### 实验1：个人介绍

**设计思路**：

首先 id为out的<div>用于作为整个页面的背景图，在CSS文件中将该<div> background: 暗绿色，并用绝对定位的方式将该div居中：div高度95%，宽度80%，绝对定位距页面左侧10%，页面上部5%

在out<div>内部，第一个head<div>，只定义了个人介绍标题以及下划线，第二个left<div>，用于显示个人照片左侧栏，第三个main<div>，是页面的主体部分包括基本资料和专业技能。

head<div>，CSS设置了文本居中，用于标题文字居中，以及padding=20px，用于页面整体看起来分配均匀，最后是设置字体大小

left<div>，通过绝对定位的方式置于页面的左侧，并设置了合适的颜色以及大小。

main<div>，也是用绝对定位放置在合适的位置

**总结**：

熟悉了html中<div>、<image>、<hn>、<hr>、<table>等标签的使用

熟悉了CSS中元素选择器、类选择器、ID选择器的使用

熟悉了CSS中内连样式表、外部样式表的使用

熟悉了CSS绝对定位的使用，已及多种表示颜色的方式

### 实验2：电子日历

#### 设计思路：

在页面打开的时候调用calendarAdd(year,month,data,day)函数，还原当前月份的日历，**还原思路**：首先得到年份月份->显示年份月份；然后是最复杂的日历的主体部分，我的思路是获取该月的第一天是星期几（weekPoint），然后从获取到的weekPoint开始增加还原日历(具体实现见函数介绍部分)，而到哪天停止这个问题，通过获取该年该月的最大日期值，在最大日期处停止就能还原日历。

当在点击下一月或者上一月的按钮时，通过调用monthChange(change)函数来页面打开时的calendarAdd(year,month,data,day)更改日历，此时传入的参数为更改后的值。而在页面打开时，我们调用calendarAdd函数传入日期值是通过JavaScript内置的new Date()获取，此时遇到一个问题，在此时应该传入的值如何确定。**确定思路**：在两个button按钮设置value值用于存放点击按钮后传入的星期值。

关于当前天的字体红色显示：在css中将id为now的字体设置为红色，背景设置为蓝色着重显示，当动态生成html时，判断加入的日期如果等于当前天，当前月，当前年，那么把id设置为now。同时实现了当前天显示在了日历月的前一月/后一月的信息值中也会红色字体显示

#### 页面框架

从*HTML*部分开始：

最外部的out<div>，out<div>内部包括：head<div>以及main<div>。

head<div>内有两个id分别为year和month的<b>标签以及两个<button>按钮标签。

main<div>内有两个table标签，第一个table用于显示星期一，二，三……日，第二个table标签id为addPoint，是日历的主体部分，用于动态的添加日期于日历

CSS部分：主要是#now设置字体颜色，背景颜色用于当前天的显示，以及#pre将前月的日期值显示灰色，#next将下一月的日期值显示为黄色

#### 主要函数：

1. updateYear(year)函数：

作用：用于更新日历的年份

传入参数：year：日历的年份

具体实现：通过更新id为year的<b>标签的innerHTML内容为year+“年”

1. updateMonth(month)函数：

作用：用于更新日历的月份。

传入参数：month：日历的月份

具体实现：通过更新id为month的<b>标签的innerHTML内容为month+“月”

1. getMaxDay(year,month)函数：

函数设计原因：还原该月的日历，就一定要知道该月一共有多少天。

作用：用于返回该年该月的天数。

参数：year年份 month月份

具体实现：首先创建一个数组值为[1,3,5,7,8,10,12]，如果month的值为其中一个，那么可以确定该月份为31天并返回；如果month是2，此时就需要判断year是否是闰年，如果是返回29不是返回28；最后都不是返回30。

1. getMonthFristDay(week, nowdate)函数：

函数设计原因：我实现日历动态还原的思路为获取该月的第一天是星期几，然后循环向后添加日期。

函数作用：返回该月的第一天是星期几。

传入参数：nowdate：一个确定星期几的日期，week：nowdate日期的星期数。0代表星期日，1代表星期一，2代表星期二……

具体实现：见代码

function getMonthFristDay(week,nowdate) {

var tem = nowdate % 7;

/\* 这里week\*1的原因是week他默认当做string识别，结果成了07会导致bug\*/

var firstDay = ((week\*1) + (8 - tem)) % 7;

return firstDay;

}

返回的firstDay就是第一天的星期数 0 – 星期日 1 – 星期一……

以上四个函数都是被动调用的，下面是两个最主要的函数：

1. calendarAdd(year,month,day,week)函数

传入参数year年份 month月份 day日期 week星期几

作用：还原日历

具体实现：

1.调用了函数updateYear和updateMonth还原了年份，月份。

2.调用函数getMonthFristDay获取该月第一天是周几记录在wekPoint，如果wekPoint为0那么改为7方便操作。

3.调用getMaxDay函数获取最大日期值记录于变量maxDay。在所有日历中不仅包括本月的值，还包括前月余下的值，所以还得获得一个前月的最大日期值(preMonthMaxDay)。

4.通过document获取增加点$table，然后把信息组合添加在$table.innerHTML中，创建一个str临时变量用于组合信息，以及一个日期指针dayPoint=1，临时星期指针temwek=wekPoint用于处理前月的信息。

5.在设计思路中提到，在点击上一月或下一月按钮时，通过按钮的value属性值获取周几，现在存放value=wekPoint – 1，就代表前月最大日期值就是周(wekPoint-1)。细节：wekPoint前面设置为1-7，所以说wekPoint-1为0-6对应周日到周6

6.从此开始往$table中添加信息

6.1日历从本月第一天(周wekPoint)开始，它之前的信息由前一月的信息构成：日历会显示前月的日期值的条件是当前月的第一天不是周1。如果是周二那么显示一条前月的信息，周3显示2条……

6.2当前月信息的处理：前月信息处理完后，正式到达wekPoint所指向的周数，dayPoint从1开始计算一直到maxDay，并且每当wekPoint=8时说明该换行输出日期

6.3输出完当前月信息，日历余下部分由下一月的值补充构成且从下一月第一天开始一直到周日结束，如果当月刚好是周日那么没有下一月的信息。

7.日历每行(wekPoint一轮)循环后加到$table的innerHTML中

6,7部分具体代码：

6. monthChange(change)函数

while (dayPoint <= maxDay) {

while(temwek-- > 1) {

/\*处理前月的值\*/

if ((preMonthMaxDay+1-temwek) == new Date().getDate() &&

(month-1 == new Date().getMonth()+1 || month + 12 == new Date().getMonth()+1) ) {

str += "<td id = 'now'>" + (preMonthMaxDay+1-temwek) + "</td>";

}else{

str += "<td id= 'pre'>" + (preMonthMaxDay+1-temwek)+ "</td>";

}

}

while (wekPoint < 8 && dayPoint <= maxDay){

if (dayPoint == new Date().getDate() && month+1 == new Date().getMonth()+1) {

str += "<td id = 'now'>" + dayPoint + "</td>";

}else{

str += "<td>" + dayPoint + "</td>";

}

dayPoint++;

wekPoint++;

}

//这里结束后只是前月以及当月的被输入到日历

//重点：需要把wekPoint当前做标记，用于下个月

var flag = 1;

for (var i = 1,j = wekPoint; j < 8; i++,j++) {

if (i == 1) {

document.getElementById("btnadd").value = (wekPoint) % 7; //标记

flag = 0;

}

str += "<td id = 'next'>" + i + "</td>";

}

if (flag == 1) {

document.getElementById("btnadd").value = 1;

}

wekPoint = 1;

str += "</tr>";

$table.innerHTML += str;

str = "<tr>";

}

作用：用于点击按钮后的处理

参数：change月份的改变如果前一月传入-1 下一月传入1

具体实现：该函数最终会调用calendarAdd函数，所以该函数最重要的是获取newYear，newMonth，newWeek，newDate变量的信息就可以完成日历下一月上一月的功能，首先newYear是多少受月份的改变的影响(月份减一后如果为12月那么年份也减1)，所以先确定newMonth，newMonth=当前日历月份的信息所以通过document来读取id=month的<b>中的值，再使用match(/\d+/g)读取数字就是当前日历月份，加上change得到newMonth，通过newMonth确定newYear的值，在calendarAdd函数，中我们提前存放了下一月第一天是周几，也存放了上一月最后一天是周几，所以根据这些信息调用calendarAdd函数，从而实现日历上一月下一月功能。

具体代码：

function monthChange(change) {

var nowMonth = document.getElementById("month").innerHTML.match(/\d+/g); //获取目前日历的月份

var newMonth = (1\*nowMonth)+change; //如果是-1就 前一月 如果是+1就 后一月

var newYear = document.getElementById("year").innerHTML.match(/\d+/g);

var $table = document.getElementById("addPoint");

$table.innerHTML = "";

if (change == 1) {

if (newMonth == 13 ) {

newMonth = 1;

newYear++;

}

var week = document.getElementById("btnadd").value;

//var week = change.target.value;

//alert(week)

calendarAdd(newYear,newMonth,1,week);

}else{

if (newMonth == 0) {

newMonth = 12;

newYear--;

}

var week = document.getElementById("btnsub").value

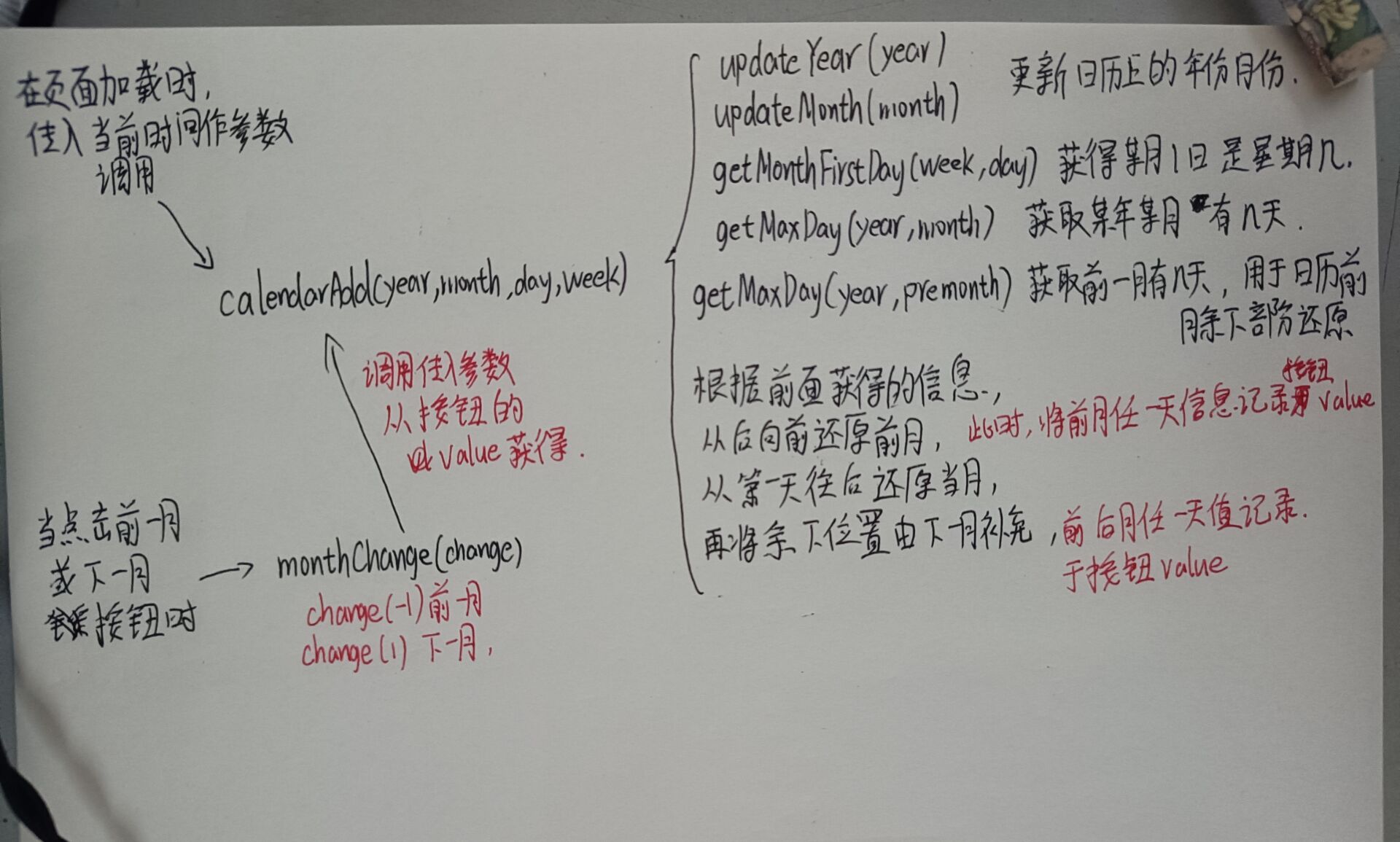
//alert(week);

calendarAdd(newYear,newMonth,getMaxDay(newYear,newMonth),week)

}

}

#### 页面显示流程图示



#### 总结：

复习了HTML以及CSS的基础知识。

熟悉了JavaScript函数的实现以及使用

熟悉了按钮绑定事件(onclick事件)

熟悉了JavaScript动态生成HTML文本

### 实验3：扑克拖放

#### 设计思路

1. 首先扑克图片在A框中是背面显示，在B框是正面显示，如何将A框中图片拖到B框后正面显示，从B框拖到A框后又变为反面显示。
   1. 解决思路：在A框时，图片的src属性指向一个背面的图片(src="poker/bei.webp")
   2. 当拖放到B框时，在ondrop事件中（$img.src = `poker/heart${num}.jpg`;）将拖放图片的src属性改为正面的url
   3. B拖放到A同理
2. 每个扑克图片的值有唯一性，即13张扑克一一对应A，2，3……K，不能有重复
   1. 我们知道，图片有value属性，所以在value属性存放扑克的大小(1-13)
   2. 每个图片的value在何时产生，可以在页面加载之前就给全部扑克随机产生value，也可以在每次拖放一个图片时给他一个不重复value，我采用的是第二种，在A框扑克拖放到B框时，在ondrop事件中有一段代码产生一个不重复的value，赋值给img.value
   3. 这时，如果一个图片第二次拖放到B框，那么会出现一个**错误，**它的图片又一次刷新了，需要在产生不重复随机值的代码前加上一个判断：$img.value==undefined
3. 在B框中图片满5张后，如何计算它是否满足顺子
   1. 在全局变量中维护了一个record数组（记录扑克各个值）以及一个sum（B框扑克数值总和），divBCount（B框扑克数目）
   2. 当divBcount等于5时，首先判断sum/5是否为整数，因为不是整数的话，可以确保此时一定不满足顺子。
   3. 如果是正数，那么循环record，找出最大值(max)，判断max-2是否等于sum/5。如果等于可以确实是顺子，游戏结束

#### 页面框架：

Html部分：

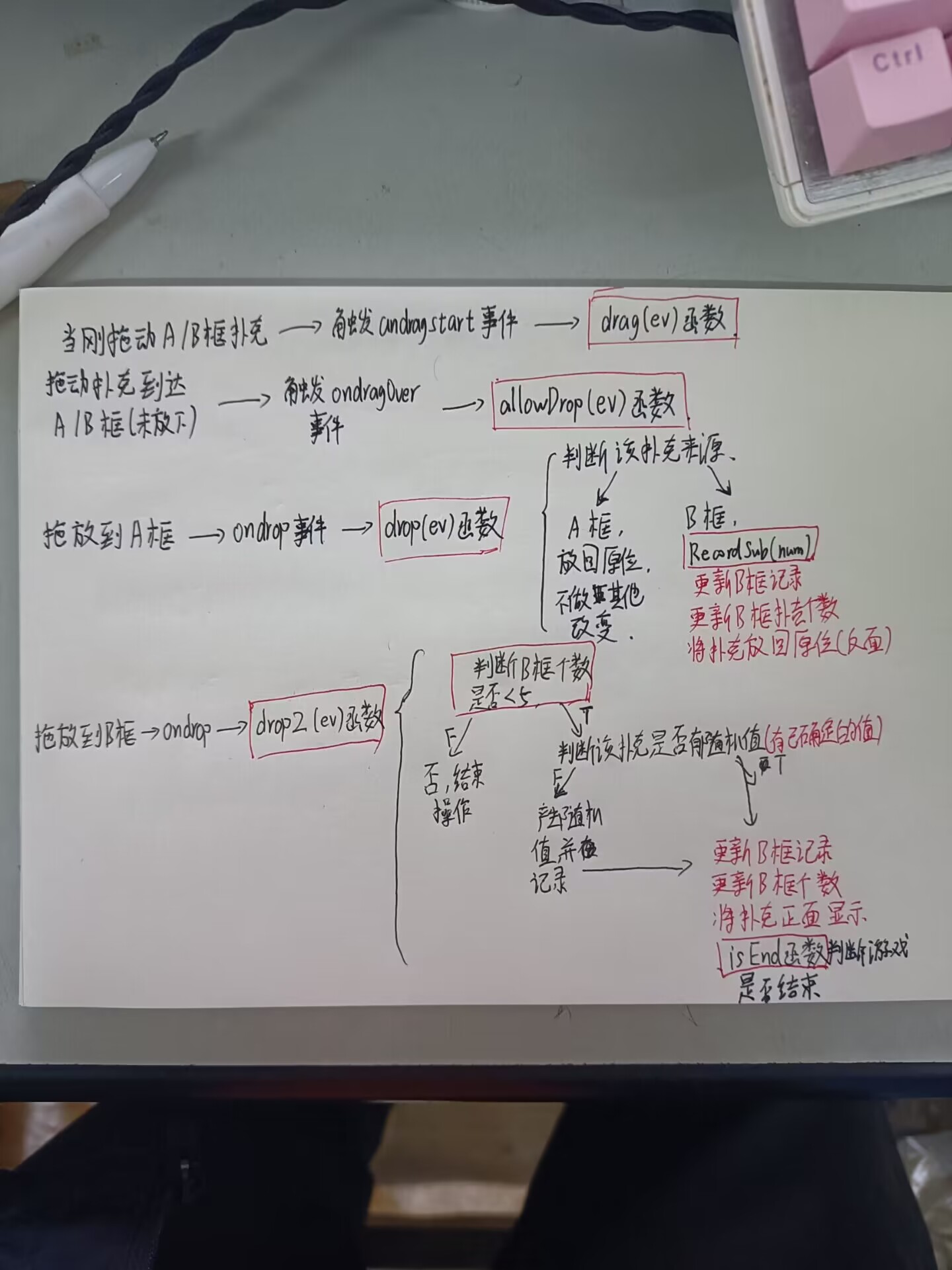
两个主要div：divA和divB，分别代表A框和B框。

* 1. DivA中包括有14个div，13个div分别放置13个扑克。第14个div显示拖动次数
  2. DivB中没有任何内容，在扑克拖动到B框时动态加入

#### 主要函数

1. drop(event):
   1. A框触发ondrop事件的操作：在思路中提到，它主要用于将图片重新反面显示
   2. 函数主要功能实现：A框触发ondrop事件，假如是B框的图片移动过来，该函数获取ondragstart事件方法存放的数据(拖动图片的id)，然后根据id获取到该图片的dom对象设置他的src值为扑克反面，然后放回A框
   3. 其他功能：count加一（用于记录拖放次数），divBcount--（记录divB的扑克数量），sum -= Number($img.value)(用于记录divB中扑克大小总和)，recordSub(Number($img.value))（用于记录divB中扑克各个的大小）
   4. 其他细节：如果是A框中图片拖到A框呢？所以应该加判断：它不是A框拖放的，具体实现有很多方法，不细说。
2. drop2(event)：
   1. B框触发ondrop事件的操作：主要用于将拖过来的图片正面显示
   2. 函数主要功能实现：同样是获取拖放图片的id，然后根据id得到img对象，设置img的value为一个随机且不重复的值（如果已经有跳过），设置它的src的值为使之显示正面（通过字符串拼接）。最后将其用appendChild到B框
   3. 该函数还会检查拖放是divBcount <5是否成立,如果不成立显然B框扑克已满，
   4. 在函数的末尾，判断divB==5是否成立，成立就调用isEnd函数来判断是否满足顺子
   5. 记录数据细节不细说，记录的数据主要用于isEnd函数判断是否为顺子
3. recordAdd(num)和recordSub(num):
   1. 在drop和drop2两个函数中被调用，更改record数组记录的情况
4. isEnd():
   1. 在divBcount等于5时被调用，判断游戏是否结束
5. Drag(event)和allowDrop(event)：实现拖放必须函数

#### 页面显示流程图示



#### 总结

1. 熟悉了HTML5拖放API的使用
2. 熟悉了HTML5拖放元素的过程
3. 加深了对JavaScript函数的理解
4. 学习到了Math. Random函数产生随机值，以及Math.trunc函数取整等等函数
5. 学到了全局变量存储数据，加快计算，以空间换时间
6. 其他问题总结：
   1. 拖放不只有A框拖放的到B框，还可能A->A，B->B框，所以应该做好拖动元素来源的判断，防止出现非预期的情况。
   2. B框5张扑克，当它们为顺子时，那么它们的平均值一定是sum/5，且sum/5一定为整数。

### 实验4：问卷调查

问卷相较于实验二：电子日历，实验三：扑克拖放

主要使用html5的API，涉及JavaScript的内容不多，逻辑简单

#### 页面框架

HTML部分：

主体div id=main，内部<h1>，<form>，然后就是简单的罗列<input>，<select>,<textarea>等API实现各种输入，其中使用了ul无序列表让它看起来更整齐。在form的最后，有一个按钮标签，在其onclick方法调用check()方法，检查输入数据格式是否正确。如果不正确将错误信息显示。

JavaScript部分：只有一个check()方法,在点击提交问卷按钮时触发，进行表单内容的检查，主要功能：

1. 检测输入的名字非空，通过name的dom对象获取value值，判断其是否等于空字符串，等于的话return false，停止表单提交操作。
2. 检测电话号码是否全为数字，同上得到value值，然后使用正则表达式匹配字符串
3. 检测email的格式是否正确，对于type为email的input标签，浏览器(大部分)会自动检查是否符合email标准格式，也可以自己通过正则表达式验证
4. 检测gender不为空，getElementsByName("gender")获取到gender对象数组，循环该数组，如果有一个是checked那么gender已被选择。
5. 最后两个判断和gender方法相同。

#### 总结

了解了HTML5表单的作用

掌握了HTML5大部分表单API

学习到了通过getElementsByName获取dom对象，以及reason[i].checked来判断一个radio或checkbox是否被选择

学习到了通过onclick="return check()"来控制表单是否提交，check函数返回false即拒绝提交，返回true即提交表单

复习了CSS的各种属性，以及相对定位的使用

### 实验5：饼状图和柱状图

#### 设计思路

为了页面在初次加载时不那么空旷，为页面提供了默认数据。

1. 饼状图：在全局变量区定义一个sum变量存储总数，以及一个numCount数组存储各个人数的值，在每次绘制饼状图时，根据numCount数组各个值占sum的比例，来给各个类型分配圆弧的大小。
2. 柱状图：柱状图的坐标，以及X,Y轴不需要多次绘制，只需要页面加载时绘制一次，只要在数据更改时，重新绘制柱状图就可以。根据数据(numCount[i])的大小获取矩形的高，通过fillRect函数绘制矩形来绘制柱状图
3. 数据的接收：<input>表单API，并把该<input> 设置为number类型，这样还可以控制输入数据的类型。定义一个按钮”确实数据”，在点击获取到表单的value值到numCount数组中，之后开始绘制饼状图和柱状图

#### 页面框架

表单<form>以及内部的<input>控制数据输入，和<button>按钮标签，触发相关方法。

两个<canvas>画布标签绘制出对应的数据图。

#### 主要函数

1. showpiechart()：

绘制饼状图，通过获取ctx画笔对象，然后根据numCount在sum的比例对应到圆弧的比例来绘制饼状图

1. showcolumn()

在页面加载时调用，初始化柱状图基本框架：X,Y轴以及箭头和提示文字。

1. fillByData()：

绘制柱状图，根据数据的值

1. clickButton()

确实数据按钮点击时触发该方法，读取输入框中的数据到numCount数组和sum，然后调用方法1和方法3绘制更新的饼状图和柱状图

1. getdata(num)

计算numCount中下标为num的元素在sum中的占比，根据该比例，返回饼状图所占

1. randomCol()

获取随机颜色

#### 页面显示流程图示

