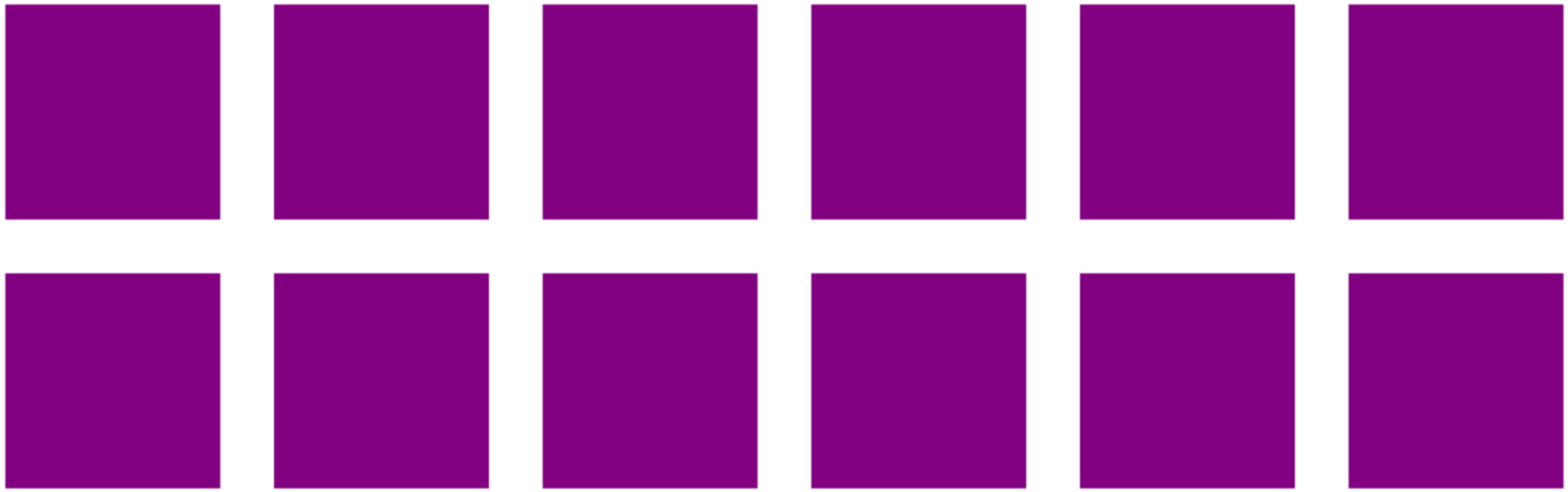
짝 맞추기 게임

게임 규칙

- 짝 맞추기 게임
  - 카드가 뒤집혀 있으면 플레이어는 한번에 카드 두개를 클릭해서 일치하는 카드 찾기 게임
  - 만약 두개의 카드가 일치하면 이 두 카드를 제거
  - 그렇지 않으면 다시 두 카드를 뒤집어 원위치 시킴
  - 플레이어가 일치하는 카드를 전부찾으면 프로그램 종료하고 게임 걸린 시간 표시

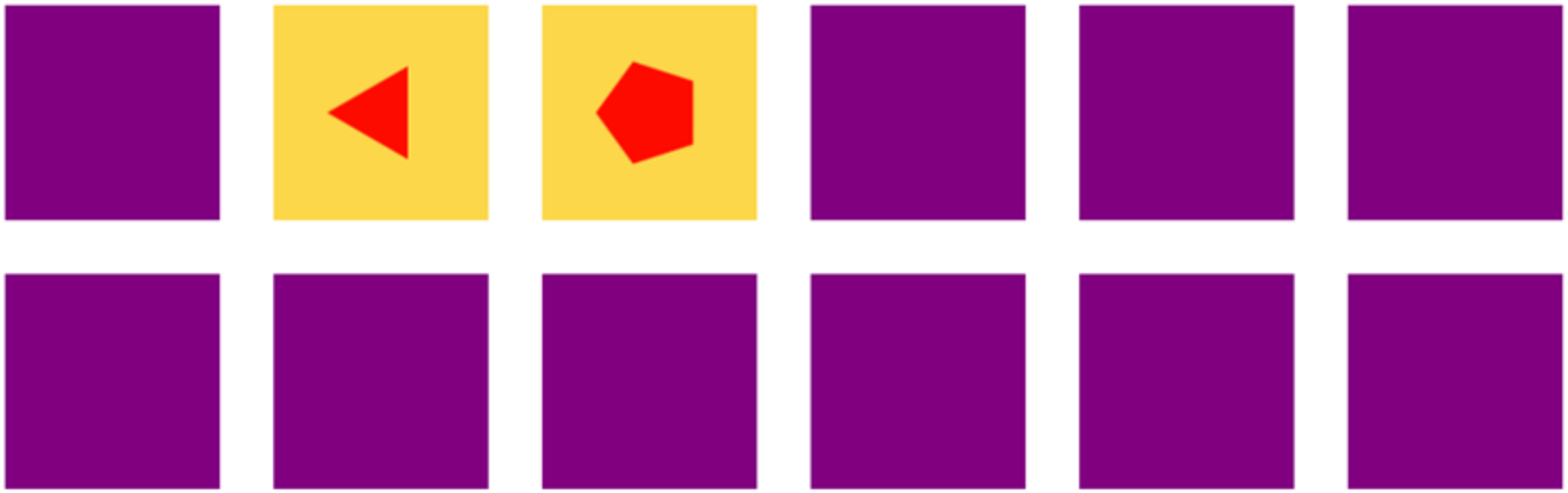
게임의 시작 예



두 카드를 클릭해서 서로 맞는지 확인하세요.

일치 횟수: 0

퍼즐 완료에 걸린 시각: 초



두 카드를 클릭해서 서로 맞는지 확인하세요.

일치 횟수: 0

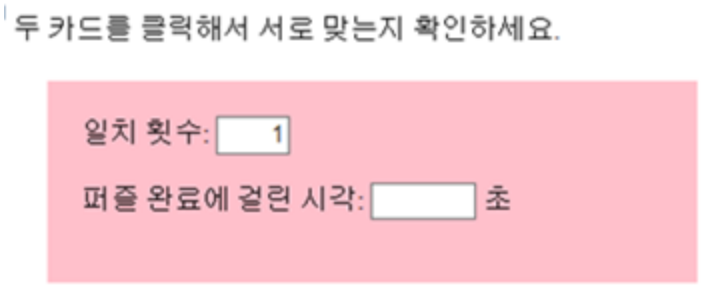
퍼즐 완료에 걸린 시각: 초

두 카드를 클릭해서 서로 맞는지 확인하세요.

일치 횟수: 1

퍼즐 완료에 걸린 시각: 초

게임의 결과 표시




```
<body onLoad="init();">
<canvas id="canvas" width="900" height="400">
이 브라우저는 HTML5의 canvas 요소를 지원하지 않습니다.
</canvas>
<br/>
두 카드를 클릭해서 서로 맞는지 확인하세요.
<form name="f">
일치 횟수: <input type="text" name="count" value="0" size="1"/>
<p>
퍼즐 완료에 걸린 시각: <input type="text" name="elapsed" value="  " size="4"/> 초
</p>
</form>
</body>
```

짝맞추기 게임

- 다각형 짝 맞추기
  - 3각형, 4각형, 5각형, 6각형, 7각형, 8각형
- 그림 짝 맞추기
  - 미리 입력한 인물 사진
  - 같은 인물 맞추기

작업해야할 사항

- 카드 뒷면 그리기
- 같은 선택 배열이 계속 나오지 않게 플레이어의 첫 선택 전에 카드 섞기
- 플레이어가 카드를 클릭할 때 감지하기, 첫번째와 두번째 클릭을 구별하기
- 클릭을 감지할 때 적절한 카드면(다각형,사진)을 표시하기
- 일치하는 쌍을 제거하기
- 일치하는 쌍의 수를 계산하기
- 게임 경과 시간을 계산하기

주요 기능

- HTML5 문서의 구조
- Canvas 요소
  - 선분으로 구성된 패스
  - 사각형, 그림, 패스 그리기
  - 사용자 정의 함수 및 내장함수
  - 사용자 객체
  - Form 요소
  - 배열
- 주요 새로운 기능
  - 시간 종료 이벤트, Date 객체
  - 캔버스에 글자 그리기

초기 화면

```
function init(){
   ctx = document.getElementById('canvas').getContext('2d');
   canvas1 = document.getElementById('canvas');

   canvas1.addEventListener('click',choose,false);

   makedeck(); // 패를 준비하는 함수, Card 객체 생성

   document.f.count.value = "0";
   document.f.elapsed.value = "";

   starttime = new Date();  // 시간을 설정하기 위한 함수
   starttime = Number(starttime.getTime());

   shuffle(); .// 카드를 섞는 함수
}
```

카드만들어 그리기

```
//generate deck of cards 6 pairs of polygons
function makedeck() {
        var i;                                                  var deck = [];
        var acard;
        var bcard;
        var cx = firstsx;
        var cy = firstsy;
        for(i=3;i<9;i++) {
                acard = new Card(cx,cy,cardwidth,cardheight,i);
                deck.push(acard);
                bcard = new Card(cx,cy+cardheight+margin,cardwidth,cardheight,i);
                deck.push(bcard);
                cx = cx+cardwidth+ margin;
                acard.draw();
                bcard.draw();
        }
        shuffle();
}
```

```
var cwidth = 900;
var cheight = 400;
var ctx;
var firstpick = true;
var firstcard;
var secondcard;
var backcolor = "rgb(128,0,128)";
var frontbgcolor = "rgb(251,215,73)";
var polycolor = "rgb(254,11,0)";
var tablecolor = "rgb(255,255,255)";
```

```
var cardrad = 30;
var deck = [];
var firstsx = 30;
var firstsy = 50;
var margin = 30;
var cardwidth = 4*cardrad;
var cardheight = 4*cardrad;
var tid;
var matched;
var starttime;
```

글자그리기

```
<html>
<head>
<title> 글꼴 </title>
<script type=“text/javascript”>
function init() {
  var ctx=document.getElementById(‘canvas’).getContext(‘2d’);
  ctx.font=“15px Lucida Handwriting”;
  ctx.fillText(“this is Lucida Handwriting”, 10,20);
  ctx.font=“italic 30px HarlemNights”;
  ctx.fillText(“italic HarlemNights”, 40,80);
 ctx.font=“bold 40px HarlemNights”;
  ctx.fillText(“HarlemNights”, 100,200);
  ctx.font=“30px Accent”;
  ctx.fillText(“Accent”+”KKK”, 200,300);
}
</script>
</head>
```

다각형 그리기

```
function Polycard(sx,sy,rad,n) {
  this.sx = sx;
  this.sy = sy;
  this.rad = rad;
  this.draw = drawpoly;
  this.n = n;
  this.angle = (2*Math.PI)/n  //parens may not be needed.
  this.moveit = generalmove;
}


 function generalmove(dx,dy) {
 this.sx +=dx;
 this.sy +=dy;
 }
```

다각형 그리기

```
function drawpoly() {
ctx.fillStyle= frontbgcolor;
ctx.strokeStyle=backcolor;
ctx.fillRect(this.sx-2*this.rad,this.sy-2*this.rad,4*this.rad,4*this.rad);
ctx.beginPath();
ctx.fillStyle=polycolor;
var i;
var rad = this.rad;
ctx.beginPath();
ctx.moveTo(this.sx+rad*Math.cos(-.5*this.angle),this.sy+rad*Math.sin(-.5*this.angle));
for (i=1;i<this.n;i++) {
  ctx.lineTo(this.sx+rad*Math.cos((i-.5)*this.angle),
                this.sy+rad*Math.sin((i-.5)*this.angle));
}
ctx.fill();
}
```

카드 객체

- 앞면(다각형, 그림), 뒤면(색칠된 사각형)

```
function Card(sx,sy,swidth,sheight,info) {
this.sx = sx;
this.sy = sy;
this.swidth = swidth;
this.sheight = sheight;
this.info = info;          // 앞면을 지정할 정보
this.draw = drawback; // 뒷면 그리기 함수
}
```

```
var backcolor = "rgb(128,0,128)";
function drawback() {
  ctx.fillStyle = backcolor;
  ctx.fillRect(this.sx,this.sy,this.swidth,this.sheight);
}
```

카드만들어 그리기

```
//generate deck of cards 6 pairs of polygons
function makedeck() {
        var i;                                                  var deck = [];
        var acard;
        var bcard;
        var cx = firstsx;
        var cy = firstsy;
        for(i=3;i<9;i++) {
                acard = new Card(cx,cy,cardwidth,cardheight,i);
                deck.push(acard);
                bcard = new Card(cx,cy+cardheight+margin,cardwidth,cardheight,i);
                deck.push(bcard);
                cx = cx+cardwidth+ margin;
                acard.draw();
                bcard.draw();
        }
        shuffle();
}
```

카드 섞기

```
function shuffle() {
//coded to resemble how I shuffled cards when playing concentration
	var i;
	var k;
	var holder;
	var dl = deck.length
	var nt;
	for (nt=0;nt<3*dl;nt++) {  //do the swap 3 times deck.length times
		i = Math.floor(Math.random()*dl);
		k = Math.floor(Math.random()*dl);
		holder = deck[i].info;
		deck[i].info = deck[k].info;
		deck[k].info = holder;
	}
}
```

카드 클릭하기

canvas1 = document.getElementById('canvas');

canvas1.addEventListener('click',choose,false);

choose 함수:

-어느 카드를 선택해서 섞을지 알아내는 코드 필요

- 플레이어가 캔버스를 클릭할 때 마우스 좌표를 반환해야 함

```
function choose(ev) {
    var mx;
    var my;
    var pick1;
    var pick2;
    var mx; var my;
    var pick1; var pick2;
    if ( ev.layerX ||  ev.layerX == 0) { // Firefox
        mx= ev.layerX; my = ev.layerY;
    } else if (ev.offsetX || ev.offsetX == 0) { // Opera
        mx = ev.offsetX; my = ev.offsetY;
    }
```

카드 클릭하기

```
for (i=0;i<deck.length;i++) {
  var card = deck[i];
  if (card.sx >=0)  //this is the way to avoid checking for clicking on this space
    if ((mx>card.sx)&&(mx<card.sx+card.swidth)&&(my>card.sy)&&
       (my<card.sy+card.sheight)) {
       //check that this isn't clicking on first card
        if ((firstpick)|| (i!=firstcard)) break;
    }
}
```

카드 클릭하기

```
if (i<deck.length) {  //clicked on a card
        if (firstpick) {
                firstcard = i;
                firstpick = false;
                // create poly to draw it on top of back
                pick1 = new
                Polycard(card.sx+cardwidth*.5,card.sy+cardheight*.5,cardrad,card.info);
                pick1.draw();
        }
        else {
            두 번째 선택
         }
}
```

카드 클릭하기

```
{
    secondcard = i;
    pick2 = new Polycard(card.sx+cardwidth*.5,card.sy+cardheight*.5,cardrad,card.info);
    pick2.draw();
    if (deck[i].info==deck[firstcard].info) {
        matched = true;
        var nm = 1+Number(document.f.count.value);
        document.f.count.value = String(nm);
        if (nm>= .5*deck.length) {
                var now = new Date();
                var nt = Number(now.getTime());
                var seconds = Math.floor(.5+(nt-starttime)/1000);
                document.f.elapsed.value = String(seconds);
                //need to fo to flipback for the last match
        }
    } else { matched = false; }
    firstpick = true;
    tid = setTimeout(flipback,1000);
}
```

일시정지효과넣기

- 플레이어가 카드 두장을 비교할 수 있게 일시정지
  - setTimeout(flipback, 1000);

firstpick

matched

```
function flipback() {
  var card;
  if (!matched) { deck[firstcard].draw();
            deck[secondcard].draw();
  } else {
     ctx.fillStyle = tablecolor;
     ctx.fillRect(deck[secondcard].sx,deck[secondcard].sy,
     deck[secondcard].swidth,deck[secondcard].sheight);
      ctx.fillRect(deck[firstcard].sx,deck[firstcard].sy,
      deck[firstcard].swidth,deck[firstcard].sheight);
      deck[secondcard].sx = -1;
      deck[firstcard].sx = -1;
   }
}
```

Date를 이용한 시간 측정

- starttime = new Date();
- starttime = Number(starttime.getTime());

```
var now=new Date();
var nt =Number(now.getTime());
var seconds=Math.floor(0.5+(nt-starttime)/1000);
```

그림짝 맞추기

맞춰볼 두 카드를 클릭하세요.

현재까지 맞춘 횟수: 0

```
var pairs = [
["allison1.jpg","allison2.jpg"],
   [ "grant1.jpg","grant2.jpg"],
  ["liam1.jpg","liam2.jpg"],
   ["aviva1.jpg","aviva2.jpg"],
   ["daniel1.jpg","daniel2.jpg"]
]
```

```
function Card(sx,sy,swidth,sheight, img, info) {
        this.sx = sx;
        this.sy = sy;
        this.swidth = swidth;
        this.sheight = sheight;
        this.info = info;
        this.img = img;
        this.draw = drawback;
}
```

```
function drawback() {
        ctx.fillStyle = backcolor;
        ctx.fillRect(this.sx,this.sy,this.swidth,this.sheight);
}
```

```
//첫번째 선택
firstcard = i;
firstpick = false;
ctx.drawImage(card.img,card.sx,card.sy,card.swidth,card.sheight);

// 두번째 선택
secondcard = i;
ctx.drawImage(card.img,card.sx,card.sy,card.swidth,card.sheight);
```

```
function makedeck() {
var i;
var acard; var bcard;
var pica; var picb;
var cx = firstsx; var cy = firstsy;
for(i=0;i<pairs.length;i++) {
        pica = new Image();
        pica.src = pairs[i][0];
        acard = new Card(cx,cy,cardwidth,cardheight,pica,i);
        deck.push(acard);
        picb = new Image();
        picb.src = pairs[i][1];
        bcard = new Card(cx,cy+cardheight+margin,cardwidth,cardheight,picb,i);
        deck.push(bcard);
        cx = cx+cardwidth+ margin;
        acard.draw();
        bcard.draw();
  }
}
```