MainActivity

onCreate

```
findViewById(R.id.btnDraw).setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View view){
    Intent intent = new Intent(MainActivity.this, DrawActivity.class);
    startActivity(intent);
    }
});
```

By RicheyHans

DrawActivity

```
RadioGroup radioColor;    // radio group
FrameLayout stage;        // 그림이 그려질 영역의 별도 레이아웃
DrawView draw;
```

onCreate

```
radioColor = (RadioGroup) findViewById(R.id.radioColor);

radioColor.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(RadioGroup radioGroup, @IdRes int i) {
        switch(i){
            case R.id.radioBlack:
                draw.setColor(Color.BLACK);
                break;
            case R.id.radioCyan:
                draw.setColor(Color.CYAN);
                break;
            case R.id.radioYellow:
                draw.setColor(Color.YELLOW);
                break;
            case R.id.radioMagenta:
                draw.setColor(Color.MAGENTA);
                break;
        }
    }
});
```

By RicheyHans

```
drawView draw;
```

```
stage = (FrameLayout)findViewById(R.id.stage);
stage.addView(draw);
```

DrawView extends View

*DrawView는 View를 상속받는 일종의 **custom view**

```
Paint paint;                          // 색상, 폰트, 스타일, 그리기 모드 지정 객체
Path currentPath;                     // 도형의 궤적 정보 보유한 객체, 좌표 정보만 보유
List<PathTool> paths = new ArrayList<>();   // Path를 다수 활용하기 위해 Path를 상속하는
                                            임의 객체 PathTool의 리스트 생성
```

Constructor

```
public DrawView(Context context) {
    super(context);
    paint = new Paint();
    init();
}
```

By RicheyHans

*Paint 객체는 통상적으로 onDraw()내에서 호출하지 않고 필드나 생성자에서 호출해서 속도 및 메모리 관리를 한다.

init()

```
paint.setStyle(Paint.Style.STROKE);
paint.setStrokeWidth(5f);
setColor(Color.BLACK);
```

By RicheyHans

*Paint 객체인 paint의 속성들을 설정하는 메서드. 관리를 위해 별도 메서드 선언해 정의한 사항이며 paint 속성의 기본값으로 인식할 것.

setColor(int Color)

```
public void setColor(int color){
    PathTool tool = new PathTool();
    tool.setColor(color);
    currentPath = tool;
    paths.add(tool);
}
```

*라디오버튼에서 전달된 색상 인수를 받아, Path상속 객체인 PathTool의 객체 필드인 color에 전달

onTouchEvent(MotionEvent event)

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    switch(event.getAction()){
        case MotionEvent.ACTION_DOWN:
            // 최초 터치 시 해당 좌표로 이동(중간을 그리지 않는다)
            // 이전 점과 현재 점 사이를 그리지 않는다.
            currentPath.moveTo(event.getX(), event.getY());
            break;
        case MotionEvent.ACTION_MOVE:
            // 이전 점과 현재 점 사이를 그린다.
            // 터치가 일어나면 패스를 해당 좌표까지 선을 긋는다.
            currentPath.lineTo(event.getX(), event.getY());
            break;
        case MotionEvent.ACTION_UP:
            break;
    }
    invalidate();
    // 리턴이 false 일 경우는 touch 이벤트를 연속해서 발생시키지 않는다
    // 즉, 드래그를 하면 onTouchEvent가 재 호출되지 않는다.
    return true;
}
```

By RicheyHans

onDraw(Canvas canvas)

```
protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);

    for(PathTool tool:paths){
        paint.setColor(tool.getColor());
        canvas.drawPath(tool, paint);
    }
}
```

By RicheyHans

* Path 타입 List에 저장된 tool객체들을 터치 이벤트가 발생하면 받아와 tool 객체에 저장된 color를 읽고, tool을 paint 포맷으로 그려준다.

PathTool extends Path

```
class PathTool extends Path {
    int color;
    public PathTool(){  }
    public void setColor(int color){
        this.color = color;
    }
    public int getColor(){
        return this.color;
    }
}
```

By RicheyHans

*통상 Path 객체를 이용해 궤적 컨트롤을 하며, Path 객체는 Canvas 클래스의 drawPath 메서드로 출력이 가능하다.
*본 예제에서는 연속 선 그리기 동작 관리/색상 변경을 위해 임의의 자식 클래스를 생성해 사용한다.