미로

류관희

- 미로 만들고 이동하는 프로그래밍 기법
  - 플레이어가 직접 미로 만들어 저장
  - 벽의 충돌 검사
- 미로 프로그램
  - 캔버스(객체, 미로의 벽)
  - 토큰(오각형)과 벽의 충돌 검사
  - 미로의 벽 그리기
    - 마우스 이용
  - 토큰의 이동(화살표를 이용)

- 1차버전 미로
  - 모든 기능을 하나의 파일
- 2차버전 미로
  - 미로 제작하는 코드 파일
  - 플레이어가 라디오 버튼으로 미로를 선택할 수 있는 코드 파일

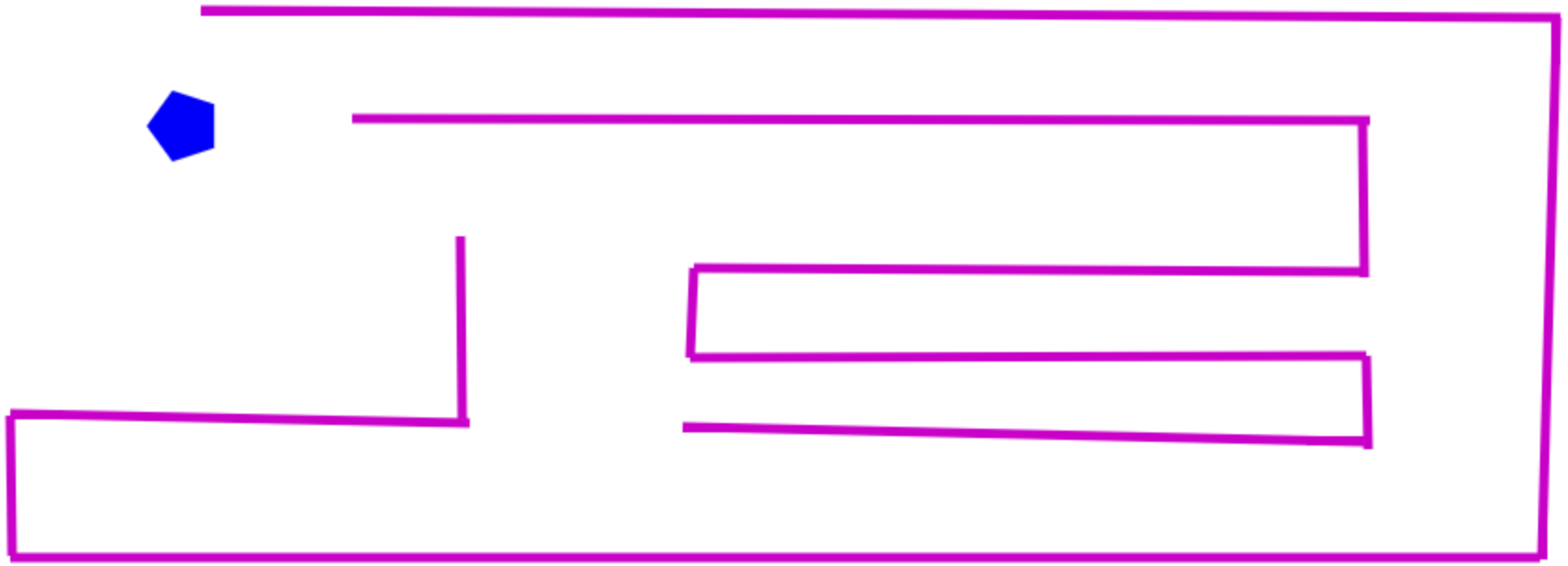
- 미로 구조 제작
- 화살표 키로 토큰 움직이기
- 충돌 검사
- 인코딩
- 저장
- 사용자 컴퓨터에서 데이터 불러오기

미로게임의 시작화면

마우스 버튼을 누르고 드래그한 후 손을 떼면 미로를 그릴 수 있습니다. 화살표를 움직이면 오각형이 이동합니다.
다른키를 누르면 키 감기를 종료하고 컴퓨터에 미로를 저장합니다.
미로를 저장할 때는 이름을 입력하고 '미로 저장' 버튼을 누르세요.
이름: maze_name 미로 저장

이름을 입력하고 '미로 불러오기' 버튼을 누르세요면 미리 저장해둔 미로가 현재 그린 미로에 추가됩니다.
이름: maze_name 미로 불러오기

미로게임 미로벽



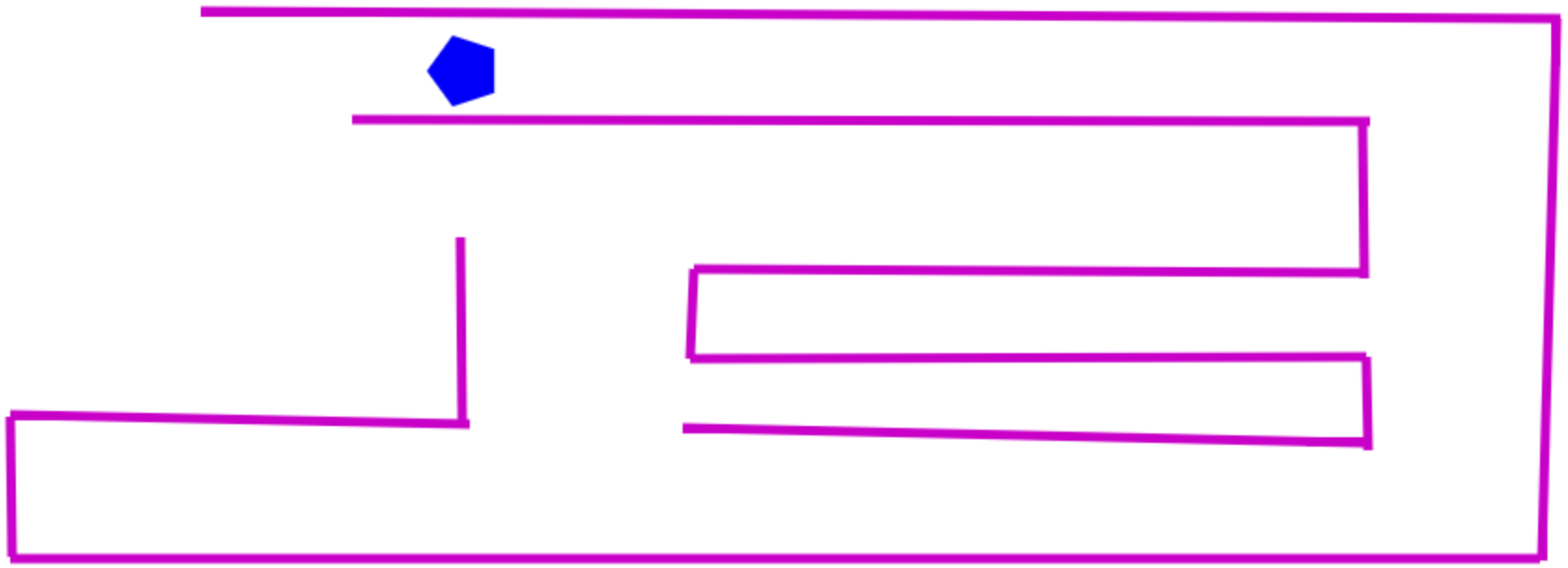
마우스 버튼을 누르고 드래그한 후 손을 떼면 미로를 그릴 수 있습니다. 화살표를 움직이면 오각형이 이동합니다.
다른키를 누르면 키 감기를 종료하고 컴퓨터에 미로를 저장합니다.
미로를 저장할 때는 이름을 입력하고 '미로 저장' 버튼을 누르세요.

이름: maze_name 미로 저장

이름을 입력하고 '미로 불러오기' 버튼을 누르세요면 미리 저장해둔 미로가 현재 그린 미로에 추가됩니다.

이름: maze_name 미로 불러오기

미로 게임 실행 과정



마우스 버튼을 누르고 드래그한 후 손을 떼면 미로를 그릴 수 있습니다. 화살표를 움직이면 오각형이 이동합니다.
다른키를 누르면 키 감기를 종료하고 컴퓨터에 미로를 저장합니다.
미로를 저장할 때는 이름을 입력하고 '미로 저장' 버튼을 누르세요.
이름: maze_name 미로 저장

이름을 입력하고 '미로 불러오기' 버튼을 누르세요면 미리 저장해둔 미로가 현재 그린 미로에 추가됩니다.
이름: maze_name 미로 불러오기

플레이어가 난이도 조절하여 게임

미로를 불러오려면 수준을 선택하고 '미로 불러오기' 버튼을 클릭하세요.

- 어려움
- 보통
- 쉬움

미로 불러오기

토큰을 이동하려면 화살표 키를 누르세요.

사전 준비 사항

- 미로 프로그램 구성요소
  - 다양한 미로 벽 제공
  - 토큰의 이동 (화살표 키이용)
  - 토큰과 벽의 충돌 검사
- 미로 벽 정보
  - 인코딩하여 저장
  - 벽(시작점과 끝점)
  - 로컬 저장소(localStorage)
  - 미로의 선택(라디오 버튼)

HTML5, CSS, 자바스크립트 기능

- HTML5 구조
  - HTML 구조
  - 사용자 정의 객체를 비롯한 사용자 정의 함수
  - 마우스를 사용하여 CANVAS 요소에 벽을 나타내는 선분을 생성하는 패스 그리기
  - 토큰의 이동을 화살표 키로 사용하기
  - 사용자가 만든 미로벽을 국부저장소에 저장하기
  - 사용자 객체
  - 배열

벽 표시하기

```
  var everything = [];
  var curwall;
  var wallwidth = 5;
  var wallstyle = "rgb(200,0,200)";
  var walls = [];


function Wall(sx,sy,fx,fy,width,stylestring) {
      this.sx = sx;
      this.sy = sy;
      this.fx = fx;
      this.fy = fy;
      this.width = width;
      this.draw = drawAline;
      this.strokestyle = stylestring;
}
```

```
function drawAline() {
      ctx.lineWidth = this.width;
      ctx.strokeStyle = this.strokestyle;
      ctx.beginPath();
      ctx.moveTo(this.sx,this.sy);
      ctx.lineTo(this.fx,this.fy);
      ctx.stroke();
}
```

벽 그리기

- 마우스이용(왼쪽 버튼을 클릭하면 시작하고 이동하면 선분을 그리고 버튼을 놓으면 끝나기)

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

   canvas1 = document.getElementById('canvas');

   canvas1.addEventListener('mousedown',startwall,false);
   canvas1.addEventListener('mousemove',stretchwall,false);
   canvas1.addEventListener('mouseup',finish,false);

    drawall();
}
```

벽 그리기

```
function startwall(ev) {
	var mx;
	var my;
	if ( ev.layerX ||  ev.layerX == 0) {
	   mx= ev.layerX;
	    my = ev.layerY;
	  } else if (ev.offsetX || ev.offsetX == 0) {
	    mx = ev.offsetX;
	    my = ev.offsetY;
	  }
	curwall = new Wall(mx,my,mx+1,my+1,wallwidth,wallstyle);
	inmotion = true;
	everything.push(curwall);
	drawall();
}
```

벽 그리기

```
function stretchwall(ev) {
        if (inmotion) {
        var mx;
        var my;
        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;
          }
         curwall.fx = mx;
         curwall.fy = my;
         drawall();
        }
}
```

벽 그리기

```
function finish(ev) {
	inmotion = false;
	walls.push(curwall);
}


 function drawall() {
	 ctx.clearRect(0,0,cwidth,cheight);
	 var i;
	 for (i=0;i<everything.length;i++) {
		everything[i].draw();
	 }
 }
```

그린 벽을 국부저장소에 저장하기

- 서버에 그린 벽을 저장하는 방식
- 국부 저장소에 그린 벽을 저장하는 방식
  - 국부 컴퓨터에 작은 쿠키 파일에 사용자나 웹 사이트 관리자의 편의를 위해 ID 혹은 암호 등의 정보를 유지
  - HTML5 localStorage(브라우저별로 국한)

국부 저장소에 날짜 저장하기

- 날짜와 시간 정보 저장
  - Date 함수로 날짜/시간 정보 저장
  - 국부저장소: 문자열이 저장되는 특정한 이름의 데이터 베이스(키: 이름, 값: 문자열, 시스템: 키와값의 쌍)

- 날짜 저장 프로그램 시작화면

```
<body>
    <button onClick="javascript:store();">날짜 정보 저장</button>
    <button onClick="javascript:fetch();">날짜 정보 불러오기</button>
    <button onClick="javascript:remove();">날짜 정보 삭제</button>
</body>
```

```
function store() {
    if (typeof(localStorage) == "undefined") {
        alert("브라우저가 HTML 로컬 저장소를 인식하지 못 합니다.");
    }
    else {
        try {
            olddate = new Date();
            localStorage.setItem("lastdate",olddate);
            alert("저장됨: "+olddate);
        }
        Catch(e) {
            alert("로컬 저장소 사용 오류: "+e);}
        }
        Return false;
    }
```

```
function fetch() {
        if (typeof(localStorage) == "undefined") {
            alert("브라우저가 HTML 로컬 저장소를 인식하지 못 합니다.");
        }
        else {
            alert("저장됨 "+localStorage.getItem('lastdate'));
        }
        return false;
    }
function remove() {
        if (typeof(localStorage) == "undefined") {
            alert("브라우저가 HTML 로컬 저장소를 인식하지 못 합니다.");
        }
        else {
            localStorage.removeItem('lastdate');
            alert("저장된 날짜가 삭제되었습니다.");
        }
        return false;
    }
```

국부 저장소에 데이터저장하기

```
function savewalls() {
    // 단순히 처음과 끝 쌍을 저장
    // 긴 문자열로 변환해야 함
    var w = [];
    var allw=[];
    var sw;
    var onewall;
    var i;
    var lsname =
    document.sf.slname.value;
    for (i=0;i<walls.length;i++) {
        w.push(walls[i].sx);
        w.push(walls[i].sy);
        w.push(walls[i].fx);
        w.push(walls[i].fy);
        onewall = w.join("+");
        allw.push(onewall);
        w = [];
    }
    sw = allw.join(";");
    //alert(sw+"를 저장하시겠습니까?");
    try {
        localStorage.setItem(lsname,sw);
        alert("벽이 "+lsname+"이라는 이름으로
                    저장되었습니다.");
        }
        catch (e) {
            alert("데이터가 저장되지 않았습니다.
                    오류: "+e);
        }
        return false;
}
```

국부 저장소에서 데이터 불러오기

```
function getwalls() {
    var swalls;
    var sw;
    var i;
    var sx;
    var sy;
    var fx;
    var fy;
    var curwall;
    var lsname =
    document.gf.glname.value;
    swalls=localStorage.getItem(lsname);
    if (swalls!=null) {
      wallstgs = swalls.split(";");
      for (i=0;i<wallstgs.length;i++) {
          sw = wallstgs[i].split("+");
          sx = Number(sw[0]);
          sy = Number(sw[1]);
          fx = Number(sw[2]);
          fy = Number(sw[3]);
          curwall = new
          Wall(sx,sy,fx,fy,wallwidth,wallstyle);
          walls.push(curwall);
          everything.push(curwall);
          }
       drawall();
    } else { alert("데이터를 불러오지 못
    했습니다.");}

    window.addEventListener('keydown',getkey
    AndMove,false);
        return false;
    }
```

토큰 만들기

- 토큰: 오각형(n=5)

```
function Token(sx,sy,rad,stylestring,n) {
  this.sx = sx;
  this.sy = sy;
  this.rad = rad;
  this.draw = drawtoken;
  this.n = n;
  this.angle = (2*Math.PI)/n  // 괄호는 생략 가능
  this.moveit = movetoken;
  this.fillstyle = stylestring;
}
```

토큰 그리기

```
function drawtoken() {
        ctx.fillStyle=this.fillstyle;
        ctx.beginPath();
        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();
}
```

토큰 움직이기

- 화살표를 사용하여 움직임
- 화살표 키 감지하기

```
var mypent = new Token(100,100,20,"rgb(0,0,250)",5);
everything.push(mypent);
function init(){
   ctx = document.getElementById('canvas').getContext('2d');
   canvas1 = document.getElementById('canvas');
   canvas1.addEventListener('mousedown',startwall,false);
   canvas1.addEventListener('mousemove',stretchwall,false);
   canvas1.addEventListener('mouseup',finish,false);
   window.addEventListener('keydown',getkeyAndMove,false);
   drawall();
}
```

화살표 키로 토큰 움직이기

```
var mypent = new Token(100,100,20,"rgb(0,0,250)",5);
everything.push(mypent);


function getkeyAndMove(event) {
  var keyCode;
  if(event == null)
  {
    keyCode = window.event.keyCode;
    window.event.preventDefault();
  }
  else
  {
    keyCode = event.keyCode;
    event.preventDefault();
  }
```

```
switch(keyCode)
 {
     case 37:  // 왼쪽 화살표
              mypent.moveit(-unit,0);
     break;
   case 38:  // 위쪽 화살표
               mypent.moveit(0,-unit);
     break;
   case 39: // 오른쪽 화살표
               mypent.moveit(unit,0);
     break;
   case 40:  // 아래쪽 화살표
              mypent.moveit(0,unit);
     break;
default:
     window.removeEventListener('keydown',getkeyAndMove,false);
 }
drawall();
}
```

```
function movetoken(dx,dy) {
	this.sx +=dx;
	this.sy +=dy;
	var i;
	var wall;
	for(i=0;i<walls.length;i++) {
		wall = walls[i];
		if (intersect(wall.sx,wall.sy,wall.fx,wall.fy,this.sx,this.sy,this.rad)) {
			this.sx -=dx;
			this.sy -=dy;
			break;
		}
	}
}
```

토큰과 벽의 충돌 감지

- Intersect 함수
  - 주어진 중심과 반경의 원이 선분과 만나는지 검사
  - 만나면 TRUE, 그렇지 않으면 FALSE

직선: (sx, sy), (fx, fy)

직선의 방정식: x = sx+t*(fx-sx), y=sy+t*(fy-sy)

원: 중심(cx,cy), 반지름 rad

```
function intersect(sx,sy,fx,fy,cx,cy,rad) {
   var dx;
   var dy;
   var t;
   var rt;
       // t가 0.0에서 1.0으로 갈 때 선분을 매개변수로 나타낸 후
       // 각 점부터 cx,cy까지의 거리 제곱을 작성하고
       // 미분해서 결과가 0이 되는 t 값을 계산하여 최솟값을 구함
        dx = fx-sx; dy = fy-sy;
        t =0.0-((sx-cx)*dx+(sy-cy)*dy)/((dx*dx)+(dy*dy));
        if (t<0.0) { t=0.0; } else if (t>1.0) { t = 1.0;}

      // 이 t에서의 거리가 반경보다 작으면, 제곱을 비교
       dx = (sx+t*(fx-sx))-cx;
       dy = (sy +t*(fy-sy))-cy;
       rt = (dx*dx) +(dy*dy);
       if (rt<(rad*rad)) {
            return true; }
       else {
            return false;}
}
```

2차 버전

미로를 불러오려면 수준을 선택하고 '미로 불러오기' 버튼을 클릭하세요.

- 어려움
- 보통
- 쉬움

미로 불러오기

토큰을 이동하려면 화살표 키를 누르세요.

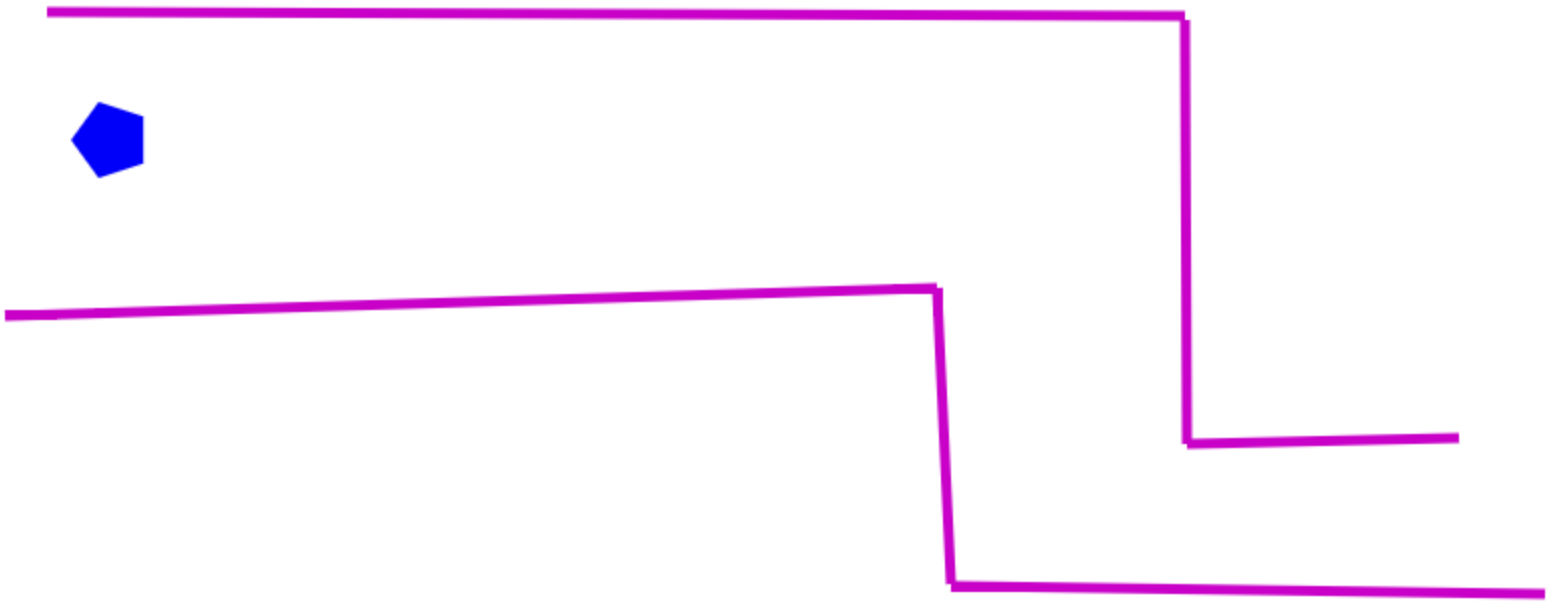
난이도 선택

```
<body onLoad="init();" >
<canvas id="canvas" width="900" height="350">
이 브라우저는 HTML5의 canvas 요소를 지원하지 않습니다.
</canvas>
<br/>

미로를 불러오려면 수준을 선택하고 '미로 불러오기' 버튼을 클릭하세요.

<form name="gf" onSubmit="return getwalls()" >
<br/>
<input type="radio" value="hard" name="level" />어려움<br/>
<input type="radio" value="moderate" name="level" />보통<br/>
<input type="radio" value="easy" name="level" />쉬움<br/>
<input type="submit" value="미로 불러오기"/><br/>
</form>
<p>
토큰을 이동하려면 화살표 키를 누르세요.
</p>
</body>
```

선택한 미로 벽 불러오기

```
function getwalls() {
        var swalls;
        var sw;
        var i;
        var sx;
        var sy;
        var fx;
        var fy;
        var curwall;
        var lsname;
        for (i=0;i<document.gf.level.length;i++) {
            if (document.gf.level[i].checked) {
                lsname= document.gf.level[i].value+"maze";
                break;
            }
        }
```

```
//alert("로컬 저장소 이름: "+lsname);
        swalls=localStorage.getItem(lsname);
        if (swalls!=null) {
            wallstgs = swalls.split(";");
            walls = [];  // walls 배열에서 벽 제거
            everything = [];
            everything.push(mypent);
            for (i=0;i<wallstgs.length;i++) {
                sw = wallstgs[i].split("+");
                sx = Number(sw[0]);
                sy = Number(sw[1]);
                fx = Number(sw[2]);
                fy = Number(sw[3]);
                curwall = new Wall(sx,sy,fx,fy,wallwidth,wallstyle);
                walls.push(curwall);
                everything.push(curwall);
            }
            drawall();
        } else {
            alert("데이터를 불러오지 못 했습니다.");
        }
        window.addEventListener('keydown',getkeyAndMove,false);
        return false;
    }
```

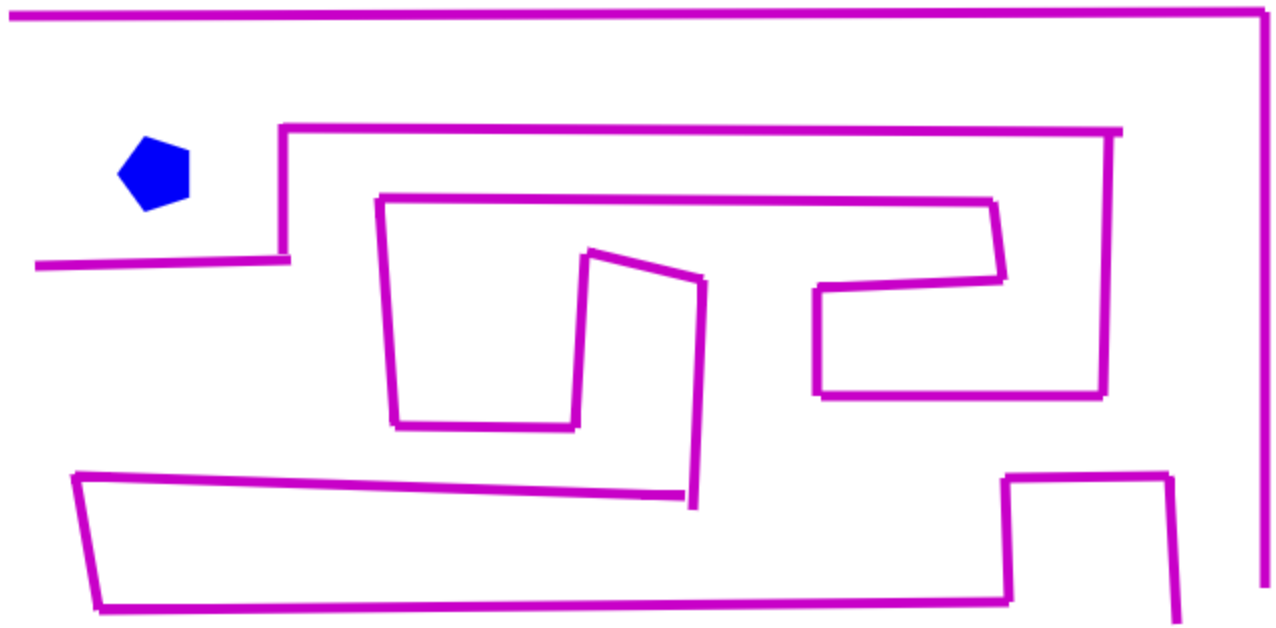


미로를 불러오려면 수준을 선택하고 '미로 불러오기' 버튼을 클릭하세요.

- 어려움
- 보통
- 쉬움

미로 불러오기

토큰을 이동하려면 화살표 키를 누르세요.



미로를 불러오려면 수준을 선택하고 '미로 불러오기' 버튼을 클릭하세요.

- (•) 어려움
- ( ) 보통
- ( ) 쉬움

[미로 불러오기]

토큰을 이동하려면 화살표 키를 누르세요.