

# Übungsblatt 5

## Lösungsvorschlag

---

### Aufgabe 1 Zugriff auf Zellen

Ich speichere das Spielfeld in der Klasse `Field` als String-Array. Ich kann einzelne Zellen lesen. Dazu schreibe ich die Methode `getChar(int x, int y)`, die das Zeichen an der Stelle  $(x, y)$  zurückgibt. Wenn die Stelle außerhalb ist, gebe ich ein Leerzeichen zurück.

```

1  final String[] field;
2
3  /**
4   * Creates a new Field.
5   * @param field the ASCII map as a String array
6   */
7
8  public Field(String[] field){
9      this.field = field;
10 }
11
12 /**
13  * Die Dateinamen der Bodengitterelemente, die direkt mit einer
14  * Rotation 0 verwendet werden können. Der Index ergibt sich
15  * aus der Summe der folgenden Zahlen:
16  * 1: In Richtung 0 (+1, 0) gibt es eine Verbindung.
17  * 2: In Richtung 1 (0, +1) gibt es eine Verbindung.

```

**Erklärung:** Ich speichere das Spielfeld-Array im Konstruktor. In `getChar` prüfe ich, ob die Koordinaten gültig sind. Wenn ja, gebe ich das Zeichen zurück, sonst ein Leerzeichen.

Ich habe auch die Methode `getCell(int x, int y)` gemacht. Sie macht das gleiche, gibt das Zeichen an der Stelle  $(x, y)$  zurück oder ein Leerzeichen, wenn es außerhalb ist.

```

1 /**
2  * Returns the character at position (x, y), or a space if out of bounds.
3  * @param x the column index
4  * @param y the row index
5  * @return the character at the given position
6  */
7 private char getChar(int x, int y){
8     if( y >= field.length || y < 0 || x >= field[y].length() || x < 0){
9         return ' ';
10    }
11    else {
12        String row = field[y];
13        return row.charAt(x);
14    }
15 };

```

**Erklärung:** Ich prüfe zuerst, ob die Koordinaten gültig sind. Wenn ja, gebe ich das Zeichen zurück. Sonst gebe ich ein Leerzeichen. Der Konstruktor speichert weiter das Spielfeld.

### Aufgabe 2 Nachbarschaft berechnen

Ich berechne für jede Zelle die Nachbarschaft. Die Methode

```
1 int getNeighborhood(int x, int y)
```

gibt eine Zahl zwischen 0 und 15. Die Zahl zeigt, welche Nachbarn da sind: rechts (1), unten (2), links (4), oben (8).

```
1 /**
2  * Calculates a neighborhood index for position (x, y).
3  * @param x the column index
4  * @param y the row index
5  * @return a number (0-15) describing the connections
6 */
7
8
9 private int getNeighborhood(int x, int y){
10     int index = 0;
11
12     if (getChar(x, y+1) != ' '){
13         index += 2;
14     }
15
16     if (getChar(x, y-1) != ' '){
17         index += 8;
18     }
19
20     if (getChar(x-1, y) != ' '){
21         index += 4;
22     }
23
24     if (getChar(x+1, y) != ' '){
25         index += 1;
26     }
27     return index;
28 }
```

### Aufgabe 3 Spielfeld aufbauen

Im Konstruktor gehe ich durch das Spielfeld. Ich berechne die Nachbarschaft und erstelle Game-Objects. Ich benutze zwei **for**-Schleifen:

- Äußere Schleife: geht über jede zweite Zeile ( $y \pm= 2$ )
- Innere Schleife: geht über jede zweite Spalte ( $x \pm= 2$ )

```
1 /** Ein Testfall, der alle Nachbarschaften enthält. */
2 static void test()
3 {
4     new GameObject.Canvas(5, 5, 96, 96);
5
6     // Einkommentieren, sobald Konstruktor vorhanden
7     Field field = new Field(new String[] {
8         "0-0-0-0 ", ,
9         "| | ", ,
10        "0 0-0-0 0",
11        "| | | | |",
12        "0-0-0-0-0",
13        "| | | | |",
14        "0 0-0-0 0",
15        " " | | ",
16        "0-0-0-0-0"
17    });
18
19
20    for (int y=0; y<field.field.length; y+=2 ) {
21        for(int x=0; x<field.field[y].length(); x+=2 ){
22            int selectedNeighborhoodIndex = field.getNeighborhood(x, y);
23            String gameObject = neighborhoodToFilename[selectedNeighborhoodIndex];
```

```
25             new GameObject(x/2, y/2, 0, gameObject);  
26     }  
27 };  
28 };  
29 };  
30 }
```

**Erklärung:** Für jede Zelle rufe ich `getNeighborhood` auf. Ich nehme die Zahl als Index in `neighborhoodToFilename`. Dann mache ich ein neues GameObject an der halbierten Position.

**Test:** Ich erstelle ein Testfeld und mache für alle Zellen GameObjects. So sehe ich, dass `getCell` und `getNeighborhood` funktionieren.