

The Document of BlackJack(21)

一.游戏介绍

1.起源

2.规则

3.技巧

二.游戏设计

1.整体UI构思

2.素材采集

3.游戏总规划

三.代码设计

0.编码规范以及优化记录

1.HTML文档结构

2.CSS布局与动画

3.JavaScript功能模块

四.游戏测试（以下所列皆可正常运行游戏）

1.PC端

2.移动端



一.游戏介绍

1.起源

21点又名黑杰克（英文：Blackjack），起源于法国，已流传到世界各地。21点，是一种使用扑克牌玩的赌博游戏。亦是唯一一种在赌场中可以在概率中战胜庄家的一种赌博游戏。

2.规则

- 21点是一张牌面朝上(叫明牌)，一张牌面朝下(叫暗牌);给自己发两张牌，一张暗牌，一张明牌。
- 玩家手中扑克点数的计算是:K、Q、J 和 10 牌都算作 10 点。
- A 牌既可算作1 点也可算作11 点，由玩家自己决定。
- 其余所有2 至9 牌均按其原面值计算。
- 如果玩家前两张牌是A 、10点牌，就拥有黑杰克(Blackjack);
- 如果庄家没有黑杰克，玩家就能赢得2倍的赌金(1赔2)。
- 没有黑杰克的玩家可以继续拿牌，可以随意要多少张。目的是尽量往21点靠，靠得越近越好。
- 如果所有的牌加起来超过21点，玩家就输了--叫爆掉(Bust)，游戏随之结束。
- 如果玩家没爆掉，又决定不再要牌了，这时庄家就把他的那张暗牌打开来。
- 一般到17点或17点以上不再拿牌，但也有可能15到16点甚至12到13点就不再拿牌或者18到19点继续拿牌。（本游戏采用1号电脑的逻辑为达到16点便不再拿牌）
- 假如庄家爆掉了，那他就输了。
- 假如他没爆掉，那么你就与他比点数大小，大为赢。一样的点数为平手。

3.技巧

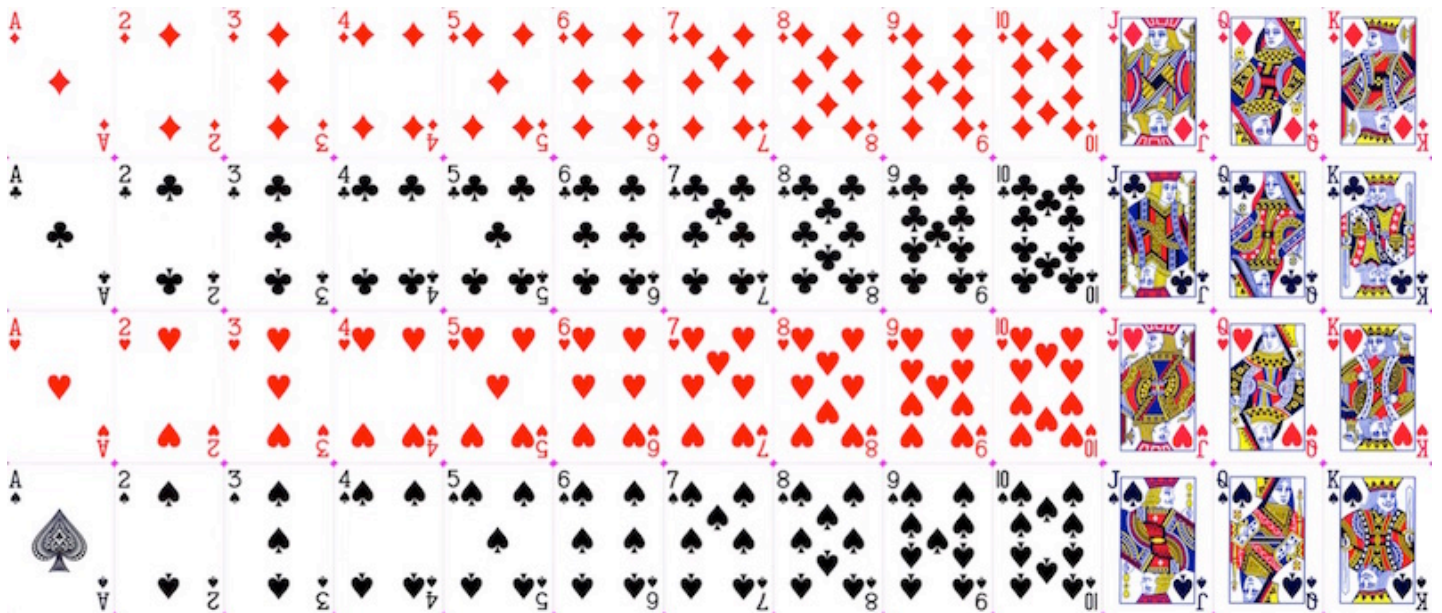
- 障眼法
 - 这种方法主要适合于闲家，而且在众多玩家参与的时候适用。
 - 一般在闲家明牌是10点的时候，如果牌底是3~7点，一般拿到这种点数特别难受，如果要牌，有50%的机率会暴，因为闲家暴点是100%输，不要牌可能也会输，我暂且把它称为尴尬点数。这时候，闲家如果观察到别的闲家点数都比较大，而且都不要牌的情况下，可以跟着不要。这样一来，就会给庄家很大的压力，以为每一位闲家手中的牌都是比较大的点数，如果庄家同样拿着尴尬点数，这种情况只能逼着庄家要牌，很明了，无形中所用的这个障眼法是把风险转移给了庄家，输赢靠天定了，看哪个运气好了，若庄家爆了，则闲家赢。
- 姜太公钓鱼法
 - 这种方法主要适合于闲家，而且在众多玩家参与的时候适用。
 - 一般在闲家明牌是10点的时候，如果牌底是3~7点，一般拿到这种点数特别难受，如果要牌，有50%的机率会暴，因为闲家暴点是100%输，不要牌可能也会输，我暂且把它称为尴尬点数。这时候，闲家如果观察到别的闲家点数都比较大，而且都不要牌的情况下，可以跟着不要。这样一来，就会给庄家很大的压力，以为每一位闲家手中的牌都是比较大的点数，如果庄家同样拿着尴尬点数，这种情况只能逼着庄家要牌，很明了，无形中所用的这个障眼法是把风险转移给了庄家，输赢靠天定了，看哪个运气好了，若庄家爆了，则闲家赢。
- 补救法
 - 这种方法庄家闲家都适用。
 - 所谓补救，就是拿到的是前面所提到的尴尬点数，输地可能性极大，在明知道牌点数就比对手点数小的情况下，我们只能要牌，因为不要也是输，而要牌还有一线希望是赢，所以我们能选择的就是奔那一线希望，寄希望总比放弃希望好。

二.游戏设计


1.整体UI构思

- 背景：我认为需要一个暗色的、非纯色的背景。
- 桌面：一张记忆中赌博赛事的标准绿色桌子，木质黄色的边。
- 按钮：于是有了简约风格的画面，接下来我考虑将按钮在桌面中。
- logo：考虑到游戏叫 21点，所以 应用了简单却不失代表性的带阴影的透明21图标。
- 候牌：做两个等牌区的底框（透明的淡色框），等待发牌，并标注玩家x号。
- 积分：考虑到筹码问题，于是在等牌区的一侧做一个不太复杂的标注，响应整体简约的风格。
- 标题：放在桌面的正中央，游戏主题的表达，BlackJack(21)足矣。
 - ⚠️ 不能贴顶，至少产生`h1 { margin-top:20px;}` 以保持美感。
 - 游戏界面的色彩搭配灵感部分来源于其他网络扑克游戏
 - （游戏素材均来自网络）

2.素材采集



3.游戏总规划

- 利用**html + css**将设计好的整体的布局 + 采集到的素材  应用到配置、搭建一个初始的页面，后期再给予精确调整控制。
 - **html**基本文档结构
 - **css**布局 根据游戏的需求来编写不同的**css动画**和**js**功能函数以贴近游戏规则、增强游戏体验。
 - **css**动画
 - 使用animation keyframes完成动画:保证每次新局首发牌一人两张，且1号电脑玩家的第二张牌为暗牌，后再发牌则进行一人一张动画。
 - 配合js控制扑克牌发出前后的显隐，特别是亮牌后1号电脑暗牌的显示。

- **JavaScript各功能函数设计**

- **实现actions系列按钮的功能**

- 新局 sendCard()
 - 要牌 sendCard()第二次开始
 - 亮牌 showSend()
 - 退出游戏 exit()
 - 初始化整副牌的数组，即洗牌，给52个数字重新分配扑克牌的值（花色，数字）
 - 本游戏设定只有一副牌，即在一个length为52的数组中抽取牌进行游戏，也就是说每次新局开始便初始化，保证其有52张牌的相应概率来进行游戏，属于不放回游戏。
 - 每张牌的选取靠js配合for控制量i对每张牌的坐标进行计算以保证每次新局初始化时都可以在random的控制下以tmp结果随机选取一张牌（剩余的牌堆中）。
 - 实现发牌后css动画结束前，计算相应偏移位置并自动添加html内容，使得动画结束之时，添加落牌刚好衔接，其中使用了for中的i控制外加setTimeout实现单次计时。

- **本游戏规则的逻辑设定声明：**

- 每次开局一人两张牌，后每次要牌均为单张发牌动画。
 - 1号电脑的设定是发牌小于16时，则在2号玩家点击要牌的同时加一张牌。若大于16，则不再要牌。2号玩家则自行判断，任意要牌。
 - Ace牌的1或11，在亮牌时进行智能判断：
 - 如果按11算不会爆牌则会按照11计算
 - 如果按照11算会爆牌则会按照1来计算
 - 黑杰克：新局初次发牌两张为1+10
 - 若为黑杰克，且庄家没有黑杰克时，则获胜时筹码加倍。
 - 庄家设定为一号电脑。
 - 如果双方有一方拿牌爆掉，便判定另一方为获胜方。
 - 如果都没有在拿牌过程中爆掉，则正常比对双方拥有的点数，小于21点且大的一方：获胜。
 - Actions操作系列按钮说明：（从左到右依次）
 - 手掌👐 为新局
 - 食指👉 要加牌
 - 两张牌交换为两牌
 - 21点的logo为退出游戏
 - 本游戏 点击发牌区的牌效果 = 点击 新局 + 要牌按钮

三.代码设计

0.编码规范以及优化记录

HTML:

标准的html结构, meta标签等

思考布局方式, 合理的结构

避免使用行内样式

事件尽量采用事件绑定

页面样式尽量处理的精致一些 (优先级以功能为主, 这些次之)

本次作品, 时间和质量的比重, 质量的权重高, 所以要优先提升质量

CSS:

1. 页面单独引用xxxx.css

2. 功能样式可以分类 (就算没必要分页、也可以按照功能写在一起), 如:

2.1 公共样式: 包括字体、h1~h6 p div等会用到的一些样式

2.2 布局样式: 就是布局相关的css写在一起, 主要处理布局、结构

2.3 功能样式: 各个子功能块样式, 如: 桌面、操作图标, 基本思想是按功能分块

2.4 编码相关: 尽可能少使用id; 最好使用class, 且其命名有功能性

(用于绑定事件)

样式性 (用户写样式), 命名语义化 (能表达出你这个样式是干什么用的,

尽量避免写flash1、flash2...) (已经改成sendCardTo)

动画相关样式可以单独写在一个文件里, 进行引用

手机上不居中可能也是这个原因, 另外手机上可以设置meta的viewport.

按钮问题: 可以给其父元素一个position:relative;

然后操作元素整体使用position:absolute;bottom:xx;进行定位

如果手机上要求访问效果和PC相同, 可以考虑样式做两套 (根据时间情况看吧)

Js:

格式得当。

尽量简化代码。

简单封装一下dom操作: 获取dom、addClass、removeClass、事件绑定等。

合理的注释, 每个方法都得有注释。

函数命名按照功能命名。

1.HTML文档结构

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title></title>
  <link rel="stylesheet" href="">
</head>
<body>
  <header></header>
  <section>
    <div></div>
    .....
    <div></div>
  </section>
  <footer></footer>
  <script></script>
</body>
</html>
```

- 本结构由html5中语义化标签构成，使得文档易读、结构清晰。
- 细节则根据题意和规则设计来完善

2.CSS布局与动画

- 动画
 - 设定路径实现简单的翻转发牌animation动画

```

/*绘制透明的绿色桌面以及木质桌面边缘*/
@keyframes sendCardTo1
{
    0% { right:-450px;top: 30px; transform: rotate(240deg); }
    100% { right: 0; transform: rotate(0); }
}
@keyframes sendCardTo2
{
    0% { right:-220px; top:30px; transform: rotate(240deg); }
    100% { right: 0; top:0; transform: rotate(0); }
}
@-moz-keyframes sendCardTo1 /* Firefox */
{sendCardTo1
    0% { right:-450px;top: 30px; transform: rotate(240deg); }
    100% { right: 0; transform: rotate(0); }
}
@-moz-keyframes sendCardTo2 /* Firefox */
{
    0% { right:-220px; top:30px; transform: rotate(240deg); }
    100% { right: 0; top:0; transform: rotate(0); }
}
@-webkit-keyframes sendCardTo1 /* Safari 和 Chrome */
{
    0% { right:-450px;top: 30px; transform: rotate(240deg); }
    100% { right: 0; transform: rotate(0); }
}
@-webkit-keyframes sendCardTo2 /* Safari 和 Chrome */
{
    0% { right:-220px; top:30px; transform: rotate(240deg); }
    100% { right: 0; top:0; transform: rotate(0); }
}
@-o-keyframes sendCardTo1 /* Opera */
{
    0% { right:-450px;top: 30px; transform: rotate(240deg); }
    100% { right: 0; transform: rotate(0); }
}
@-o-keyframes sendCardTo2 /* Opera */
{
    0% { right:-220px; top:30px; transform: rotate(240deg); }
    100% { right: 0; top:0; transform: rotate(0); }
}

```

• 布局

- 绘制桌面主体以及发牌区牌及牌堆
- 控制各个部件定位，样式
- 调整至可以正常在移动端（手机端）进行游戏


```
.circle {
    background: rgba(7,121,5,0.7);
    border-radius: 50%;
    position: absolute;
    top: -460px;
    left: 145px;
    width: 900px;
    height: 900px;
    border: 30px solid rgba(85,72,4,0.9);
}
/*发牌区动画的出发点以及牌*/
.send {
    position: absolute;
    top: 531px;
    right: 50px;
    height: 85px;
    width: 60px;
    -ms-transform: rotate(35deg); /* IE 9 */
    -webkit-transform: rotate(35deg); /* Safari and Chrome */
    -o-transform: rotate(35deg); /* Opera */
    -moz-transform: rotate(35deg); /* Firefox */
    transform: rotate(35deg);
    background-color: white;
    z-index: 2;
    border-top: 3px solid white;
    border-right: 2px solid white;
}
```

3.JavaScript功能模块

- sendCard():
 - 利用num来实现第一次发牌为一人两张，第二次开始皆为一张。
 - 使用setTimeout函数来实现一次性的定时操作：使得在动画结束后的已设定的时刻，依次调用realSend()函数以衔接发牌动作。
 - 实现了防治未开始游戏便点击亮牌的错误逻辑：新游戏开始调用sendCard(id)则begin++，若未执行此处，则要牌和亮牌均无法执行。

//此为核心代码，重复部分已省去，详情见源代码。

```
var num = 1;
if (count[1].count == 0) {
    num = 2;
    //AI
    for (var i = 0; i < num; ++i) {
        setTimeout(function() {realSend(1)}, i * 100);
        setTimeout(function() {
            var n = document.getElementById('p_1')
                .getElementsByClassName('send-card1');
            if (n.length > 0)
                n[0].classList.remove('send-card1');
        }, i * 100 + 800);
    }
}
```

- showCard():
 - 实现了防止未开始游戏便点击亮牌的错误逻辑
 - 实现了亮牌后的一系列函数动作:
 - getMax()、alert(count[id].sum)、lose() (已分别注释)
 - 以及根据结果判断输赢并弹出提示。

//此为核心代码，重复部分已省去，详情见源代码。

var begain = 0; //防止没开始游戏直接点击要牌和亮牌的按钮

```
function showCard() {
    if (begain == 0) {} else {
        var hidden = document.getElementById('p_1')
            .getElementsByClassName('card-hidden')[0];
        if (typeof(hidden) != "undefined")
            hidden.result();
        if (over)
            return;
        count[1].sum = getMax(1);
        count[2].sum = getMax(2);
        ... ..
        if(){alert("... .."); }else{alert("... .."); };
    }
}
```

- realSend()
 - 通过getPorker()、getPorkerPos()获得初始扑克牌位置真实发牌.
 - 并进行相关的dom操作html以完成真实发牌动作。
 - 如果一开始两张牌为1和10则为黑杰克。
 - 如果2号玩家拥有黑杰克而庄家没有则筹码翻倍。

- 根据count[id].sum > 21与否对lose()进行传递id以获得相应提示。

JavaScript

```
if (id == 1 && count[1].count == 1) {
    /* 庄家第二章暗牌 通过属性k, v来实现明牌和计算 */
    newNode.className = "send-card" + id + " card-hidden";
    newNode.setAttribute('k', pos);
    newNode.result = function() {
        this.className = "card";
        this.style.backgroundPosition =
            this.getAttribute('k');
    }
} else {
    newNode.className = "send-card" + id + " card";
    newNode.style.backgroundPosition = pos;
}
if (count[id].count == 1)
    newNode.setAttribute('v', card.value);
node.appendChild(newNode);

++count[id].count;
/* 对A单独处理 */
if (card.value != 1) {
    if (card.value > 10)
        count[id].sum += 10;
    else
        count[id].sum += card.value;
} else ++count[id].A;

var ai = count[1];
var pl = count[2];
if (pl.A == 1 && pl.count == 2 && pl.sum == 10)
    //如果一开始两张牌为1和10则为黑杰克。
    blackjack = 1;
if (blackjack ==
1 && ai.A != 1 && ai.count != 2 && ai.sum != 10)
    scale = 2;
    //如果2号玩家拥有黑杰克而庄家没有则筹码翻倍。

/* 先判定是否已经结束 否则调用机器人函数 */
if (count[id].sum > 21)
    lose(id);
else if (count[id].sum + count[id].A > 21)
    lose(id);
else if (id != 1 && count[2].count > 2)
    AI();
```

- getPorker()

- 抽牌函数。
 - 每次从开局时定义好的52个数的数组中不放回（porker.pop();）抽牌。
 - 返回tmp关联数组供getPorkerPos调用。
- getPorkerPos()
 - 接受传入的tmp参数以获取背景坐标

```
function getPorkerPos(tmp) {  
  // console.log(tmp.value);  
  return porkerPos[tmp.suit.toString()  
    + tmp.value.toString()];  
}
```

JavaScript

- getMax()
 - 在sendCard中调用，为的是实现每次传入id值算总和时智能判定将A算作1或者11，获得最大的优势。
 - 如果把A算作11大于21,则将其算为1，否则，算11。
 - 随后返回sum。
- lose()
 - 接受由realCard()传入的id参数，以衔接弹窗提示判断输赢。
 - 根据不同情况，编写不同的获胜提示，以二号玩家为操作玩家的角度提示。
- init()
 - 初始化游戏，将候牌区的清空并替换成玩家编号提示。
 - 重置blackjack，scale，over值。
- AI()
 - 函数功能：当sendCard()触发AI执行逻辑：
 - 若1号电脑拿牌总和小于16或者A牌经过getMax()智能处理后依旧小于16时，执行realSend(1)以继续要牌，这是一个智能判断，增加了游戏的可玩性。
 - 同时配合动画删除发牌区中的待发牌，然后一张随机抽取的牌显示在相应的位置。

```
function AI() {
    if (noneed)
        return;
    if (count[1].sum + count[1].A < 16 || getMax(1) < 16) {
        realSend(1);
        setTimeout(function() {
            var n = document.getElementById('p_1')
                .getElementsByClassName('send-card1');
            if (n.length > 0)
                n[0].classList.remove('send-card1');
        }, 800);
    } else
        noneed = 1;
}
```

- DOM操作实现Actions按钮及发牌堆首张牌的点击功能函数的事件绑定：

```
// 给 [btnName] 按钮 添加fn()功能并发牌功能
function addFn(btnName, fn) {
    var btnName = document.getElementById("btnName");
    btnName.onclick = function() {
        if (begain == 0) {} else {
            fn();
        }
    }
}
```

四.游戏测试（以下所列皆可正常运行游戏）

1.PC端

- Mac os x
 - chrome
 - firefox
 - safari
 - IE11
 - Edge
- Windows 10
 - chrome
 - firefox
 - safari
 - IE11

- Edge

2.移动端

- Android:5.1
 - 自带浏览器
- ios:9.3
 - Safari

五.番外篇

异常：若遇到任何问题导致无法正常浏览源代码

- 请您及时与我联系
 - 手机号：15594980508
 - 邮箱：451323138@qq.com
- 您也可以访问我的个人小站进行测试下载：
 - [个人小站](http://thqy39.GitHub.io) <http://thqy39.GitHub.io>
- 同时这份产品文档也将置顶在我的技术博客之中：
 - [技术博客](http://www.cnblogs.com/thqy39/) <http://www.cnblogs.com/thqy39/>

致谢：

感谢北京奇虎360可以给我这次展示自己的机会，无论结果如何。

版权：

本游戏仅为奇虎360公司前端星计划编写，任何人未征得本人同意之前请勿使用本代码作商业用途。