大家好，我是李芳媛，我的web汇报主题是冒泡排序的实现

->

首先给大家演示一下这个网站做出来后的整体效果（打开），3 2 4 1 5排序

大家所看到的网页是通过canvas画布来实现的。这里为了加快速度，我把定时器的时间间隔设置的小了点。但是我相信大家对冒泡排序的过程已经很了解了，应该跟得上这个节奏。

->

然后我们来看一下这个网页是怎么实现的，这个目录也就是我做这个网页的流程，首先我们可以在程序内部设定好元素的数值，尝试调换两个元素的位置；然后在这个的基础上完成一次比较完整的从小到大排序，但是这个待排序的数值仍然是我们程序内部设定的；那下一步我们就设置一个文本框用来输入待排序元素；然后对元素的移动过程进行优化，增加从大到小排序的功能，最后把整个页面优化一下，就达到我们刚刚所看到的最终的效果了。

因为这个目录后面的小节都是在他前面小节的基础上优化的，也就是不断地迭代。所以我前面章节讲的内容会多一点，后面的小节只会挑重点说

->

那我们来看看怎么调换两个元素的位置。在这个版本完成的功能是获取窗口的大小，从而调整画布、柱状图长度以及柱状图宽度并形成相应的图案，并调换指定两个元素的位置，这里我指定的是调换第一个元素和第三个元素的位置。

首先看一下v1.0版本的效果。

(关闭)

然后我们细讲一下是怎么实现的

->

首先定义一个canvas画布，把他声明为var类型，动态创建canvas元素并把它作为body标签下的第一个孩子节点。然后我们可以看到方法start中，根据窗口的大小设置了canvas的宽度和高度。从而使画布能够适应窗口的变化。

->

Start函数将被页面自动加载函数startLoad调用，每次加载只调用一次，也就是说每次刷新，画布就适应窗口一次，我们看看效果。

(演示“适应一次”和不断适应)

然后我们还在start中定义了定时器，所以大家可以看到只要一打开这个网页，它就自动开始排序。

->

->

然后我们看看画布上面的柱状图以及柱状图里面的数值是怎么形成的，他们都属于component类，左边这张图定义了他们的各个属性，id表明它在当前数组中的位置，每一次调换两个元素的位置，都要对id进行调整；一般我们都用x，y表示矩形块的位置，但是我们这里用originMiddle代替X，它记录的是上次调换前矩形块x轴的中点，主要是考虑到除了矩形块还有标注它数值大小的文本块，因为两者移动是分开的方便两者后续移动，（移动过程中 文本块始终位于矩形块的中间）。然后可以看到还有参数type，他用来表示component元素是矩形块还是数值。

->

在componet类中，还有update方法，用于不断更新小矩形块和文本块。

->

还有newpos方法，这个方法是小矩形块移动的关键，首先看这里有select1，select2两个变量用于标注两个即将交换位置的小矩形块的id，我们设定select1为后面一个元素,select2为前面一个元素。移动过程也分了步骤，我们来具体看看他们的移动步骤

->

这里的黑色小框圈住的是select1、蓝色小框圈住的是select2. select1移动过程分为4个状态，第一个状态是在本位还没有开始移动，第二个状态是向下移动到底部，第三个状态是向左移动到select2原位置的正下方，第四个状态是向上移动到select2的原位置。

select2的移动过程分为2个状态，第一个状态是在本位没有开始移动，第二个状态是向右移动到select1的原位置。这些状态我们用flag来表示，每一个componet元素都有一个flag表示状态

->

这个是移动具体的代码实现过程，判定flag数值并调整x和y的位置。最后等两个元素完成调换之后，调整他们的id和flag，并将select1、select2置为0。

->

然后我们利用刚刚说的componet类来创建具体的小矩形块

这里设置了三个数组来实现。Mybox数组用来填入小矩形块对象，value数组用来填入小矩形块对应的数值文本对象，还有一个gao数组，里面设定了一组待排序的数字。

我把这三个数组的0位置空出来，不填入正式的数据，id为0。那排序完成以后，我可以把select1、select2值置为0，也就停止排序了。

->

还在外部定义了一个函数，这个函数就是定时器每次更新都会调用的那个函数，用于更新画布。

->

然后我们实现一次较为完整的排序

->

主要原理其实很简单，在上一步中，我们的select1、select2分别设定初值为3 和1 完成一次调换后，select1、select2都置为了0，停止调换。那我们只需要在完成一次调换后，修改select1、select2为下一次需调换元素的id值就可以实现一次完整的排序过程了。那我们来演示一下效果

（点击链接演示）

->

下面我们来看看具体的实现过程

因为我们要从小到大排序，所以select1、select2初始化为1、0，在每次updateArea的时候，都重置select1、select2的值，如果select1前面有元素比它大，就置select2为该元素的id，否则select1数值加1，由此找出select1和select2数值。如果select1已经扫描到数组尾部，说明排序完成了，那就置select1、select2为0。

->

每完成一次调换就要将置select1、select2的flag为1，便于下次调换。

这就完成了一次完整的排序过程，但是我们发现这个待排序数值在程序内部已经设定好了，外部没法改变，而且一打开网页就自动开始排序，有点莫名其妙。

->

于是我们在v2.0版本的基础上增加输入排序数值、点击开始排序的功能。

->

让我们看一下效果。

(演示 输入3 2 1)

->

然后我们来看一下具体怎么实现的

首先增加input文本框，可以看到这里是使用了正则表达式的，只能输入数字和空格。（演示一下，输入字母，中括号）

然后增加了两个按钮

->

点击生成柱状图按钮，会调用produce函数，我们将之前在窗口加载自动调用的创建小矩形块、文本块功能移到这里，并且获取文本输入框的内容，放入gao数组中，根据这些文本框的数值生成适当高度和宽度的小矩形块、文本块。

（点击链接，输入 20 40 10 4 2 生成）

可以看到高度内部会自己调节，不管输入的数值差异有多大，都不会超出画布的高度

基础上输入1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9）

可以看到宽度内部会自己调节，不管输入的数值个数有多少，都不会超出画布的宽度

->

按钮点击开始对应的函数是intervalStart ，将myarea里面的定时器注释掉，添加到intervalStart函数中即可。

->

然后进行移动过程的优化

->

其实功能很简单，就是标注一下哪两个元素被选中要交换位置。

->

我们演示一下

（点击链接 3 2 1）

可以看到当前交换位置的两个小矩形块上方出现了不同颜色的长条，表示他们被选中要交换位置。

->

具体代码只需要在componet类的update方法中，给select1、select2选中的元素上方画一个很扁的小矩形框就可以，每交换一次select1、select2置为0，所以每完成一次交换，他们上方的小矩形框就会消失

清空画布比较简单，我就不讲了。

->

->

然后为页面添加从大到小排序的功能，

->

演示一下(1 2 3 从大到小排序 从小到大排序)

->

因为有了从小到大排序的代码基础，所以从大到小排序其实很好做。首先设定两个按钮，分别表示开始从小到大排序和开始从大到小排序。分别绑定各自的函数。

每个函数中都要清空定时器，并设置direction的值，表示不同的排序方向。然后启动定时器。 这里清空定时器还是非常必要的，如果不清空，点多少次就有多少个定时器，我们来看一下如果不清空定时器会怎么样。

（点击链接不清空定时器 1 2 3， 我们可以看到如果不清空定时器多次点击按钮，矩形块移动越来越快）

然后我们看一下清空定时器后的效果

（点击清空定时器 1 2 3 ，我们可以看到清空定时器以后，多次点击按钮，矩形块移动没有什么变化）

->

然后我们增加从大到小排序的功能，在updataarea里面判定direction的值，如果是1就从小到大排序，如果是2就从大到小排序。改变选择select1、select2的方式，小矩形块、文本块移动只需要改变方向即可。

->

然后我们对这个界面整体优化一下，推出最终版本。

->

（复制 点击链接 ）

主要用css进行优化，布局采用 grid进行优化，

按钮改变了大小以及hover效果

输入框选中会低调的发光

除此之外，我们在利用输入框中的信息生成矩形块和文本块时，只能生成一次，这里我设置了禁用按钮。如果想要再次生成就先清空画布，在文本框内输入信息，点击生成。  
 在排序过程中，我将从小到大排序和从大到小排序按钮禁用了，防止排序过程中断，保证排序过程的连续性，但是在排序期间，我们可以点击暂停和继续来控制排序过程。

然后我们在最上方添加了处理信息，方便我们知道排序进行到哪一步了。

最下方的暂停的实现原理是清除定时器，然后下方的滑块可以控制小矩形块的移速，可以通过鼠标控制也可以用键盘左右方向键来控制

输入（3 44 38 5 47 15 36 26 27 2 46 4 19 50 48）

总结一下，这个冒泡排序网页可以实现用户输入文本内容，选择按钮进行从大到小排序或从小到大排序，排序过程中不可以再次点击从大到小排序或从小到大排序，同时会在页面显示排序状态，（是正在排序还是已完成排序）可以通过调节下方的拖块来控制排序的速度，或者点击暂停和继续来控制排序流程。其中对我来说比较繁琐的还是调节小矩形块和文本块位置的工作，因为这个是两个对象，所以他们的位置在移动过程中要保持对齐。所以在中途调试过程中，我遇到了很多和我预想运行过程不太一样的bug，然后通过consule 输入出到控制台，debug找到问题并解决。强烈推荐用控制台辅助我们调试程序，效果真的很不错！

->

->

->

->

这就是我实现冒泡排序的全部内容，谢谢大家。