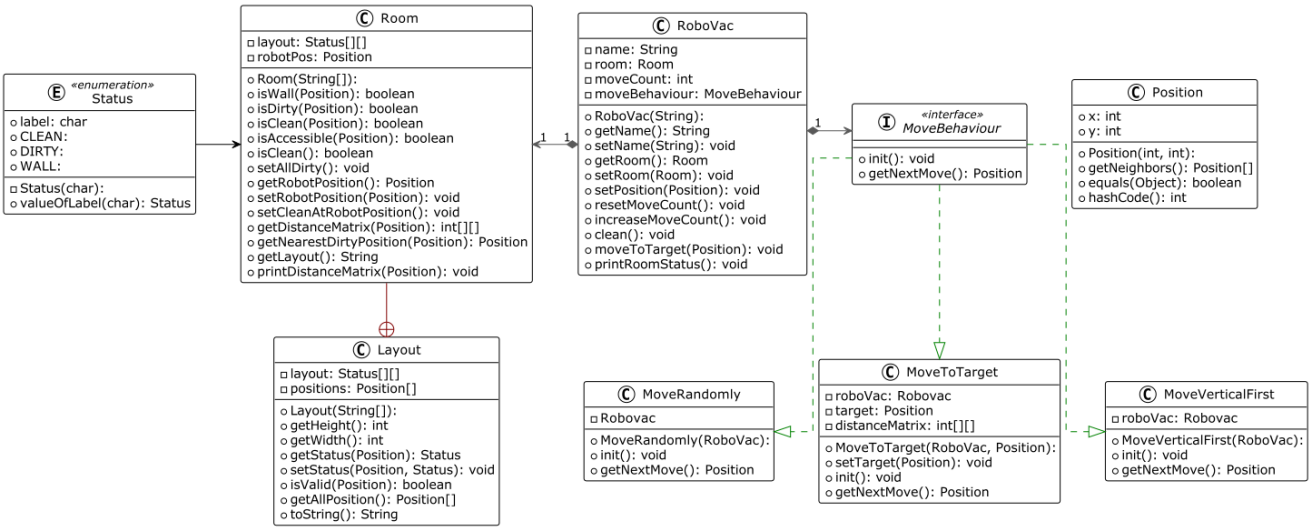
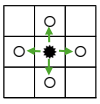
RoboVac 1.21

Überarbeiten Sie Ihre Implementierung des Staubsauger-Roboters. Orientieren Sie sich dabei an folgendem Klassendiagramm:



Beachten Sie dabei folgende Änderungen:

- Machen Klassen wurden umbenannt: *RoomLayout -> Room, RandomMove -> MoveRandomly*, …
- Es wurde eine neue Klasse *Position* eingeführt, die eine Position im Raum beschreibt. Interessant sind hier insbesondere:
  - die Methode *getNeighbors()*, die die Positionen aller 4 erreichbaren Nachbarfelder liefert
  - die Methode *equals()*.



- Die Klasse *Raum* wurde mit zahlreichen Methoden angereichert, z. B.
  - *isValid(Position p)*: prüft, ob eine gegebene Position innerhalb des Raumlayouts liegt
  - *isAccessible(Position p)*: prüft, ob eine gegebene Position vom Roboter erreichbar ist, d. h. es ist eine valide Position und keine Wand.
  - *setCleanAtRobotPosition()*: Säubert die Raumposition, an der sich der Roboter befindet.
- Das Room-Layout wurde in eine innere Klasse *Layout* ausgelagert. Dies hat den Vorteil, dass die *Room*-Klasse ausschließlich mit *Position*-Objekten arbeitet und die Umsetzung von Positionen auf Raum-Koordinaten konzentriert in der *Layout*-Klasse stattfindet.
- Der Roboter wurde um eine Funktion erweitert, um auf dem kürzesten Weg eine bestimmte Zielposition im Raum zu erreichen. Dazu wurden
  - in der Klasse *RoboVac* die Methode *moveToTarget()* eingeführt,
  - eine neue Bewegungsstrategie *MoveToTarget()* hinzugefügt und
  - in der *Room* Klasse eine Methode zur Berechnung einer Distanzmatrix vorgesehen.
- Die *move*-Methode im *MoveBehaviour*-Interface wurde zu einer *getNextMove*-Methode umgearbeitet. Damit liegt es nicht mehr in der Verantwortung dieser Methode, den Roboter auch tatsächlich zu bewegen, sondern nur die nächste Position vorzuschlagen.
  Hintergrund dieser Veränderung: Der Roboter kann nun diesen Vorschlag bewerten und entscheiden, ob er diese Bewegung tatsächlich ausführt oder eventuell auf eine andere Bewegungsstrategie umschaltet.
  Beispielsituation:
  - Der Roboter befindet sich in einer Ecke.
  - Der Vorschlag der aktiven *MoveVerticalFirst*-Strategie führt in auf ein Feld, das bereits gereinigt ist, was in der weiteren Folge zu unnötigen Umwegen führt.
  - Daher ändert der Roboter sein Verhalten: Er sucht sich den nächstliegenden Punkt, welcher nicht gereinigt ist und fährt diesen mittels *MoveToTarget*-Strategie direkt an, um dann wieder in die *MoveVerticalFirst*-Strategie zu wechseln und so weiterzuarbeiten.

- Implementieren Sie eine verbesserte Umsetzung der *clean*-Methode:
    - Der Roboter soll zunächst nach der *moveVerticalFirst*-Strategie arbeiten. Er erhält hier bei jedem *getNextMove()* einen Vorschlag für den nächsten Schritt.
    - Führt ihn dieser Vorschlag auf ein schmutziges Feld, so vollzieht er den Schritt.
    - Würde ihn dieser Vorschlag auf ein gereinigtes Feld führe, so nutzt er die *moveToTarget*-Methode, um die nächstliegenden schmutzige Position direkt anzufahren (Methodenaufruf). Dabei wird ein Wechsel der Bewegungsstrategie durchgeführt.
    - Wenn er die Zielposition erreicht hat, so wird in der clean-Methode mit der *moveVertical-First*-Strategie fortgefahren.
    - Die Methode endet erst, wenn der gesamte Raum gereinigt wurde.

Sie können und sollen bei Bedarf weitere Klassenmember (Methoden, Eigenschaften) einführen.