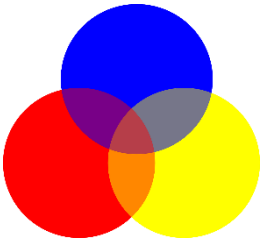
Interaktion von Objekten

In der Dokumentation der Klassen `Actor` und `World` (s. Blätter der Einführungsphase) könnt ihr einige Methoden finden, welche euch bei der Interaktion von Objekten behilflich sein können. Die Wichtigsten und ihre Anwendung werden euch nun kurz vorgestellt:

```
protected java.util.List getIntersectingObjects (java.lang.Class cls)
```



Wenn ihr diese Methode in der Klasse eines Objektes aufruft, liefert diese euch eine Liste der Objekte zurück, welche sich grafisch mit eurem überschneiden (hier reicht bereits ein einziges gemeinsames Pixel aus). Als Parameter übergebt ihr die Art der Objekte, die dabei berücksichtigt werden sollen. Angenommen ihr habt Objekte vom Typ `Waggon` und `Kugel` in eurem Projekt (beides Unterklassen von `Actor`). Nun wollt ihr alle Objekte vom Typ `Waggon` erhalten, welche sich mit eurem Objekt `Waggon` grafisch überschneiden. Der Aufruf in der Klasse sähe dann wie folgt aus:

```
List ueberschneidendeObjekte = getIntersectingObjectWaggon.class)
```

Um in dieser Liste zu navigieren, benötigt ihr nun einen Iterator:

```
Iterator i = ueberschneidendeObjekte.iterator()
```

Dieser Iterator bietet unter anderem zwei Methoden an:

`boolean hasNext()` und `E next()`

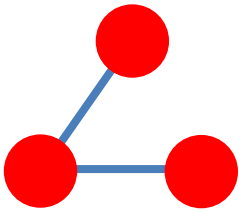
Die erste Methode gibt euch an, ob das aktuelle Element der Liste einen Nachfolger hat (also ob es noch ein weiteres Objekt in der Liste gibt). Der Iterator startet vor dem ersten Element und gibt somit beim Aufruf auf einer leeren Liste false zurück. Die zweite Methode liefert euch das nächste Element in der Liste

Achtung: existiert kein weiteres Element, wird eine Fehlermeldung geworfen.
Prüft also immer zuerst, ob es ein nachfolgendes Element in der Liste gibt!

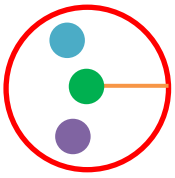
Der Rückgabetyp `E` steht hierbei als Platzhalter für den Typ der Listenelemente.

```
protected java.util.List getNeighbours (int distance, Boolean diagonal,
java.lang.Class cls)
```

Diese Methode liefert euch eine Liste der benachbarten Objekte mit einem maximalen Abstand zu eurem Objekt. Diesen maximalen Abstand legt ihr mit dem Parameter distance fest. Mit dem Parameter diagonal gebt ihr an, ob auch diagonale Abstände dabei berücksichtigt werden sollen. Der letzte Parameter gibt wieder an, welche Objekttypen dabei berücksichtigt werden sollen.



```
protected java.util.List getObjectsInRange (int radius, java.lang.Class cls)
```



Diese Methode liefert euch eine Liste der benachbarten Objekte innerhalb eines Radius um euer Objekt. Diesen Radius legt ihr mit dem Parameter radius fest. Der letzte Parameter gibt wieder an, welche Objekttypen dabei berücksichtigt werden sollen.

```
protected boolean intersects (Actor other)
```

Wenn ihr herausfinden wollt, ob sich euer Objekt mit einem bestimmten grafisch überschneidet, hilft euch diese Methode weiter. Übergebt als Parameter das andere Objekt.

Quellenverzeichnis:

Farbkreis – Quelle: https://pixabay.com/, Autor: ClkerFreeVectorImages (CC0)
Abstand von Objekten – Quelle: InfoSphere
Radius um Objekt – Quelle: InfoSphere

 angefertigt vom InfoSphere-Team