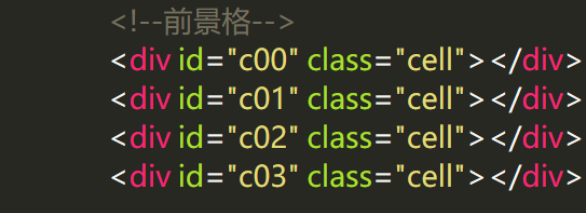
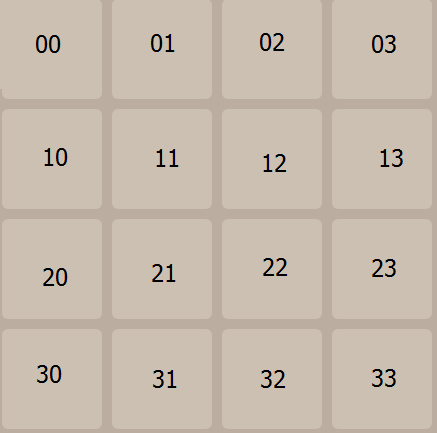
# 编写PC端网页单机版2048(2)

在上一篇中我们完成了大部分的html和css文件中的代码，这次我们来编写2048的游戏逻辑，只需要用到js文件。

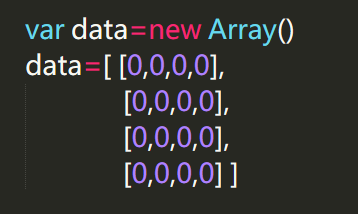
在上一篇文章中，我们在html文件中将前景格的元素都赋予了ID。



赋予ID是为了方便的控制他们，所以我们赋值之后的结构图如下。

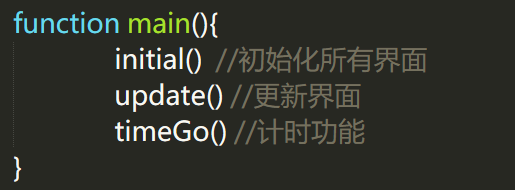


接下来我们需要一个二维数组跟其对应上，用来储存每一个格子对应的值



创建好了之后，我们可以正式开始进入编写阶段了。

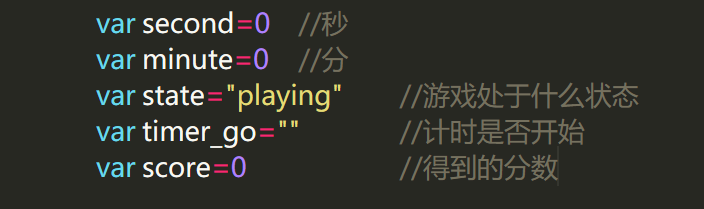
首先，我们先写一个主函数(总得有一个程序的入口)，想一想我们需要什么功能的函数。



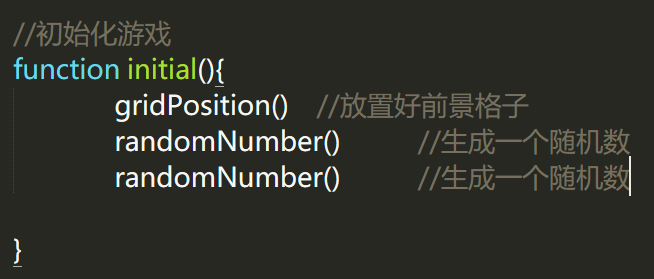
虽然我不知道我编写程序的方法和思路是不是正确的，但是所有的事情都是先有需求才有实现，先写上我们需要的函数，然后再想怎么去实现函数的功能。

我们先来想一想有什么变量，我们需要的数值：分钟、秒数、游戏状态、计时开关、分数。

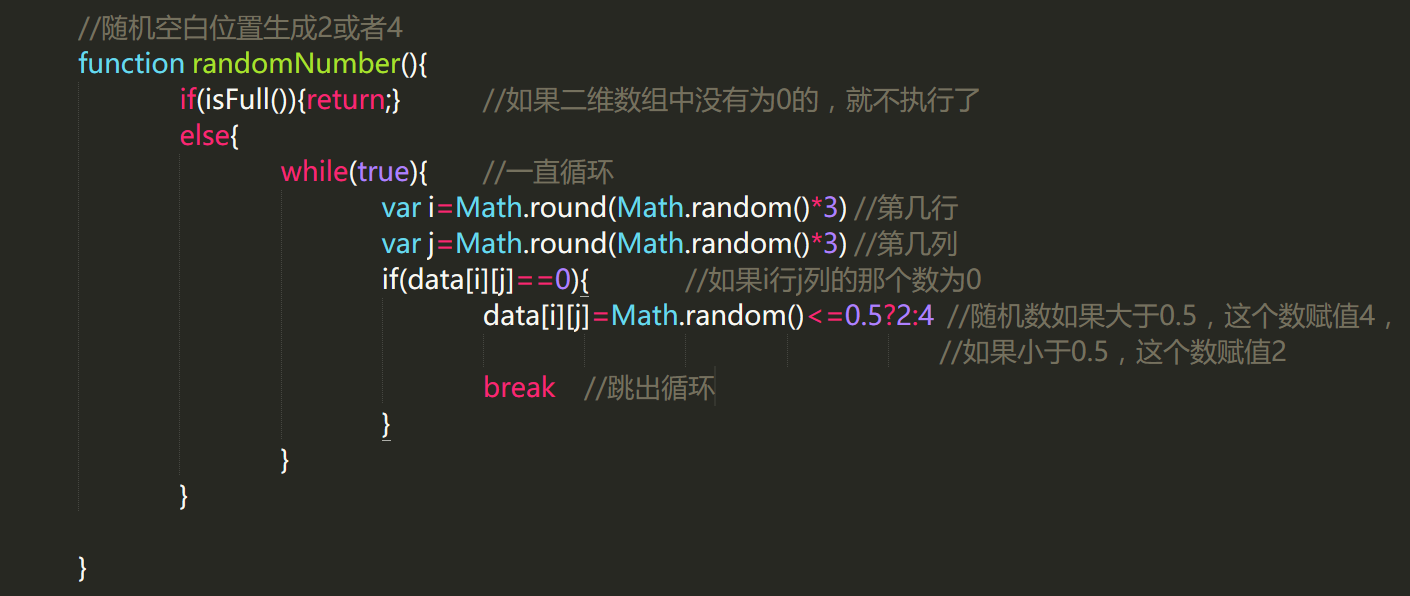
暂时能想到的就是这些，那么我们先来写上。



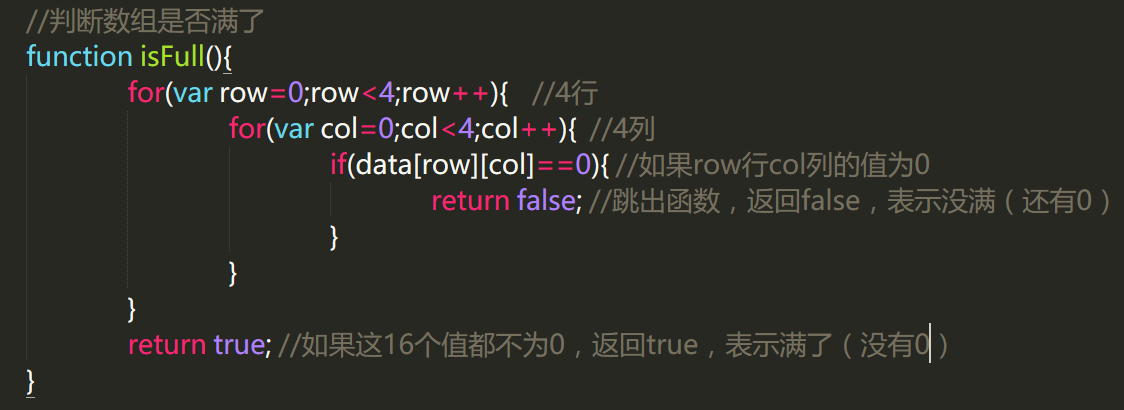
我们再来想一下初始化界面都有哪些需求：1.将16个前景格子放到界面上2.在16个方格中任意两个格子生成数字（2或者4）。



怎么想的就怎么去写。放置前景格子的方法(gridposition)，我们在上一篇已经写好了,这里只需要调用它就好了，下面我们去实现生成随机数的函数。



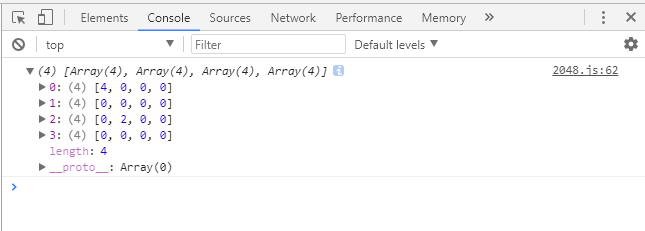
这里的Math.random()是js自带的函数，调用它会得到0-1之间的随机小数，我们乘以3之后，就会得到0-3之间的数了。Math.round()的作用是四舍五入。这里我们又想到了先得判断数组中是不是都不为0，如果都不为0了，那么我们就没有位置去生成新的数字了。去实现isFull()吧。



好了，写好之后，我们可以加入一句话，来查看我们的函数正不正常。



我们可以在浏览器中运行一下html文件，然后按下F12（谷歌浏览器），找到Console。再按一下F5刷新，我们就可以看到是否有正常生成两个数字和界面是否正常。

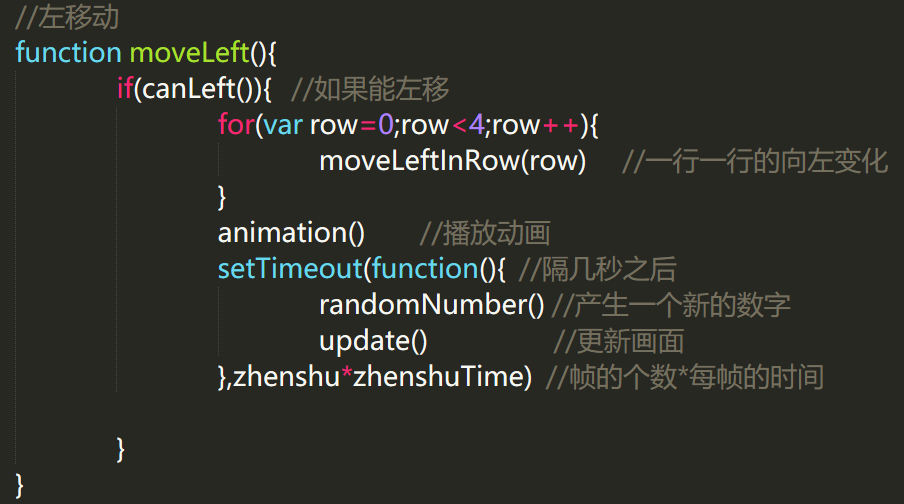


接下来我想到了按键控制，有上下左右4个键来控制游戏。



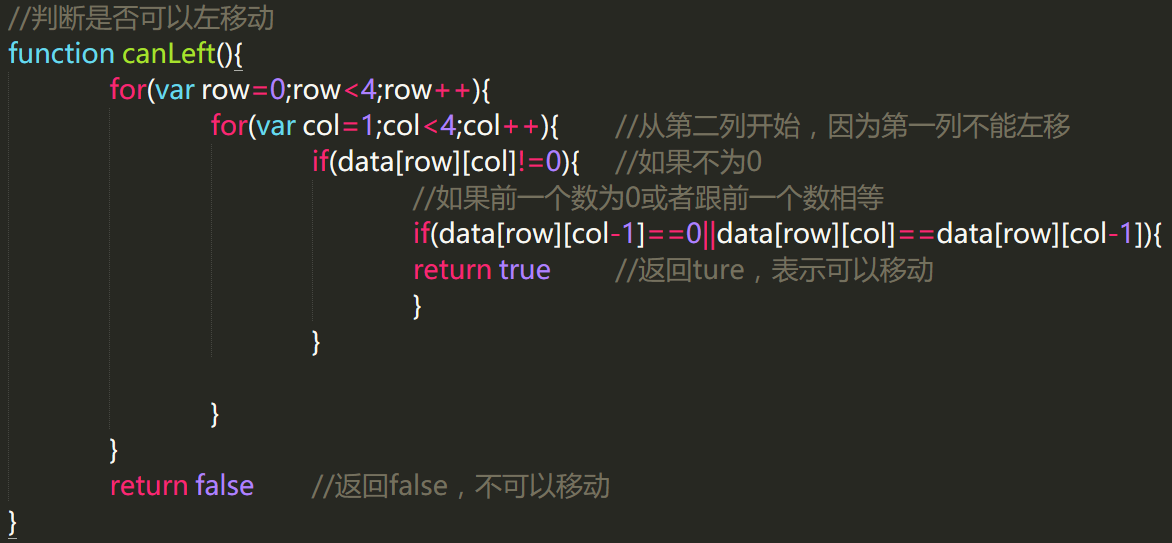
我们的逻辑是刚开始不计时，所以得需要按下一个键时，才会开始计时，state的变量是用来避免在播放动画时按键，导致动画没播放完（不明白没关系，往后看就好了）。在这中间我加了大量的console.log是为了查看我写的东西是否生效。不要怕麻烦，写一些，检查一些，就不容易出错。

接上来我们来实现上下左右移动的函数(moveUp,moveDown,moveLeft,moveRight)，因为4个方法类似，我就贴出来其中一个。拿左移来举例:

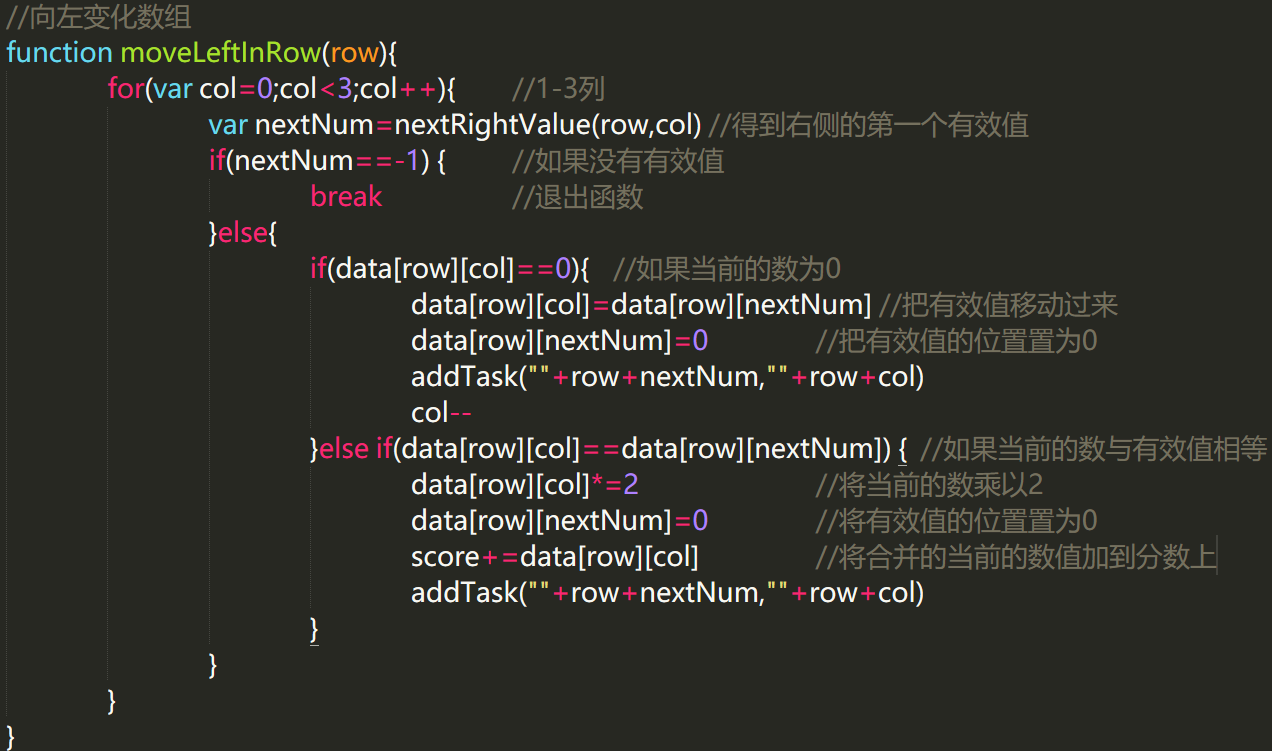


我们需要判断是否能左移(canLeft),需要变化一行(moveLeftInRow)。再加上更新画面(update)和随机产生数字(randomNumber)。写这些这就可以了。因为主要是分享2048游戏逻辑的实现。所以播放动画的部分如果想了解，可以向大鱼要源码。

canLeft函数的实现:

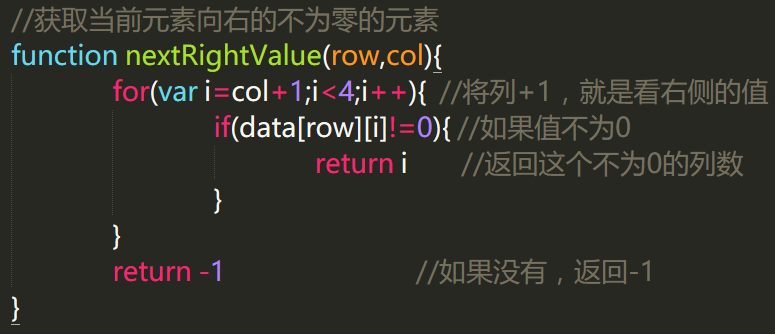


moveLeftInRow函数的实现:



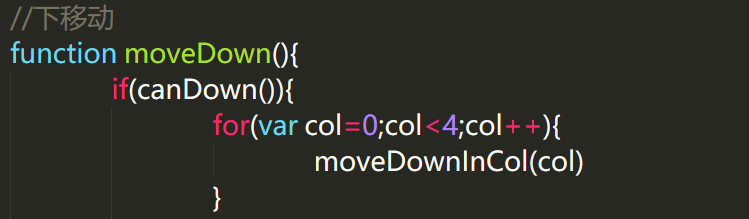
没有注释的是跟动画有关系的部分，将注释的写上就可以。其中有一个nextRightValue的函数，这个函数是为了获得同一行中，当前数右面的有效值。

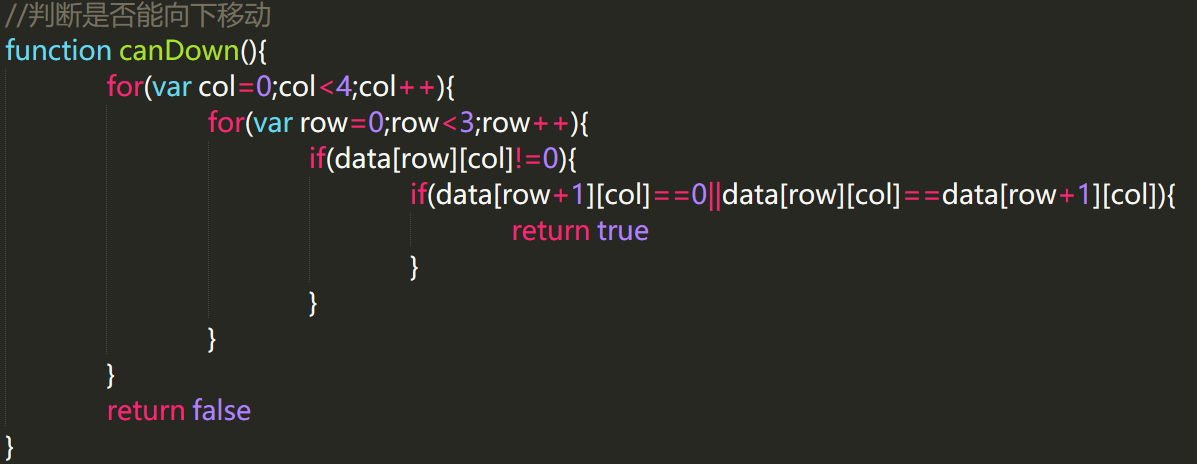
nextRightValue的实现:

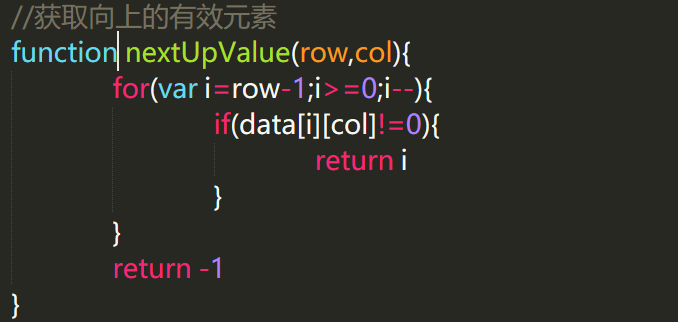


我们就完成了左移的所有函数,因为右移和左移是相对的，所以就不放上代码了，可以借鉴左移来实现，改一些for循环的方式就可以。下面将下移的具体代码分享出来，就不一句句的注释了，大致都是相同的。

下移的代码：





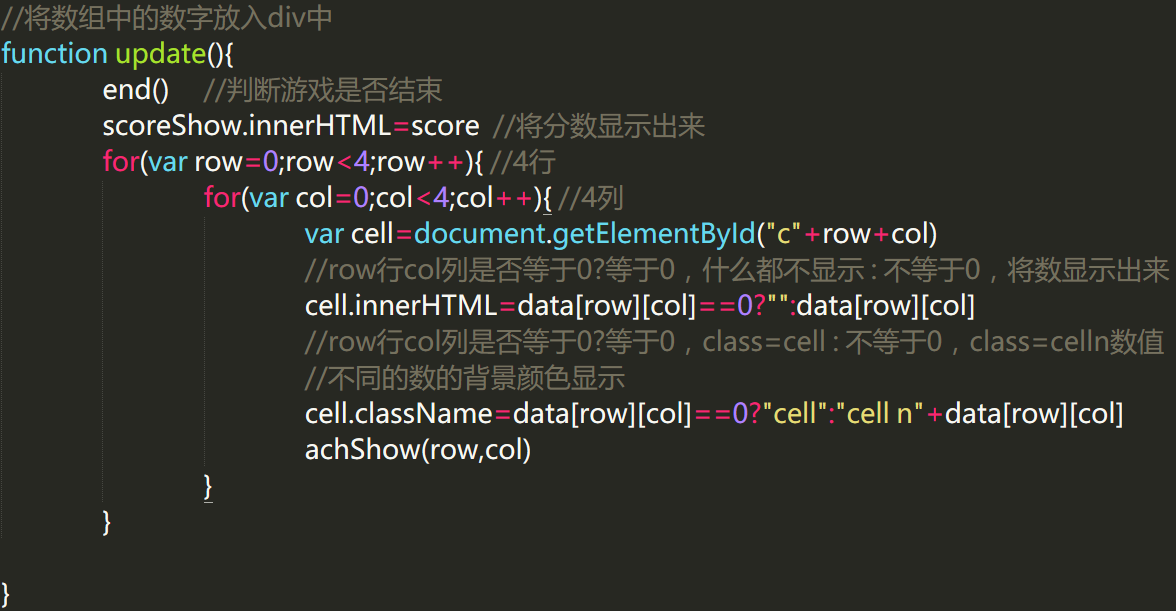




借鉴下移，我们也可以写出来上移的代码。这时，我们就完成了所有移动的函数。

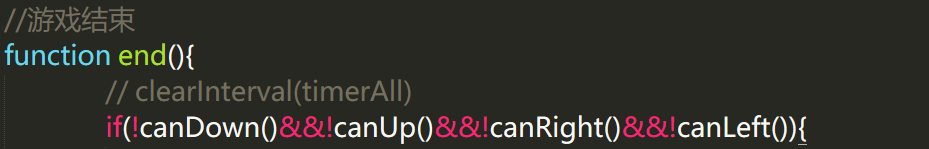
下面我们只需要再实现更新函数(update)就可以了。

update函数的实现:



achShow也是关于动画的部分，就不展示出来了。

End函数的实现:



如果不能上下左右移动就算结束了，你在if中添加console.log(“结束了”)来查看是否结束了,或者自己想添加一些其他的功能，例如显示一个弹窗出来。

最后就是计时功能了



这些都完成了，我们就实现了2048的所有游戏的基本逻辑了。