# A\*

De gekozen afstandsfunctie bevindt zich in Map.cpp en heet findPath().

# Eindigetoestandsautomaten

**Klasse**: Artist

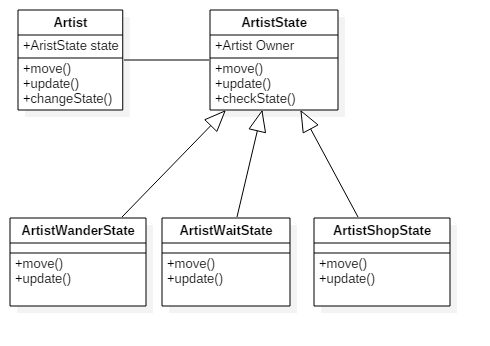
**Toestanden**:

* ArtistWanderState
  + In deze toestand loopt de artist willekeurig rond door de stad.
  + Bij elke stap verliest de artist 20 euro. Als de artist minder dan 100 euro op zak heeft en te horen krijgt van de manager dat deze naar hem toe komt transioneert zijn state naar ArtistWaitState
* ArtistWaitState
  + In deze toestand staat de artist stil en wacht hij op een bezoek van de manager
  + Als de manager hem heeft bereikt transioneert zijn state naar ArtistWanderState
* ArtistShopState
  + In deze toestand loopt de Artiest naar een van de 4 winkels.
  + Als de Artist de winkel heeft bereikt wacht hij 3 periodes en transioneert dan zijn state naar ArtistWanderState

**Klasse**: Manager

**Toestanden**:

* ManagerWanderState
  + In deze toestand loopt de manager willekeurig rond door de stad.
  + Elke stap checkt wordt er gecheckt of een van de artiesten de manager nodig heeft. Als dit zo is transioneert de manager naar ManagerVisitArtistState.
* ManagerVisitArtistState
  + In deze toestand loopt de manager naar de artist die hulp nodig heeft.
  + Als de manager bij de artiest is aangekomen transioneert de state van de artiest naar ArtistShopState en de state van de manager naar ArtistWanderState.



# Force-driven entities

**Klasse:** Fan

**Krachten:**

* CohesionVector
  + CohesionVector wordt berekend door voor iedere nabije fan een vector naar die fan toe te berekenen en deze bij elkaar op te tellen tot 1 vector.
* SeparationVector
  + SeparationVector wordt berekend door voor iedere nabije fan een vector van die fan af te berekenen en deze bij elkaar op te tellen tot 1 vector.
* AlignmentVector
  + AlignmentVector wordt berekend door voor de richtingsvectoren van nabije fans bij elkaar op te tellen tot 1 vector.
* AttractedToArtistsVector
  + AttractedToArtistsVector wordt berekend door voor iedere artiest een vector naar die artiest toe te berekenen en deze bij elkaar op te tellen tot 1 vector. (Als een fan niet aangetrokken is tot een bepaalde artiest kunnen deze vectoren ook vectoren van de artiesten af worden.)
* SteeringVector
  + Steering wordt berekend door een vierkant te tekenen in de richting waar de fan naartoe wil en daarop een willekeurig punt te pakken als vector.

Al deze vectoren worden bij elkaar opgeteld om een vector te creeëren die de richting van de Fan bepaald.

# Genetisch Algoritme

Aan het begin van de eerste simulatie genereerd het programma 100 fans met ieder een random chromosoom.

Voor de fitness van de fans wordt de tijd die ze in de buurt van artiesten hebben doorgebracht gebruikt.

Aan het einde van iedere simulatie wordt een nieuwe populatie fans gegenereerd middels Fitness Proportionate Selection. Dit algoritme is geïmplementeerd met behulp van [deze](https://en.wikipedia.org/wiki/Fitness_proportionate_selection) webpagina.

Ouders worden bepaald met behulp van roulette selection. Ieder fan heeft een kans om gekozen te worden als ouder maar de fans met een hogere fitness hebben een hogere kans.

Nakomelingen worden gegenereerd door de chromosomen van 2 ouders op een willekeurig punt te snijden en aan elkaar te koppelen tot een nieuw chromosoom.

De nieuwe populatie is 2/3e van de vorige populatie.