

Starten met programmeren in .NET voor AutoCAD

Starten met
programmeren in .NET
voor AutoCAD

Anton Huizinga

© Anton Huizinga

www.huiz.net

www.linkedin.com/in/antonhuizinga/

ISBN: 9789402166965

Autodesk®, AutoCAD®, DWG® and the DWG logo are registered trademarks or trademarks of Autodesk, Inc., and/or its subsidiaries and/or affiliates in the USA and other countries.

AutoCAD screen shots reprinted courtesy of Autodesk, Inc.

Microsoft® and Visual Studio® are trademarks of the Microsoft group of companies.

Visual Studio screen shots reprinted courtesy of Microsoft.

All other trademarks are the property of their respective owners.

VOORWOORD

Als jongen raakte ik al snel gefascineerd door computers. Het zogenaamde computertijdperk was toen nog niet begonnen. Bedrijven gebruikten kaartenbakken en typemachines. Brieven gingen per post. Contact per telefoon, en daarmee wordt geen smartphone bedoeld.

Mijn eerste ervaring met programmeren was met een MSX, dat was ergens rond 1985. Internet zoals we dat nu kennen bestond niet eens. De MSX was een computer die je aansloot op een televisie voor het beeld, en gekoppeld werd aan een cassettespeler waar je de code op kon slaan. Dit waren dezelfde cassettes die toen gebruikt werden om muziek van af te spelen.

De MSX kon je programmeren met de taal BASIC. Dit is een vrij simpele taal waarin je de computer opdrachten kon geven om uit te voeren. Mijn eerste echte applicatie was een fruitautomaat. Op het scherm draaiden allerlei symbolen rond in drie blokken en geheel random stopten deze. Bij drie dezelfde kreeg je punten.

Het werkte fantastisch. Behalve dat de kans op drie gelijke symbolen veel te klein was (op echte fruitautomaten ook, maar dat terzijde). Gelukkig kon ik de formules aanpassen zodat de kans op drie gelijke symbolen veel groter werd. Binnen no-time had ik miljoenen punten. Meer dan op het beeldscherm paste. Dat was ook weer niet de bedoeling. En de lol van het spelen ging er heel snel vanaf want winnen deed je tenslotte altijd. Daarna heb ik er niet meer naar omgekeken. Spelen was niet het doel, maar het schrijven

van de code tot iets dat werkt.

Wat heb ik ervan geleerd? Dat computerspelletjes leuk zijn? Nee, ik vond het programmeren ervan leuker. Het omvormen van een idee tot code, zodat de computer snapt wat je wilt doen. Het kunnen aanpassen en optimaliseren.

Nadat ik mijn eerste PC gekocht had, was de overstap op andere talen snel gemaakt. BASIC was niets voor een PC. Een betere taal was Pascal. Ik heb daar toen heel wat tools in gemaakt. Daarna wat uitstapjes naar andere talen en technieken zoals webdesign waarbij de code in PHP geschreven werd en gebruik gemaakt werd van JavaScript. Beide talen hebben een sterke overeenkomst met C++.

Het programmeren was nog steeds een hobby. Beroepsmatig had ik ook met computers te maken. Ik was namelijk landmeter geworden en werkte op kantoor veel met AutoCAD om metingen om te zetten naar een tekening. Uit hobbyïsme heb ik me verdiept in AutoLISP en VBA om AutoCAD naar mijn hand te zetten. Diverse tijdrovende handelingen konden ineens stukken sneller met een handig tooltje.

Ik heb altijd gezegd dat ik van mijn hobby nooit mijn beroep zou willen maken. Want dan zou ik daarna geen hobby meer hebben.

In 2011 ben ik ondanks mijn voornemen toch professioneel programmeur geworden. Nu ontwikkel ik commerciële applicaties die binnen AutoCAD, Civil 3D of Map 3D draaien, vooral bedoeld

voor de civieltechnische markt.

Sindsdien heb ik geen hobby meer.

In 2015 begon het toch te kriebelen. Het had mij al heel lang leuk geleken om boeken te gaan schrijven. Na een aantal succesvolle romans en spionageboeken werd het tijd voor het echte werk. Ik had een nieuwe hobby gevonden: boeken schrijven.

Over programmeren zijn vele boeken te vinden. Over AutoCAD ook. Maar over programmeren in AutoCAD is er bijna niets te vinden. Ik heb veel zelf moeten uitzoeken en uitproberen. Ik had graag gewild dat ik boeken had gehad zoals deze. Gelukkig kan ik mijn kennis nu bundelen in dit boek zodat anderen er hulp bij hebben om in AutoCAD te leren programmeren.

HOE DIT BOEK TE LEZEN

Je hoeft nog niet veel ervaring te hebben in programmeren maar wel de *drive* hebben om te willen programmeren.

Er wordt van je verwacht dat je een redelijke computerkennis hebt en zelfredzaam bent. Afwachtende types die roepen: "Hij doet het niet", die kunnen dit boek beter niet lezen. Doet iets het niet? Ga dan op zoek naar de oorzaak en los het op. *That's the spirit!* Voor zulke types is dit boek geschreven.

Het boek is redelijk rechttoe-rechtaan te lezen. De volgorde van de hoofdstukken is zodanig dat er afwisselend een stukje code wordt beschreven, daarna (indien nodig) een wat gedetailleerdere uitleg

over dat stukje code en tussen de verschillende hoofdstukken af en toe een deel theorie.

De voortgang van het boek is gericht op praktijk, om iets werkend te krijgen in AutoCAD. Er wordt niet tot in detail uitgelegd wat er met .NET mogelijk is, daarvoor zijn de mogelijkheden te omvangrijk. In elk stuk code worden voldoende functies en constructies besproken om je .NET eigen te maken. Hoe meer je weet, hoe gemakkelijker het wordt om zelf op te zoeken naar wat je nog niet weet.

Het is geen boek voor Dummies, dus je zult geen tussenblokjes, infoblokjes, tips of aandachtspunten vinden. Wel staat aan het eind een overzicht met termen die gebruikt worden met een korte uitleg. Dit kan een handig naslagwerk zijn.

Tevens zal alle broncode te downloaden zijn via de website *www.huiz.net* zodat je niet alles over hoeft te typen.

Deze nieuwe druk is nu bijgewerkt tot het gebruik van Visual Studio 2019 en AutoCAD 2021.

Hopelijk vind je het een leerzaam boek en is het voor jou elke euro waard! Succes met leren!

Anton Huizinga

*Pipi Langkous: "Ik heb het nog nooit gedaan,
dus ik denk dat ik het wel kan."*

INHOUDSOPGAVE

Voorwoord	5
<i>Hoe dit boek te lezen</i>	7
Inhoudsopgave.....	11
Beginnen met programmeren.....	15
<i>Programmeren voor AutoCAD</i>	16
<i>Kiezen voor de programmeertaal</i>	16
<i>Hoe leer je programmeren?</i>	19
<i>.NET enzo</i>	20
<i>De editor</i>	22
Applicatie in AutoCAD	24
<i>Een Project aanmaken</i>	24
<i>Bibliotheken koppelen</i>	29
<i>Code schrijven</i>	31
<i>Debuggen</i>	33
Uitleg: Applicatie in AutoCAD.....	39
<i>References</i>	39
<i>Code tekst</i>	40
<i>Debuggen</i>	43
Theorie: Visual Studio.....	45
<i>De IDE van Visual Studio</i>	45
<i>Debug en Release</i>	46
<i>Applicatie versiebeheer</i>	46
<i>Frameworks en talen</i>	49
<i>IntelliSense</i>	52
Interactie met AutoCAD	55
<i>Project instellingen</i>	55
<i>Selecties maken</i>	57
<i>Een lus of een loop</i>	63

<i>Rekenen met getallen</i>	66
<i>Eigenschappen uitlezen van Entities</i>	71
Uitleg: Interactie met AutoCAD	77
<i>Namespaces of niet?</i>	77
<i>Objecten of Entities?</i>	79
<i>Copy Local</i>	80
<i>Code syntax afspraken</i>	81
<i>Transactions gebruiken</i>	82
<i>Using Transaction</i>	83
<i>ObjectId's en Handles</i>	86
Theorie: Objecten en waarden	88
<i>Objecten aanmaken en initialiseren</i>	88
<i>Constanten gebruiken</i>	92
<i>Rekenen met variabelen</i>	93
<i>Converteren van object-type</i>	97
<i>Werken met Operators</i>	100
<i>Arrays en lijsten</i>	105
Theorie: controle en fouten.....	108
<i>If - Else constructie</i>	109
<i>Switch constructie</i>	110
<i>Lus onderbreken met Break of Continue</i>	111
<i>Try, Catch</i>	112
<i>Breekpunten en debuggen</i>	114
Vensters en Palettes	121
<i>Vertrouwde applicatie</i>	122
<i>Project automatisch laden in AutoCAD</i>	123
<i>Organiseren van je project</i>	124
<i>PaletteSet maken</i>	126
<i>Dialogvenster maken</i>	139

Uitleg: Vensters en Palettes.....	147
<i>AutoCAD Plugins</i>	147
<i>Naamgeving van objecten</i>	149
<i>Font, DPI, 4K, resoluties</i>	150
<i>Events op Controls</i>	151
Theorie en praktijk: OOP	154
<i>Zonder objecten werken</i>	155
<i>Met objecten werken</i>	156
<i>Vershil tussen objectbeschrijving en object</i>	156
<i>Constructor en functies</i>	165
<i>Toegankelijkheid van functies en variabelen</i>	171
<i>Functies die wel of geen waarden teruggeven</i>	177
Applicatie PerceelEigenaar.....	179
<i>De User Interface</i>	181
<i>De Classes</i>	186
<i>Get/Set Properties</i>	188
<i>Comments met functionaliteit</i>	191
<i>Constructor met parameters</i>	194
<i>Vershil functie en eigenschap</i>	195
<i>Static Properties</i>	195
<i>Object maken van het object-type EigendomObject</i>	199
<i>Eigenschappen weergeven</i>	202
<i>Controle op selectie</i>	206
<i>Wijzigen van perceeleigenschappen</i>	209
<i>Tot zover gekomen?</i>	215
Theorie: Coding Standard	216
<i>Handboek bij het programmeren</i>	217
Highlight objecten.....	221
<i>Document lock voor bewerking</i>	223

<i>Tekenveld actief maken</i>	226
Import en export Excel.....	229
<i>Export naar CSV</i>	229
<i>Bestandsnaam opvragen</i>	231
<i>AutoCAD functies</i>	233
<i>Import uit Excel</i>	235
Uitleg: Import en Export Excel.....	238
<i>String functies</i>	238
<i>Werken met Arrays</i>	240
<i>OpenFileDialog en SaveFileDialog</i>	241
Objecten tekenen	244
<i>Functie om laagnamen aan te maken</i>	246
<i>Functie om tekst toe te voegen</i>	251
<i>Teksten plaatsen in de tekening</i>	254
<i>Functie om cirkels te tekenen</i>	256
<i>Cirkels plaatsen in de tekening</i>	258
Release maken.....	262
<i>Bundle maken voor AutoCAD</i>	264
<i>Inhoud PackageContents.xml</i>	265
Overzicht termen	268
<i>Gebruikte termen</i>	268
Nawoord bij dit boek.....	271
Index van moeilijke woorden	273
Bijlagen (downloads)	276

BEGINNEN MET PROGRAMMEREN

De meningen zijn verdeeld over wat de beste manier is om te leren programmeren. Sommige mensen zullen zeggen dat je het beste een goed boek over programmeren kunt lezen. Dat roept weer de vraag op of het dan een algemeen boek over programmeren moet zijn of een boek over de taal waarin geprogrammeerd gaat worden.

Ik zeg: "programmeren moet in je zitten." Anders kun je net zo goed stoppen. Autorijden kun je ook niet uit een boekje leren. De techniek van autorijden wel, zoals schakelen, het bedienen van de richtingaanwijzers, en zelfs hoe je de olie peilt. Maar het verkrijgen van verkeersinzicht, het onbewust kunnen bedienen van de auto, en het vermogen tot anticiperen op medeweggebruikers zal nooit iets worden als het niet in je zit om goed te kunnen autorijden.

Zo werkt het met programmeren ook. Als een computer slechts een stuk gereedschap is dat hooguit tijdens kantooruren aan moet, als je niet nieuwsgierig bent om dat bijzondere apparaat verder te doorgronden, als je geen gedrevenheid voelt om dieper te graven in de wondere wereld van computers, dan kun je beter stoppen met dit boek en je richten op een andere hobby. Het gaat nooit iets worden. Succes met je verdere leven!

Mocht je nu toch verder lezen, dan neem ik aan dat jij wel de gedrevenheid bezit om te leren programmeren, en het mogelijk in je zit om daar goed in te worden. Succes gewenst dan maar! We beginnen dit hoofdstuk met wat saaie kost. Hopelijk kom je er doorheen.

PROGRAMMEREN VOOR AUTOCAD

Dit boek is gericht op het programmeren in AutoCAD. Het is geen boek met alle ins en outs over programmeren van Windows applicaties of mobiele applicaties, maar over het maken van plug-ins voor AutoCAD. Wil je je verder verdiepen in specifieke programmeeronderdelen, dan zijn daar talloze boeken van te verkrijgen. Maar helaas zijn er maar weinig boeken over programmeren voor AutoCAD. Daarom heb ik de tijd genomen om er zelf een te schrijven in de hoop andere mensen ermee te kunnen helpen.

KIEZEN VOOR DE PROGRAMMEERTAAL

Hoewel er verschillende talen beschikbaar zijn om in te gaan programmeren, is er niet een echte keuze. De beste taal om te leren is C#. Dit wordt uitgesproken als *sie sjarp*. En waarom is dit de beste keuze?

In AutoCAD is het mogelijk om te programmeren in de volgende talen (en nog wel meer):

- AutoLISP
- VBA (Visual Basic for Applications)
- VB.NET
- C++
- C#

AutoLISP is heel krachtig en je kunt er bijna alles mee maken. Maar

de Lisp-routines zijn niet of nauwelijks te beveiligen en zijn dus te bewerken door derden. Verder is de syntax van Lisp in niets gelijkend op andere gangbare programmeertalen en daarom is het moeilijker om een andere taal aan te leren of te switchen tussen verschillende talen. Leuk voor kleine klusjes in AutoCAD en zeker handig als je het kent, maar laat toch maar even links liggen.

VBA is ook zeer krachtig en je kunt er nog veel meer mee dan met Lisp, omdat je met VBA ook Windows functies kunt uitvoeren. VBA heeft als nadeel dat het een oudere taal is en al jaren doodverklaard is maar telkens weer (half) tot leven komt. Bovendien is het een Visual Basic taal en wordt het door echte (of beter gezegd: succesvolle) programmeurs niet serieus genomen.

VB.NET is een moderne programmeertaal en kan alles wat ook met C# mogelijk is. Beide zijn .NET varianten en je kunt ze gebruiken om zeer krachtige applicaties te maken. Op zich is er geen verschil tussen beide talen, afgezien van de syntax, maar ook hier geldt dat deze taal niet serieus genomen wordt door succesvolle (lees: niet persé betere) programmeurs. Je moet zelf weten als je liever VB.NET wilt leren, misschien omdat je het eenvoudiger lijkt omdat je ooit iets met VBA gedaan hebt, maar ik raad het toch af. De syntax van VB.NET wijkt best wel af van VBA waardoor je je net zo goed gelijk kunt verdiepen in C#. Ik ken maar één succesvolle programmeur die alles in VB.NET doet. Dat is Stephen Preston van Autodesk. Komende vanuit de oudere C-talen moest hij een keuze gaan maken toen .NET uitkwam, wordt het VB.NET of C#? Zijn vrienden voelden zich superieur ten opzichte van VB.NET

programmeurs dus vanuit een sterke eigenwijsheid heeft Stephen toen gekozen voor VB.NET om een punt te maken. Toch lijkt hij in een interview een klein beetje spijt te hebben van zijn keuze: <http://adndevblog.typepad.com/autocad/2012/05/vbnet-or-c-a-personal-perspective.html>

C++ is een vrij ingewikkelde taal om te leren, nauwelijks geschikt om mee te beginnen. Het is ook wat krachtiger dan C#. Omdat het heel complex is, en je snel echte storingen kunt veroorzaken, is het niet een geschikte taal om mee te beginnen.

Dan komen we bij C#, dat is een taal die serieus genomen wordt, en een taal is die qua syntax erg lijkt op andere talen zoals Java, PHP, Javascript en C++, waardoor het gemakkelijker is om een andere taal erbij te leren of te kunnen begrijpen. Als je C# kunt lezen, dan kun je op het internet ook een voorbeeldstukje JavaScript of C++ snappen en dus kunnen implementeren in je eigen code. Bovendien is er veel meer voorbeeldcode te vinden in C# of C-achtige talen dan bijvoorbeeld VB.NET.

C# heeft alles om snel en eenvoudig zeer krachtige applicaties te kunnen bouwen. Of dit nu Windows applicaties zijn of AutoCAD plug-ins, deze taal is uitstekend geschikt.

Daarom is C# de (beste) keuze om in te programmeren.

Ook al kun je met VB.NET exact hetzelfde, toch worden C#-programmeurs door vrijwel iedereen serieuzer genomen en vinden ze ook sneller een goede baan. Er is geen enkele goede reden voor

om C#-programmeurs voor te trekken op VB.NET-programmeurs maar toch gebeurt het. Wil je verder komen, carrière maken? Kies dan voor C#.

HOE LEER JE PROGRAMMEREN?

Dat kan een lastige vraag zijn en is voor iedereen anders te beantwoorden. Hoe je *niet* leert te programmeren, is veel eenvoudiger te beantwoorden. Toch zal ik een aantal punten noemen die volgens mij voor iedereen noodzakelijk zijn om te kunnen leren programmeren.

- Zorg dat je geïnteresseerd raakt in programmeren.
- Ga het doen. Probeer wat. Zoek uit op internet. Doen doet leren.
- Leer van anderen. Lees broncode en probeer te snappen wat ze bedoelen. Laurendo Almeida zei eens: "Stealing from one source is plagiarism. Stealing from many sources is research."
- Als je de tijd en mogelijkheid hebt, volg een opleiding die een certificaat of diploma oplevert. Altijd handig voor bepaalde banen waar dit van je verlangd wordt.
- Ga samenwerken. Soms ben je de beste, soms ben je de slechtste in de groep. Leer ervan.
- Bouw programma's na. Verzin zelf hoe je tot oplossingen komt.
- Leer ook andere talen.
- Plan een *Coding Standard* voor jezelf. Niet om je te beperken maar om meer gestructureerd te kunnen werken.
- Je kunt geen taal leren in 24 uur. Reken eerder in jaren.

.NET ENZO

Wat is .NET (of zoals het uitgesproken wordt: *dot net*) dan eigenlijk? Misschien heb je er al meer van gehoord, en anders leg ik het graag even in het kort uit. Wil je nog meer weten, dan zijn daar vast goede boeken over te vinden.

Vroeger, als je een applicatie wilde maken, dan moest je alles zelf uitwerken. Bijvoorbeeld de manier om je beeldscherm aan te sturen, het toetsenbord uit te lezen, muisbewegingen te onderscheppen, enzovoort. Vaak kon je wel kant en klare stukken code of modules vinden of kopen die dat voor jou deden, toch was het altijd een ingewikkelde klus om een applicatie te maken die op elke computer werkte omdat elk apparaat anders geconfigureerd was. Fabrikanten hadden hun eigen methodes om hun apparatuur aan te spreken en die waren onderling niet compatible. Je kon als programmeur niet alle mogelijke voorkomende configuraties testen en dan kon het gebeuren dat jouw applicatie niet goed werkte met muis van het merk X of monitor van het merk Y.

Toen kwam Java op de markt. Dat was een *Framework* waarbinnen applicaties konden draaien die gebruik maakten van de beschikbare Java functies. Dat scheelde enorm veel tijd in programmeren, want je kon nu rechtstreeks functies aanroepen die het echte werk uit handen namen. Omdat Microsoft geen Java mocht implementeren in Windows, heeft Microsoft zelf een Framework gemaakt, gestoeld op het idee van Java maar dan beter. Dit Framework werd .NET genoemd. Het Framework diende als

basis voor verschillende talen die onderling uitwisselbaar zijn.

Applicaties die in een .NET taal geschreven zijn, kunnen onderling namelijk met elkaar communiceren en er zijn enorm veel functiebibliotheken beschikbaar om te gebruiken in de code. En daarom is .NET succesvol geworden. Het maakt niet uit of je C++.NET, VB.NET, F# of C# gebruikt, het is allemaal uitwisselbaar. Ook de grote hoeveelheid beschikbare functies is een zegen voor programmeurs. Met één regel code kun je een meldingsvenster tonen met de juiste knoppen en iconen. Met een paar regels code roep je een Save-dialoogvenster op dat zelfstandig voor jou kan controleren of een pad bestaat, of een bestand overschreven gaat worden, enzovoort.

.NET is zelf geen taal, maar een verzameling bibliotheken vol met functies die gebruikt kunnen worden in de verschillende .NET talen.

Autodesk heeft het mogelijk gemaakt om .NET applicaties in AutoCAD te kunnen laden die gebruik kunnen maken van de AutoCAD functies. Een .NET applicatie in AutoCAD kan dus zowel Windows functies aanroepen als AutoCAD functies. Autodesk heeft als het ware .NET uitgebreid met een bibliotheek vol met AutoCAD functies. En zo zijn er meer bedrijven die hun applicaties openstellen in .NET. Zo'n beschikbaar gestelde bibliotheek wordt een API genoemd: *application programming interface*.

DE EDITOR

Nog een belangrijk ding om te noemen voordat we echt iets gaan maken, is de editor. Dit wordt ook wel een IDE (*integrated development environment*) genoemd, want het is meer dan alleen de editor zelf waarin je de code schrijft. Uiteraard kun je prima C# in Kladblok schrijven maar dat is net zo omslachtig en tijdrovend als het paard in de wagen zetten en zelf de kar gaan trekken. Om snel en eenvoudig te kunnen programmeren, heb je een IDE nodig met daarin een editor die toegespitst is op de door jou gekozen taal. C# dus. En hiervoor zijn hele mooie IDE's.

We beperken het nu even tot Visual Studio van Microsoft. Hiervan is een gratis versie te verkrijgen waarmee je mag programmeren zolang je aan bepaalde voorwaarden voldoet. Ben je een commercieel bedrijf en verdien je ook aan je applicaties, dan zal je Visual Studio moeten kopen, of anders een andere IDE moeten zoeken. Er zijn ook Open-Source IDE's.

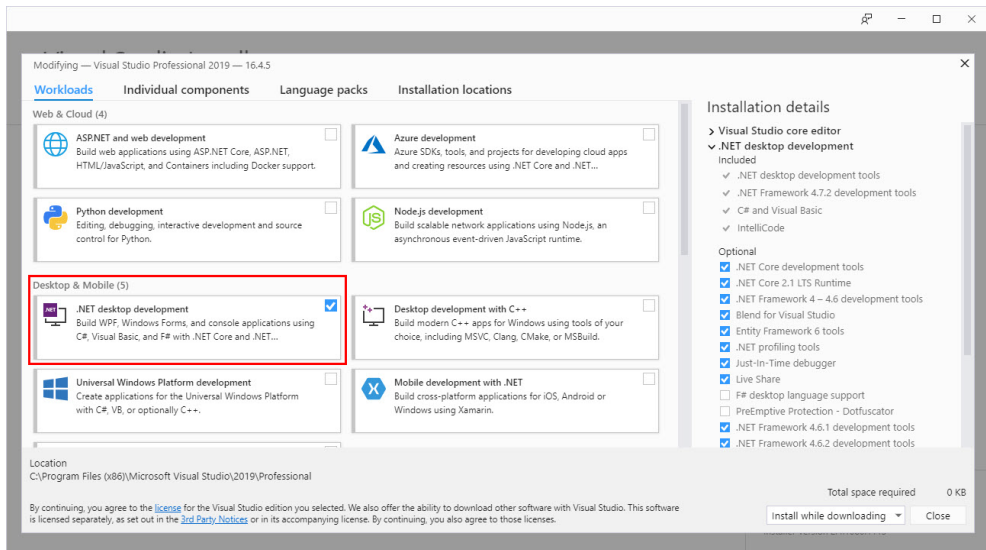
Op dit moment is Visual Studio 2019 de meest recente IDE van Microsoft. Je kunt een gratis versie downloaden genaamd 'Visual Studio 2019 Community' via onderstaande link:

<https://visualstudio.microsoft.com/downloads/>

Kies voor de Community Edition.

Als het je niet lukt om Visual Studio te installeren, dan kun je het programmeren misschien maar beter opgeven. Ik ga er vanuit dat het je lukt, en zie je in het volgende hoofdstuk terug.

Ik zal je nog wel een tip geven. Tijdens de installatie kun je aangeven welke onderdelen je wilt installeren. Omdat we simpele C# applicaties gaan ontwikkelen, kiezen we ook alleen maar voor de optie '.NET desktop development' onder de categorie 'Desktop and Mobile':



Het is voldoende om alleen dit onderdeel aan te vinken en alle voorgestelde subonderdelen rechts te accepteren zoals het wordt voorgesteld.

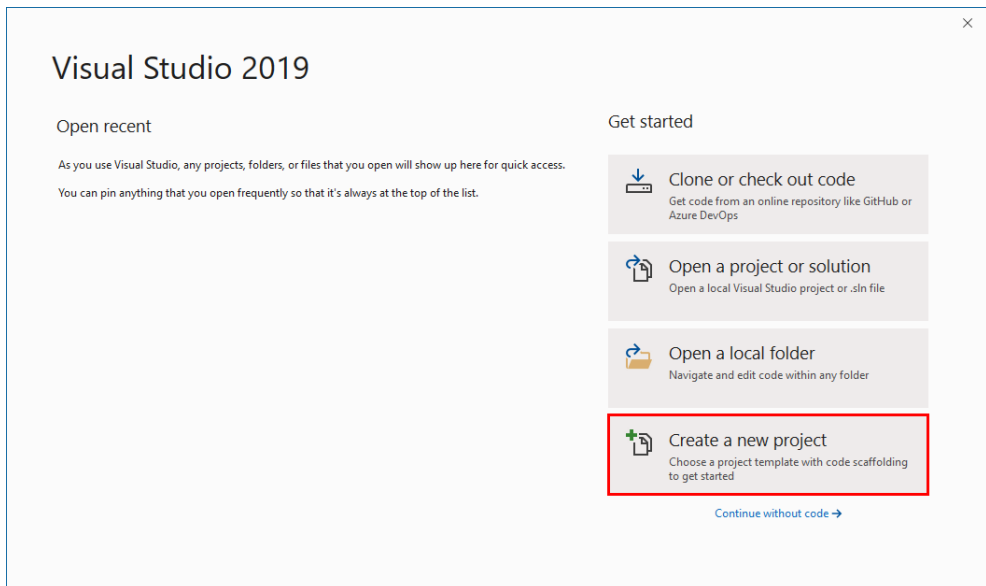
Voor nu moet dat ruim voldoende zijn. Aanvullende onderdelen zijn altijd later nog te installeren maar die zijn voor AutoCAD plug-ins niet nodig.

APPLICATIE IN AUTOCAD

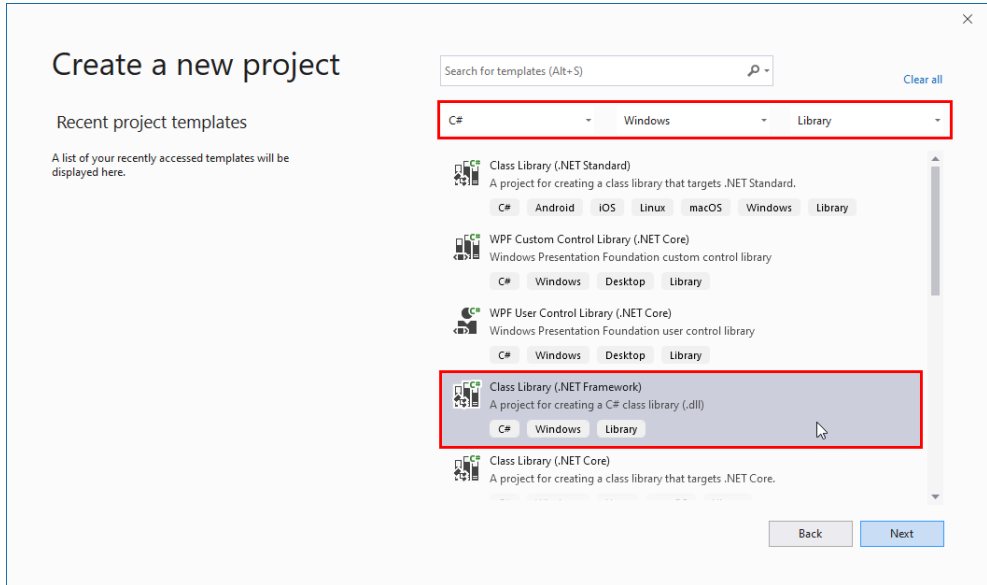
In dit hoofdstuk gaan we een kleine applicatie maken die geladen wordt in AutoCAD en waarmee je de beginselen van programmeren voor AutoCAD leert. Je hebt Visual Studio geïnstalleerd en uiteraard ook AutoCAD.

EEN PROJECT AANMAKEN

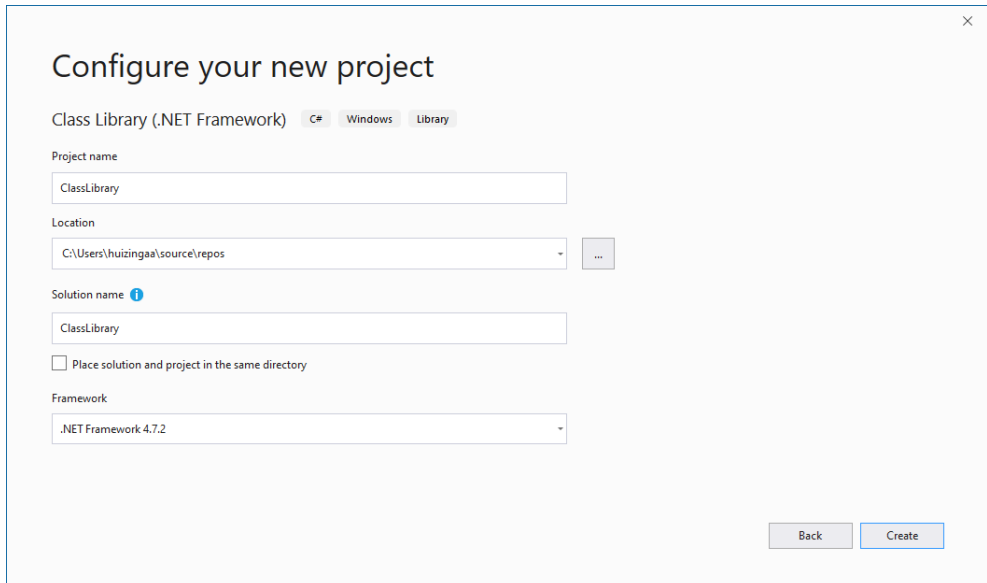
Een plug-in in AutoCAD is van het type *Class Library*. In het opstartscherm van Visual Studio kun je kiezen voor *Create a New project*:



In het volgende venster kies je voor een *Class Library (.NET Framework)*. Let op dat je je niet vergist, er is ook nog een *.NET Standard* en een *.NET Core* variant.



Je kunt bovenin filteren op 'C#', 'Windows' en 'Library'. Ga verder naar de volgende stap. Je geeft een naam op en kiest de locatie waar je project wordt opgeslagen.



Tot zover niks bijzonders, behalve dat je nu een paar nieuwe zaken moet leren voor je verder gaat.

Voor elke applicatie die je maakt, heb je een *Project* nodig. Het project bevat alle programmacode, de *Resources* zoals de afbeeldingen die je gebruikt op je dialoogvensters, de verwijzingen naar de verschillende functiebibliotheken en de instellingen die gelden voor deze applicatie. Alles bij elkaar is een *Project*.

Uit één Project volgt één applicatie. Het Project is dus de applicatie.

Je kunt meerdere Projects in een *Solution* hebben. Een Solution is daarmee een verzameling bij elkaar horende Projects. Als de projecten onderling van elkaar gebruik maken, dan is een Solution met meerdere Projects zeker handig. In dit stadium hanteren we de regel dat elk Project in een eigen Solution zit. En misschien zal je in de toekomst ook nooit anders doen dan dat.

In bovenstaand geval kun je dus rustig de Solution naam gelijk houden aan de Project naam. De Solution en het Project hoeven niet in dezelfde directory worden geplaatst maar het is ook geen probleem als je dit wel aanvinkt.

Kies vervolgens de juiste .NET Framework versie. Dit kun je eventueel later aanpassen. Het is handig om te weten welke versie je nodig hebt. Voor elke AutoCAD versie kan dit anders zijn:

Release	.NET Framework
AutoCAD 2021	4.8

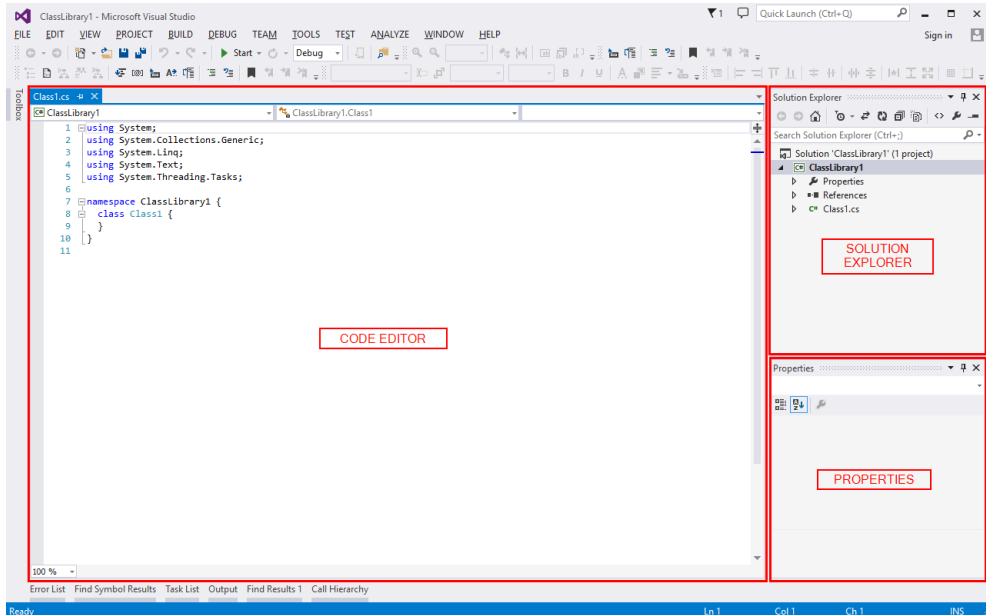
AutoCAD 2020	4.7
AutoCAD 2019	4.7
AutoCAD 2018	4.6
AutoCAD 2017	4.6
AutoCAD 2016	4.5
AutoCAD 2015	4.5

Wil je een applicatie maken voor AutoCAD 2021, dan zal jouw project moeten verwijzen naar .NET Framework 4.8, maar als je iets voor AutoCAD 2015 wilt maken dan kies je .NET Framework 4.5.

Klik op de knop *Create* en de editor zal verschijnen met het nieuw gemaakte Project van het type *Class Library*.

De indeling op de schermafdrucken kan hier en daar anders zijn, maar in grote delen moet het overeen komen. Mis je een onderdeel, dan kan dit meestal via het menu View worden opgezocht en worden getoond.

Het grote veld links/midden is de editor waar je de codebestanden aanpast. Rechts heb je de *Solution Explorer*, daar staan alle bestanden die bij het project horen. Daaronder heb je een *Properties* venster waar je de eigenschappen kunt aanpassen van geselecteerde objecten of bestanden. Voor dit boek gebruik ik een licht thema voor de weergave van de IDE, dit werkt beter met screenshots. Je kunt het thema ook op donker laten, dit heeft op de werking van de IDE geen invloed.



In het grote codeveld zijn de geopende bestanden zichtbaar. De tabs bovenaan tonen de geopende bestanden en je kunt eenvoudig switchen, sluiten of bestanden naast elkaar plaatsen.

De code wordt door middel van kleuren getoond zodat snel inzichtelijk is wat de verschillende onderdelen van de code voorstellen. Tevens zijn er enkele belangrijke kenmerken aan de code:

1. Alles is hoofdlettergevoelig,
2. Elke regel behalve de accolades eindigen met een puntkomma,
3. Coderegels kunnen gegroepeerd worden tussen accolades, dit zijn codeblokken die bij elkaar horen.