# JUnit Test van de Class Map

In de test werd gekeken naar de volgende methodes:

* playerAllowedToMoveUp()
* playerAllowedToMoveDown()
* playerAllowedToMoveLeft()
* playerAllowedToMoveRight()

Deze methodes zijn belangrijk voor de collision check van de Player. Zonder deze methode’s wordt er niet gekeken of de player wel die kant op mag lopen.

De eerder genoemde methodes bevatten een if/else-statement waarin gekeken wordt of de positie van de player voldoet om een bepaalde richting op te lopen. Zie de volgende twee condities.

x >= 0 && x <= Game.HORIZONTAL\_AMOUNT - 1

..waarin ‘x’ de x-coördinaat is en ‘y’ de y-coördinaat van de player op dat moment wanneer de methode aangeroepen wordt. ‘Game.HORIZONTAL\_AMOUNT’ is het aantal tiles wat zich in de breedte bevindt van het speelveld.

In de eerste conditie wordt gekeken of de x-coördinaat groter is of gelijk aan 0. Dit wil zeggen dat de speler in het speelveld moet staan. Dit is ook zo bij de tweede conditie, alleen beginnen de coördinaten bij 0, dus moet er één van de breedte van het speelveld afgetrokken worden. Hetzelfde gebeurd ook voor de volgende twee condities, maar dan voor het y-coördinaat.

Als de speler niet in het speelveld mocht staat, kan deze niet lopen omdat er dan false wordt geretourneerd. Mocht de speler wel in het speelveld staan dan wordt de eigenschap isSolid van de tile waar de speler naar toe kijkt inverted teruggegeven. Als isSolid de waarde true heeft dan kan de speler alsnog niet die kant op lopen, aangezien hij dan door bijv. een muur zou lopen. Als isSolid de waarde false heeft, wordt dit inverted geretourneerd en wordt er toegestaand dat de speler die kant op loopt.

In de test die op de eerste methode uitgevoerd is, playerAllowedToMoveUp(), wordt gekeken op plaats (0,0) in het speelveld of de speler omhoog mag lopen. Dit is natuurlijk niet het geval en daarom wordt als verwachte uitkomst false ingesteld. Dezelfde methode wordt ook drie plaatsen in het speelveld naar beneden getest, want daar hoort de speler wel omhoog te kunnen lopen. Hierdoor worden beide uitkomsten getest en dit betekent dat de decision coverage gedekt is. (Bijna)