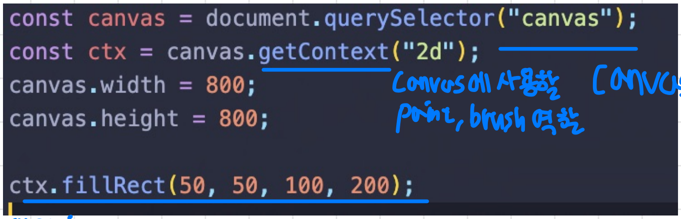
# Canvas API

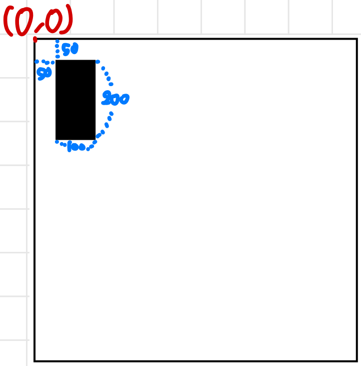


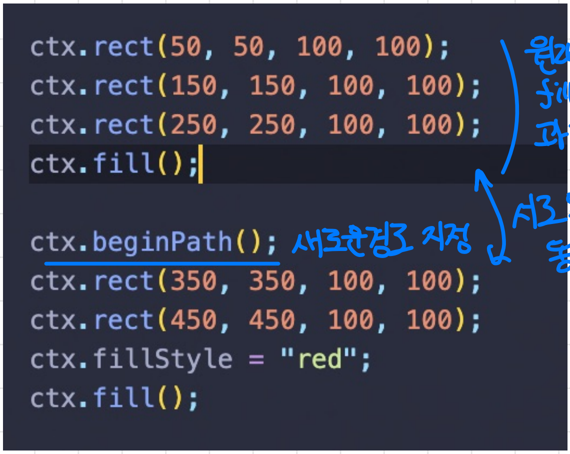
```
const canvas = document.querySelector("canvas");
const ctx = canvas.getContext("2d");
canvas.width = 800;
canvas.height = 800;

ctx.fillRect(50, 50, 100, 200);
```

canvas에 사용할 paint, brush 역할

canvas TAG 선택

채워진 사각형 만들기



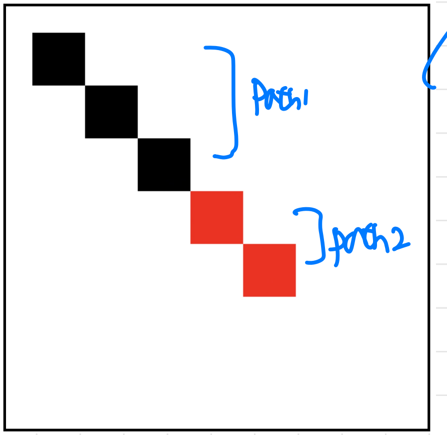


```
ctx.rect(50, 50, 100, 100);
ctx.rect(150, 150, 100, 100);
ctx.rect(250, 250, 100, 100);
ctx.fill();

ctx.beginPath();
ctx.rect(350, 350, 100, 100);
ctx.rect(450, 450, 100, 100);
ctx.fillStyle = "red";
ctx.fill();
```

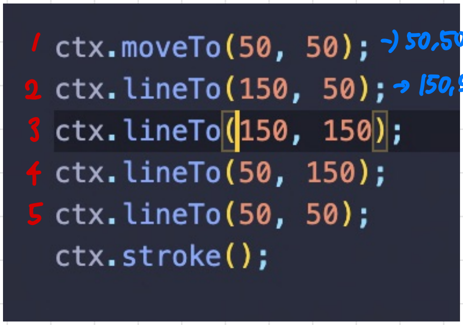
원래 fillrect의 과정

새로운경로 지정

서로 독립적으로 동작



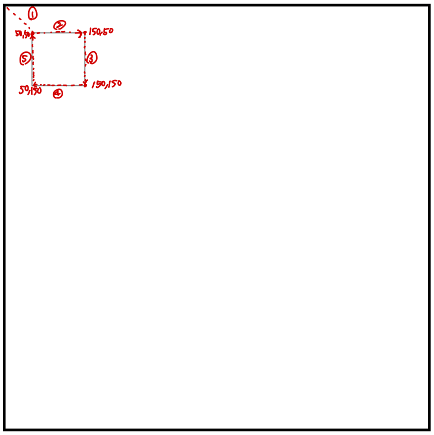
rect2.



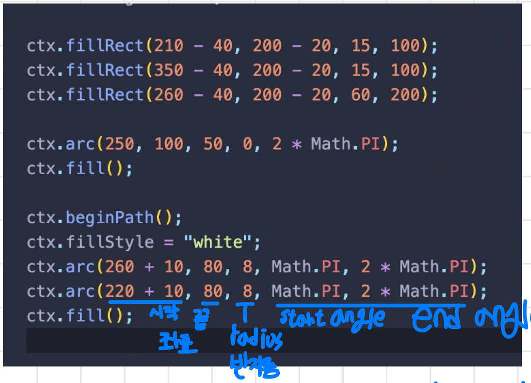
```
1 ctx.moveTo(50, 50);
2 ctx.lineTo(150, 50);
3 ctx.lineTo(150, 150);
4 ctx.lineTo(50, 150);
5 ctx.lineTo(50, 50);
  ctx.stroke();
```

-) 50,50 으로 좌표이동(시작점)

→ 150,50 까지 선긋기

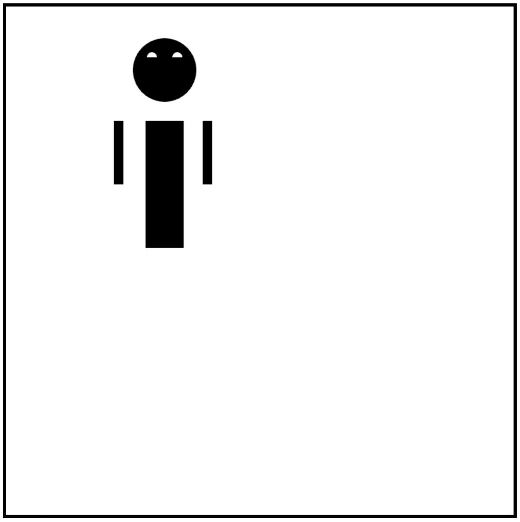


## Arc



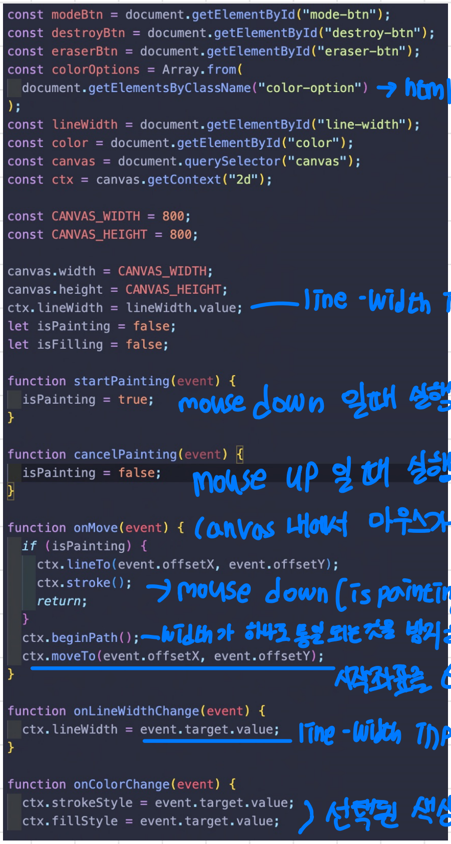
```
ctx.fillRect(210 - 40, 200 - 20, 15, 100);
ctx.fillRect(350 - 40, 200 - 20, 15, 100);
ctx.fillRect(260 - 40, 200 - 20, 60, 200);

ctx.arc(250, 100, 50, 0, 2 * Math.PI);
ctx.fill();

ctx.beginPath();
ctx.fillStyle = "white";
ctx.arc(260 + 10, 80, 8, Math.PI, 2 * Math.PI);
ctx.arc(220 + 10, 80, 8, Math.PI, 2 * Math.PI);
ctx.fill();
```

시작 좌표 끝 radius 반지름 start angle end angle



style을 바꾸기전 항상 경로지정을 어떻게 할지 생각.

## Painting Board



```
const modeBtn = document.getElementById("mode-btn");
const destroyBtn = document.getElementById("destroy-btn");
const eraserBtn = document.getElementById("eraser-btn");
const colorOptions = Array.from(
  document.getElementsByClassName("color-option")
);
const lineWidth = document.getElementById("line-width");
const color = document.getElementById("color");
const canvas = document.querySelector("canvas");
const ctx = canvas.getContext("2d");

const CANVAS_WIDTH = 800;
const CANVAS_HEIGHT = 800;

canvas.width = CANVAS_WIDTH;
canvas.height = CANVAS_HEIGHT;
ctx.lineWidth = lineWidth.value;
let isPainting = false;
let isFilling = false;

function startPainting(event) {
  isPainting = true;
}

function cancelPainting(event) {
  isPainting = false;
}

function onMove(event) {
  if (isPainting) {
    ctx.lineTo(event.offsetX, event.offsetY);
    ctx.stroke();
    return;
  }
  ctx.beginPath();
  ctx.moveTo(event.offsetX, event.offsetY);
}

function onLineWidthChange(event) {
  ctx.lineWidth = event.target.value;
}

function onColorChange(event) {
  ctx.strokeStyle = event.target.value;
  ctx.fillStyle = event.target.value;
```

→ html collection을 배열로 만듦 (for Each 사용하기 위해)

— line-width input의 value값받아 초기값으로 설정

mouse down 일때 실행

mouse up 일 때 실행

canvas 내에서 마우스가 move될때

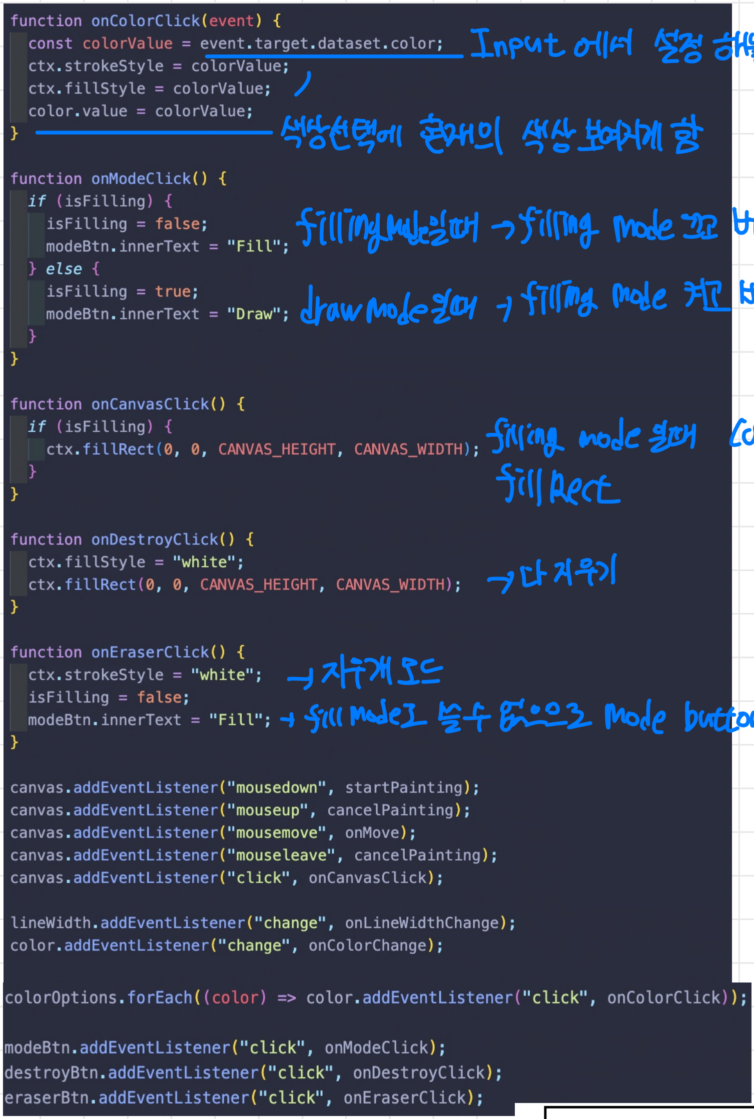
→ mouse down (is painting이 true 일때) move event의 x, y 좌표에 선긋기

— width가 하나로 통일 되는 것을 방지하기위해 new path 추가

시작좌표를 event의 x, y좌표로 설정 (ispainting이 false 일때 에만실행)

line-width input의 value값

) 선택된 색상 value



```
function onColorClick(event) {
  const colorValue = event.target.dataset.color;
  ctx.strokeStyle = colorValue;
  ctx.fillStyle = colorValue;
  color.value = colorValue;
}

function onModeClick() {
  if (isFilling) {
    isFilling = false;
    modeBtn.innerText = "Fill";
  } else {
    isFilling = true;
    modeBtn.innerText = "Draw";
  }
}

function onCanvasClick() {
  if (isFilling) {
    ctx.fillRect(0, 0, CANVAS_HEIGHT, CANVAS_WIDTH);
  }
}

function onDestroyClick() {
  ctx.fillStyle = "white";
  ctx.fillRect(0, 0, CANVAS_HEIGHT, CANVAS_WIDTH);
}

function onEraserClick() {
  ctx.strokeStyle = "white";
  isFilling = false;
  modeBtn.innerText = "Fill";
}

canvas.addEventListener("mousedown", startPainting);
canvas.addEventListener("mouseup", cancelPainting);
canvas.addEventListener("mousemove", onMove);
canvas.addEventListener("mouseleave", cancelPainting);
canvas.addEventListener("click", onCanvasClick);

lineWidth.addEventListener("change", onLineWidthChange);
color.addEventListener("change", onColorChange);

colorOptions.forEach((color) => color.addEventListener("click", onColorClick));

modeBtn.addEventListener("click", onModeClick);
destroyBtn.addEventListener("click", onDestroyClick);
eraserBtn.addEventListener("click", onEraserClick);
```

Input 에서 설정 해놓은 dataset 값(헥사코드)

색상선택에 현재의 색상 보여지게 함

filling mode일때 → filling mode 끄고 버튼 이름 바꾸기

draw mode일때 → filling mode 켜고 버튼이름 바꾸기

mode button 클릭시 실행

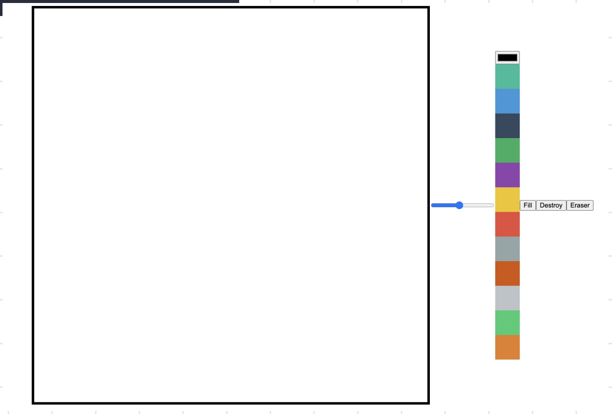
filling mode 일때 Canvas 클릭 시 캔버스 전체영역을 fill Rect

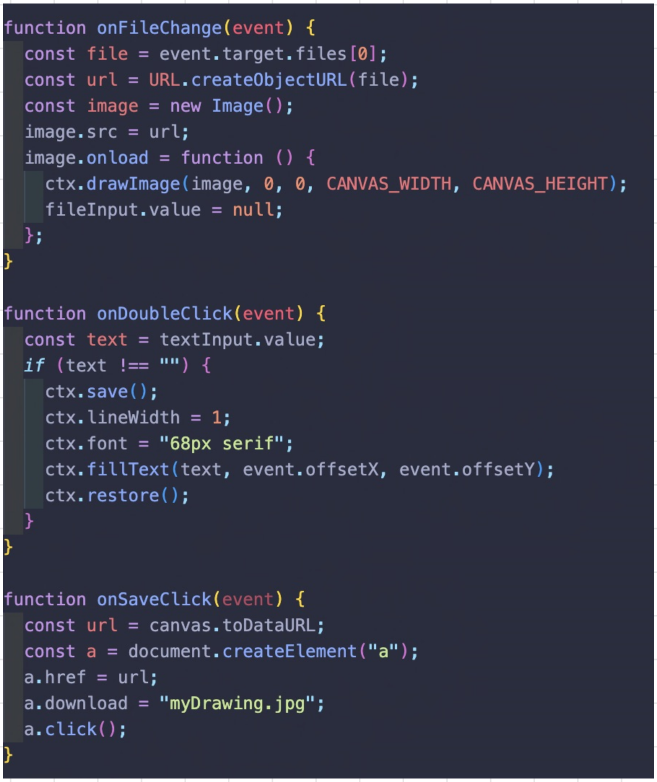
→ 다 지우기

→ 지우개 모드

→ fill mode로 쓸수 없으므로 mode button 명 변경

Color-options 의 자식요소들에





```
function onFileChange(event) {
  const file = event.target.files[0];
  const url = URL.createObjectURL(file);
  const image = new Image();
  image.src = url;
  image.onload = function () {
    ctx.drawImage(image, 0, 0, CANVAS_WIDTH, CANVAS_HEIGHT);
    fileInput.value = null;
  };
}

function onDoubleClick(event) {
  const text = textInput.value;
  if (text !== "") {
    ctx.save();
    ctx.lineWidth = 1;
    ctx.font = "68px serif";
    ctx.fillText(text, event.offsetX, event.offsetY);
    ctx.restore();
  }
}

function onSaveClick(event) {
  const url = canvas.toDataURL;
  const a = document.createElement("a");
  a.href = url;
  a.download = "myDrawing.jpg";
  a.click();
}
```