

Programmier-Projekt

Das Programmier-Projekt

In den folgenden Wochen werdet ihr - erneut - an einem Spiel arbeiten. Das Spiel soll folgende Voraussetzungen erfüllen:

- Mindestens eine **Spieler-Klasse**, über die der Spieler gesteuert werden kann. Wer mag, gerne auch ein Mehrspieler-Spiel z.B. Spieler 1 WASD, Spieler 2 Pfeiltasten.
- Mehrere **weitere Klassen**, z.B. eine Klasse pro **Level** / **Gegner** / **Hindernis** / **Gegenstand**, etc.
- Ein Klasse, die den **Startbildschirm** mit Infos zum Spiel und den Autoren darstellt.

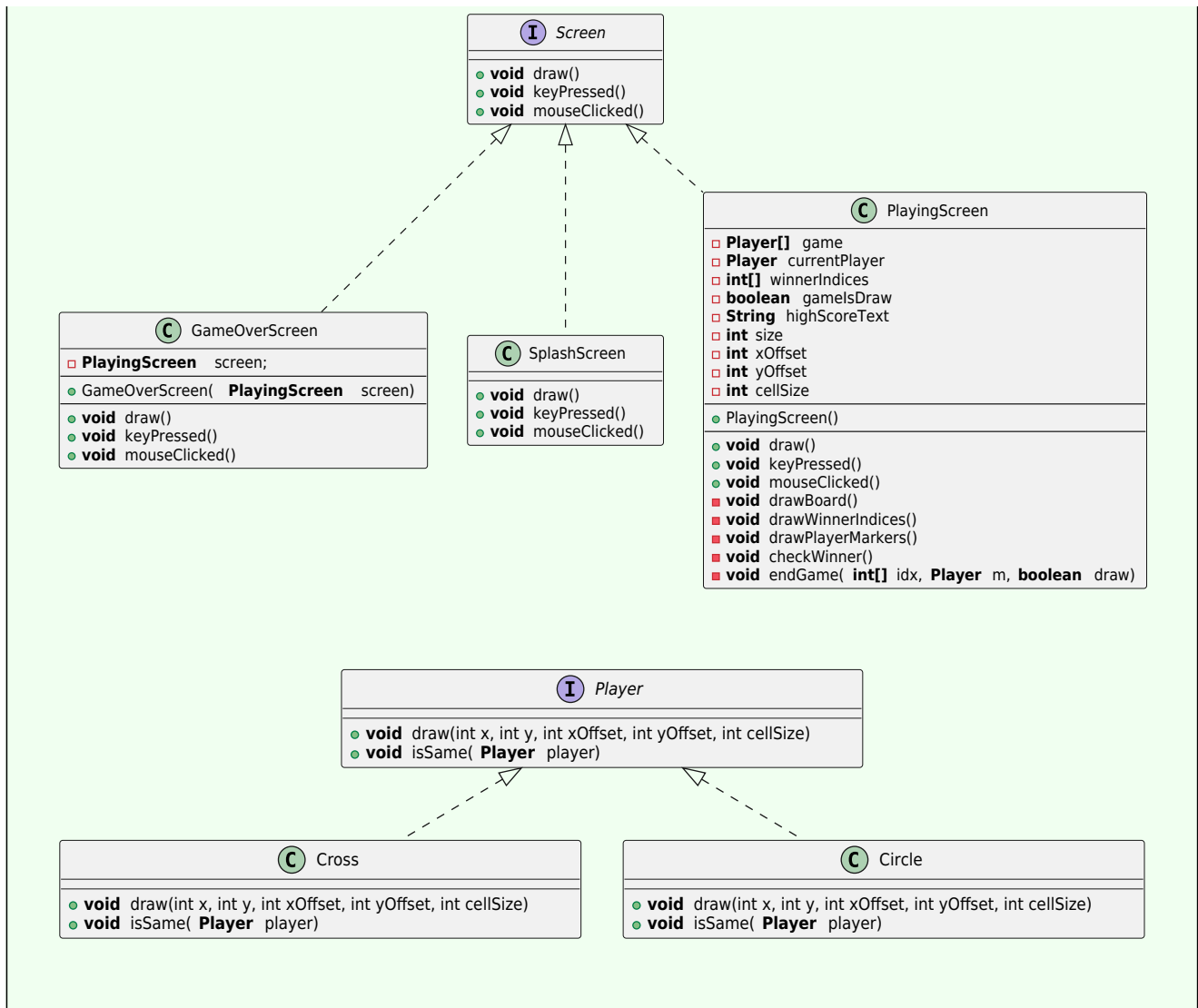
Beispiel: TicTacToe

Nehmen wir als Beispiel das TicTacToe-Projekt aus der E-Phase:

- Hier gibt es mehrere Screens:
 - Der **SplashScreen** zeigt am Anfang Informationen über das Spiel.
 - Der **PlayingScreen** zeichnet das eigentliche Spiel.
 - Der **GameOverScreen** erscheint, wenn das Spiel beendet ist.
 - Alle drei Screens haben die drei Methoden **void draw()**, **void keyPressed()** und **void mouseClicked()** gemeinsam. Das kann man in dem so genannten Interface **interface Screen** als Gemeinsamkeiten zusammenfassen.
 - Lediglich der **PlayingScreen** hat noch mehr Methoden, die für die eigentliche Spiellogik verantwortlich sind.
 - Außerdem muss der **GameOverScreen** den zugehörigen **PlayingScreen** kennen, da er diesen im Hintergrund zeichnet.
- Ebenso gibt es zwei Spieler, **Cross** und **Circle**.
 - Diese haben ebenfalls Gemeinsamkeiten: Sie können mit **void draw()** gezeichnet werden und man kann mit **boolean isSame(Player player)** feststellen, ob ein anderer Spieler zur gleichen "Kategorie" gehört.

Demo: TicTacToe

[screencast_from_18.06.2022_18_32_06.webm](#)



Code zur objektorientierten Variante von TicTacToe

TicTacToeOOP

TicTacToeOOP.pde

```

Screen screen;

PFont headingFont;
PFont textFont;

void setup() {
  size(500, 500);

  headingFont = createFont("Liberation Sans Bold", 32);
  textFont = createFont("Liberation Sans", 16);

  rectMode(CENTER);
  ellipseMode(CENTER);
  textAlign(CENTER, CENTER);

```

```
screen = new SplashScreen();  
}  
  
void draw() {  
    screen.draw();  
}  
  
void keyPressed() {  
    screen.keyPressed();  
}  
  
void mouseClicked() {  
    screen.mouseClicked();  
}  
  
void setScreen(Screen newScreen) {  
    screen = newScreen;  
}
```

Screen

Screen.pde

```
interface Screen {  
    public void draw();  
  
    public void keyPressed();  
  
    public void mouseClicked();  
}
```

Player

Player.pde

```
interface Player {  
    void draw(int x, int y, int xoffset, int yoffset, int  
        cellSize);  
    boolean isSame(Player player);  
}
```

Cross

[Cross.pde](#)

```
class Cross implements Player {
    void draw(int x, int y, int xOffset, int yOffset, int cellSize)
    {
        line(xOffset + x * cellSize + cellSize / 6, yOffset + y *
            cellSize + cellSize / 6, xOffset + (x + 1) * cellSize - cellSize
            / 6, yOffset + (y + 1) * cellSize - cellSize / 6);
        line(xOffset + (x + 1) * cellSize - cellSize / 6, yOffset + y
            * cellSize + cellSize / 6, xOffset + x * cellSize + cellSize / 6,
            yOffset + (y + 1) * cellSize - cellSize / 6);
    }

    boolean isSame(Player player) {
        return player != null && player instanceof Cross;
    }
}
```

Circle

[Circle.pde](#)

```
class Circle implements Player {
    void draw(int x, int y, int xOffset, int yOffset, int cellSize)
    {
        ellipse(xOffset + x * cellSize + cellSize / 2, yOffset + y *
            cellSize + cellSize / 2, 2*cellSize / 3, 2*cellSize / 3);
    }

    boolean isSame(Player player) {
        return player != null && player instanceof Circle;
    }
}
```

Playing

[Playing.pde](#)

```
class PlayingScreen implements Screen {
    final int[][] WINNING_COMBINATIONS = {
        { 0, 1, 2 }, // first row
    }
```

```
        { 3, 4, 5 }, // second row
        { 6, 7, 8 }, // third row
        { 0, 3, 6 }, // first column
        { 1, 4, 7 }, // second column
        { 2, 5, 8 }, // third column
        { 0, 4, 8 }, // first diagonal
        { 2, 4, 6 } // second diagonal
    };

    Player[] game = {
        null, null, null,
        null, null, null,
        null, null, null
    };

    Player currentPlayer = null;
    int[] winnerIndices = null;
    boolean gameIsDraw = false;
    String highScoreText = "";
    int size;
    int xOffset;
    int yOffset;
    int cellSize;

    public PlayingScreen() {
        size = min(width, height);
        xOffset = (width - size) / 2;
        yOffset = (height - size) / 2;
        cellSize = size / 3;

        gameIsDraw = false;
        currentPlayer = null;
        game = new Player[] {
            null, null, null,
            null, null, null,
            null, null, null
        };
        winnerIndices = null;
    }

    public void draw() {
        background(#CCCCCC);

        drawWinnerIndices();
        drawBoard();
        drawPlayerMarkers();
    }

    public void keyPressed() { }

    public void mouseClicked() {
        int x = (mouseX - xOffset) / cellSize;
        int y = (mouseY - yOffset) / cellSize;
```

```
        int idx = y * 3 + x;

        if (game[idx] != null) return;

    if (currentPlayer == null || currentPlayer instanceof Circle)
    {
        currentPlayer = new Cross();
    } else {
        currentPlayer = new Circle();
    }

    game[idx] = currentPlayer;

    checkWinner();
}

void drawBoard() {
    stroke(#000000);
    strokeWeight(5);

    line(xOffset + cellSize, yOffset, xOffset + cellSize, height
        - yOffset);
    line(xOffset + 2*cellSize, yOffset, xOffset + 2*cellSize,
        height - yOffset);
    line(xOffset, yOffset + cellSize, width - xOffset, yOffset +
        cellSize);
    line(xOffset, yOffset + 2*cellSize, width - xOffset, yOffset
        + 2*cellSize);
}

void drawWinnerIndices() {
    if (winnerIndices == null || winnerIndices.length <= 0)
        return;

    for (int i = 0; i < winnerIndices.length; i++) {
        int x = winnerIndices[i] % 3;
        int y = winnerIndices[i] / 3;

        noStroke();
        fill(#AACCEE);
        rect(xOffset + x * cellSize + cellSize / 2, yOffset + y *
            cellSize + cellSize / 2, cellSize, cellSize);
    }
}

void drawPlayerMarkers() {
    for (int i = 0; i < game.length; i++) {
        int x = i % 3;
        int y = i / 3;

        Player marker = game[i];
```

```
        noFill();
        strokeWeight(5);
        stroke(#000000);

        if (marker != null) {
            marker.draw(x, y, xOffset, yOffset, cellSize);
        }
    }
}

void checkWinner() {
    for (int[] indices : WINNING_COMBINATIONS) {
        Player firstMarker = game[indices[0]];
        boolean failed = false;

        if (firstMarker == null) continue;

        for (int i = 1; i < indices.length; i++) {
            Player marker = game[indices[i]];

            if (!firstMarker.isSame(marker)) {
                failed = true;
            }
        }

        if ((firstMarker instanceof Cross || firstMarker instanceof
            Circle) && !failed) {
            endGame(indices, firstMarker, false);
            return;
        }
    }

    int usedCells = 0;

    for (int i = 0; i < game.length; i++) {
        if (game[i] != null) {
            usedCells++;
        }
    }

    if (usedCells == game.length) {
        endGame(null, null, true);
    }
}

void endGame(int[] indices, Player marker, boolean isDraw) {
    winnerIndices = indices;
    gameIsDraw = isDraw;

    StringList highScores = new StringList();
```

```

        if (dataFile("highscores.txt").isFile()) {
            highScores.append(loadStrings("data/highscores.txt"));
        }

        String date = nf(year(), 4) + "-" + nf(month(), 2) + "-" +
            nf(day(), 2) + " " + nf(hour(), 2) + ":" + nf(minute(), 2) + ":"
                + nf(second(), 2);
        String text = isDraw
            ? "draw"
            : ((marker instanceof Cross ? "cross" : "circle") + "@" +
                join(nf(indices, 1), ","));

        highScores.append(date + " / " + text);

        saveStrings("data/highscores.txt", highScores.array());

        if (highScores.size() > 0) {
            int crossWins = 0;
            int circleWins = 0;
            int draw = 0;

            for (String s : highScores) {
                String winner = s.substring(22);

                if (winner.indexOf("@") > 0) {
                    winner = winner.substring(0, winner.indexOf("@"));
                }

                if (winner.equals("cross")) crossWins++;
                else if (winner.equals("circle")) circleWins++;
                else if (winner.equals("draw")) draw++;
            }

            highScoreText = "\n\nHighScores:\nCross: " + crossWins +
                "\nCircle: " + circleWins + "\nDraw: " + draw;

            setScreen(new GameOverScreen(this));
        }
    }
}

```

GameOver

GameOver.pde

```

class GameOverScreen implements Screen {
    private PlayingScreen playingScreen;
}

```



```
public GameOverScreen(PlayingScreen playingScreen) {
    this.playingScreen = playingScreen;
}

public void draw() {
    background(#CCCCCC);

    playingScreen.draw();

    fill(#77000000);
    noStroke();
    rect(playingScreen.xOffset, playingScreen.yOffset, 6 *
    playingScreen.cellSize, 6 * playingScreen.cellSize);

    fill(#FFFFFF);
    textFont(headingFont);
    text("Game Over", width / 2, height / 4);

    textFont(textFont);
    String text = "The player using " +
    (playingScreen.currentPlayer instanceof Cross ? "crosses" :
    "circles") + " has won the game.";

    if (playingScreen.gameIsDraw) {
        text = "The game is a draw.";
    }

    text(text + "\n\nPress any key or click anywhere to play
    again." + playingScreen.highScoreText, width / 2, height / 2);
}

public void keyPressed() {
    setScreen(new PlayingScreen());
}

public void mouseClicked() {
    setScreen(new PlayingScreen());
}
}
```

SplashScreen

[SplashScreen.pde](#)

```
class SplashScreen implements Screen {
    public void draw() {
        background(#000000);
    }
}
```

```
fill(#FFFFFF);

textFont(headingFont);
text("TikTakToe", width / 2, height / 4);

textFont(textFont);
text("A two-player game\n\nPress any key or click anywhere to
start playing\n\n© 2022 Christian Weber", width / 2, height / 2);
}

public void keyPressed() {
setScreen(new PlayingScreen());
}

public void mouseClicked() {
setScreen(new PlayingScreen());
}
}
```

From:

<https://wiki-mathe-info.de/> - Wiki: Mathe und Info

Permanent link:

<https://wiki-mathe-info.de/info/sek-ii/q1/oop/l5-projekt?rev=1700999481>

Last update: **2023-11-26 12:51**

