

電子計算機研究会

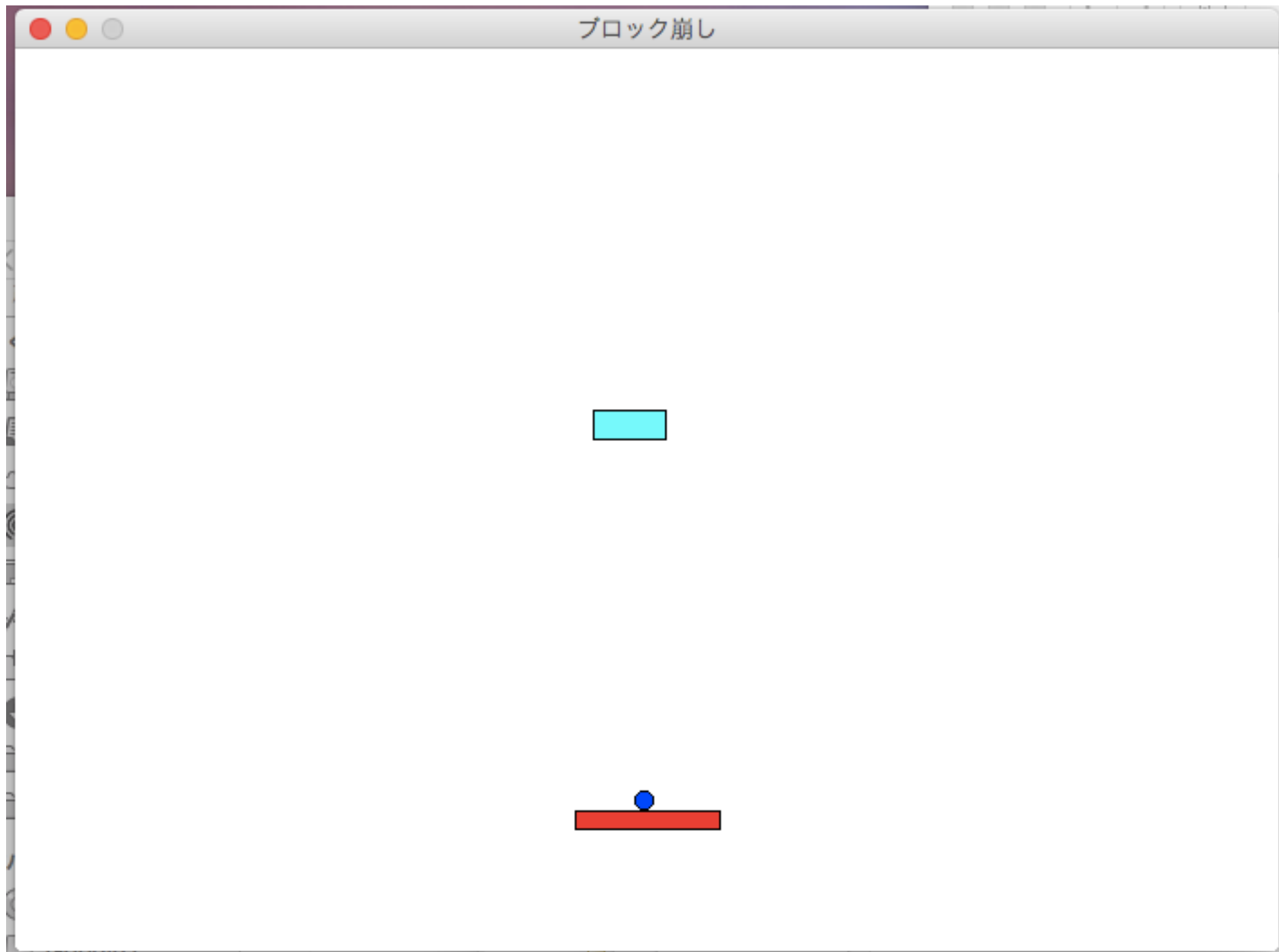
第8回オープン授業

ゲーム作成

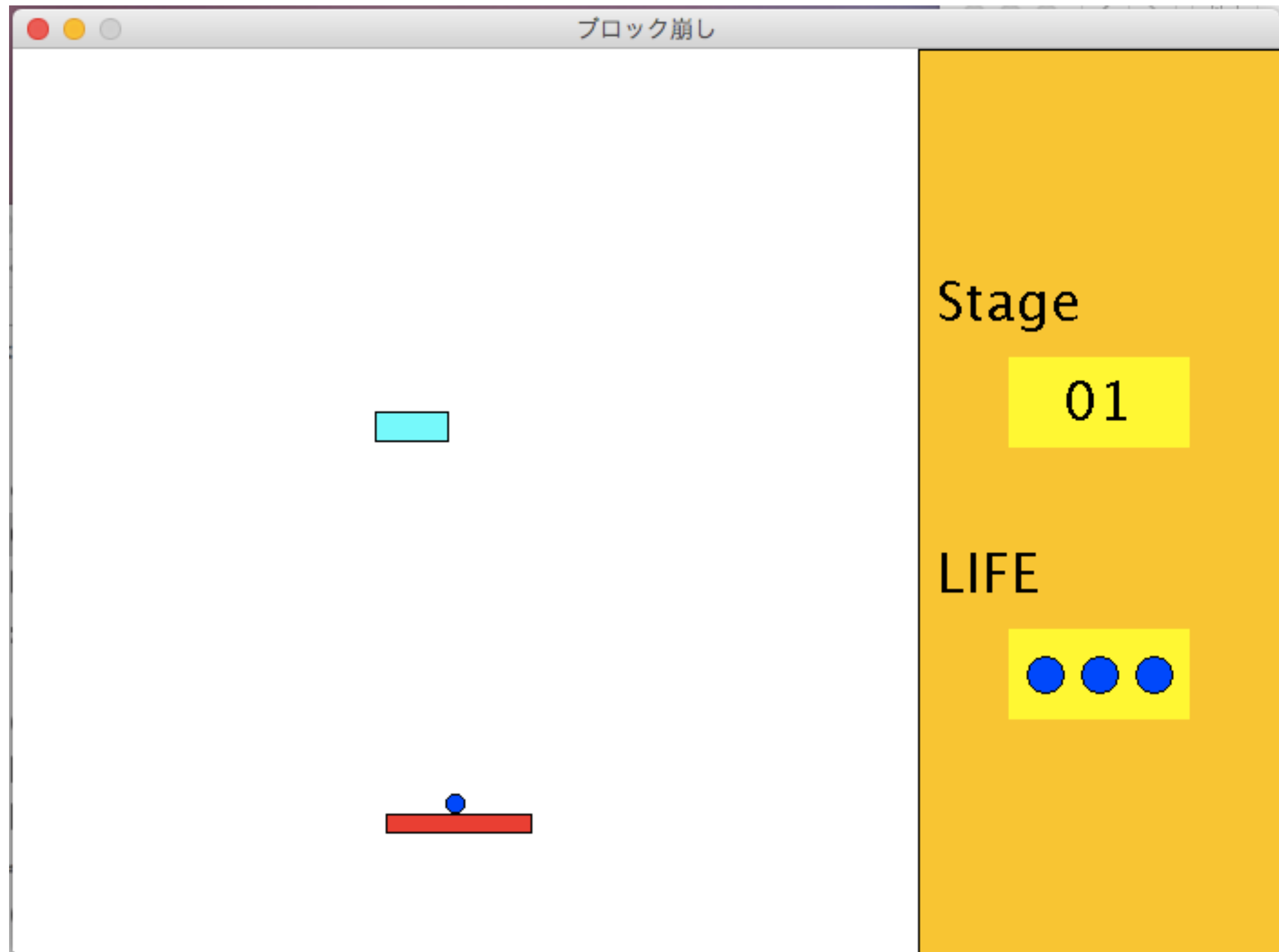
ゲーム作成の手順

1. 電算Wikiのオープン授業から今回のオープン授業のファイルをダウンロードする。
2. 電算WikiのDJGL-更新履歴からDensanJavaGameLibrary01_01aをダウンロード
3. eclipseを起動し画面右上のファイルの項目からインポートを選択し、既存プロジェクトをワークスペースへを選択する。そしてダウンロードの項目からオープン授業のファイルを選択し、完了をクリック。
4. eclipse上のbreakBlock1の項目で右クリックし、ビルドパスの項目から、外部アーカイブの追加をクリックする。そして、ダウンロードの項目からDensanJavaGameLibrary01_01aを選択し、開くをクリックする。

ブロック崩し(制作途中)



ブロック崩し(完成版)



breakBlock2の構成

- ・ main…ゲームの基本的な部分
- ・ object…ゲーム上の物の動作の処理
- ・ stage…ステージで行われる動作の処理

今回の課題

1. main内のクラス、
ClearScene,ContinueScene,GameOverScene,Title
の一部が穴埋めとなっているのでその部分を作成する。
2. stage内のクラス,MainStageにボード表を表示し、
それにobject内のクラス、Ball,Racketを対応させる。
3. stage内のクラス,MainStageに画面の切り替えの処理
を付け加える。

画面の表示の確認

```
public Main(){  
    //最初にこの部分のクラスの表示が行われる  
    nowScene = new Title();  
}
```

- ・ 上文の場合だと最初にタイトル画面が表示される。
new Title()の部分をnew Stage1()などに変更することで最初に表示する画面を変えることができる。

※コンティニュー画面の表示の場合、下文のようになる。
(コンティニューを選択した場合Stage1に画面が変更される)

```
public Main(){  
    //最初にこの部分のクラスの表示が行われる  
    nowScene = new ContinueScene(new Stage1());  
}
```

ClearSceneクラスへの追加

- ・ クリア画面の描画(drawメソッド)
- ・ 一定時間経った後にクリアシーンを終了する
(updateメソッド)

drawメソッド(ClearScene)

```
public void draw(Drawer d) {  
    d.setColor(Color.black);  
    d.setFont(new Font(Font.DIALOG, Font.ITALIC, 30));  
    d.drawStringCenter("GameClear", GameManager.getInstance().getFrameWidth()/2,  
                       GameManager.getInstance().getFrameHeight()/2);  
}
```

- ・ 黒色でイタリック体のフォントの30の大きさのGameClearという文字を、画面中央に表示する。

updateメソッド(ClearScene)

```
public void update() {  
    //400フレーム後クリア画面を終了  
    if(count >= 400)  
        System.exit(0);  
    count++;  
}
```

- ・ 400フレームに達するまでcountを足し続けて、400フレームに達した時点で実行を終了する。

ContinueSceneクラスへの追加

- ・ コンティニュー画面の描画(drawメソッド)
- ・ コンティニューをするかどうかの判断(updateメソッド)

drawメソッド(ContinueScene)

```
public void draw(Drawer d) {  
    d.setColor(Color.black);  
    d.setFont(new Font(Font.DIALOG, Font.ITALIC, 30));  
    d.drawStringCenter("Continue?", GameManager.getInstance()  
        .getFrameWidth() / 2, GameManager.getInstance()  
        .getFrameHeight() / 3);  
    d.drawStringCenter("Yes",  
        GameManager.getInstance().getFrameWidth() / 3, GameManager  
        .getInstance().getFrameHeight() * 2 / 3);  
    d.drawStringCenter("No",  
        GameManager.getInstance().getFrameWidth() * 2 / 3, GameManager  
        .getInstance().getFrameHeight() * 2 / 3);  
  
    d.drawStringCenter("→", GameManager.getInstance().getFrameWidth() * n  
        / 3 - 40, GameManager.getInstance().getFrameHeight() * 2 / 3);  
  
    d.drawStringCenter("" + count / 60, GameManager.getInstance()  
        .getFrameWidth() / 2, GameManager.getInstance()  
        .getFrameHeight() / 2);  
}
```

- ・ 黒色のイタリック体の文字をそれぞれ指定の位置に表示させている。矢印の位置はnの値によって変化し、カウントの表示はcountが60減っていくごとに常時が変化していく。

updateメソッド(ContinueScene)

```
public void update() {  
    //Yesの状態  
    if(n==1){  
        if(KeyInput.isPress(KeyEvent.VK_ENTER)||KeyInput.isPress(KeyEvent.VK_SPACE)){  
            Main.changeScene(nowScene);  
        }else if(KeyInput.isPress(KeyEvent.VK_RIGHT)){  
            n++;  
        }  
        //Noの状態  
    }else if(n==2){  
        if(KeyInput.isPress(KeyEvent.VK_ENTER)||KeyInput.isPress(KeyEvent.VK_SPACE)){  
            Main.changeScene(new GameOverScene());  
        }else if(KeyInput.isPress(KeyEvent.VK_LEFT)){  
            n--;  
        }  
    }  
  
    //10秒後ゲームオーバーの画面へ  
    if(count/60<=0)  
        Main.changeScene(new GameOverScene());  
  
    count--;  
}
```

・ Yesにカーソルがあっている時は右矢印でNoにカーソルを移動、エンターキーもしくはスペースキーでコンティニューを行う。Noにカーソルがあっている時は左矢印でYesにカーソルを移動、エンターキーもしくはスペースキーでゲームオーバー画面に移動する。また、10秒経った場合もゲームオーバー画面に移動する。

GameOverSceneクラスへの追加

- ・ ゲームオーバー画面の描画(drawメソッド)
- ・ GameOverという文字の移動(updateメソッド)

drawメソッド(GameOverScene)

```
public void draw(Drawer d) {  
    d.setColor(Color.BLACK);  
    d.fillRect(0, 0, GameManager.getInstance().getFrameWidth()  
        , GameManager.getInstance().getFrameHeight());  
  
    d.setColor(Color.RED);  
    d.setFont(new Font(Font.DIALOG, Font.BOLD, 50));  
    d.drawString("G", gX, gY);  
    d.drawString("a", gX+space, aY);  
    d.drawString("m", gX+space*2-8, mY);  
    d.drawString("e", gX+space*3, eY);  
    d.drawString("0", gX+space*4, oY);  
    d.drawString("v", gX+space*5, vY);  
    d.drawString("e", gX+space*6-10, e2Y);  
    d.drawString("r", gX+space*7-20, rY);  
  
}
```

- drawString(表示文字,表示x座標,表示y座標)となっている。

updateメソッド(GameOverScene)

```
public void update() {  
    if(count>=0&&count<=200){  
        gY++;  
        oY--;  
    }  
    if(count>=countSpace&&count<=200+countSpace){  
        aY++;  
        vY--;  
    }  
    if(count>=countSpace*2&&count<=200+countSpace*2){  
        mY++;  
        e2Y--;  
    }  
    if(count>=countSpace*3&&count<=200+countSpace*3){  
        eY++;  
        rY--;  
    }  
  
    if(count>=200+countSpace*7)  
        System.exit(0);  
    count++;  
}
```

- ・ 各フレームごとに文字の移動を始める処理をまとめている

Titleクラスへの追加

- ・ タイトル画面の描画(drawメソッド)
- ・ ステージ画面への移動(updateメソッド)

drawメソッド(Title)

```
public void draw(Drawer d) {
    d.setColor(Color.black);
    d.setFont(new Font(Font.DIALOG,Font.BOLD,50));
    d.drawStringCenter("ブロック崩し", GameManager.getInstance().getFrameWidth()/2,
                       GameManager.getInstance().getFrameHeight()/3);
    d.setFont(new Font(Font.DIALOG,Font.ROMAN_BASELINE,20));
    d.drawString("操作説明",100, 300);
    d.drawString("← →で左右にバーを移動", 130, 330);
    d.drawString("スペースキーでボールを発射", 130, 360);

    //60カウントごとに表示したり消えたりする
    if((count/60)%2<1 ){
        d.setFont(new Font(Font.DIALOG,Font.BOLD,15));
        d.drawStringCenter("スペースキーでスタート", GameManager.getInstance().getFrameWidth()/2,
                           GameManager.getInstance().getFrameHeight()*4/5);
    }
}
```

- ・タイトル画面に必要な情報の表示を行う。スペースキーでスタートという文字の部分は点滅して表示させる。

updateメソッド(Title)

```
public void update() {  
    if (KeyInput.isPress(KeyEvent.VK_SPACE))  
  
        Main.changeScene(new Stage1());  
    count += 1;  
}
```

- ・スペースキーを入力した場合にステージ1に移動する。

MainStageクラスへの追加

- ・ ボード表の処理を追加(drawメソッド)
- ・ 残機が0になった場合の処理とステージをクリアした場合の処理を追加(updateメソッド)

ボード表の追加(drawメソッド)

//ボード表

```
d.setColor(Color.ORANGE);
d.fillRect(GameManager.getInstance().getFrameWidth()-BOARD_WIDTH, 0, BOARD_WIDTH,
GameManager.getInstance().getFrameHeight());
d.setColor(Color.BLACK);
d.drawRect(GameManager.getInstance().getFrameWidth()-BOARD_WIDTH, 0, BOARD_WIDTH,
GameManager.getInstance().getFrameHeight());
d.setColor(Color.BLACK);
d.setFont(new Font(Font.DIALOG,Font.ROMAN_BASELINE,30));
d.drawString("LIFE", 510, 300);
d.setColor(Color.YELLOW);
d.fillRect(550, 320, 100, 50);
//残機の表示
for(int i=1;i<=remainingLives;i++){
    d.setColor(Color.BLUE);
    d.fillCircle(540+30*i,345,10);
    // 枠線を描画
    d.setColor(Color.BLACK);
    d.drawCircle(540+30*i,345,10);
}
d.setColor(Color.BLACK);
d.setFont(new Font(Font.DIALOG,Font.ROMAN_BASELINE,30));
d.drawString("Stage", 510, 150);
d.setColor(Color.YELLOW);
d.fillRect(550, 170, 100, 50);
if(count>START_TIME){
    d.setColor(Color.BLACK);
    d.setFont(new Font(Font.DIALOG,Font.ROMAN_BASELINE,30));
    if(stageNumber<10)
        d.drawString("0"+stageNumber, 580, 205);
    else d.drawString(""+stageNumber, 580, 205);
}
```

- ・ ボード表を表示し、残機とステージ名を表示する。

Ballクラスへの追加

```
public void update(){
    addX(vx);
    addY(vy);

    // 左または右に当たったらx方向速度の符号を反転させる
    if (getX() < 0 || getX() >
GameManager.getInstance().getFrameWidth()-MainStage.BOARD_WIDTH -
getWidth()) {
        vx = -vx;
    }

    // 上に当たったらy方向速度の符号を反転させる
    if (getY() < 0){
        vy = -vy;
    }
}
```

- ・赤字の部分が追加部分であり、ボールがボード表の部分にはみ出さないようになっている。

Racketクラスへの追加

```
public void update(){
    if(KeyInput.isPressing(KeyEvent.VK_LEFT)&&getX()>0)
        addX(-5);
    else if(KeyInput.isPressing(KeyEvent.VK_RIGHT)&&getX()
+WIDTH<GameManager.getInstance().getFrameWidth()-MainStage.BOARD_WIDTH)
        addX(5);
}
```

- ・赤字の部分が追加部分であり、ラケットがボード表の部分にはみ出さないようになっている。

残機が0になった場合の処理の追加 (updateメソッド)

```
if(remainingLives==0)//残機が0になった場合コンティニュー画面へ変更  
    Main.changeScene(new ContinueScene(newStage()));
```

- ・ 残機が0になったときにコンティニュー画面へ変更する

ステージをクリアした場合の処理の追加 (updateメソッド)

```
//ブロックの消えた回数とステージ上のブロックの数と同じ
if(deadCount==numBlock)
    //次のステージがあるなら
    if(nextStage()!=null){
        //ステージ移動
        Main.changeScene(nextStage());
        stageNumber++;
    }else
        //ないならクリア画面
        Main.changeScene(new ClearScene());
}
```

- ・ ステージ内のブロックを全てこわした場合、次のステージに移動、最後のステージだった場合はクリア画面へ移動する。

オープン授業おわり