캐논볼과 슬링샷

류관희

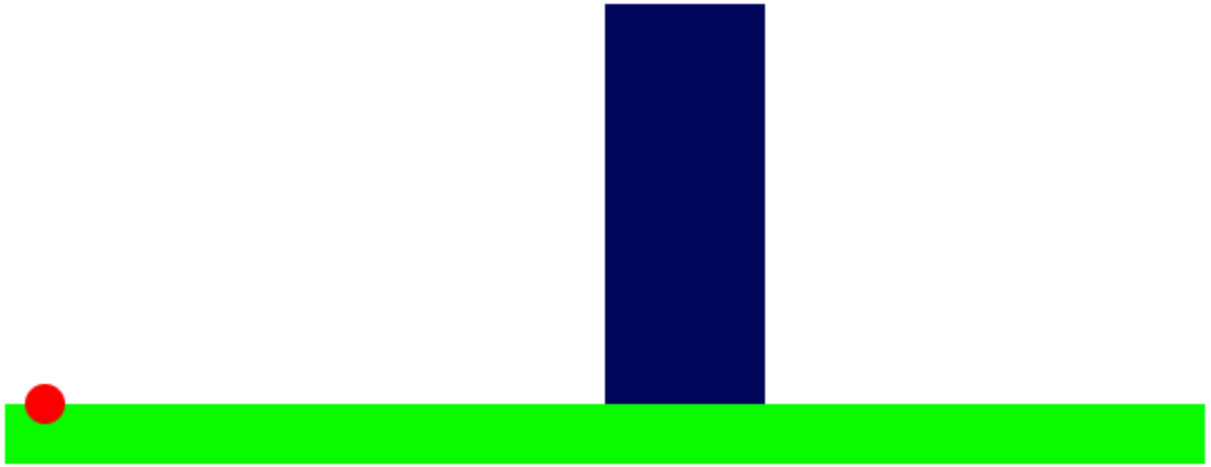
배우는 내용

- 포탄의 탄도 시뮬레이션
  - 포탄
  - 포물선 운동
  - 수평변위
  - 수직변위(중력의 영향)
  - 포탄이 지면 혹은 목표물에 닿으면 정지

주요 내용

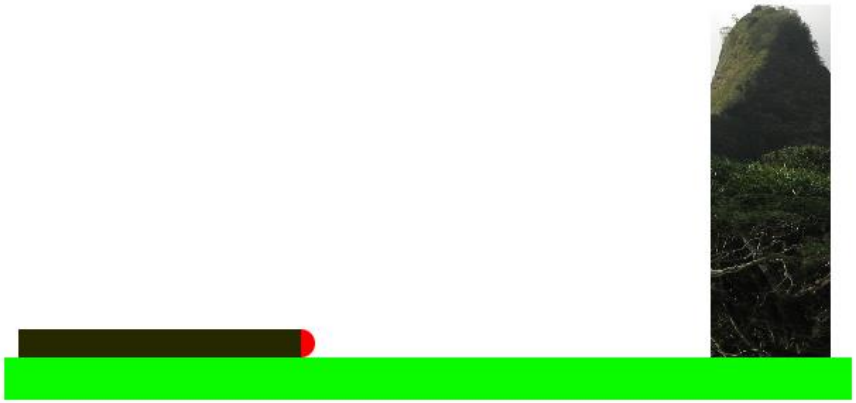
- 포탄의 기본 탄도 시뮬레이션
  - 포탄 발사(호를 그리며 이동, 입력: 수평속도, 수직속도 – 폼 입력 필드)
  - 포탄이 지면 혹은 목표물에 닿으면 정지
- 캐논볼(cannon ball)
  - 사각형 대포(특정각도로 기울어짐)
  - 매개변수: 대포에서 발사하는 속도, 각도
- 슬링샷(slingshot)
  - 새총 막대에 묶인 공
  - 공의 속도: 새총에 있는 공에서부터 한지점까지의 거리
  - 각도: 새총의 그 지점까지의 수평선에서부터 형성된 각도

대포가 없는 기본 탄도 프로그램



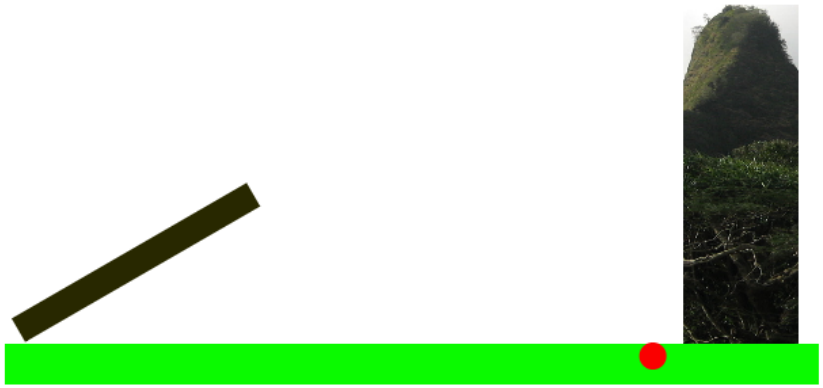
속도를 지정하고 포탄을 발사
수평 변위 10
초기 수직 변위 -25 발사

대포가 있는 캐논볼 프로그램



속도, 각도를 지정하고 대포를 발사
대포에서 발사되는 속도 10
각도 0 발사

대포가 있는 캐논볼 프로그램



속도, 각도를 지정하고 대포를 발사
대포에서 발사되는 속도 20
각도 30 발사

속도, 각도를 지정하고 대포를 발사
대포에서 발사되는 속도 40
각도 30 발사

슬링샷

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

공통 구현 사항

- 일정시간마다 이벤트 설정 후 공의 애니메이션 작동하고 공을 재배치
- 공의 포물선 운동 시뮬레이션
  - 수평변위(dx): dx=horvelocity(변하지 않음)
  - 수직변위(dy):
    - 가속량(중력): gravity
    - 구간처음의수직속도: verticalvel1
    - 구간끝의수직속도: verticalvel2=verticalvel1+gravity
    - dy=(verticalvel1+verticalvel2)/2
- 목표물에 닿았을 때 목표물 대신 다른 그림으로 대체하는 기능

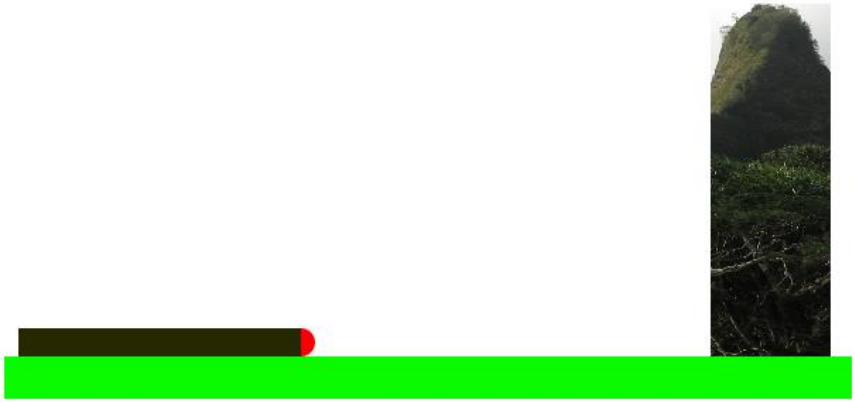
캐논 볼 구현

- 캐논볼
  - 초기값
  - 대포 발사 속도
  - 대포 발사 각도
  - 대포의 회전

슬링 샷 구현

- 슬링샷
  - 마우스 버튼을 누른채 새총 고무줄에 붙어 있는 공을 끌어 당긴 후 마우스 버튼을 놓으면 공을 발사
  - 새총 머리부터 공까지의 거리와 각도를 통한 공의 움직임 계산

Form 관리

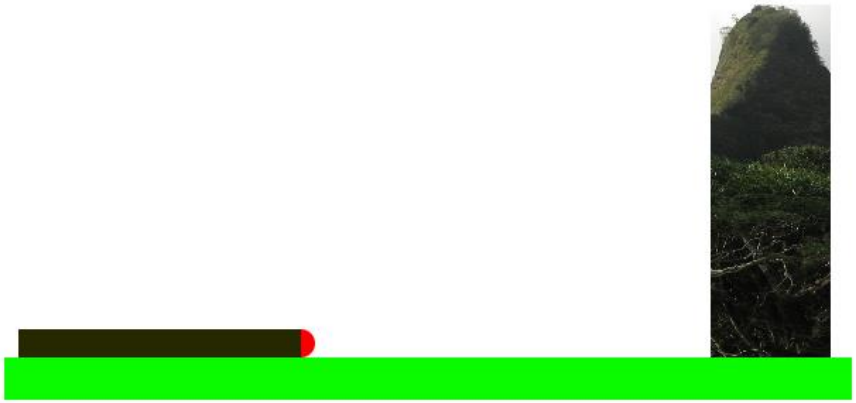



```
<form name="f" id="f" onSubmit="return fire();">
        속도, 각도를 지정하고 대포를 발사<br/>
        대포에서 발사되는 속도 <input name="vo" id="vo"
value="10" type="number" min="-100" max="100" />
        <br>
        각도 <input name="ang" id="ang" value="0"
type="number" min="0" max="80"/>
        <input type="submit" value="발사"/>
    </form>
```

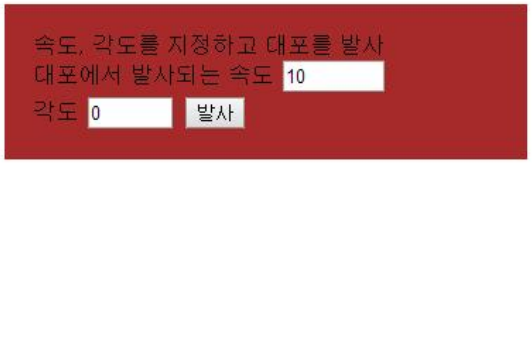
객체 관리

- 캔버스에 그려질 객체(항목의 모음이나 세트와 같은 개념)
- 객체
  - 속성과 메서드
  - Ball, Picture, Myrectangle, Sling
  - 속성: draw 메서드 정의와 위치, 크기를 설정하는 속성
  - 메서드: 회전

everything 객체



everything (var everything = [];)




```
var target = new Myrectangle(300,100,80,200,"rgb(0,5,90)")
var ground = new Myrectangle(0,300,600,30,"rgb(10,250,0)
```

```
everything.push(target);
everything.push(ground);
everything.push(cball);
```

Ball 객체

```
function Ball(sx,sy,rad,stylestring) {
        this.sx = sx;
        this.sy = sy;
        this.rad = rad;
        this.draw = drawball;
        this.moveit = moveball;
        this.fillstyle = stylestring;
    }

    function drawball() {
        ctx.fillStyle=this.fillstyle;
        ctx.beginPath();
        // ctx.fillStyle= rgb(0,0,0);
        ctx.arc(this.sx,this.sy,this.rad,0,Math.PI*2,true);
        ctx.fill();
    }
```

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

```
var cball = new
Ball(iballx,ibally,10,"rgb(250,0,0)");
```

Myrectangle 객체

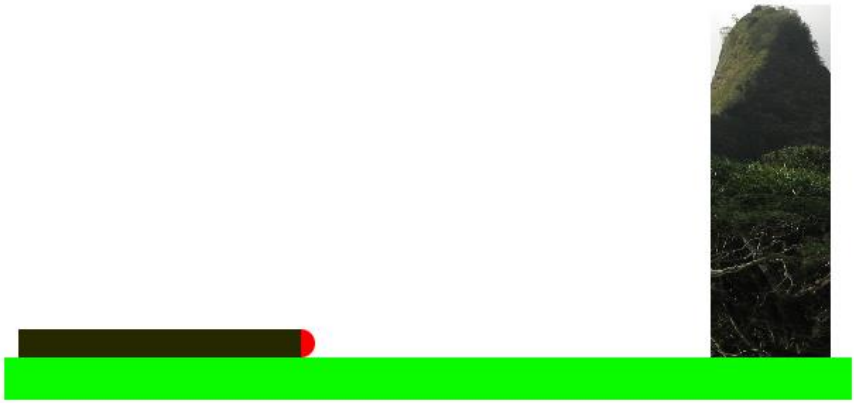
```
function Myrectangle(sx,sy,swidth,sheight,stylestring) {
    this.sx = sx;
    this.sy = sy;
    this.swidth = swidth;
    this.sheight = sheight;
    this.fillstyle = stylestring;
    this.draw = drawrects;
    this.moveit = moveball;
}

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

```
var target = new
Myrectangle(300,100,80,200,"rgb(0,5,90)");

var ground = new
Myrectangle(0,300,600,30,"rgb(10,250,0)");
```

Picture 객체

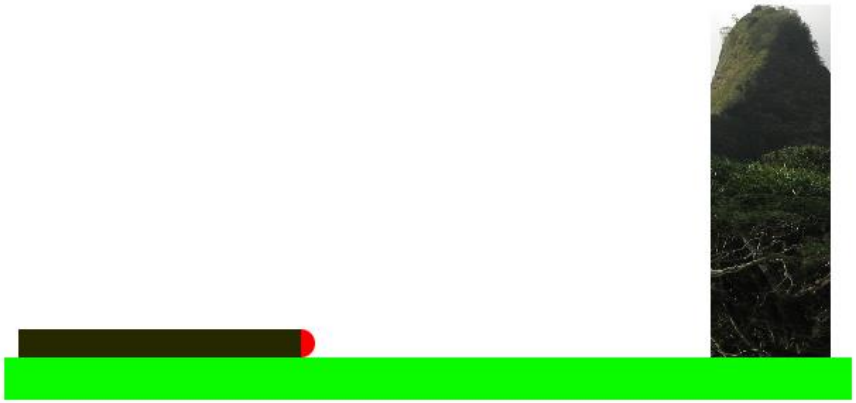


everything (var everything = [];)

```
function drawAnImage() {
    ctx.drawImage(this.img,this.sx,
    this.sy,this.swidth,this.sheight);
}
```


```
function Picture (sx,sy,swidth,sheight,filen) {
        var imga = new Image();
        imga.src=filen;
        this.sx = sx;
        this.sy = sy;
        this.img = imga;
        this.swidth = swidth;
        this.sheight = sheight;
        this.draw = drawAnImage;
        this.moveit = moveball;
}
```

everything 객체(1)



everything (var everything = [];)

```
var cball = new
Ball(iballx,ibally,10,"rgb(250,0,0)");
var target = new
Picture(targetx,targety,targetw,targeth,"hill.j
pg");
var htarget = new Picture(htargetx, htargety,
htargetw, htargeth, "plateau.jpg");
var ground = new
Myrectangle(0,300,600,30,"rgb(10,250,0)");
 var cannon = new
Myrectangle(cannonx,cannony,cannonleng
th,cannonht,"rgb(40,40,0)");
```

everything 객체(2)

```
var targetindex = everything.length;
everything.push([target,false]);

everything.push([ground,false]);

var ballindex = everything.length;
everything.push([cball,false]);

var cannonindex = everything.length;  // 나중 사용을 위해 저장
everything.push([cannon,true,0,cannonx,cannony+cannonht*.5]);
// 나중에 회전을 지정
```

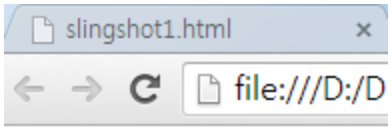
대포각도에 따른 대포의 위치변화

속도, 각도를 지정하고 대포를 발사
대포에서 발사되는 속도 40
각도 30 발사

사각형 그리기(1)

```
<html>
<head>
<title>사각형</title>
<script type="text/javascript">
    var ctx;
    function init(){
        ctx = document.getElementById('canvas').getContext('2d');
        ctx.fillStyle = "rgb(250,0,0)";
        ctx.fillRect(50,50,100,200);
        ctx.fillStyle = "rgb(0,0,250)";
        ctx.fillRect(50,50,5,5);
    }
</script>
</head>

<body onLoad="init();">
    <canvas id="canvas" width="400" height="300">
        이 브라우저에서는 HTML5의 canvas 요소가 지원되지 않습니다.
    </canvas>
</body>
</html>
```

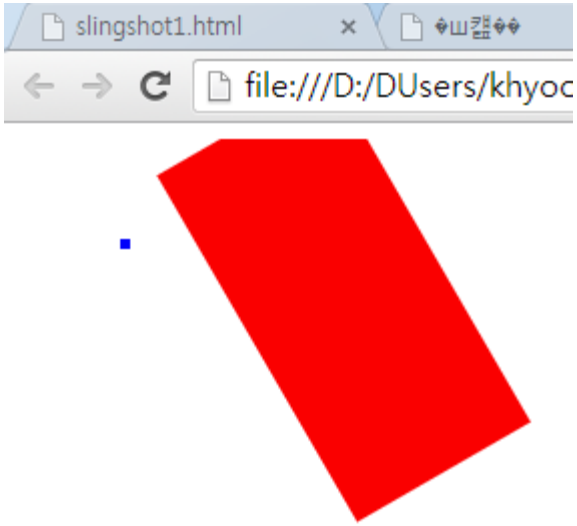


사각형 그리기(2)

```
<html>
<head>
<title>사각형</title>
<script type="text/javascript">
    var ctx;
    function init(){
        ctx = document.getElementById('canvas').getContext('2d');
        ctx.fillStyle = "rgb(250,0,0)";
        ctx.rotate(-Math.PI/6);
        ctx.fillRect(50,50,100,200);
        ctx.rotate(Math.PI/6);
        ctx.fillStyle = "rgb(0,0,250)";
        ctx.fillRect(50,50,5,5);
    }
</script>
</head>

<body onLoad="init();">
    <canvas id="canvas" width="400" height="300">
        이 브라우저에서는 HTML5의 canvas 요소가 지원되지 않습니다.
    </canvas>
</body>
</html>
```

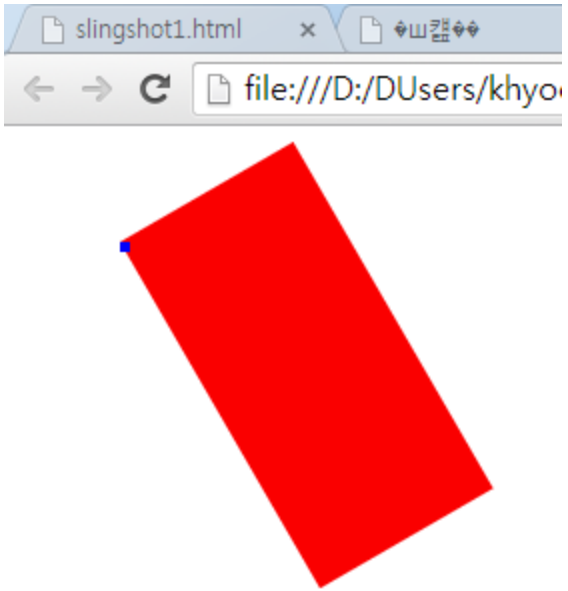
rotate
회전의 기준점:0,0
라디안 단위의 시계방향



사각형 그리기(3)

```
<html>
<head>
<title>사각형</title>
<script type="text/javascript">
    var ctx;
    function init(){
        ctx = document.getElementById('canvas').getContext('2d');
        ctx.fillStyle = "rgb(250,0,0)“;
        ctx.save(); //현재의 좌표계(위치와 축 방향 저장)
        ctx.translate(50,50);
        ctx.rotate(-Math.PI/6);
        ctx.translate(-50,-50);
        ctx.fillRect(50,50,100,200);
        ctx.restore(); // 저장한 좌표계의 복원
        ctx.fillStyle = "rgb(0,0,250)";
        ctx.fillRect(50,50,5,5);
    }
</script>
</head>
<body onLoad="init();">
    <canvas id="canvas" width="400" height="300">
        이 브라우저에서는 HTML5의 canvas 요소가 지원되지 않습니다.
    </canvas>
</body>
</html>
```

Translate
평행이동



공의 움직임(1)

```
<body onLoad="init();">
    <canvas id="canvas" width="600" height="400">
        이 브라우저에서는 HTML5의 canvas 요소가 지원되지
않습니다.
    </canvas>
    <br/>
    <form name="f" id="f" onSubmit="return fire();">
        속도, 각도를 지정하고 대포를 발사<br/>
        대포에서 발사되는 속도 <input name="vo" id="vo"
value="10" type="number" min="-100" max="100" />
        <br>
        각도 <input name="ang" id="ang" value="0"
type="number" min="0" max="80"/>
        <input type="submit" value="발사"/>
    </form>
</body>
```

```
function init(){
        ctx =
document.getElementById('canvas').getCo
ntext('2d');
        drawall();
    }
```

공의 움직임(2)

```
function drawall() {
    ctx.clearRect(0,0,cwidth,cheight);
    var i;
    for (i=0;i<everything.length;i++) {
        var ob = everything[i];
        if (ob[1]) {  // 평행이동과 회전에 필요
            ctx.save();
            ctx.translate(ob[3],ob[4]);
            ctx.rotate(ob[2]);
            ctx.translate(-ob[3],-ob[4]);
            ob[0].draw();
            ctx.restore(); }
        else {
          ob[0].draw();
        }
    }
```

공의 움직임(3)

```
function fire() {
        var angle = Number(document.f.ang.value);
        var outofcannon = Number(document.f.vo.value);
        var angleradians = angle*Math.PI/180;

        horvelocity =  outofcannon*Math.cos(angleradians);
        verticalvel1 = - outofcannon*Math.sin(angleradians);

        everything[cannonindex][2]= - angleradians;
        cball.sx = cannonx + cannonlength*Math.cos(angleradians);
        cball.sy = cannony+cannonht*.5 - cannonlength*Math.sin(angleradians);

        drawall();

        tid = setInterval(change,100);
        return false;
    }
```

공의 움직임(4)

```
function change() {
        var dx = horvelocity;
        verticalvel2 = verticalvel1 + gravity;
        var dy = (verticalvel1 + verticalvel2)*.5;
        verticalvel1 = verticalvel2;
        cball.moveit(dx,dy);
        // 목표물에 닿는지 검사
        var bx = cball.sx;
        var by = cball.sy;
        if ((bx>=target.sx)&&(bx<=(target.sx+target.swidth))&&
        (by>=target.sy)&&(by<=(target.sy+target.sheight))) {
            clearInterval(tid);
            // target을 제거하고 htarget을 삽입
            everything.splice(targetindex,1,[htarget,false]);
            everything.splice(ballindex,1);
            drawall();
        }
        // 공이 지면 영역을 벗어났는지 검사
        if (by>=ground.sy) {  clearInterval(tid); }
        drawall();
    }
```

대포 발사 결과

속도, 각도를 지정하고 대포를 발사
대포에서 발사되는 속도 40
각도 30 발사

슬링샷

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

슬링샷

var everything = [];

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

```
var target = new Picture(700,210,209,179,chicken);
var ground = new myrectangle(0,370,1200,30,"rgb(10,250,0)");
var cball = new Ball(startrockx,startrocky,ballrad,"rgb(250,0,0)");
var mysling= new Sling(startrockx,startrocky,startrockx+80,startrocky-
10,startrockx+80,startrocky+10,startrockx+70,startrocky+180,"rgb(120,20,10)");

everything.push(target);
everything.push(ground);
everything.push(mysling);
everything.push(cball);
```

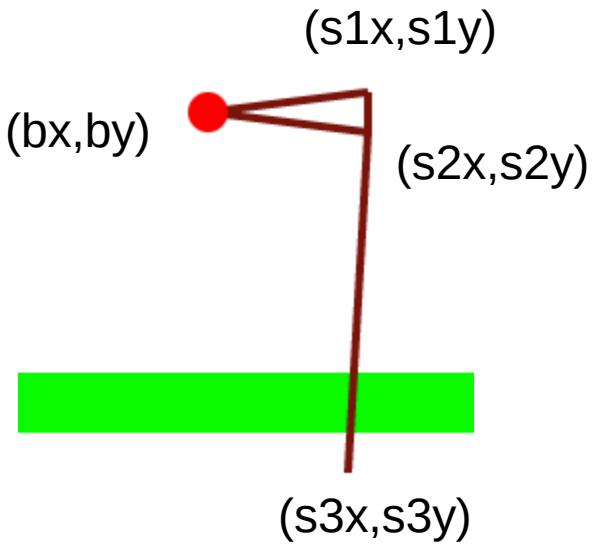
슬링샷

var everything = [];

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

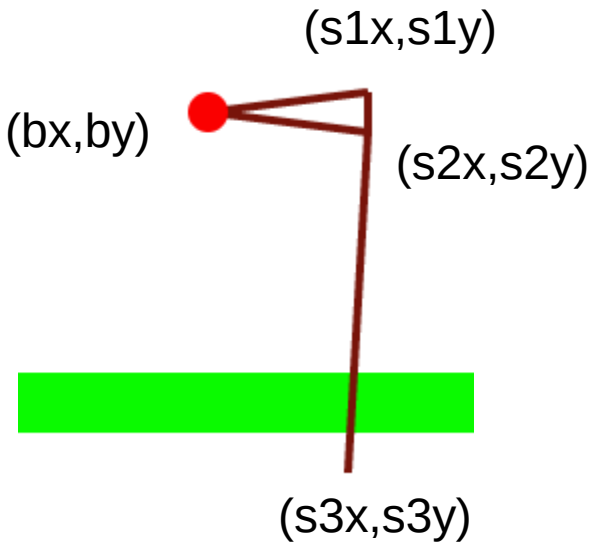
```
function drawall() {
    // drawall 함수는 캔버스 전체를 지운 후, everything 배열의 모든 원소를 그림
    ctx.clearRect(0,0,cwidth,cheight);
    var i;
    for (i=0;i<everything.length;i++) {
        everything[i].draw();
    }
}
```

새총 그리기(1)




```
function
Sling(bx,by,s1x,s1y,s2x,s2y,s3x,s3y,stylestring) {
        this.bx = bx;
        this.by = by;
        this.s1x = s1x;
        this.s1y = s1y;
        this.s2x = s2x;
        this.s2y = s2y;
        this.s3x = s3x;
        this.s3y = s3y;
        this.strokeStyle = stylestring;
        this.draw = drawsling;
        this.moveit = movesling;
    }
```

새총 그리기(2)




```
function drawsling() {
        ctx.strokeStyle = this.strokeStyle;
        ctx.lineWidth = 4;
        ctx.beginPath();

        ctx.moveTo(this.bx,this.by);
        ctx.lineTo(this.s1x,this.s1y);

        ctx.moveTo(this.bx,this.by);
        ctx.lineTo(this.s2x,this.s2y);

        ctx.moveTo(this.s1x,this.s1y);
        ctx.lineTo(this.s2x,this.s2y);
        ctx.lineTo(this.s3x,this.s3y);
        ctx.stroke();
    }
```

마우스의 작동

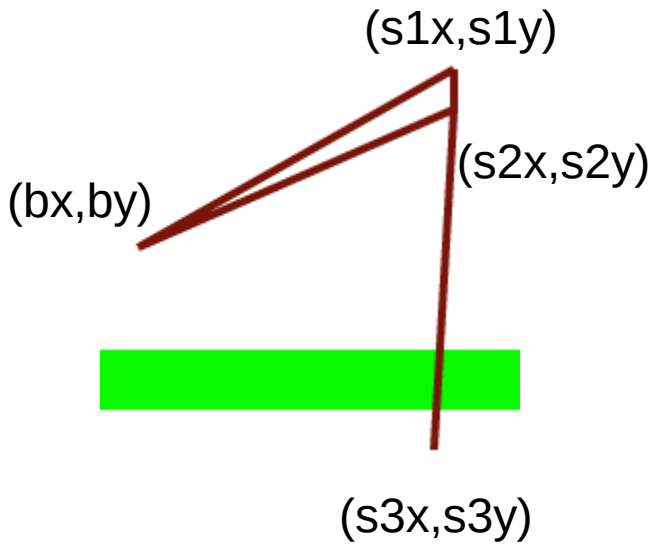
```
function init(){
        ctx = document.getElementById('canvas').getContext('2d');
        canvas1 = document.getElementById('canvas');
        canvas1.addEventListener('mousedown',findball,false);
        canvas1.addEventListener('mousemove',moveit,false);
        canvas1.addEventListener('mouseup',finish,false);
        // 처음 그리기
        drawall();
    }
```

```
<body onLoad="init();">
    <canvas id="canvas" width="1200" height="600">
        이 브라우저에서는 HTML5의 canvas 요소가 지원되지 않습니다.
    </canvas>
    <br/>
        마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다.
        새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침
하세요.
</body>
```

마우스에 의한 볼의 선택

```
function findball(ev) {
        var mx; var my;
        if ( ev.layerX ||  ev.layerX == 0) { // 파이어폭스, 크롬
            mx= ev.layerX;
            my = ev.layerY;
        } else if (ev.offsetX || ev.offsetX == 0) { // 오페라
            mx = ev.offsetX;
            my = ev.offsetY;
        }
        if (distsq(mx,my, cball.sx,cball.sy)<ballradsq) {
            inmotion = true;
            drawall();
        }
    }
    // 거리의 세제곱을 계산에 이용
    function distsq(x1,y1,x2,y2) {
        return (x1-x2)*(x1-x2)+(y1-y2)*(y1-y2);
    }
```

마우스에 의한 선택한 볼의 움직임




```
// 탄알 드래그와 mysling 변경을 위한 함수
   function moveit(ev) {
      var mx;
      var my;
      if (inmotion) {
         if ( ev.layerX ||  ev.layerX == 0) { // 파이어폭스
            mx= ev.layerX; my = ev.layerY;
         } else if (ev.offsetX || ev.offsetX == 0) { // 오페라
            mx = ev.offsetX; my = ev.offsetY;
         }
         cball.sx = mx;
         cball.sy = my;
         mysling.bx = mx;
         mysling.by = my;
         drawall();
      }
   }
```

마우스에 의해 결정된 볼의 속도

```
function finish(ev) {
    // 마우스에서 손을 뗄 때, 탄알과 mysling이 드래그돼 있다면,
    // 탄알이 포물선을 그리며 운동하게끔 지정
    if (inmotion) {
        inmotion = false;
        // 최초 속도를 길이만큼 증가시켜야 함, 편의상 정사각형으로 만듬
        // 700은 그럴싸한 호가 형성되도록 임의로 정한 값입니다.
        var outofcannon = distsq(mysling.bx,mysling.by,mysling.s1x,mysling.s1y)/700;

        // 선 간격 bx, by, s1x, s1y, 새총의 상박을 토대로 구한 각도를 이용
        var angleradians = -Math.atan2(mysling.s1y-mysling.by,mysling.s1x-mysling.bx);
        horvelocity =  outofcannon*Math.cos(angleradians);
        verticalvel1 = - outofcannon*Math.sin(angleradians);

        drawall();
        tid = setInterval(change,100);
    }
}
```

공의 애니메이션

```
function change() {
    // 이 함수는 새총으로부터 목표물이나 지면까지의 탄알의 움직임을 형성함
    var dx = horvelocity; verticalvel2 = verticalvel1 + gravity;
    var dy = (verticalvel1 + verticalvel2)*.5; verticalvel1 = verticalvel2;
    cball.moveit(dx,dy);

    // 목표물에 닿았는지 검사
    var bx = cball.sx; var by = cball.sy;
    // 목표물의 내부를 검사
    // 타격 영역을 좁히기 위해 경계를 40씩 조정
    if ((bx>=target.sx+40)&&(bx<=(target.sx+target.swidth-40))&&
    (by>=target.sy+40)&&(by<=(target.sy+target.sheight-40))) {
        // clearInterval(tid);
        // 기존 이미지를 깃털 이미지로 대체
        target.img = feathers;
    }

    // 지면 영역을 벗어났는지 검사지면 영역을 벗어났는지 검사
    if (by>=ground.sy) { clearInterval(tid); }
    drawall();
}
```

슬링샷 결과

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

마우스를 누른 채 탄알을 드래그하세요. 마우스 버튼을 놓으면 새총이 발사됩니다. 새총은 최종 위치에 그대로 있게 됩니다. 게임을 다시 하려면 페이지를 새로고침 하세요.

표시항목 변경을 위한 배열의 함수

- splice
  - 원소를 몇 개든 제거할 수도 있고 제거한 후 삽입할 수 있음

```
everything.splice(targetindex, 1, [htarget, false]);
everything.splice(ballindex, 1);
```