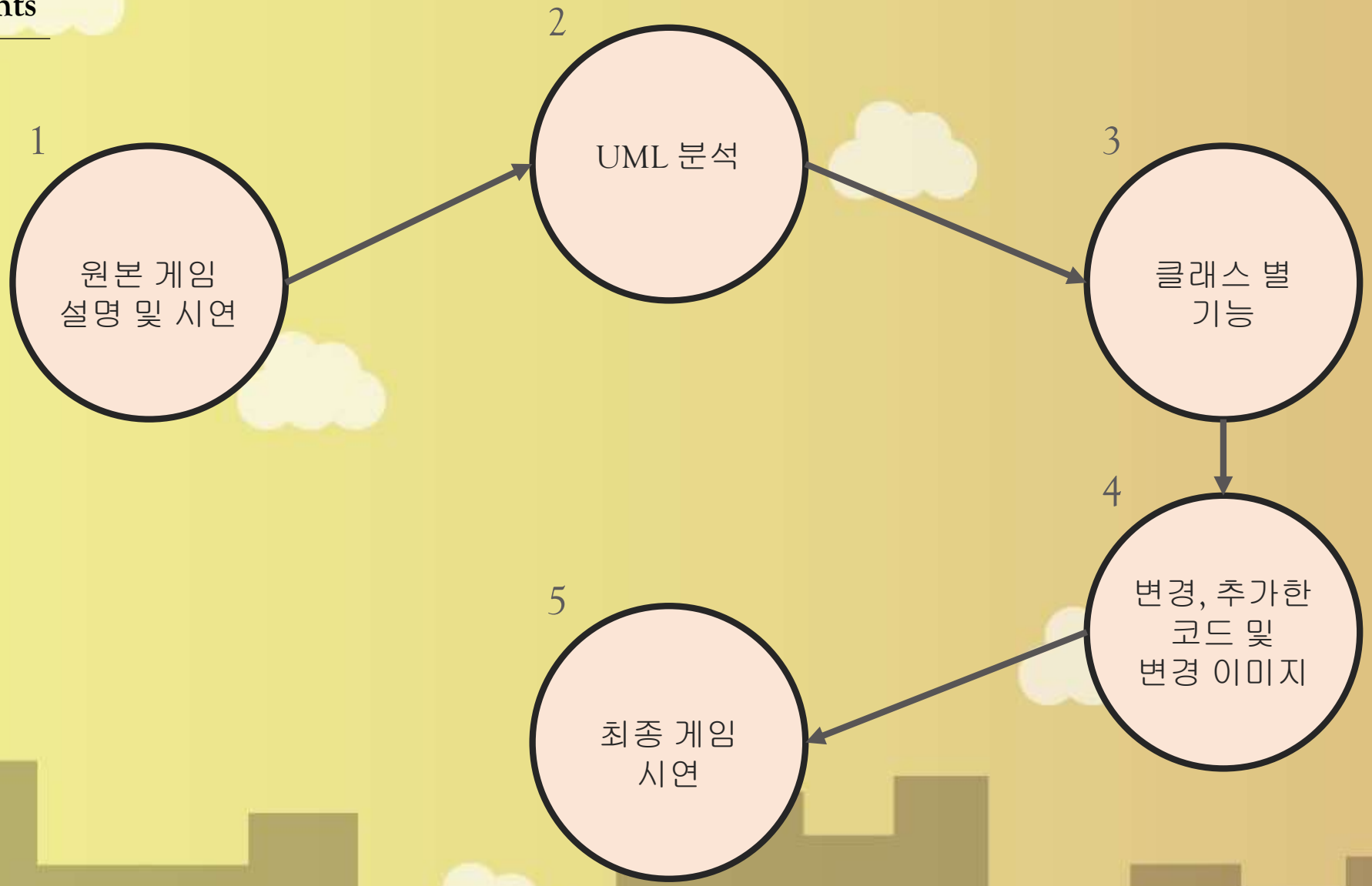

JUMPER

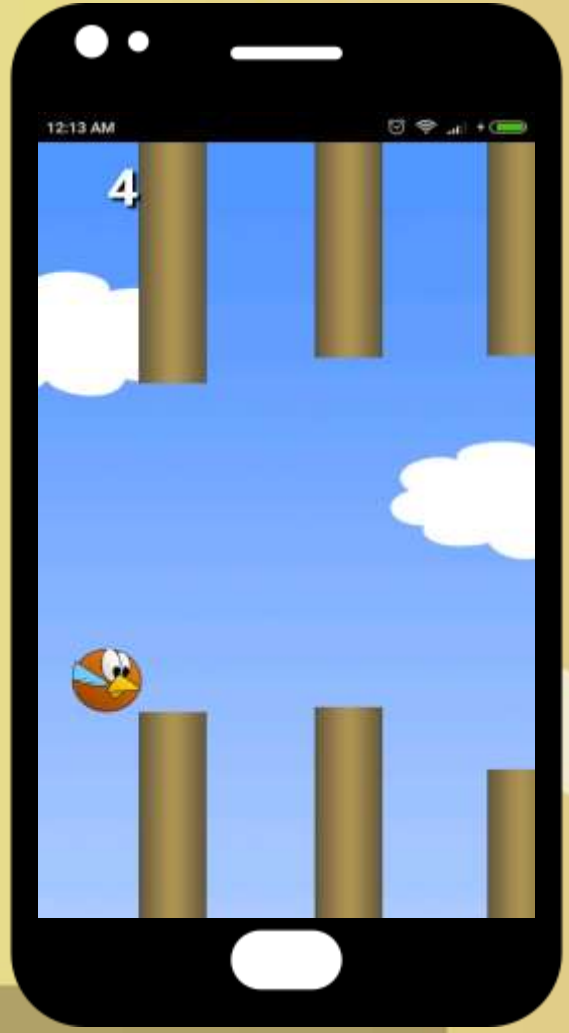
JUMPER

Contents



1) 원본 게임 설명 및 시연

1) 원본 게임 설명 및 시연



게임 설명

위, 아래로 장애물인 파이프들 존재

새 캐릭터로 파이프 사이를 통과해야 함

새 캐릭터는 화면을 터치 할 때마다 위로 튀어 오름

화면 터치를 하지 않고 있으면 캐릭터가 아래로 떨어짐

최대한 많은 기둥을 지나가야 함

장애물을 하나 지날 때마다 1점을 얻음

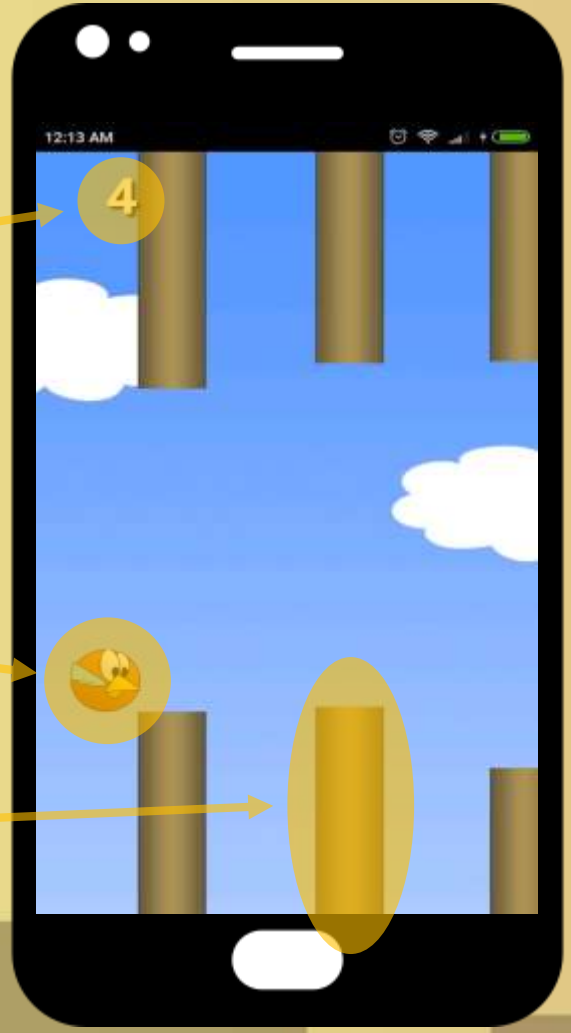
장애물에 새 캐릭터가 닿으면 게임 종료

1) 원본 게임 설명 및 시연

점수

캐릭터

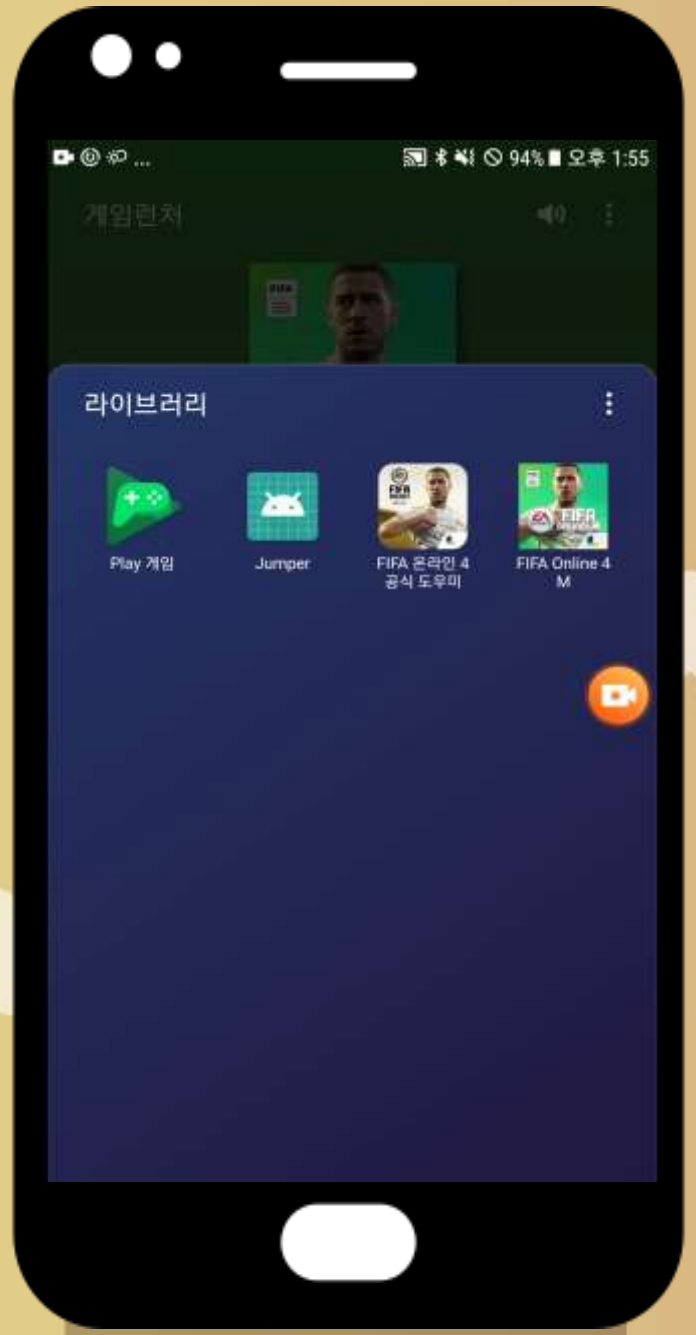
장애물



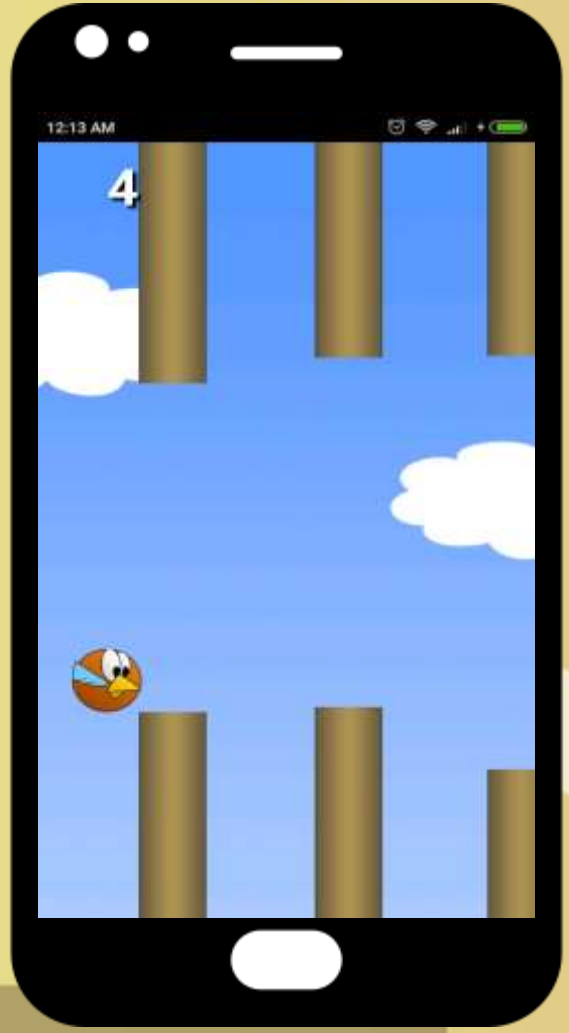
게임 실행 화면

게임 오버 화면

1) 원본 게임 설명 및 시연


94% 오후 1:55
게임런처
라이브러리
Play 게임
Jumper
FIFA 온라인 4 공식 도우미
FIFA Online 4 M

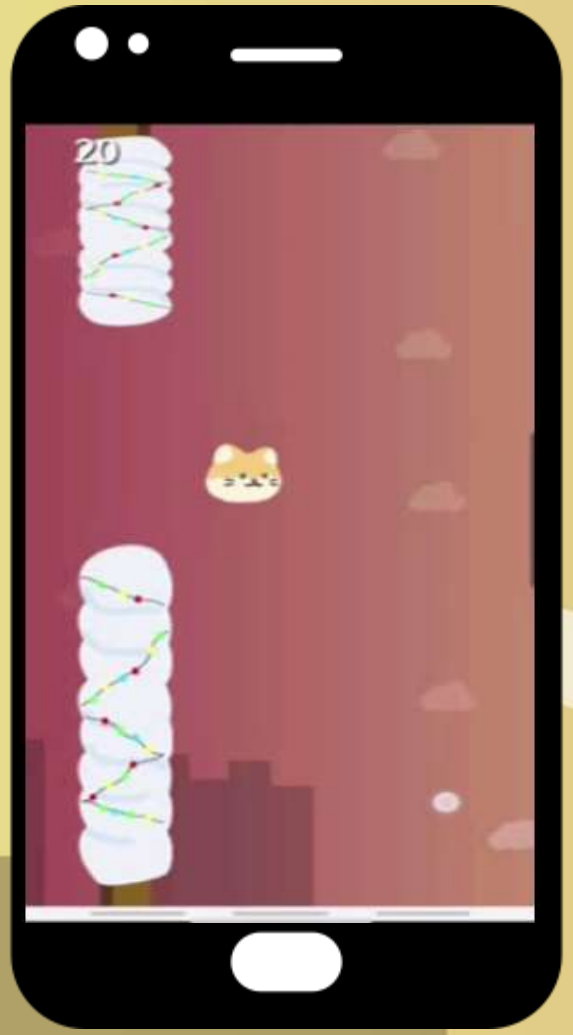
1) 원본 게임 설명 및 시연



원본 게임의 특징

1. 캐릭터의 속도가 매우 느림
2. 장애물인 나무 기둥이 고정되어 있고 단조로움
3. 장애물 사이의 상하 간격은 너무 넓고, 좌우 간격은 너무 좁음
4. 게임시작 버튼이 없음
5. 게임오버 시 재시작 버튼이 없음

1) 개발 게임 설명



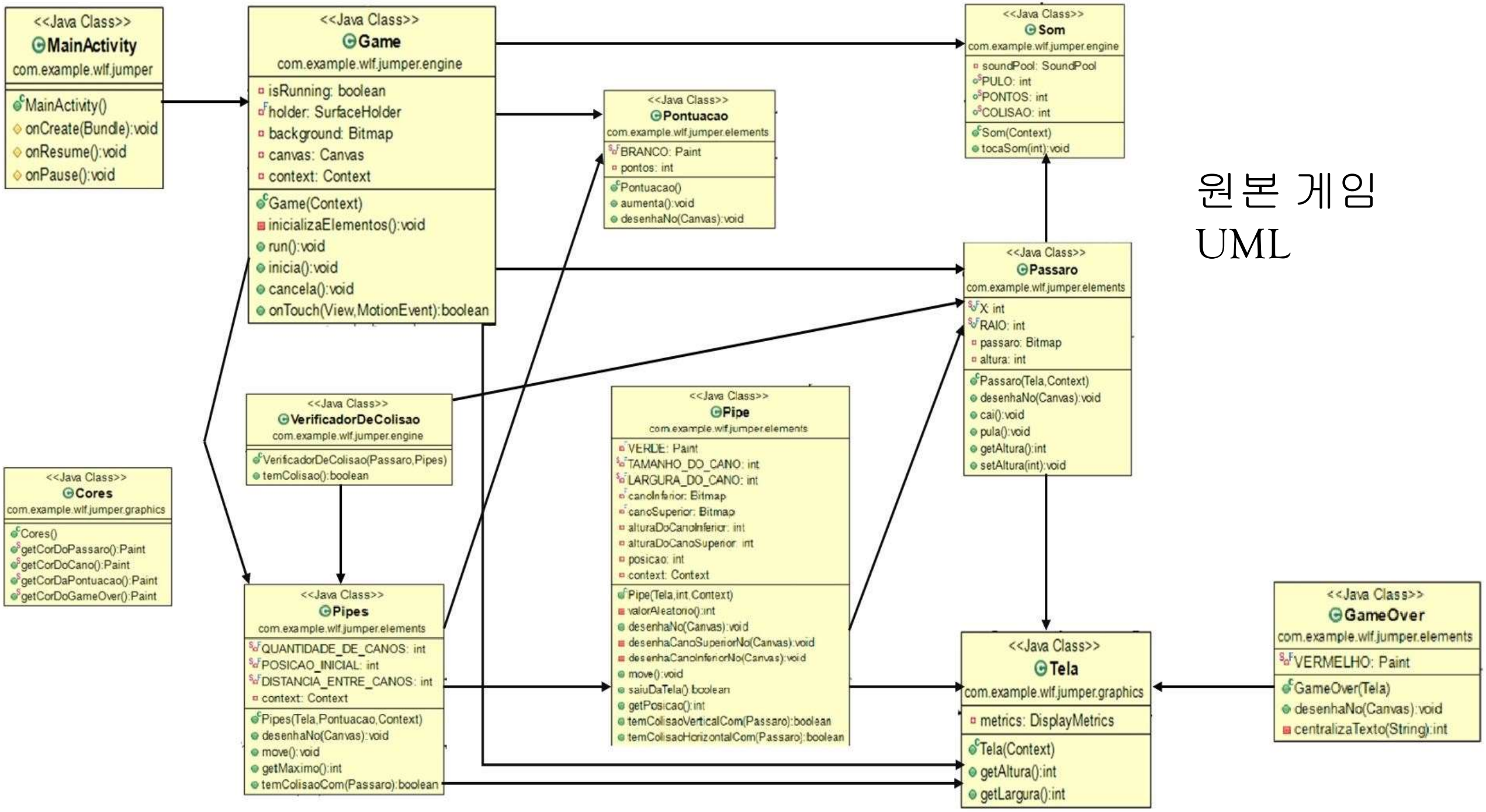
개발 게임의 변화

1. 캐릭터의 포물선 점프
2. 장애물들의 이동

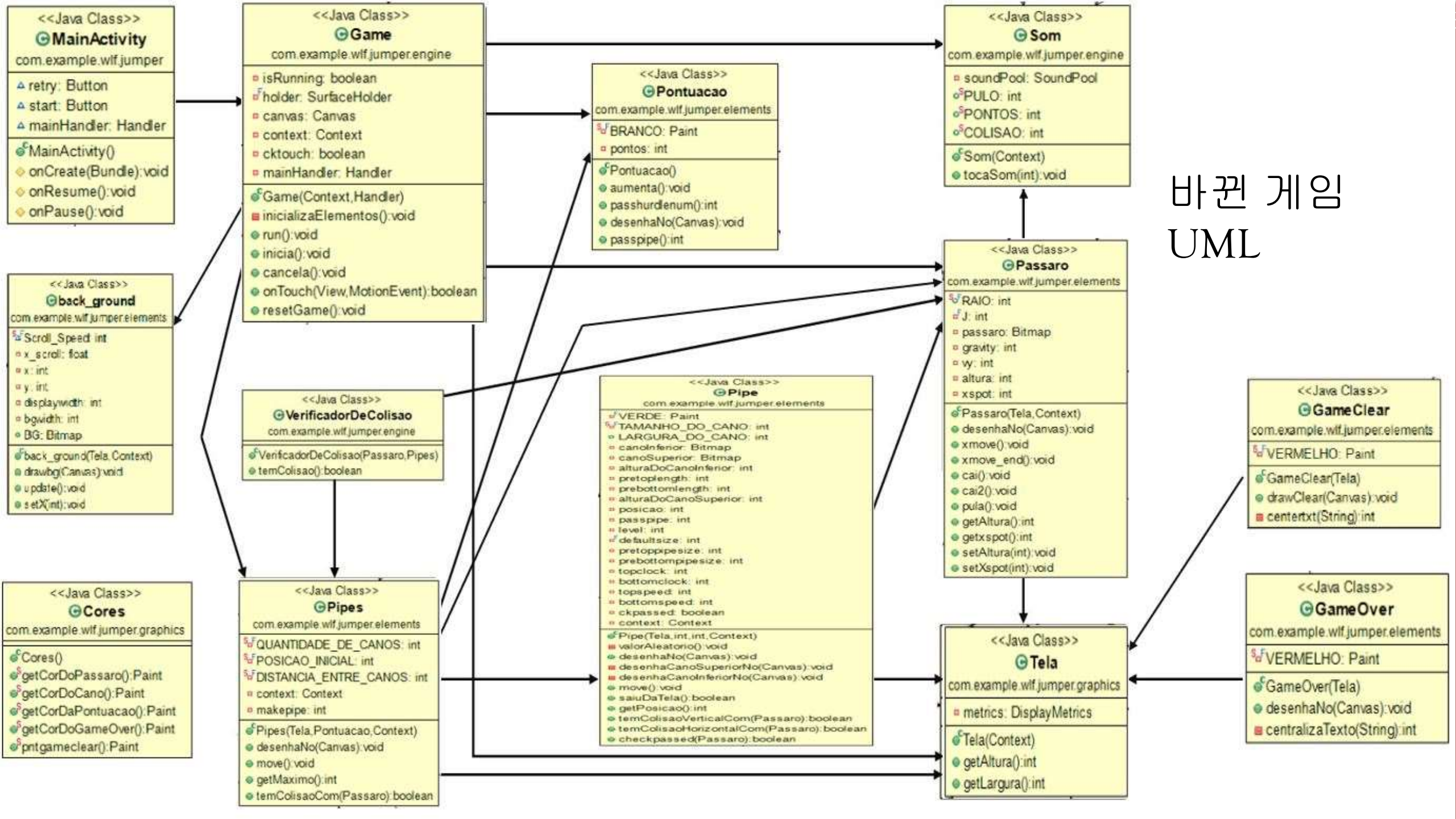
   -> 장애물을 총 40개로 이루어져 있으며 20개를 넘었을 시 장애물의 움직임이 추가.(늘어났다 줄어듬),
3. 배경의 흐름
4. 난이도 추가

   -> 장애물 5개를 넘을 때마다 통과할 수 있는 장애물 사이의 공간이 줄어들어 난이도의 개념 도입.

2) UML 분석


<<Java Class>>
MainActivity
com.example.wlf.jumper
MainActivity()
onCreate(Bundle):void
onResume():void
onPause():void
<<Java Class>>
Game
com.example.wlf.jumper.engine
isRunning: boolean
holder: SurfaceHolder
background: Bitmap
canvas: Canvas
context: Context
Game(Context)
inicializaElementos():void
run():void
inicia():void
cancela():void
onTouch(View,MotionEvent):boolean
<<Java Class>>
Som
com.example.wlf.jumper.engine
soundPool: SoundPool
PULO: int
PONTOS: int
COLISAO: int
Som(Context)
tocaSom(int):void
<<Java Class>>
Pontuacao
com.example.wlf.jumper.elements
BRANCO: Paint
pontos: int
Pontuacao()
aumenta():void
desenhaNo(Canvas):void
<<Java Class>>
Passaro
com.example.wlf.jumper.elements
X: int
RAIO: int
passaro: Bitmap
altura: int
Passaro(Tela,Context)
desenhaNo(Canvas):void
cai():void
pula():void
getAltura():int
setAltura(int):void
<<Java Class>>
VerificadorDeColisao
com.example.wlf.jumper.engine
VerificadorDeColisao(Passaro,Pipes)
temColisao():boolean
<<Java Class>>
Pipe
com.example.wlf.jumper.elements
VERDE: Paint
TAMANHO_DO_CANO: int
LARGURA_DO_CANO: int
canoInferior: Bitmap
canoSuperior: Bitmap
alturaDoCanoInferior: int
alturaDoCanoSuperior: int
posicao: int
context: Context
Pipe(Tela,int,Context)
valorAleatorio():int
desenhaNo(Canvas):void
desenhaCanoSuperiorNo(Canvas):void
desenhaCanoInferiorNo(Canvas):void
move():void
saiuDaTela():boolean
getPosicao():int
temColisaoVerticalCom(Passaro):boolean
temColisaoHorizontalCom(Passaro):boolean
<<Java Class>>
Cores
com.example.wlf.jumper.graphics
Cores()
getCorDoPassaro():Paint
getCorDoCano():Paint
getCorDaPontuacao():Paint
getCorDoGameOver():Paint
<<Java Class>>
Pipes
com.example.wlf.jumper.elements
QUANTIDADE_DE_CANOS: int
POSICAO_INICIAL: int
DISTANCIA_ENTRE_CANOS: int
context: Context
Pipes(Tela,Pontuacao,Context)
desenhaNo(Canvas):void
move():void
getMaximo():int
temColisaoCom(Passaro):boolean
<<Java Class>>
Tela
com.example.wlf.jumper.graphics
metrics: DisplayMetrics
Tela(Context)
getAltura():int
getLargura():int
<<Java Class>>
GameOver
com.example.wlf.jumper.elements
VERMELHO: Paint
GameOver(Tela)
desenhaNo(Canvas):void
centralizaTexto(String):int

원본 게임
UML


<<Java Class>>
MainActivity
com.example.wlf.jumper
retry: Button
start: Button
mainHandler: Handler
MainActivity()
onCreate(Bundle):void
onResume():void
onPause():void
<<Java Class>>
Game
com.example.wlf.jumper.engine
isRunning: boolean
holder: SurfaceHolder
canvas: Canvas
context: Context
cktouch: boolean
mainHandler: Handler
Game(Context,Handler)
inicializaElementos():void
run():void
inicia():void
cancela():void
onTouch(View,MotionEvent):boolean
resetGame():void
<<Java Class>>
Pontuacao
com.example.wlf.jumper.elements
BRANCO: Paint
pontos: int
Pontuacao()
aumenta():void
passhurdlenum():int
desenhaNo(Canvas):void
passpipe():int
<<Java Class>>
Som
com.example.wlf.jumper.engine
soundPool: SoundPool
PULO: int
PONTOS: int
COLISAO: int
Som(Context)
tocaSom(int):void
<<Java Class>>
back_ground
com.example.wlf.jumper.elements
Scroll_Speed: int
x_scroll: float
x: int
y: int
displaywidth: int
bgwidth: int
BG: Bitmap
back_ground(Tela,Context)
drawbg(Canvas):void
update():void
setX(int):void
<<Java Class>>
Passaro
com.example.wlf.jumper.elements
RAIO: int
J: int
passaro: Bitmap
gravity: int
vy: int
altura: int
xspot: int
Passaro(Tela,Context)
desenhaNo(Canvas):void
xmove():void
xmove_end():void
cai():void
cai2():void
pula():void
getAltura():int
getxspot():int
setAltura(int):void
setXspot(int):void
<<Java Class>>
VerificadorDeColisao
com.example.wlf.jumper.engine
VerificadorDeColisao(Passaro,Pipes)
temColisao():boolean
<<Java Class>>
Pipe
com.example.wlf.jumper.elements
VERDE: Paint
TAMANHO_DO_CANO: int
LARGURA_DO_CANO: int
canoInferior: Bitmap
canoSuperior: Bitmap
alturaDoCanoInferior: int
pretoplength: int
prebottomlength: int
alturaDoCanoSuperior: int
posicao: int
passpipe: int
level: int
defaultsize: int
pretoppipesize: int
prebottompipesize: int
topclock: int
bottomclock: int
topspeed: int
bottomspeed: int
ckpassed: boolean
context: Context
Pipe(Tela,int,int,Context)
valorAleatorio():void
desenhaNo(Canvas):void
desenhaCanoSuperiorNo(Canvas):void
desenhaCanoInferiorNo(Canvas):void
move():void
saiuDaTela():boolean
getPosicao():int
temColisaoVerticalCom(Passaro):boolean
temColisaoHorizontalCom(Passaro):boolean
checkpassed(Passaro):boolean
<<Java Class>>
GameClear
com.example.wlf.jumper.elements
VERMELHO: Paint
GameClear(Tela)
drawClear(Canvas):void
centertxt(String):int
<<Java Class>>
Cores
com.example.wlf.jumper.graphics
Cores()
getCorDoPassaro():Paint
getCorDoCano():Paint
getCorDaPontuacao():Paint
getCorDoGameOver():Paint
pntgameclear():Paint
<<Java Class>>
Pipes
com.example.wlf.jumper.elements
QUANTIDADE_DE_CANOS: int
POSICAO_INICIAL: int
DISTANCIA_ENTRE_CANOS: int
context: Context
makepipe: int
Pipes(Tela,Pontuacao,Context)
desenhaNo(Canvas):void
move():void
getMaximo():int
temColisaoCom(Passaro):boolean
<<Java Class>>
Tela
com.example.wlf.jumper.graphics
metrics: DisplayMetrics
Tela(Context)
getAltura():int
getLargura():int
<<Java Class>>
GameOver
com.example.wlf.jumper.elements
VERMELHO: Paint
GameOver(Tela)
desenhaNo(Canvas):void
centralizaTexto(String):int

바뀐 게임
UML

3) 클래스 별 기능

3) 클래스 별 기능

MainActivity

처음 실행되고 프레임을 생성

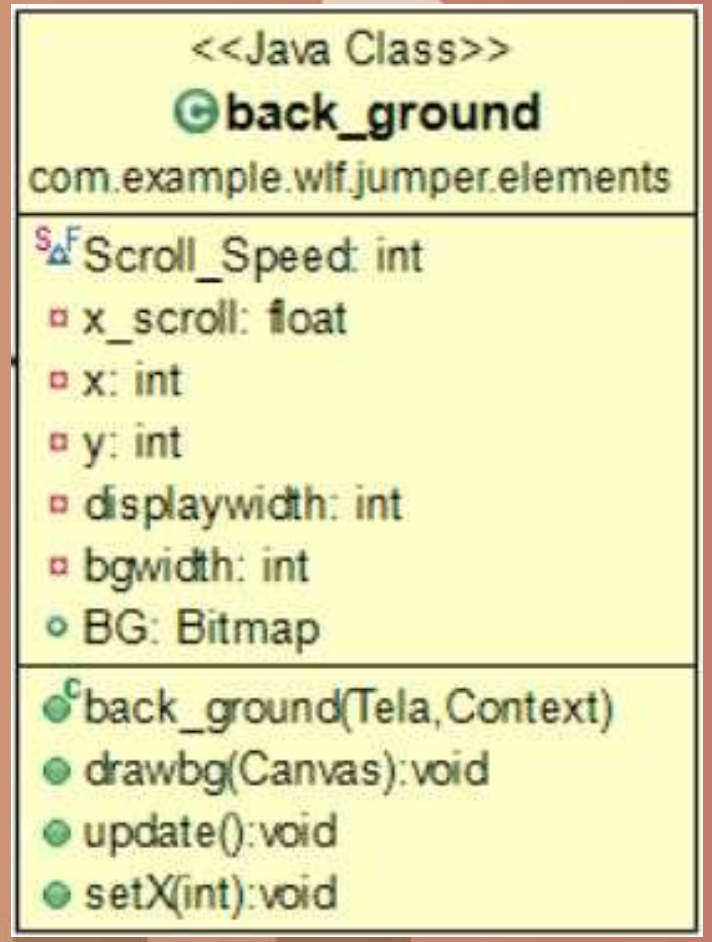
게임 클래스를 실행하는 클래스

3) 클래스 별 기능

Game

전반적인 게임에 필요한 클래스 생성, 게임 시작 화면부터, 캐릭터, 장애물 등 대부분의 클래스와 연관되어 게임 오버 조건까지 실행하는 클래스

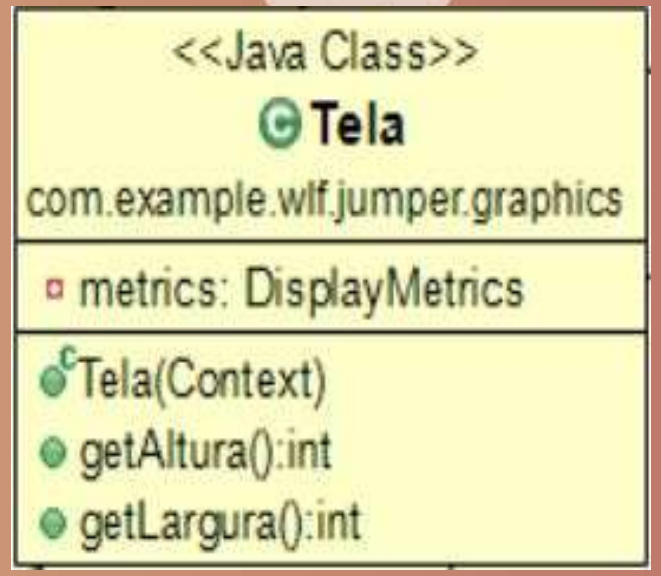
3) 클래스 별 기능



Back_ground

게임 배경에 전반적으로 관련된 클래스

3) 클래스 별 기능



Tela

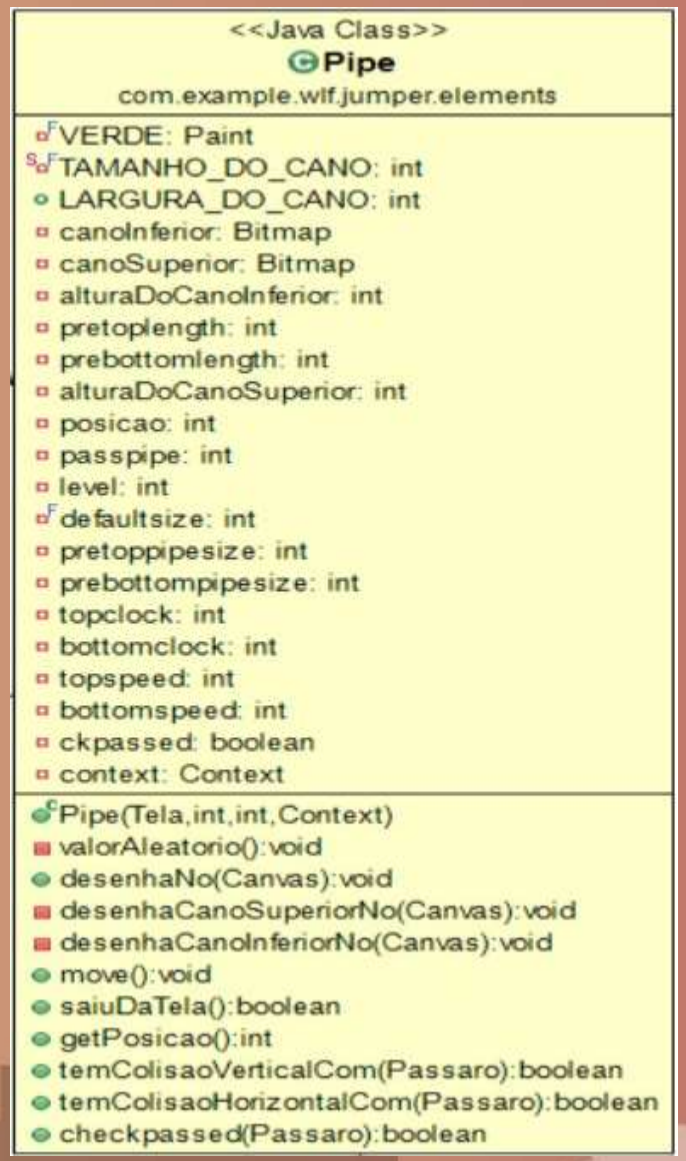
화면 크기에 관한 클래스

3) 클래스 별 기능

Passaro

캐릭터를 화면에 나타내고, 캐릭터의 이동, 화면 터치 시 점프까지 캐릭터의 모든 것에 관련된 클래스

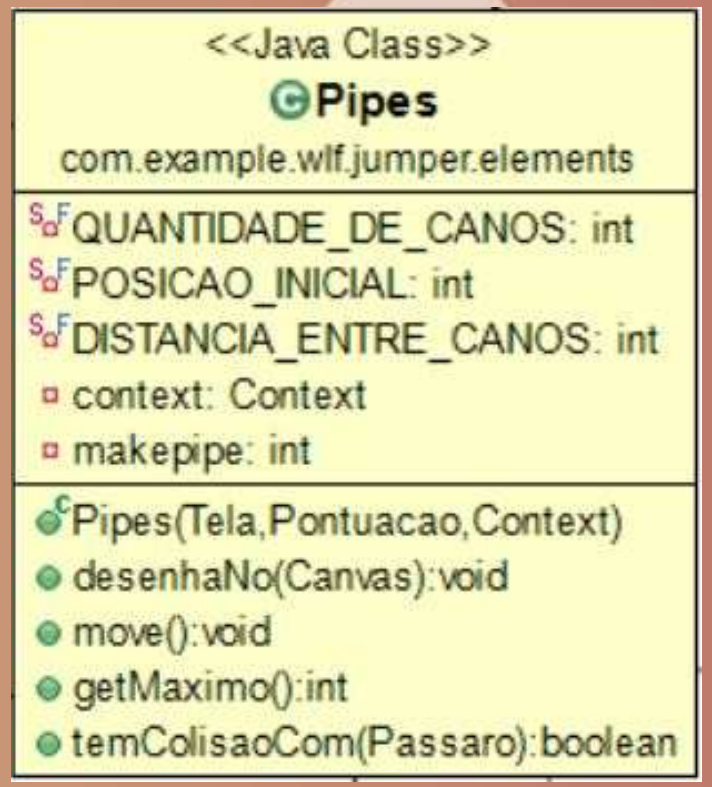
3) 클래스 별 기능



Pipe

파이프(장애물)을 생성하고 단계별 길이와 움직임까지 담당하는 클래스

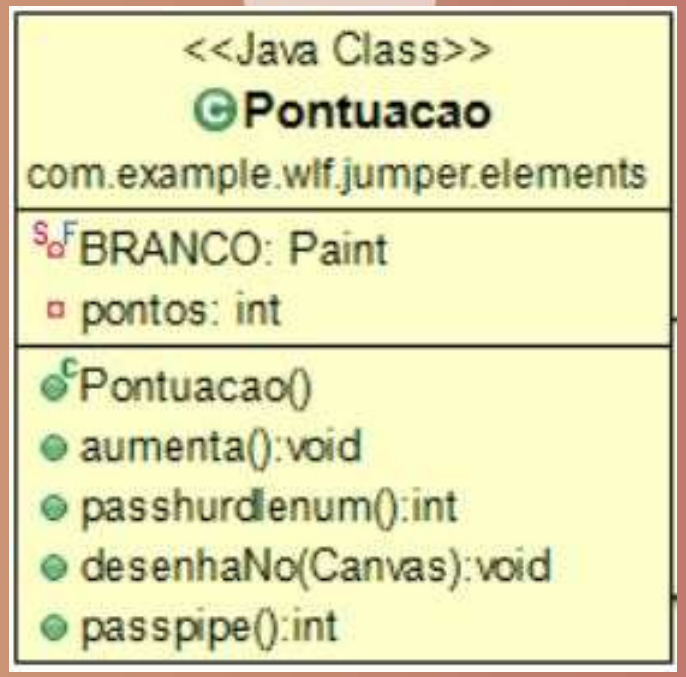
3) 클래스 별 기능



Pipes

게임 화면의 파이프(장애물)을 보여주는 클래스

3) 클래스 별 기능



Pontuacao

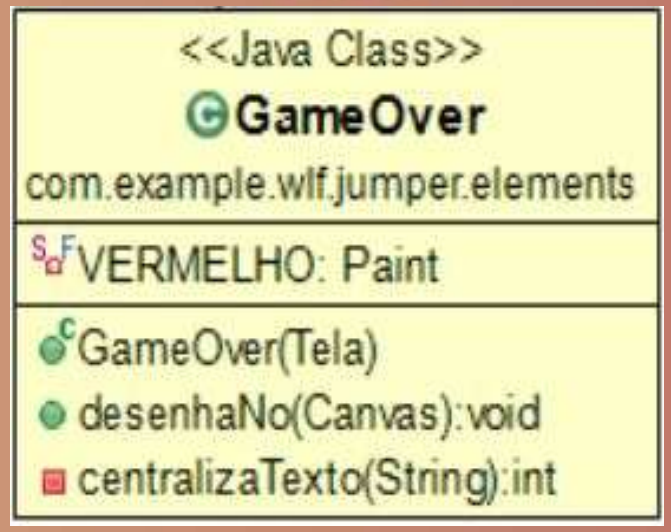
뛰어넘은 장애물의 개수를 카운팅하고 화면에 표시하는 클래스

3) 클래스 별 기능

VerificadorDeColisao

파이프(장애물)와 접촉이 되었는지를 판단하는 클래스

3) 클래스 별 기능



GameOver

게임이 끝나면 'gameover'를 출력하고, 다시하기 버튼을 호출하는 클래스

3) 클래스 별 기능

GameClear

게임의 마지막 단계인 40개를 다 넘어가면 'gameclear'를 출력하고, 다시하기 버튼을 호출하는 클래스

4) 변경, 추가한 코드 및 변경한 이미지

Passaro

게임 속도 변환 및 캐릭터의
움직임을 포물선 운동으로 변환

4) 변경, 추가한 코드 및 변경한 이미지

```
public class Passaro {


    //public static final int X = 100;
    public static final  int RAIO = 80;
    private final int J=-40;
    private Tela tela;
    private Bitmap passaro;
    private Som som;
    private int gravity=2;
    private int vy=J;
```

치삼이 캐릭터가 뛰어 오를 때 처음 올라가는 속도
바뀌지 않도록 final 선언

중력값

값을 바꾸기 쉽게 변수로 선언

Passaro

게임 속도 변환 및 캐릭터의
움직임을 포물선 운동으로 변환

4) 변경, 추가한 코드 및 변경한 이미지

```
public void xmove(){
    boolean checkcenter=getxspot()+RAIO<=(tela.getLargura()/2);
    if(checkcenter){
        setXspot(getxspot()+3);
    }
}
public void cai()
{
    boolean checouNoChao = getAltura()  > tela.getAltura();

    if ( ! checouNoChao )
    {
        this.vy+=this.gravity;
        setAltura(getAltura() + this.vy);

    }
}
public void cai2()
{
    boolean checouNoChao = getAltura()  > tela.getAltura();

    if ( ! checouNoChao )
    {
        setAltura(getAltura() + 15);

    }
}
```

치삼이 캐릭터가 화면 크기의 중앙으로 이동할 때 까지 캐릭터의 x좌표가 3씩 더해짐

한 번 이상의 터치를 실행하고 터치할 때 마다 실행
치삼이 y좌표가 vy만큼 변화하는 메소드

터치를 하지 않으면 떨어지는 메소드

4) 변경, 추가한 코드 및 변경한 이미지

Passaro

게임 클리어 시 캐릭터의 움직임 추가

```
public void xmove_end(){
    boolean checkcenter=getxspot()-RAIO<=tela.getLargura();
    if(checkcenter){
        setXspot(getxspot()+5);
    }
}
```

클리어 후 캐릭터가 화면 오른쪽으로 이동하여 나가는 움직임 구현

4) 변경, 추가한 코드 및 변경한 이미지

Pipe

파이프 모션 추가

```
private void valorAleatorio()
{


    int prelevel=(this.level%8);
    int gap=(10-prelevel)*150;
    int range =(int)(Math.random()*(tela.getAltura()-gap-TAMANHO_DO_CANO*2))+TAMANHO_DO_CANO;

    int topspot=range;
    int bottomspot=tela.getAltura()-(range+gap);

    this.alturaDoCanoInferior = bottomspot;
    this.alturaDoCanoSuperior = topspot; //길이


}
```

level을 설정하여 level에 따라 gap이 (위 장애물과 아래 장애물 사이의 거리가) 줄어 난이도 조절 가능하도록 구현

장애물 5개를 지날 때 마다 1level씩 증가

gap의 크기가 정해지면 위 장애물과 아래 장애물의 길이가 결정됨

4) 변경, 추가한 코드 및 변경한 이미지

Pipe

파이프 모션 추가

파이프가 위아래에서 생기는 메소드

```
private void desenhaCanoSuperiorNo( Canvas canvas )
{
   // canvas.drawRect(posicao, 0,  posicao + LARGURA_DO_CANO,alturaDoCanoSuperior, VERDE);
    int width=tela.getLargura();



    if(posicao>=(width-(width/5))){
        canvas.drawBitmap( canoSuperior, posicao, 0-alturaDoCanoSuperior+defaultsize, null );
        this.pretoplength=defaultsize;
    }else{
        if(this.level<8) {
            int preheight = -alturaDoCanoSuperior + defaultsize;
            if (preheight + this.pretoppipesize < 0) {
                this.pretoppipesize += ((this.alturaDoCanoSuperior - defaultsize) / 15);
                this.pretoplength=defaultsize+ pretoppipesize;
                canvas.drawBitmap(canoSuperior, posicao, preheight + pretoppipesize, null);

            } else {
                //this.pretoppipesize=this.alturaDoCanoSuperior;
                this.pretoplength=alturaDoCanoSuperior+defaultsize;
                canvas.drawBitmap(canoSuperior, posicao, 0, null);

            }
        }else{
            if(this.topclock==1){
                int preheight = -alturaDoCanoSuperior + defaultsize;
                if (preheight + this.pretoppipesize < 0) {
                    this.pretoppipesize += ((this.alturaDoCanoSuperior - defaultsize) / (this.topspeed*5));
                    this.pretoplength=defaultsize+ pretoppipesize;
                    canvas.drawBitmap(canoSuperior, posicao, preheight + pretoppipesize, null);
                } else {
                    //this.pretoppipesize=this.alturaDoCanoSuperior;
                    this.pretoplength=alturaDoCanoSuperior+defaultsize;
                    canvas.drawBitmap(canoSuperior, posicao, 0, null);
                    this.topclock=0;
                }
            }else{
                int preheight = -alturaDoCanoSuperior + defaultsize;
                if((preheight + this.pretoppipesize) >(-alturaDoCanoSuperior + defaultsize)){
                    this.pretoppipesize -= ((this.alturaDoCanoSuperior - defaultsize) / (this.topspeed*5));
                    this.pretoplength=defaultsize+ pretoppipesize;
                    canvas.drawBitmap(canoSuperior, posicao, preheight + pretoppipesize, null);

                }else{
                    this.pretoplength=defaultsize;
                    canvas.drawBitmap(canoSuperior, posicao, preheight, null);
                this.topclock=1;
                }
            }

        }
    }
```

4) 변경, 추가한 코드 및 변경한 이미지

Pipe

파이프 모션 추가

```
private void desenhaCanoInferiorNo( Canvas canvas )
{
    //canvas.drawRect(posicao, alturaDoCanoInferior,posicao + LARGURA_DO_CANO, tela.getAltura(), VERDE );
    int width=tela.getLargura();

    if(posicao>=(width-(width/5))){
        canvas.drawBitmap( canoInferior, posicao,tela.getAltura()-defaultsize, null );
        this.prebottomlength=tela.getAltura()-defaultsize;
    }else{
        if(this.level<8){
        int preheight=tela.getAltura()-defaultsize;
        if(preheight-this.prebottompipesize>tela.getAltura()-this.alturaDoCanoInferior){
            this.prebottompipesize+=((this.alturaDoCanoInferior-defaultsize)/15);
            this.prebottomlength=tela.getAltura()-defaultsize-this.prebottompipesize;
            canvas.drawBitmap( canoInferior, posicao,preheight-prebottompipesize , null );
        }else{

            //this.pretoppipesize=this.alturaDoCanoSuperior;
            this.prebottomlength=tela.getAltura()-this.alturaDoCanoInferior;
            canvas.drawBitmap(canoInferior, posicao,tela.getAltura()- this.alturaDoCanoInferior, null);
        }
    }else{
        if(this.bottomclock==1){
            int preheight=tela.getAltura()-defaultsize;
            if(preheight-this.prebottompipesize>tela.getAltura()-this.alturaDoCanoInferior){
                this.prebottompipesize+=((this.alturaDoCanoInferior-defaultsize)/(this.bottomspeed*5));
                this.prebottomlength=tela.getAltura()-defaultsize-this.prebottompipesize;
                canvas.drawBitmap( canoInferior, posicao,preheight-prebottompipesize , null );
            }else{
                this.prebottomlength=tela.getAltura()-this.alturaDoCanoInferior;
                canvas.drawBitmap(canoInferior, posicao,tela.getAltura()- this.alturaDoCanoInferior, null);
                this.bottomclock=0;
            }

        }else{
            int preheight=tela.getAltura()-defaultsize;
            if(preheight-this.prebottompipesize<tela.getAltura()-defaultsize){
                this.prebottompipesize-=((this.alturaDoCanoInferior-defaultsize)/(this.bottomspeed*5));
                this.prebottomlength=tela.getAltura()-this.prebottompipesize-defaultsize;
                canvas.drawBitmap( canoInferior, posicao,preheight-prebottompipesize , null );
            }else{
                this.prebottomlength=tela.getAltura()-defaultsize;
                canvas.drawBitmap( canoInferior, posicao,tela.getAltura()-defaultsize, null );
                this.bottomclock=1;
            }

        }
    }
}
```

4) 변경, 추가한 코드 및 변경한 이미지

Pipe

접촉함수 완성 및
레벨에 따른 파이프 연속 움직임 추가

```
private void desenhaCanoSuperiorNo( Canvas canvas )
{
   // canvas.drawRect(posicao, 0,  posicao + LARGURA_DO_CANO,alturaDoCanoSuperior, VERDE);
    int width=tela.getLargura();



    if(posicao>=(width-(width/5))){
        canvas.drawBitmap( canoSuperior, posicao, 0-alturaDoCanoSuperior+defaultsize, null );
        this.pretoplength=defaultsize;
    }else{
        if(this.level<8) {
            int preheight = -alturaDoCanoSuperior + defaultsize;
            if (preheight + this.pretoppipesize < 0) {
                this.pretoppipesize += ((this.alturaDoCanoSuperior - defaultsize) / 15);
                this.pretoplength=defaultsize+ pretoppipesize;
                canvas.drawBitmap(canoSuperior, posicao, preheight + pretoppipesize, null);

            } else {
                //this.pretoppipesize=this.alturaDoCanoSuperior;
                this.pretoplength=alturaDoCanoSuperior+defaultsize;
                canvas.drawBitmap(canoSuperior, posicao, 0, null);

            }
```

```
        }else{
            if(this.topclock==1){
                int preheight = -alturaDoCanoSuperior + defaultsize;
                if (preheight + this.pretoppipesize < 0) {
                    this.pretoppipesize += ((this.alturaDoCanoSuperior - defaultsize) / (this.topspeed*5));
                    this.pretoplength=defaultsize+ pretoppipesize;
                    canvas.drawBitmap(canoSuperior, posicao, preheight + pretoppipesize, null);
                } else {
                    //this.pretoppipesize=this.alturaDoCanoSuperior;
                    this.pretoplength=alturaDoCanoSuperior+defaultsize;
                    canvas.drawBitmap(canoSuperior, posicao, 0, null);
                    this.topclock=0;
                }
            }else{
                int preheight = -alturaDoCanoSuperior + defaultsize;
                if((preheight + this.pretoppipesize) >(-alturaDoCanoSuperior + defaultsize)){
                    this.pretoppipesize -= ((this.alturaDoCanoSuperior - defaultsize) / (this.topspeed*5));
                    this.pretoplength=defaultsize+ pretoppipesize;
                    canvas.drawBitmap(canoSuperior, posicao, preheight + pretoppipesize, null);

                }else{
                    this.pretoplength=defaultsize;
                    canvas.drawBitmap(canoSuperior, posicao, preheight, null);
                this.topclock=1;
                }
            }

        }

    }
```

4) 변경, 추가한 코드 및 변경한 이미지

Pipe

접촉함수 완성 및
레벨에 따른 파이프 연속 움직임 추가

```
public boolean temColisaoVerticalCom( Passaro passaro )
{
    int yspot=passaro.getAltura();
    if(yspot< this.pretoplength ||
            yspot+passaro.RAIO> this.prebottomlength+10||yspot >=tela.getAltura() ){
        return true;
    }else return false;

}
public boolean temColisaoHorizontalCom( Passaro passaro )
{

    if( passaro.getxspot()-passaro.RAIO<=getPosicao()+LARGURA_DO_CANO &&
    passaro.getxspot()+passaro.RAIO>=getPosicao()||passaro.getAltura()>=tela.getAltura()){
        return true;
    }else return false;
    //return this.posicao - passaro.getxspot() < passaro.RAIO;
}
```

접촉함수의 세로 확인

접촉함수의 가로 확인

두 메소드가 모두 참일 경우 접촉

4) 변경, 추가한 코드 및 변경한 이미지

Pipe

캐릭터의 위치가 변화로 인한
장애물 카운팅 방법 변경

```
public boolean checkpassed(Passaro passaro){
    if(passaro.getxspot()-passaro.RAIO>getPosicao()+LARGURA_DO_CANO&&!this.ckpassed){

        this.ckpassed=true;
        return true;
    }else return false;

}
```

캐릭터의 x좌표와 장애물의 위치에 따라
장애물을 넘으면 해당 장애물 카운팅
표기 후 true값 retrun

4) 변경, 추가한 코드 및 변경한 이미지

Pipes

파이프가 40개만 생성되도록 수정

```
if(this.makepipe<40) {
    this.makepipe++;
    Pipe outroCano =
            new Pipe(tela, getMaximo() + DISTANCIA_ENTRE_CANOS, this.pontuacao.passpipe(), context);//
    iterator.add(outroCano);
}
```

파이프가 40개 까지만 생성되도록 함

4) 변경, 추가한 코드 및 변경한 이미지

Back_ground

배경 흐름 및 게임 클리어 표시 추가

```
public back_ground(Tela tela,Context context){
    // BitmapFactory.Options options=new BitmapFactory.Options();
    //  options.inSampleSize=2;
      this.displaywidth=tela.getLargura();
       Bitmap bg =BitmapFactory.decodeResource(context.getResources(),R.drawable.background4 );
       this.bgwidth=bg.getWidth();
       this.BG = Bitmap.createScaledBitmap( bg, bg.getWidth(), tela.getAltura(), false );
}

public void drawbg( Canvas canvas )
{
    //canvas.drawCircle(X, getAltura(), RAIO, vermelho);
    canvas.drawBitmap( this.BG,x, y, null );
}
public void update(){

    if(this.x>-(this.bgwidth-this.displaywidth)){
        this.x=this.x+Scroll_Speed;
    }
```

백그라운드 생성자로 이미지 생성

해당 화면 그리기

위치 변화

배경이 흘러가 끝을 만나면 더 이상 흐름을 표현하지 않음

4) 변경, 추가한 코드 및 변경한 이미지

Game

게임 클리어 부분 구현

```
public void run() {
    while ( isRunning ){

        if ( ! holder.getSurface().isValid() )
        {
            continue;
        }

        canvas = holder.lockCanvas();
        BG.update();

        BG.drawbg(canvas);

        //canvas.drawBitmap(background, 0, 0, null);


        if(!cktouch)passaro.cai2();
        else{
            passaro.cai();
        }


        if(pontuacao.passhurdlenum()>=40&&passaro.getxspot()<tela.getLargura()){
            passaro.xmove_end();//게임 클리어

            canos.move();
            canos.desenhaNo(canvas);
```

40개를 넘으면 게임 클리어

4) 변경, 추가한 코드 및 변경한 이미지

GameClear

게임 클리어 클래스를 새롭게 생성

```
public class GameClear {
    private final Tela tela;
    private static final Paint VERMELHO = Cores.pntgameclear();
    public GameClear( Tela tela )
    {
        this.tela = tela;
    }

    public void drawClear( Canvas canvas )
    {
        String gameClear = "Game Clear";
        int centHorizontal = centertxt( gameClear );

        canvas.drawText( gameClear, centHorizontal,tela.getAltura() / 2, VERMELHO );
    }

    private int centertxt( String texto )
    {
        Rect limiteDoTexto = new Rect();
        VERMELHO.getTextBounds(texto, 0, texto.length(), limiteDoTexto);
        int centerHorizontal = tela.getLargura()/2 - ( limiteDoTexto.right - limiteDoTexto.left ) /2;
        return centerHorizontal;
    }

}
```

4) 변경, 추가한 코드 및 변경한 이미지

Cores

게임 클리어 시 조건 메소드 추가

```
public static Paint pntgameclear() {
    Paint vermelho = new Paint();
    vermelho.setColor(0xFF006600);
    vermelho.setTextSize(100);
    vermelho.setTypeface( Typeface.DEFAULT_BOLD );
    vermelho.setShadowLayer(2, 3, 3, 0xFF000000 );

    return vermelho;
}
```

게임 클리어 화면을 그릴 때
사용하는 조건 메소드

4) 변경, 추가한 코드 및 변경한 이미지

MainActivity

게임 시작 및 재시작 버튼, 파이프 추가

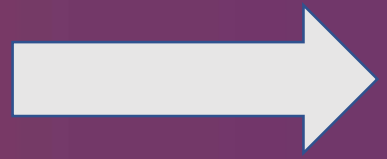
```
public class MainActivity extends AppCompatActivity {

    Game game;
    Button retry;
    Button start;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        final FrameLayout container = (FrameLayout) findViewById(R.id.container);

        game = new Game(this, mainHandler);
        //container.addView(game);

       // startbg=(ImageView) findViewById(R.id.startbg);
        retry = findViewById(R.id.retry);
        start = findViewById(R.id.start);

        retry.setVisibility(View.GONE);
        //startbg.setVisibility(View.VISIBLE);
        start.setVisibility(View.VISIBLE);

        final Thread thread = new Thread(game);
        start.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                        container.addView(game);
                thread.start();
                start.setVisibility(View.GONE);
            }
        });
    }
```

재시작 버튼, 게임 시작 버튼 구현

4) 변경, 추가한 코드 및 변경한 이미지

이미지

아이콘



4) 변경, 추가한 코드 및 변경한 이미지

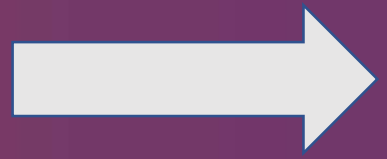
이미지

시작화면

4) 변경, 추가한 코드 및 변경한 이미지

이미지

캐릭터



4) 변경, 추가한 코드 및 변경한 이미지

이미지

배경

4) 변경, 추가한 코드 및 변경한 이미지

이미지

장애물

4) 변경, 추가한 코드 및 변경한 이미지

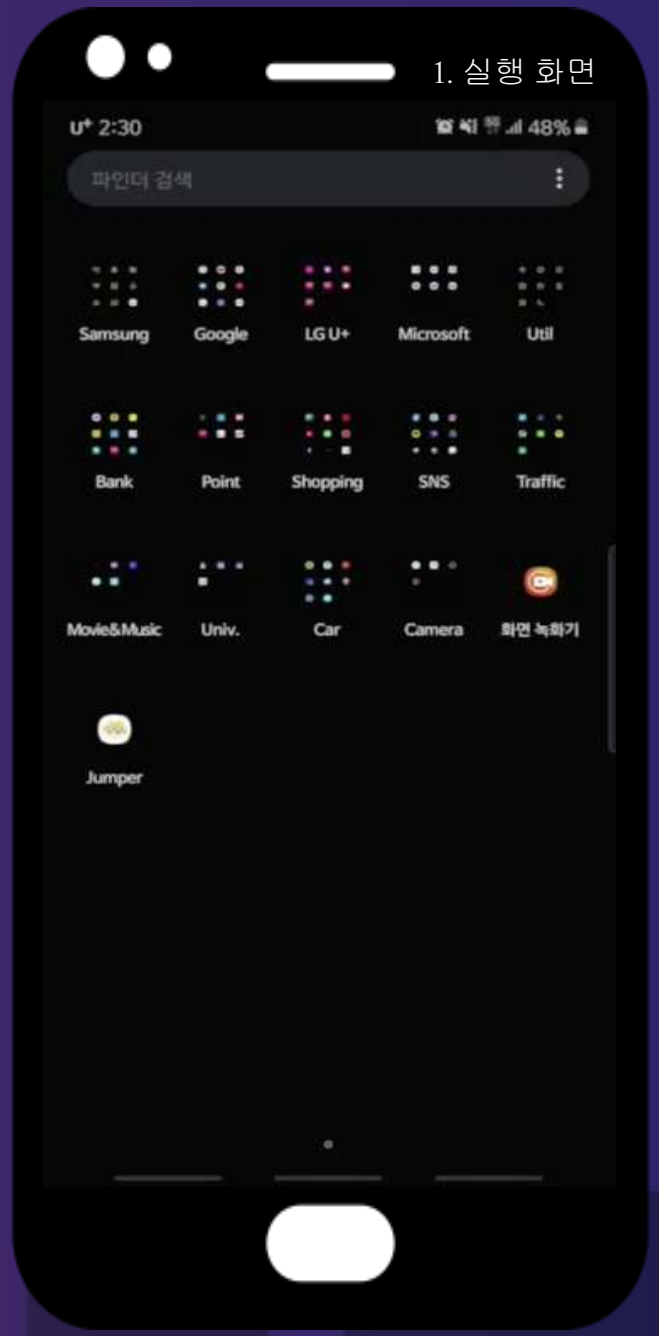
이미지

시작 버튼 및 재시작 버튼

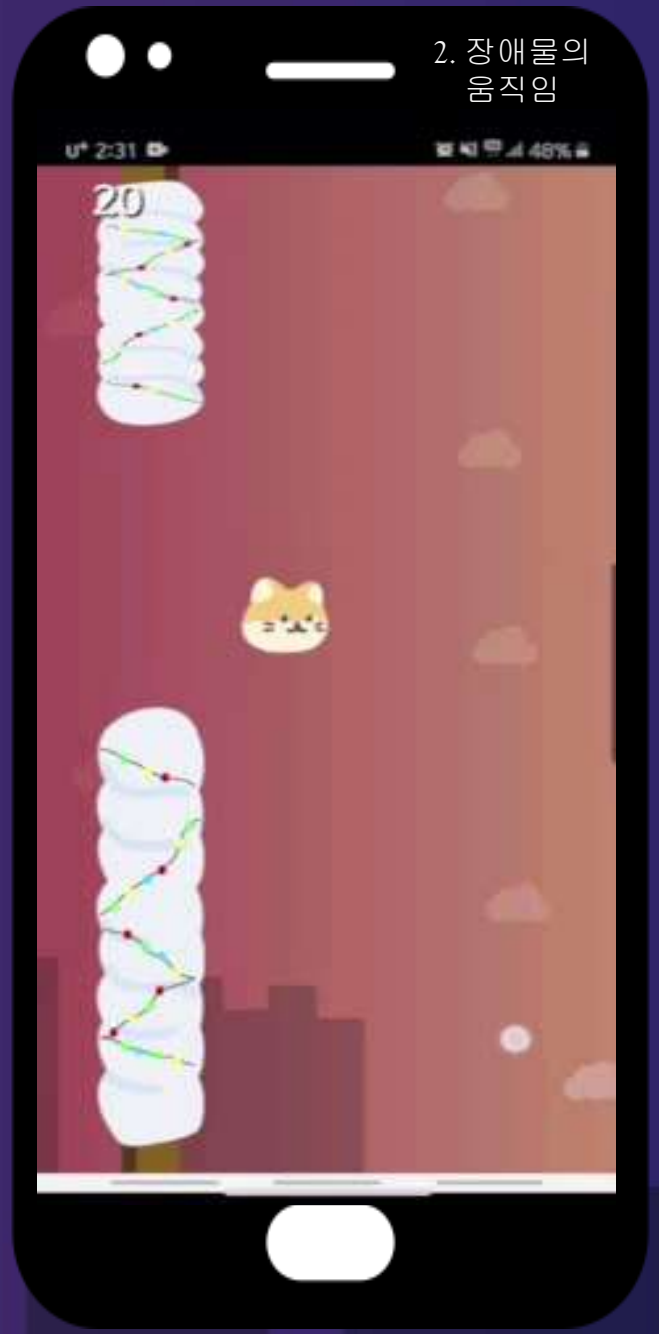
5) 최종 게임 시연

5) 최종 게임 시연

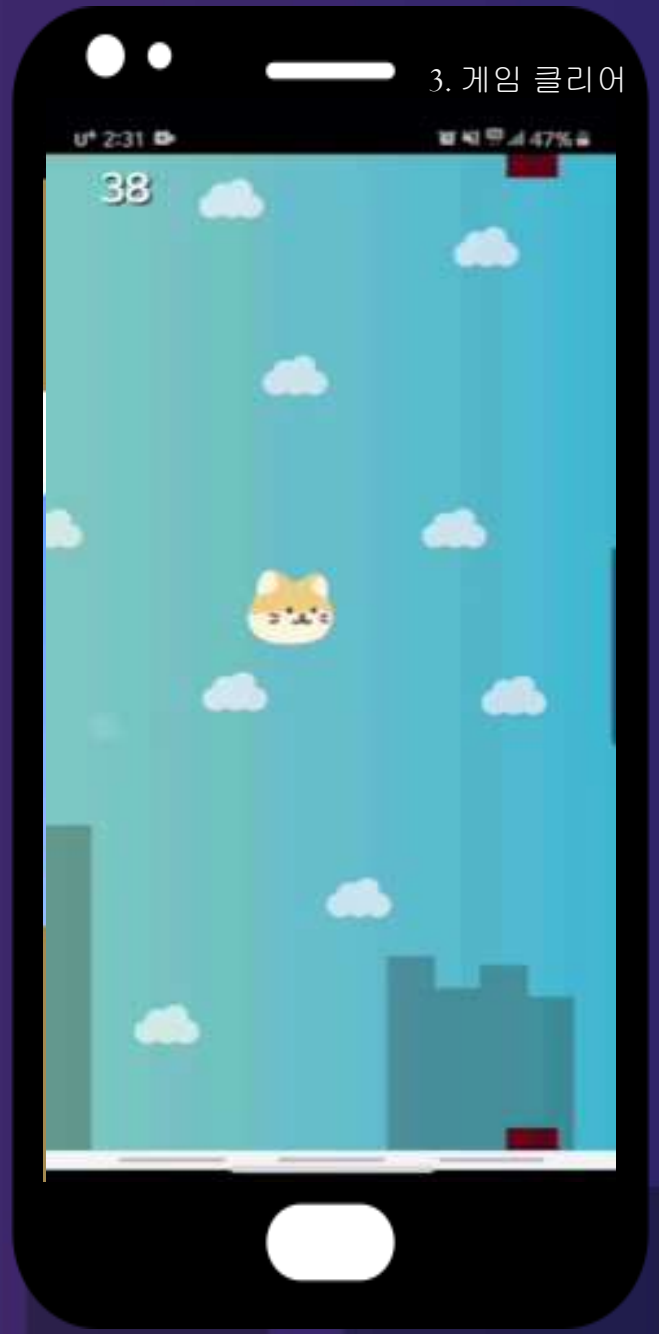
1. 실행 화면



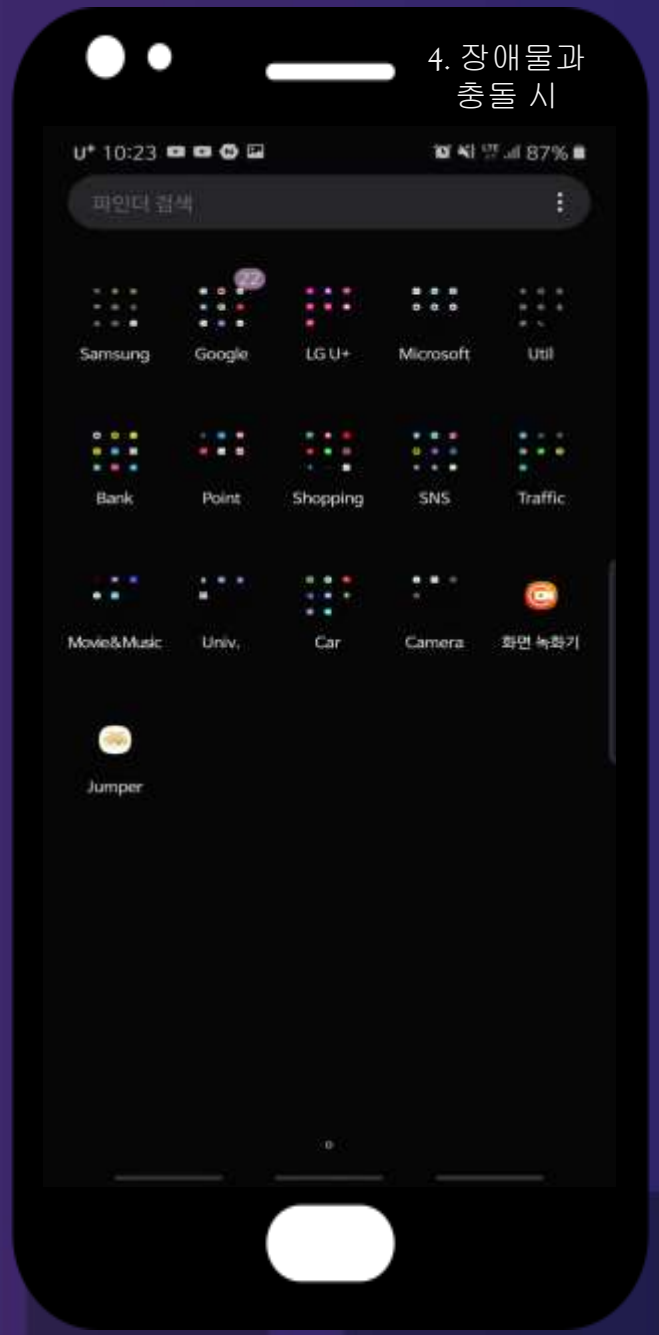
5) 최종 게임 시연



5) 최종 게임 시연



5) 최종 게임 시연


4. 장애물과
충돌 시
U+ 10:23
87%
파인더 검색
Samsung
Google
LG U+
Microsoft
Util
Bank
Point
Shopping
SNS
Traffic
Movie&Music
Univ.
Car
Camera
화면 녹화기
Jumper

JUMPE

원본 파일 : https://github.com/anji314/jumper

개발 파일 : https://github.com/anji314/JumperRemake

감사합니다