## Lesbrief 7 Classes

| * Probeer de voorbeelden uit. * Voer de opdrachten uit. |
| --- |

Een klasse, ook wel Class genoemd, is een datatype net zoals int, float, string etc. Een int representeert een geheel getal, een float representeert een komma getal, een string representeert een woord. Een int, float of string, ook wel primitieve types genoemd, zijn niet voldoende om complexe zaken te beschrijven zoals een boek of een telefoon.

Om complexe zaken te beschrijven maak je gebruik van Objecten of **Classes**.

**Voorbeeld**

Stel je wilt een programma schrijven dat iets doet met een boek. Hoe zou je een boek omschrijven?

Allereerst begin je met de kenmerken van een boek die voor jouw programma relevant zijn, bijvoorbeeld:

* Een titel
* Een auteur
* Een uitgever
* Het aantal pagina’s

Als we bovenstaande omschrijving van een boek in ons programma willen schrijven dan voeren we de volgende stappen uit:

### Maak een class Boek

De *class* boek bevat alleen de kenmerken van een boek. Voor het beschrijven van de kenmerken maak je gebruik van variabelen en datatypen. De *class* is dus alleen maar een beschrijving van het object boek.

Om een class te maken in Unity doe je het volgende:

* Selecteer van het menu *Add -> New File*.
* Selecteer *Empty Class*.
* En geef je class een naam, bijvoorbeeld Boek. LET OP in C# beginnen namen van Classes met een Hoofdletter.

Je zult merken dat C# naast de Program.cs ook een bestand genaamd Boek.cs heeft gemaakt.

In de *Class* Boek typ je de kenmerken van een boek.

In **Boek.cs**

| using System;  namespace Bibliotheek {  public class Boek  {  public string titel;  public string auteur;  public string uitgever;  public int aantalPaginas;  } } |
| --- |

Nu heb je een **definitie** gemaakt van een Boek.

### 2. Maak een instantie van een Boek

In de Program.cs kun je vervolgens een “echt” boek definiëren zoals Harry Potter of Lord of the Rings. Met andere woorden je maakt een instantie van een boek en geeft de eigenschappen van het boek een waarde.

In **Program.cs**

| using System;  namespace Bibliotheek {  class Program  {    static void Main(string[] args)  {  Boek boek1 = new Boek();  boek1.titel = "Harry Potter";  boek1.auteur = "JK Rowling";  boek1.uitgever = "Uitgeverij de Harmonie";  boek1.aantalPaginas = 500;   Console.WriteLine(boek1.titel);   Console.ReadLine();  }  } } |
| --- |

In bovenstaande code definieer je het boek Harry Potter, door

* Eerst een instantie van een boek te maken. Dit doe doe je op dezelfde manier als wanneer je een variabele maakt vb. *int getal*. Je schrijft eerst het datatype op en dan de variabele naam: Boek boek1.   
  Omdat we te maken hebben met een Object gebruik je het keyword **new** om de variabele boek1 een instantie van het Boek te geven.

| Boek boek1 = new Boek(); |
| --- |

* Vervolgens kun je de kenmerken van het boek een waarde geven. Dit doe je door de variabele op te schrijven gevolgd door een “.” en vervolgens het kenmerk en de bijbehorende waarde:

| boek1.titel = "Harry Potter"; boek1.auteur = "JK Rowling"; boek1.uitgever = "Uitgeverij de Harmonie"; boek1.aantalPaginas = 500; |
| --- |

Je kunt ook meerdere boeken creëren:

| using System;  namespace Bibliotheek {  class Program  {    static void Main(string[] args)  {  Boek boek1 = new Boek();  boek1.titel = "Harry Potter";  boek1.auteur = "JK Rowling";  boek1.uitgever = "Uitgeverij de Harmonie";  boek1.aantalPaginas = 500;   Boek boek2 = new Boek();  boek2.titel = "Lord of the Rings";  boek2.auteur = "Tolkien";  boek2.uitgever = "Uitgeverij Unieboek-Het Spectrum";  boek2.aantalPaginas = 700;   Console.WriteLine(boek1.titel);  Console.WriteLine(boek2.titel);   Console.ReadLine();  }  } } |
| --- |

| **Opdracht**   * Maak een klasse Mobiel. Geef deze klasse de volgende kenmerken:   + Type   + Model   + OS (operating system) * Maak vervolgens een programma die een instantie van een Mobiel met de volgende waarden creëert:   + Type: iPhone   + Model: iPhone XS   + OS: Apple IOS |
| --- |

### De Constructor

Een constructor is een methode die dezelfde naam heeft als de klasse. Bij het creëren van een instantie, zoals:

| Boek boek1 = new Boek(); |
| --- |

is de constructor methode de **eerste** methode die wordt aangeroepen.

Voorbeeld **Boek.cs**

| using System;  namespace Bibliotheek {  public class Boek  {  public string titel;  public string auteur;  public string uitgever;  public int aantalPaginas;   public Boek()  {  *//dit is de constructor*  }  } } |
| --- |

Aan de constructor kun je ook parameters meegeven. Deze parameters geven de kenmerken van een Class meteen een waarde.

**Boek.cs**

| using System;  namespace Bibliotheek {  public class Boek  {  public string titel;  public string auteur;  public string uitgever;  public int aantalPaginas;   public Boek(string nTitel, string nAuteur, string nUitgever, int aantal)  {  titel = nTitel;  auteur = nAuteur;  uitgever = nUitgever;  aantalPaginas = aantal;  }  } } |
| --- |

Hierdoor kun je in je programma het maken van instanties van boeken vereenvoudigen:

**Program.cs**

| using System;  namespace Bibliotheek {  class Program  {    static void Main(string[] args)  {  Boek boek1 = new Boek("Harry Potter", "JK Rowling", "Uitgeverij de Harmonie", 500);  Boek boek2 = new Boek("Lord of the Rings", "Tolkien", "Uitgeverij Unieboek-Het Spectrum", 700);   Console.WriteLine(boek1.titel);  Console.WriteLine(boek2.titel);   Console.ReadLine();  }  } } |
| --- |

Nadat je een instantie of Object hebt aangemaakt kun je de kenmerken alsnog wijzigen, mits de “Access Level” (zie hieronder) dit toestaat.

| using System;  namespace Bibliotheek {  class Program  {    static void Main(string[] args)  {  Boek boek1 = new Boek("Harry Potter", "JK Rowling", "Uitgeverij de Harmonie", 500);  Boek boek2 = new Boek("Lord of the Rings", "Tolkien", "Uitgeverij Unieboek-Het Spectrum", 700);   Console.WriteLine(boek1.titel);  Console.WriteLine(boek2.titel);   boek2.titel = "The Hobbit";   Console.WriteLine(boek2.titel);   Console.ReadLine();  }  } } |
| --- |

| **Opdracht**   * Verander de klasse Mobiel. Maak een constructor die de kenmerken als parameter meekrijgt. Pas ook de Program.cs aan. |
| --- |

### Programmeren met Objecten

Je kunt objecten toekennen, vergelijken of met een loop doorheen gaan net zoals je dat bij de primitieve types (int, float, string etc) hebt gedaan. Je kunt objecten ook in een array stoppen.

Voorbeeld

**Program.cs**

| using System;  namespace Bibliotheek {  class Program  {   static void Main(string[] args)  {  Boek boek1 = new Boek("Harry Potter", "JK Rowling", "Uitgeverij de Harmonie", 500);  Boek boek2 = new Boek("Lord of the Rings", "Tolkien", "Uitgeverij Unieboek-Het Spectrum", 700);   Console.WriteLine(boek1.titel);  Console.WriteLine(boek2.titel);   Boek[] boeken = new Boek[2];  boeken[0] = boek1;  boeken[1] = boek2;   for (int i = 0; i < 2; i ++)  {  if (boeken[i].titel == "Harry Potter")  {  Console.WriteLine(boeken[i].titel + ", dat is een mooi boek");  }  else if (boeken[i].titel == "Lord of the Rings")  {  Console.WriteLine(boeken[i].titel + " is een fantastisch boek");  }  }   Console.ReadLine();  }  } } |
| --- |

| **Opdracht**   * Breidt jouw Mobiel programma uit:   + Voeg de oprichter of CEO van het bedrijf toe aan de definitie van een mobiel.   + Maak 3 instanties van Mobieltjes (bv iPhone, Samsung, Google Pixel) en stop deze in een array.   + Druk van iedere instantie van een Mobiel, de oprichter of CEO van het desbetreffende bedrijf af, bijv, bij de iPhone druk je Steve Jobs af. |
| --- |

### Object Method

In een klasse kun je naast variabelen, ook methoden definieren.

Voorbeeld

Stel we willen van een boek de prijs incl. btw weten.

**Boek.cs**

| using System;  namespace Bibliotheek {  public class Boek  {  const float Btw = 0.21f;  public float prijs = 3.95f;  public string titel;  public string auteur;  public string uitgever;  public int aantalPaginas;   public Boek(string nTitel, string nAuteur, string nUitgever, int aantal, float nPrijs)  {  titel = nTitel;  auteur = nAuteur;  uitgever = nUitgever;  aantalPaginas = aantal;  prijs = nPrijs;   }   public float BTW(int aantal)  {  return aantal \* prijs + ((aantal \* prijs) \* Btw);  }  } } |
| --- |

In het Programma maken we 2 boeken aan. Vervolgens roepen we van elke instantie de methode aan die de prijs retourneert.

**Program.cs**

| using System;  namespace Bibliotheek {  class Program  {   static void Main(string[] args)  {  Boek boek1 = new Boek("Harry Potter", "JK Rowling", "Uitgeverij de Harmonie", 500, 4.95f);  Boek boek2 = new Boek("Lord of the Rings", "Tolkien", "Uitgeverij Unieboek-Het Spectrum", 700, 5.95f);   Console.WriteLine("Hoeveel boeken " + boek1.titel + " wilt u kopen?");  int aantalHP = Convert.ToInt32(Console.ReadLine());  Console.WriteLine("Hoeveel boeken " + boek2.titel + " wilt u kopen?");  int aantalLOTR = Convert.ToInt32(Console.ReadLine());   float totaalPrijsHP = boek1.BTW(aantalHP);  float totaalPrijsLOTR = boek2.BTW(aantalLOTR);   Console.WriteLine(aantalHP + " boeken Harry Potter kosten in totaal " + totaalPrijsHP);  Console.WriteLine(aantalLOTR + " boeken LOTR kosten in totaal " + totaalPrijsLOTR);   Console.ReadLine();  }  } } |
| --- |

### Access levels van variabelen

Alle variabelen en methoden in C# kennen een **access** level. **Access** levels geven aan of een variabele of methode ook buiten de klasse gebruikt mag worden.

Enkele access levels zijn:

| Public | Mag door andere klassen gebruikt worden. |
| --- | --- |
| Private | Mag alleen in de klasse van definitie gebruikt worden. |
| Protected | Mag alleen in de klasse van definitie of in een subklasse worden gebruikt. |

In het eerste voorbeeld van de Class Boek...

| using System;  namespace Bibliotheek {  public class Boek  {  public string titel;  public string auteur;  public string uitgever;  public int aantalPaginas;   public Boek()  {  *//dit is de constructor*  }  } } |
| --- |

...zijn alle variabelen “public” gedefinieerd. Dit houdt in dat deze variabelen direct door de **Program.cs** gewijzigd mogen worden:

| Boek boek1 = new Boek();  boek1.titel = "Harry Potter";  boek1.auteur = "JK Rowling";  boek1.uitgever = "Uitgeverij de Harmonie";  boek1.aantalPaginas = 500; |
| --- |

Als we gebruik maken van Constructors, dan geven we de variabelen in de Class reeds een waarde bij het instantiëren.

In Object georiënteerde programmeertalen zoals C# is het gebruikelijk dat de variabelen van een Class private zijn. De variabelen van een Class krijgen een waarde door middel van de **Constructor** of door middel van een **Set** methode. Voor het opvragen van een waarde van een variabele uit een Class gebruik je een **Get** methode. Het gebruik van Get - en Set methoden wordt ook wel **Property** genoemd.

Voorbeeld **Boek.cs**:  
Private variabelen, geen Property, waarden worden via de Constructor gezet.

| using System;  namespace Bibliotheek {  public class Boek  {  const float Btw = 0.21f;  private float prijs;   private string titel;  private string auteur;  private string uitgever;  private int aantalPaginas;   public Boek(string nTitel, string nAuteur, string nUitgever, int aantal, float nPrijs)  {  titel = nTitel;  auteur = nAuteur;  uitgever = nUitgever;  aantalPaginas = aantal;  prijs = nPrijs;  }   public float Btw(int aantal)  {  return aantal \* prijs + ((aantal \* prijs) \* Btw);  }   } } |
| --- |

In het programma geef je een instantie van een boek een waarde via de constructor:

In **Program.cs**

| Boek boek1 = new Boek("Harry Potter", "JK Rowling", "Uitgeverij de Harmonie", 500, 4.95f);  Boek boek2 = new Boek("Lord of the Rings", "Tolkien", "Uitgeverij Unieboek-Het Spectrum", 700, 5.95f); |
| --- |

Maar kun je niet meer de waarde van een variabele van een instantie van een Boek uitlezen of aanpassen:

| // Dit kan niet meer: boek1.titel en boek2.titel, omdat titel een private  // variabele is.  Console.WriteLine("Hoeveel boeken " + boek1.titel + " wilt u kopen?"); Console.WriteLine("Hoeveel boeken " + boek2.titel + " wilt u kopen?");  // Dit kan ook niet meer, omdat boek1.prijs een private variabele is.  boek1.prijs = 6.95f |
| --- |

### Properties

Wil je in jouw programma de waarde van een variabele in een *Class* uitlezen of aanpassen, dan zul je gebruik moeten maken van **Properties**. Properties hebben een “get” en een “set” methode voor het lezen resp. aanpassen van een klasse variabele.

**Voorbeeld met Properties**

| using System;  namespace Bibliotheek {  public class Boek  {  const float Btw = 0.21f;  private float prijs;  private string titel;  private string auteur;  private string uitgever;  private int aantalPaginas;   public Boek(string nTitel, string nAuteur, string nUitgever, int aantal, float nPrijs)  {  titel = nTitel;  auteur = nAuteur;  uitgever = nUitgever;  aantalPaginas = aantal;  prijs = nPrijs;  }   public float Btw(int aantal)  {  return aantal \* prijs + ((aantal \* prijs) \* Btw);  }  *//Property*  public float Prijs  {  get { return prijs; }  set { prijs = value; }  }  *//Property*  public string Titel  {  get { return titel; }  set { titel = value; }  }   *// enzovoorts*  } } |
| --- |

Prijs en Titel zijn properties en kunnen dus in Program.cs worden gelezen en aangepast.

C# kan ook automatisch de constructor en de *Properties* voor je aanmaken. In dit geval kun je de *Class* Boek als volgt opschrijven:

**Voorbeeld Automatische Properties:**

| using System;  namespace HelloWorld {  public class Boek  {  const float Btw = 0.21f;   public float Prijs  { get; set; }   public string Titel  { get; set; }   public string Auteur  { get; set; }   public string Uitgever  { get; set; }   public int AantalPaginas  { get; set; }   public float BTW(int aantal)  {  return aantal \* Prijs + ((aantal \* Prijs) \* Btw);  }  } } |
| --- |

Met *properties* kun je in de *Program.cs* variabelen van de *Class* Boek een initiële waarde geven of uitlezen of aanpassen.

**Voorbeeld Program.cs: waarden geven en lezen m.b.v. Properties:**

| Boek boek1 = new Boek(); *// of new Boek("Harry Potter", "JK Rowling", "Uitgeverij de Harmonie", 500, 4.95f)* boek1.Titel = "Harry Potter"; boek1.Prijs = 5.95f; Console.WriteLine(boek1.Titel); |
| --- |

**Voorbeeld** **Program.cs: waarden geven en lezen m.b.v. Automatische Properties:**

| Boek boek1 = new Boek {Prijs=3.95f, Titel= "Harry Potter", Auteur= "JK Rowling", Uitgever= "Uitgeverij de Harmonie", AantalPaginas=500}; Console.WriteLine(boek1.Titel); boek1.Prijs = 4.95f; Console.WriteLine(boek1.Prijs); |
| --- |