Code Snippets

1. Attribuut

2. Methode

3. Object

1. **public (type var.) NaamVanAttribuut**
vb. public string Naam;

2. **public (type var.) NaamVanAttribuut**
vb. public int AfstandWinkel {**return** meter - snelheid}

4. (Main) **NaamKlasse NaamObj = new NaamKlasse()**
vb. MasterChef Alexander = new MasterChef();
NaamObj.NaamMethode → vb. Alexander.KookKunsten;

Code Snippets 2

1. Enum

2. Full property

5. (in nieuwe klasse) **public enum NaamEnum { voorwerp1, voorwerp2 }**

3. Roep aan met **fullprop**
private int *naamProp*; (camelCase)
public int *NaamProp* (PascalCase)
{
get { return *naamProp*; }
set { *naamProp* = value; }
}

Klasse

Pascal case	vb. ClassName
Voorgesteld?	Rechthoek

een klasse hebben gedefinieerd die de algemene eigenschappen (properties).

Objecten

Waar?	Main
instantiëren -> new	vb. Auto mijnEerste = new Auto()

→ een verzameling van alle eigenschappen en methoden in een klasse die je dan makkelijk kan oproepen (vb.MyCar.Oliepeil -> geeft de oliepeil van het object genaamd MyCar)

Attributen

Andere Term	eigenschappen
scope	een klasse of een object van een klasse
Aanmaken	vb. public int Kilometers
Waarde geven	public Auto() {this.Kilometers = 5;}

• → stukjes data die je bijhoudt in objecten
• **encapsulatie** → netjes groeperen
• **static** → deel van je klasse maar zijn geen deel van het object.

Methoden

Termen	Instantiemethoden, objectmethoden
voorbeeld	public void Voltanken() {Benzine = 30.0;}

Methoden (cont)

Static methode	data van een object kan je niet veranderen

→ staat toe om functionaliteit toe te voegen aan objecten van een bepaalde klasse. = **gedrag** van de objecten
• (klein) uitvoerbare stukjes code die je dan makkelijk overal kan oproepen.
vb.
public int *naam*()
{
(code die *waarde* berekent)
return (*waarde*);
}

Datatypes

enum	vb. Weekdagen {Ma, Di, Wo}
variabelen toewijzen	dagKeuze = Weekdagen.Ma;

enumeratie (*enum*). Een enumeratie is een opsomming van alle mogelijke waarden voor een variabele van een bepaald type. Schrijven zoals een Klasse (onder *Namespace*)

Access modifiers

private	niet "van buitenuit" aangeroepen
	niet rechtstreeks wijzigen
	camel case
public	Pascal case

• → bepaalt welke code door welke andere code mag worden uitgevoerd of aangepast.
• Private methode → één code op meerdere plekken wordt gebruikt.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

Properties

get	
set	
propfull	(krijg formaat)

• zien er uit zoals attributen, maar werken zoals methoden.
• Code ziet er zo uit:
private int *naam*
public int *Naam*{ get {return *naam*} set {*naam* = value}

• **write-only** (SET) → enkel *naam* een waarde geven, maar niet van buitenuit lezen.
• **read-only** (GET)

• **Read-only property** met **private set** → private set
• **Read-only Get-omvormers** → iets berekenen en dat teruggeven via een getter.

DateTime

datum nu	DateTime nu = DateTime.Now;
Aangegeven datum	DateTime naam = new DateTime(year, month, day);
Toevoegen	DateTime naam = naam.AddDays(getal) | .AddMonths | .AddYears
Bereken van schrikkeljaar (if, else)	IsLeapYear
Specifiek van de datum	DayOfYear | Day | Month
Formateren	naam.ToString("yyyy", "d", "M")

DateTime (cont)

Datum in specifieke taal	CultureInfo taal = new CultureInfo("land-taal");

using System.Globalization; → nodig voor CultureInfo
using System.Threading.Tasks; → nodig voor System.Threading

System.Threading.Thread.Sleep(getal);
1000 = seconde

By **Lina Stappaerts** (HoneyLavender)
cheatography.com/honeylavender/

Not published yet.
Last updated 14th March, 2022.
Page 2 of 2.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com