显示结果以及猜测次数

二、实验过程

设计思路

1. 通过单机Go字符，开启程序，随机产生一个0~9之间的整数作为正确答案，存储在Storage中；并激活数字符号（将数字变色）
2. 通过单机数字符号操作激发一个触发事件，得到用户选择的数字（将数字变色），并将单机次数记录在Storage中
3. 比较用户选择的数字与第一步中产生的随机数字
 - 如果相等，则跳转页面，输出正确答案、总共猜测的次数以及一些提示语
 - 如果不相等则在当前页面跳出相应比较结果提示框；继续选择数字，此时对于已经猜过的数字不能再选择，若选择了，则返回提示语并不将此次操作记录总的猜测次数中。

知识点罗列

1. 随机数的产生

```
Math.random()//产生[0,1)之间的一个浮点型随机数
```

2. 取整

```
Math.floor()//向下取整  
Math.ceil()//向上取整
```

3. 跳出提示窗口的设置

```
wx.showToast({
  title: 'aaa',
  icon: 'none',
  duration:100
})
```

4. 触发事件的设置与绑定

- 首先在组件中绑定事件处理函数

```
bindtap="tapName"
```

- 在相应的page中定义该事件处理函数

```
Page({
  tapName: function(event) {
    //具体处理
  }
})
```

5. 修改data中的属性值 运用setData函数

```
this.setData({
  //a1:...,
  //a2:...
})
```

若要对data中的属性值进行数值计算，则

- 首先需要将该属性值赋值给一个变量
- 然后对该变量进行数值计算
- 最后将该变量经过数值计算后得到的结果赋值给原来的属性值
- e.g.下面对data中count属性值进行+1操作

```

Page({
  data:{
    count=0
  },
  change:function(){
    var cnt = this.data.count;
    cnt = cnt + 1;
    this.setData({
      count:cnt
    })
  }
})

```

6. 数组的变量 运用forEach函数

```

array.forEach((item,index)=>{//其中item是数组array中的每一个元素，index
是array中的每一个索引值
  //对item的操作
})

```

7. 数据缓存与获取

```

wx.setStorage({
  key: 'keyName',
  data:data1
})
//将数据data1存储在本地缓存中指定的keyName 中

```

```

wx.getStorage({
  key: 'Keyname',
  success:res=>{
    //对获取得到对数据res.data进行处理
  }
})//从本地缓存中异步获取指定 keyName的内容

```

8. 页面跳转

```
wx.navigateTo({
  url: ''
})
```

二、实验结果

1. 初始页面

- 界面介绍
 - 由0~9这十个浅灰色的数字字符以及一个Go字符组成；
 - 其中1~3，4~6，7~9各为一组，一次从上而下排列，紧接着0为水平居中布置在下面，Go字符为水平居中放在页面的底端。

以下为guess.wxml以及guess.wxss的代码实现

```
<!--pages/guess/guess.wxml-->
<view >
  <image id="1" bindtap="click" src="/images/{{c[1]}}/shuzi1.png" class="tp"
  ></image>
  <image id="2" bindtap="click" src="/images/{{c[2]}}/shuzi2.png" class="tp"
  ></image>
  <image id="3" bindtap="click" src="/images/{{c[3]}}/shuzi3.png" class="tp"
  ></image>
</view>
```

```
<view>
  <image id="4" bindtap="click" src="/images/{{c[4]}}/shuzi4.png" class="tp"
  ></image>
  <image id="5" bindtap="click" src="/images/{{c[5]}}/shuzi5.png" class="tp"
  ></image>
  <image id="6" bindtap="click" src="/images/{{c[6]}}/shuzi6.png" class="tp"
  ></image>
</view>
```

```

<view>
<image id="7" bindtap="click" src="/images/{{c[7]}}/shuzi7.png" class="tp"
></image>
<image id="8" bindtap="click" src="/images/{{c[8]}}/shuzi8.png" class="tp"
></image>
<image id="9" bindtap="click" src="/images/{{c[9]}}/shuzi9.png" class="tp"
></image>
</view>

<view class="t0" >
<image id="0" bindtap="click" src="/images/{{c[0]}}/shuzi0.png"
class="tp"></image>
</view>
<view class="t0" >
<image bindtap="click0" src="/images/go.png" class="go" ></image>
</view>

```

```

/* pages/guess/guess.wxss */

```

```

.tp{
  width: 230rpx;
  height: 200rpx;
  border: 1px solid #F40;
  border-radius: 15%;
  margin: 5rpx
}
.t0{
  display: flex;
  justify-content: center
}
.go{
  width: 230rpx;
  height: 200rpx;
}

```

- 功能介绍

- 首先，通过单机Go字符作为开始触发操作，当单机Go字符，程序产生一个随机数存储在Storage中，在当前页面显示提示语"ready go!"，同时所有数字字符变为红色。若在没有触发Go字符之前，单机数字字符，会出现提示语"You have not begin"，并不会发生其他任何变化。
- 接着，通过单机数字字符进行猜数字操作，若选择的数字大于随机产生的数字，则在当前页面显示提示语"too big"，并将该数字转成灰色,同时记录次数；若选择的数字小于随机产生的数字，则在当前页面显示提示语"too small"，并将该数字转成灰色，同时记录次数；若选择的数字等于随机产生的数字，则跳转页面，同时记录次数。
- 其中当选择已经被选择过的数字字符，即已经为灰色的数字字符，此时在当前页面显示提示语"You have chose it",但不记录该次次数。

- 功能实现 以下为在guess.js中的代码

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

  /**
   * Page initial data
   */
  data: {
    clicked:[0,0,0],
    myColor:["red","gray"],
    c: ["light gray", "light gray", "light gray", "light gray", "light gray",
    "light gray", "light gray", "light gray", "light gray", "light gray"],
    targetNumber: 'none', //当没有触发开始时，还没有产生随机数
    count:0
  },

  //设置一个点击go触发的一个开始动作，且在此之前，点击数字符号不会发生任何事件
  click0:function(event){
    //当触发开始，则产生一个随机数
    var target = Math.floor(Math.random() * 10)
    this.setData({
      targetNumber:target
    })
    var array = this.data.c;
    //当触发开始，则将所有数字符号变红激活，且提示可以开始
    array.forEach((item,index)=>{//——获取c数组中的每一个元素，并——对它们进行修改
      wx.showToast({
```

```

        title: 'Ready go!',
        icon: 'none',
        duration: 500
    });
    var sItem="c["+index+"]";
    this.setData({
        [sItem]: "red"
    })

    })

},
//设置一个点击数字符号触发的动作
click: function(event){
    var index = event.currentTarget.id;
    var array = this.data.c;
    //
    if (array[index] == "light gray") {
        wx.showToast({
            title: 'You have not begin!',
            icon: 'none'
        })
        return
    }
    /**首先判断该数字是否已经猜过，即当它的颜色为gray时候，此时显示提示语，
    然后返回，不进行任何操作直接返回，注意不会被记录猜测次数
    */
    if (array[index] == "gray") {
        wx.showToast({
            title: 'You have chose it!',
            icon: 'none'
        })
        return
    }
    var cnt = this.data.count;
    cnt = cnt + 1;
    //对选中的数字字符进行修改，同时记录次数
    array[index]="gray";
    this.setData({
        c: array,
        count: cnt
    })
    //进行比较
    if (index>this.data.targetNumber){
        wx.showToast({
            title: 'too big',
            icon: 'none'

```

```

    })
  }else{
    if (index==this.data.targetNumber){
      wx.setStorage({
        key: 'counts',
        data: cnt
      })
      wx.setStorage({
        key: 'target',
        data: this.data.targetNumber
      })
      wx.navigateTo({
        url: '/pages/result/result'
      })
    }else{
      wx.showToast({
        title: 'too small',
        icon: 'none'
      })
    }
  }
}
})

```

2. 跳转界面

- 界面介绍 显示正确答案以及总共的猜测次数，并配有胜利字符以及图示。

以下为result.wxml以及result.wxss中的代码


```

<!--pages/result/result.wxml-->
<view class="myCenter">
  <view>
    <text class="test1">正确答案为{{target}}</text>
  </view>
  <view>
    <text class="test1">一共猜测{{total}}次</text>
  </view>
</view>
<image src="/images/success.png" class="myPhoto"></image>
<view class="myCenter">
  <text class="test2">胜利</text>
</view>

```

```

/* pages/result/result.wxss */
.test1{
  color: black;
  font-size: 68rpx;
}
.test2{
  color: red;
  font-size: 68rpx;
}
.myCenter{
  text-align: center;
  line-height: 200rpx
}
.myPhoto{
  margin-left: 300;
  width: 300;
  height: 400
}

```

- 实现 其中正确答案以及总共的猜测次数的值通过wx.getStorage从本地缓存中异步获取得到

以下为result.js中的代码

```

// pages/result/result.js
Page({

  /**
   * Page initial data
   */
  data: {
    total:0,//记录总共猜测的次数
    target:0//记录正确答案的值
  },

  /**
   * Lifecycle function--Called when page load
   */
  onLoad: function (options) {
    //获取Storage中记录的猜测次数
    wx.getStorage({
      key: 'counts',
      success: res => {
        this.setData({
          total:res.data
        });
        console.log(res.data)
      },
    }),
    //获取Storage中记录的正确值
    wx.getStorage({
      key: 'target',
      success: res => {
        this.setData({
          target:res.data
        });
        console.log(res.data)
      },
    })
  },

  /**
   * Lifecycle function--Called when page show
   */
  onShow: function () {
    wx.showToast({
      title: 'Congraduation'
    })
  }
})

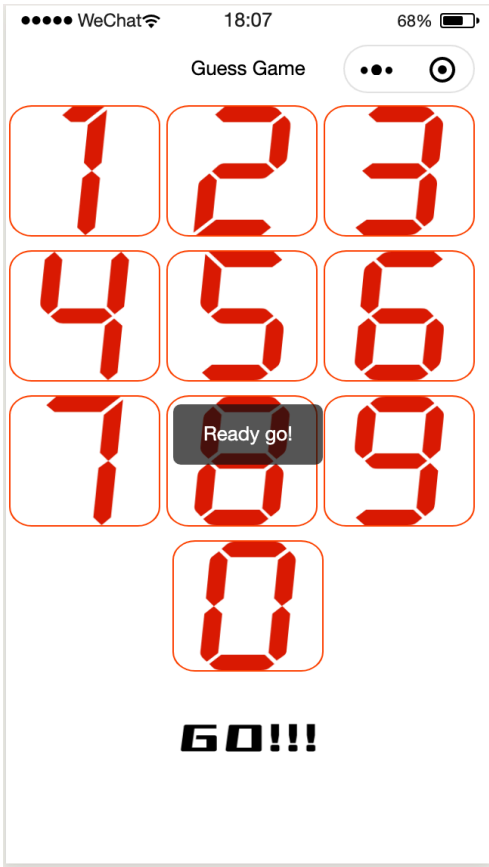
```

三、测试结果

初始界面



启动界面



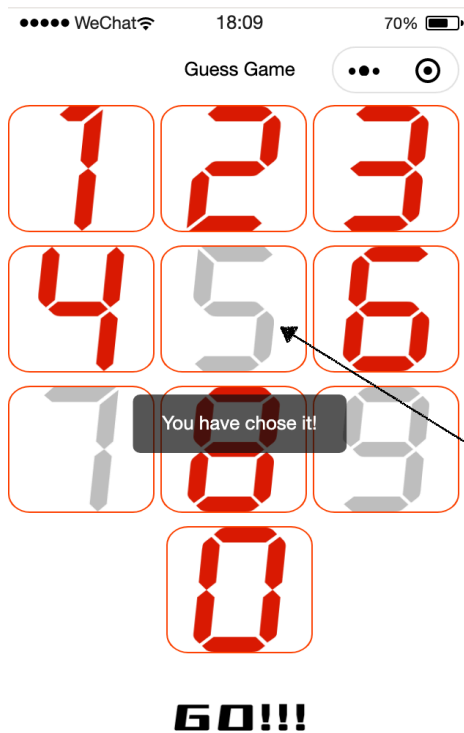
测试情况一



测试情况二

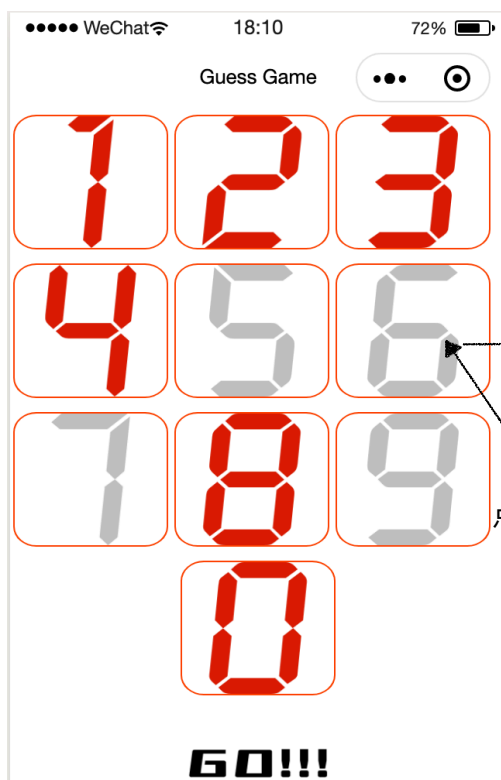


测试情况三



再次点击数字字符5

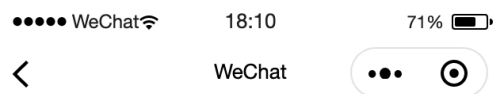
测试情况四



跳转至跳转页面

点击数字字符6

跳转页面



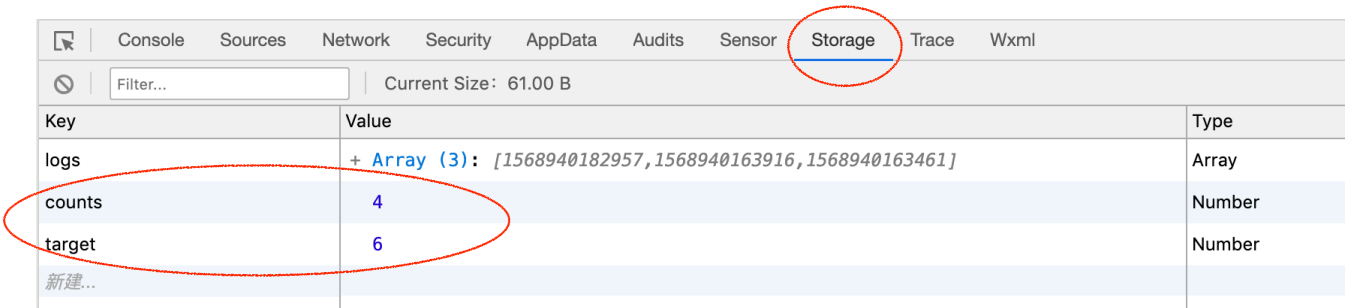
正确答案为6

一共猜测4次



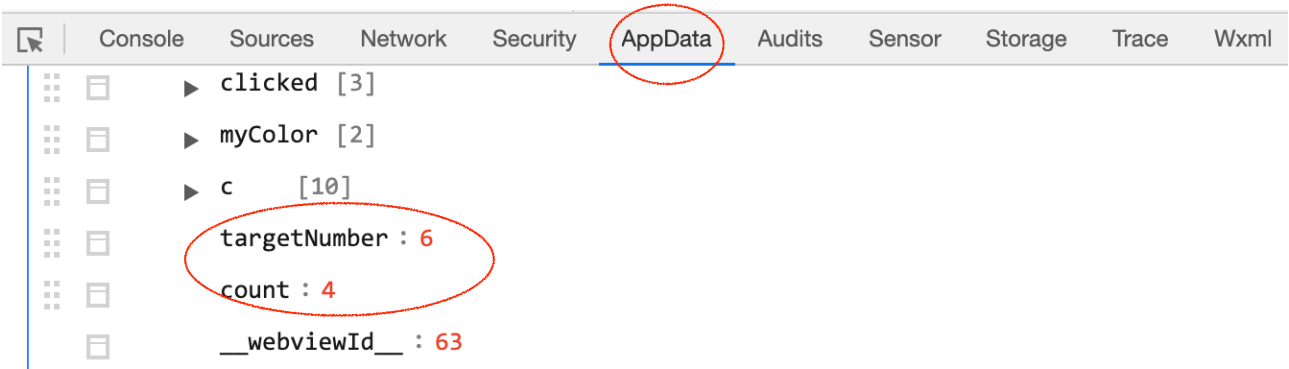
胜利

并可以通过Wechat Devtools 中的AppData、Storage中可以看出，初始产生的随机数的确为6，且最后得到的总的次数为4



The screenshot shows the 'Storage' tab in Wechat Devtools. The 'Storage' tab is circled in red. Below the tabs, there is a filter input and a 'Current Size: 61.00 B' indicator. The main table has three columns: 'Key', 'Value', and 'Type'. The 'logs' key has a value of '+ Array (3): [1568940182957, 1568940163916, 1568940163461]' and is of type 'Array'. The 'counts' key has a value of '4' and is of type 'Number'. The 'target' key has a value of '6' and is of type 'Number'. A red oval highlights the 'counts' and 'target' rows.

Key	Value	Type
logs	+ Array (3): [1568940182957, 1568940163916, 1568940163461]	Array
counts	4	Number
target	6	Number



The screenshot shows the 'AppData' tab in Wechat Devtools. The 'AppData' tab is circled in red. The table lists application data with columns for key and value. The 'targetNumber' key has a value of '6' and is circled in red. The 'count' key has a value of '4'. The 'clicked' key has a value of '[3]', 'myColor' has '[2]', 'c' has '[10]', and '__webViewId__' has '63'.

Key	Value
clicked	[3]
myColor	[2]
c	[10]
targetNumber	6
count	4
__webViewId__	63