TWENTE'S GOT TALENT

Paintcode

Programmeer handleiding

Uitleg van aan te roepen en over te erven functies en codevoorbeelden

mede mogelijk gemaakt door provincie Overijssel

Basis

In dit deel van de handleiding wordt ingegaan op de basis van de Template. Hoe is het opgezet en wat kan men programmeren.

Nieuw project

Zodra de template als nieuw project geopend wordt, zijn er 2 code-bestanden, namelijk [projectnaam].cs en [projectnaam].designer.cs. Het designer-bestand bevat code die nodig is om de nieuw gemaakte bot te laten werken in het programma PaintCode. Het is niet de bedoeling om in dit bestand wijzigingen aan te brengen. Alle wijzigingen in de AI kunnen worden ingevoerd in het bestand [projectnaam].cs.

Op ieder gewens moment kan men op F5 drukken om te kijken of de funcionaliteit van de bot werkt. Het programma PaintCode wordt dan gestart, met een aantal van de originele bots en 1 bot van het nieuwe type. Dit type wordt duidelijker gemaakt door z'n naam die er boven zweeft.

Aanroepbare basisfuncties

PaintBots hebben een aantal standaard functies die altijd aangeroepen kunnen worden. Deze functies kunnen een vraag of informatie zijn, andere zorgen ervoor dat de bot iets gaat doen. Sommige functies die de bot iets laten doen, roepen daarmee weer functies aan die je kunt overschrijven (zie 1.3).

Een functie-aanroep is niet moeilijk. Alle basisfuncties die je zelf aan kunt roepen worden hier beneden genoemd en uitgelegd.

AantalHitsVoorIkAfBen

De functie AantalHitsVoorIkAfBen() geeft terug hoe vaak de bot nog geraakt kan worden voordat je bot uit het spel is. Het kan zijn dat je wilt kijken of er nog maar 1 hit over is om te bepalen of je liever gaat vluchten of dat je gaat aanvallen. Daarvoor kun je bijvoorbeeld intypen:

```
if (AantalHitsVoorIkAfBen() == 1)
```

Hieronder kun je dan een aanroep naar een functie doen.

NogDichterbijKomen

De functie NogDichterbijKomen roep je aan op het moment dat je een vijand al genaderd bent. Je kunt hem daarom het beste gebruiken in de functie DoelGenaderd (zie onderdeel XXX). De aanroep gaat als volgt:

```
NogDichterbijKomen(15, vijand);
```

Dit zorgt ervoor dat je bot de vijand zal proberen te naderen tot een afstand van 15. De normale afstand om te naderen is 90% van de vuurafstand, die standaard 140 is.

Ontwijken

De functie Ontwijken roep je aan als je weg wil lopen van een bepaalde bot. Er wordt dan gekeken wat de toekomstige positie van die bot zal zijn, en daar zal jouw bot dan van weg vluchten. De aanroep is vrij simpel:

```
Ontwijken(vijand);
```

Richten
De functie Richten zorgt ervoor dat je bot draait naar een opgegeven positie en zijn wapen daarop gericht houdt. De opgegeven positie is een Vector. De aanroep kan daarom op 2 manieren:

```
Richten(vijand.Position);
```

Deze manier kun je gebruiken om op een bepaalde bot te richten.

```
Richten(richting);
```

Deze manier kun je gebruiken in de functie BenGeraakt, om te draaien naar de richting waaruit je geraakt bent.

Rondlopen
De functie Rondlopen zorgt ervoor dat je bot gewoon willekeurig door de arena loopt. Als je deze functie niet gebruikt en je bot ziet geen tegenstanders en er gebeurt verder ook niets met je bot, dan blijft je bot gewoon staan en gebeurt er dus helemaal niets. Het is daarom verstandig om, als je bot geen tegenstanders ziet, altijd Rondlopen aan te roepen. De aanroep gaat simpelweg:

```
RondLopen();
```

Schieten
De functie schieten doet niets anders dan ervoor zorgen dat je bot 1 verfkogel afvuurt in de richting waar hij naar kijkt. Hij zal niet proberen specifiek ergens op te richten. LET OP: er zit een minimale tijd tussen twee schoten. Schiet dus alleen als je een tegenstander ziet, anders kun je misschien niet meer schieten wanneer het moet!

```
Schieten();
```

Verstoppen
De functie Verstoppen zorgt ervoor dat je bot probeert om naar een plek te lopen waar een vijand hem niet kan zien, bijvoorbeeld door achter een krat te lopen. De aanroep is vrij simpel:

```
Verstoppen(vijand);
```

VijandZoeken
De functie VijandZoeken zorgt ervoor dat je bot gaat kijken of hij een vijand ziet. Als hij een vijand ziet, wordt de overschrijfbare functie VijandGezien aangeroepen. Ziet hij geen vijanden roept hij GeenVijandenGevonden aan. LET OP: deze functie stuurt je bot niet, de bot kijkt in de richting waar hij naartoe staat en laat je weten of er, binnen de zichtafstand een bot in het gezichtsveld staat, niet meer en niet minder. De aanroep:

```
VijandZoeken();
```

Volgen
De functie Volgen zorgt ervoor dat je bot een andere bot die hij gezien heeft gaat volgen tot hij hem uit het zicht verliest. De aanroep is als volgt:

```
Volgen(vijand);
```

ZietVijandMij
De functie ZietVijandMij vraagt of een vijand die je gezien hebt, jou ook ziet. In tegenstelling tot VijandZoeken, roept ZietVijandMij geen functie aan, maar geeft hij een waar of niet waar terug.

```
if (ZietVijandMij(vijand))
```

Overschrijfbare basisfuncties

PaintBots hebben een aantal overschrijfbare basisfuncties waar iemand zonder programmeerervaring direct mee aan de slag kan. Om een functie te overschrijven typt men gewoon buiten een bestaande functie “public override “. Zodra de spatie als laatste is ingevoerd, zou een pulldown-menu moeten verschijnen met de functies die mogelijk zijn.

Run

De functie Run is al standaard overschreven. Deze functie is de functie die continue aangeroepen wordt tijdens het draaien van het programma. Als deze functie leeg blijft, doe de bot helemaal niets. Om ervoor te zorgen dat je bot alleen maar rondloopt en verder niets onderneemt, kun je bijvoorbeeld de functie als volgt maken:

```
public override void Run()
{
    RondLopen();
}
```

BenGeraakt

De functie BenGeraakt wordt aangeroepen als de bot door een kogel geraakt wordt. De functie krijgt een Vector mee die aangeeft waar de kogel vandaan kwam. Door deze functie te overschrijven kun je ervoor zorgen dat je bot hierop reageert.

De functie ontvangt gegevens over de richting waarvandaan je geraakt bent, een Vector met de naam “uitrichting”. Een mogelijkheid om van BenGeraakt te maken is:

```
public override void BenGeraakt(Vector uitrichting)
{
   Richten(uitrichting);
}
```

DoelGenaderd

De functie DoelGenaderd wordt aangeroepen als je opdracht hebt gegeven een bot te volgen en er dichtbij genoeg bent. In deze functie kun je besluiten wat je in dat geval wilt doen. De meest simpele actie is schieten. In dit geval maak je DoelGenaderd:

```
public override void DoelGenaderd(PaintBot doel)
{
    Schieten();
}
```

DoelInVizier

De functie DoelInVizier wordt aangeroepen als je opdracht hebt gegeven om te richten en je bot inmiddels gericht is op dat doel. De meest eenvoudige actie is ook nu om te schieten, in dat geval maak je de functie:

```
public override void DoelInVizier(Vector doel)
{
    Schieten();
```

```
}
```

GeenVijandGevonden

GeenVijandGevonden wordt aangeroepen als je opdracht hebt gegeven om te zoeken naar vijanden en er geen vijand in je gezichtsveld aanwezig is. De meest simpele actie is nu om gewoon rond te gaan lopen:

```
public override void GeenVijandGevonden()
{
    RondLopen();
}
```

Verstopt

De functie Verstopt wordt aangeroepen op het moment dat de bot een obstakel (krat) tussen zichzelf en de bot waarvoor hij zich verstopt heeft zitten. Dit kan dus alleen voorkomen op het moment dat je Verstoppen ergens aanroept. Het meest logische is om nu weer normaal rond te gaan lopen:

```
public override void Verstopt()
{
    RondLopen();
}
```

VijandGezien

De functie VijandGezien wordt aangeroepen op het moment dat de bot bij VijandZoeken een vijand heeft gezien. De functie krijgt gegevens over de vijand mee. Een mogelijkheid is om vervolgens de vijand te gaan volgen:

```
public override void VijandGezien(PaintBot vijand)
{
    Volgen(vijand);
}
```

Een stap verder

Met de informatie uit het hoofdstuk Basis kun je al een beetje aanpassen aan AI. Het blijft alleen heel erg beperkt. Op het moment dat je wat handiger omgaat met informatie die je krijgt, kun je je bot slimmer maken. In dit hoofstuk kun je lezen welke informatie je op kunt vragen en hoe je ermee kunt werken.

Beschikbare informatie

Elke bot heeft bepaalde informatie beschikbaar, informatie waar je in je strategie misschien rekening mee kunt houden. Hieronder vind je een tabel met gegevens die je kunt opvragen:

Eigenschap	**Betekenis**	**Type**
Leeft	Geeft aan of de vijand wel of niet nog in het spel is.	Waar / nietwaar
Mass	De massa van de bot (iedereen kan z'n eigen massa aanpassen)	Gebroken getal
MaxForce	De kracht van de bot (kan ook iedereen zelf aanpassen)	Gebroken getal
MaxSpeed	De maximale snelheid van de bot (kan ook iedereen zelf aanpassen)	Gebroken getal
Position	De daadwerkelijke positie van de tegenstander	Vector
SchietAfstand	De afstand waarop de tegenstander gaat schieten	Gebroken getal
ZichtAfstand	De afstand die de tegenstander kijkt	Gebroken getal

Gebruik van informatie

Sommige informatie kun je gebruiken om tot besluiten te komen. Zo kun je kijken of jouw bot sneller is dan een andere. Je kunt ook ervoor zorgen dat je bot net buiten de zichtafstand van een tegenstander blijft. Hieronder een paar voorbeelden die je zou kunnen gebruiken:

```
if (MaxSpeed > vijand.MaxSpeed)
      NogDichterbijKomen(vijand.SchietAfstand + 4, vijand);
```

Probeer zelf maar dingen uit.

Tweaking

Het merendeel van wat hiervoor beschreven is, bestaat uit redelijk vaststaande dingen. Heel veel maakt gebruik van eigenschappen die in params.ini beschreven staan. Maar het is ook mogelijk om zelf te bepalen wat de waarden moeten zijn. De waarden die je wilt hebben moet je opgeven in de functie Init die al leeg opgegeven staat.

```
this._mass = 1;          // massa van de bot
this._maxforce = 1;      // maximale kracht van de bot
this._maxSpeed = 1;      // maximale snelheid van de bot
this._turnrate = 1;      // maximale draaisnelheid van de bot
this.FiringRange = 140; // afstand waarop de bot denkt te kunnen schieten
this.SightRadius = 200; // afstand die de bot kan overzien
```

Hoewel het misschien slim lijkt om bijvoorbeeld de SightRadius op bijna oneindig te zetten, kan het ervoor zorgen dat je bot achter een bot aangaat die heel ver weg is, terwijl er 1 dichterbij op hem staat te schieten. FiringRange heel hoog maken betekent dat de kans groot is dat de andere bot weg is voor de kogel aankomt. Heel laag zetten betekent dat hij dichtbij zal moeten komen om te schieten, waardoor de ander hem misschien eerder raakt.

Bot voorbeelden

Hieronder vind je een aantal voorbeelden van bots zoals die te programmeren zijn. Sommige zijn heel simpel, andere zijn ingewikkelder.

Simpele bot

```
using PaintAI;

namespace MorfBot
{
    public partial class MorfBot : PaintBot
    {
        public void Init()
        {

        }

        protected override void Run()
        {
            VijandZoeken();
        }

        protected override void GeenVijandGevonden()
        {
            RondLopen();
        }

        protected override void DoelInVizier(Vector doel)
        {
            Schieten();
        }

        protected override void DoelGenaderd(PaintBot doel)
        {
            NogDichterbijKomen(40, doel);
        }

        protected override void VijandGezien(PaintBot vijand)
        {
            Volgen(vijand);
            Richten(vijand.Position);
        }
    }
}
```

Een andere simpele bot

```
using PaintAI;

namespace SimpleBot
{
    public partial class SimpleBot : PaintBot
    {
        public void Init()
        {
```

```
        }

        protected override void Run()
        {
            RondLopen();
            VijandZoeken();
        }

        protected override void VijandGezien(PaintBot vijand)
        {
            Richten(vijand.Position);
        }

        protected override void DoelInVizier(Vector doel)
        {
            Schieten();
        }
    }
}
```

Een iets ingewikkelder bot

```
using PaintAI;

namespace Smarter
{
    public partial class Smarter : PaintBot
    {
        public void Init()
        {

        }

        protected override void Run()
        {
            VijandZoeken();
        }

        protected override void VijandGezien(PaintBot vijand)
        {
            if (ZietVijandMij(vijand))
            {
                if (AantalHitsVoorIkAfBen() == 1)
                {
                    Ontwijken(vijand);
                    Richten(vijand.Position);
                    Schieten();
                }
                else
                {
                    RondLopen();
                    Volgen(vijand);
                }
            }
            else
            {
                Volgen(vijand);
            }
        }

        protected override void DoelGenaderd(PaintBot doel)
        {
```

```
            Richten(doel.Position);
        }

        protected override void DoelInVizier(Vector doel)
        {
            Schieten();
        }

        protected override void GeenVijandGevonden()
        {
            RondLopen();
        }
    }
}
```

Alles gebruiken

```
using PaintAI;

namespace UsingAll
{
    public partial class UsingAll : PaintBot
    {
        public void Init()
        {
            this._mass = 1;
            this._maxforce = 1;
            this._maxSpeed = 3;
            this.FiringRange = 100;
            this.SightRadius = 160;
            this._turnrate = 2;
        }

        protected override void Run()
        {
            VijandZoeken();
        }

        protected override void VijandGezien(PaintBot vijand)
        {
            if (ZietVijandMij(vijand))
            {
                if (AantalHitsVoorIkAfBen() == 1)
                {
                    Verstoppen(vijand);
                    Ontwijken(vijand);
                    Richten(vijand.Position);
                    Schieten();
                }
                else
                {
                    Verstoppen(vijand);
                    Richten(vijand.Position);
                }
            }
            else
            {
                Volgen(vijand);
            }
```

```
        }

        protected override void DoelInVizier(Vector doel)
        {
            Schieten();
        }

        protected override void DoelGenaderd(PaintBot doel)
        {
            Schieten();
        }

        protected override void GeenVijandGevonden()
        {
            RondLopen();
        }


    }
}
```