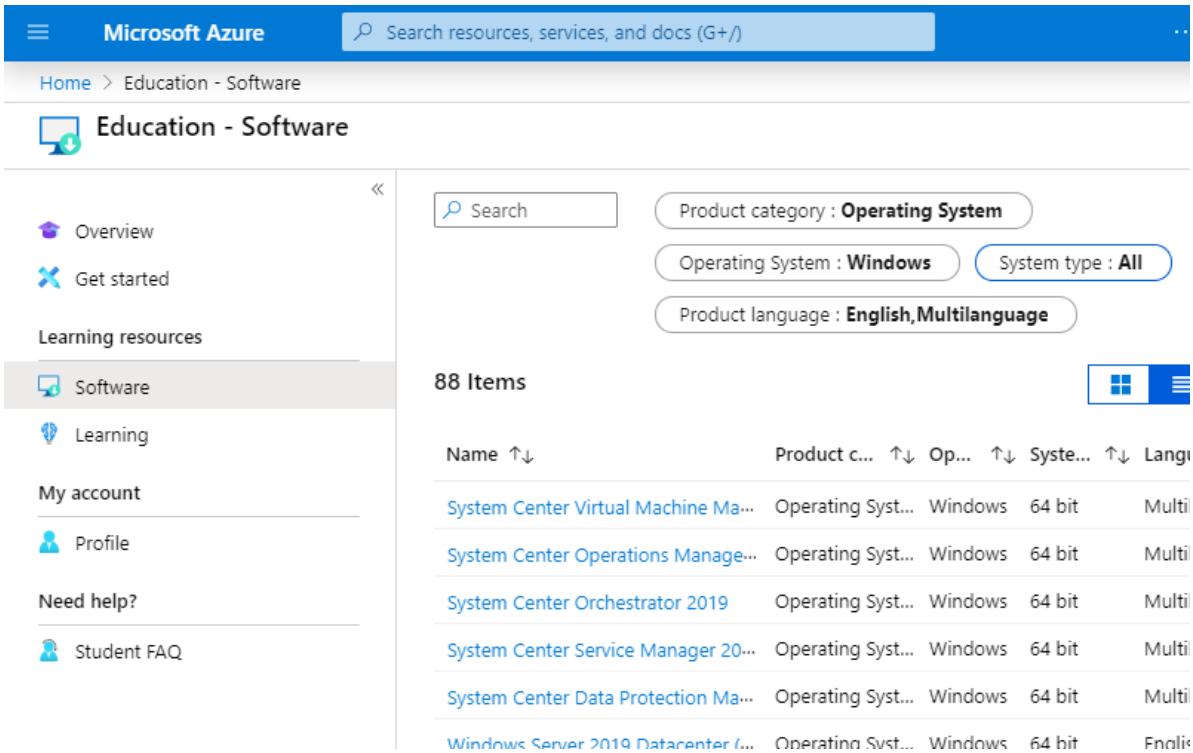
Opgaver Lektion_01

Du skal have Windows installeret i dette kursus for at køre de ting, vi skal bruge.
Hvis du ikke har Windows på din laptop, så er det tid at få installeret dette operativ system. Licens hertil finder du på linket https://aka.ms/devtoolsforteaching hvor du kan logge ind med din skole konto klikke dig frem til gratis Windows licens.

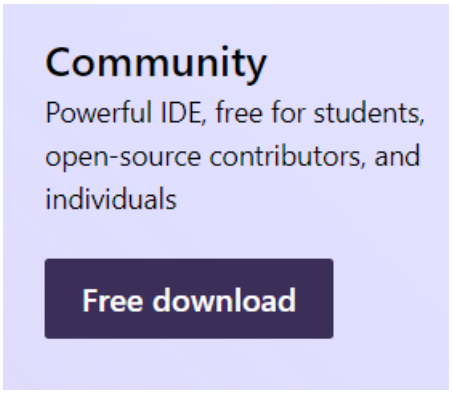


Du kan vælge mellem:

- Windows som eneste operativsystem på din computer.
- Windows blandt de mulige opstartssystemer på din computer via dual boot.
- Virtuelt miljø med Windows installation.
  På Linux har du mulighed for at installere Windows i f.eks. VirtualBox eller KVM virtuelt miljø.
  På Mac-computer kan du f.eks. vælge VirtualBox, Bootcamp eller Parallels som virtuelt miljø for installation af Windows.

Opgave 1:

- Install Visual Studio (VS) (Community Edition er fin).
- https://visualstudio.microsoft.com/downloads/



- Du skal bruge
  - .Net desktop development
  - ASP.NET and web development (MVC)

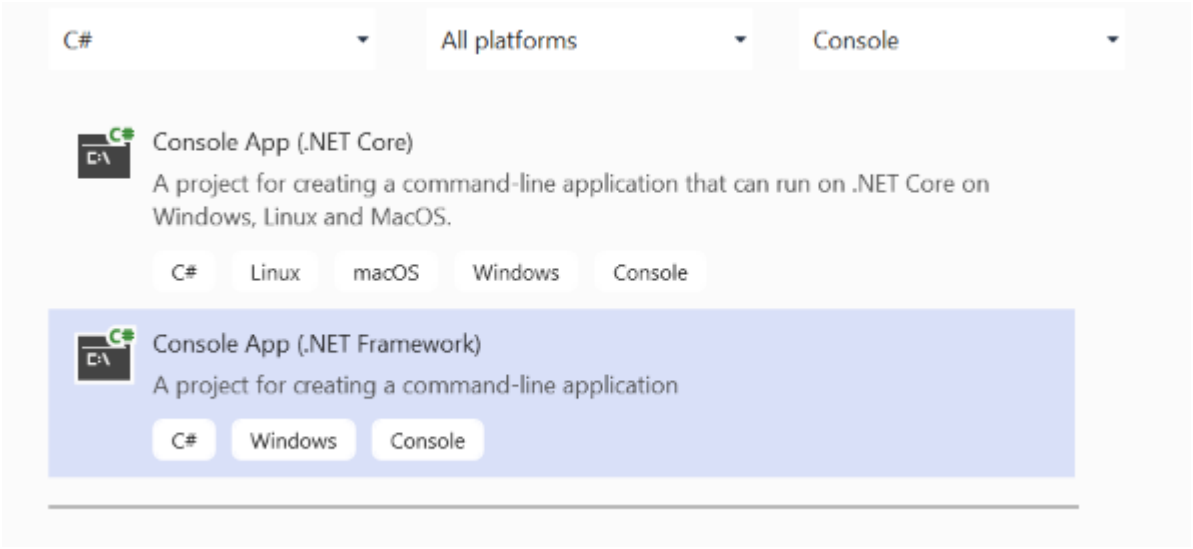
  - (Vi vil ikke bruge .NET Core)

Start VS for at se om det virker!

- Hvis du ikke har Git installeret, så installer den, da vi skal bruge det senere:
- https://git-scm.com/downloads

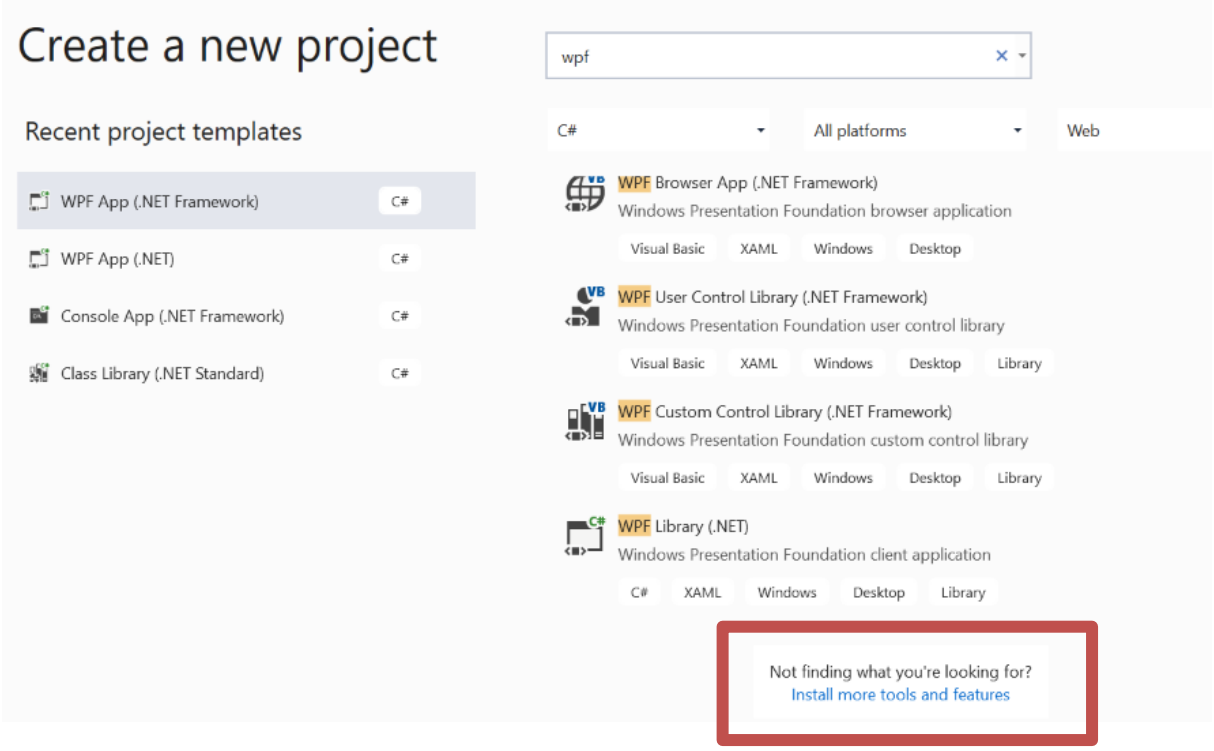
Opgave 2:

Lav en konsol applikation i Visual studio
(Brug *"Console App (.Net Framework)"* template) – se screenshot:

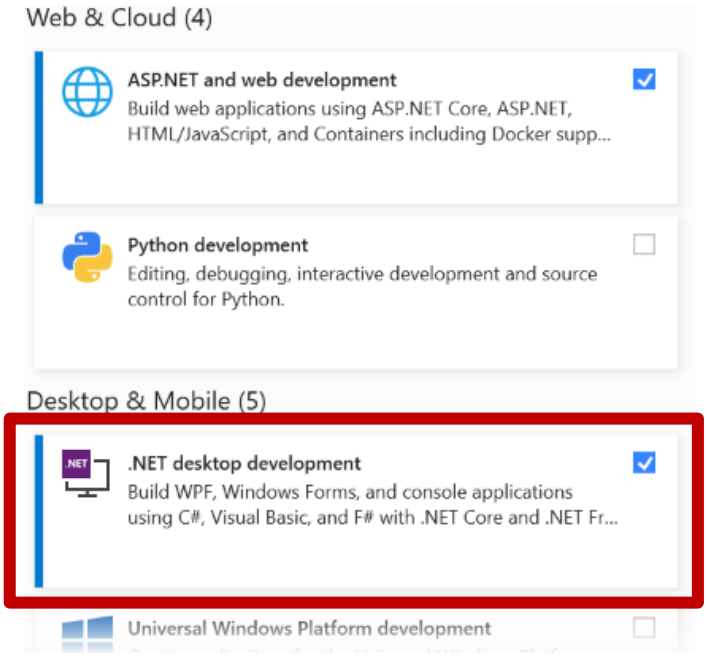


MEGET VIGTIG – Bemærk at der står "Console App (.NET Framework)". Der skal IKKE stå
Console App (.Net) eller Console App (.Net core).

Hvis du ikke har muligheden for at vælge Console App med (.NET Framework, så kan du under File->New Project scrolle ned og installere det – se følgende screenshot:



Så vælg "Install more tools and features" og du vil så få dette screenshot:

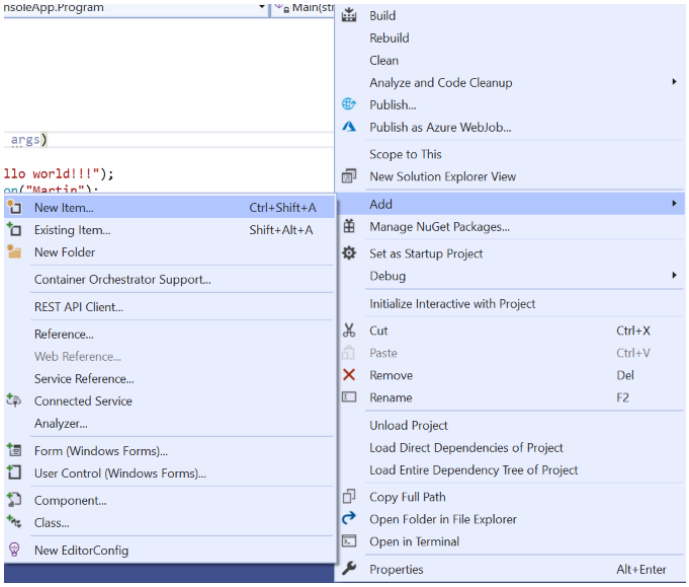


Vælg at installere .NET desktop Development.

Få den til at udskrive "Hello World" til konsollen – hint : `Console.WriteLine…kan bruges`
Kør både fra Visual studio (prøv at se forskellen på at starte programmet med F5 og Ctrl-F5) og et konsol vindue (altså en command line prompt – i Windows kan du starte det med "cmd"). Du finder det eksekverbare program i bin\Debug eller bin\Release folderen under dit projekt. Hvis du mangle en "pause" i programmet, kan du bruge Console.ReadKey() f.eks.

Opgave 3:

Lav en klasse "Person" som tager et string argument (som er personens navn) i constructoren. Du kan tilføje nye klasser ved at højre-klikke på på dit projekt i Solution Explorer og så vælge add->new item og så vælge en c# klasse:



Opgave 4:

Læs denne side om properties overfladisk:
https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/properties
(Det er ikke nødvendigt at forstå ALLE detaljer – vigtigst er at se under Auto Implemented Properties, men prøv at læse det andet først for at forstå forskellen mellem get og set)
Giv din Person klasse en auto implemented property og sæt dens værdi i constructoren til det argument som kommer som input til constructuren.

Derefter skal vi lave en ny ToString metode (faktisk override den fra object klassen, som er superklasse til alle klasser i C#) – se https://docs.microsoft.com/en-us/dotnet/api/system.object.tostring?view=net-5.0

Din kode skal kunne bruges således til sidst:

```
var person = new Person("martin");
```

```
Console.WriteLine(person);
person.Name = Benny;
Console.WriteLine(person);
```

Output skal så minde om dette
Name is Benny
Name is Martin

Opgave 5:

Du kan få hjælp her i denne tutorial, hvor du kan se hvordan man tilføjer et ClassLibrary til en eksisterende løsning (bemærk at du ikke skal lave en ny løsning, da vi jo allerede har dette – vores consol projekt, men blot et nyt ClassLibray inden i dette projekt):
https://docs.microsoft.com/en-us/dotnet/core/tutorials/library-with-visual-studio

I VS solution explorer: lav et nyt ClassLibrary project (kald det evt MyLibrary). I dette project skal defineres et interface IAnimal og en klasse Animal som implementerer interfacet IAnimal. Bemærk at Animal klassen skal være public – så signaturen for klassen vil således ud:

```
public class Animal : IAnimal  // Så : betyder nedarver/implementerer i C#
```

- Animal skal have et argument til constructoren, som er en string og som f.eks. kan hedde specie. Så hvis dyret er en hund kan du lave en ny hund som ”var hund = new Animal(”hund”)”
- Interfacet IAnimal skal (mindst) have metoden

  ```
  bool IsDog();   // som returnerer true hvis det er  ”hund” ellers false
  ```

(I virkeligheden behøver vi ikke et interface for dette selvfølgelig, men blot lige for at træne at lave et Interface).

- I implementationen af isDog skal bruges en if-sætning, som er identisk med en Java-if-sætning.
- Din nye klasse Animal skal nu bruges fra dit primære projekt - på denne måde:

  ```
  var fido = new Animal("hund");
  Console.WriteLine("fido er en hund? " + fido.IsDog());
  ```

- Du skal finde ud af at indsætte referencer og en ”using statement” for at få det til at virke med at bruge dit nye library. Referencer indsættes ved at højre klikke på References i dit hovedprojekt og vælg *”Add Reference”*, vælg *”Projects”* og marker MyLibrary projektet.
- Tilret, så det bliver muligt at spørge brugeren om, hvilke Animal’s skal oprettes. Du skal bruge Console.ReadLine().
- Du kan evt. lave en while-løkke (ja, lige som i Java), der muliggør, at brugeren bliver ved med at indtaste indtil der skrives en termineringsværdi

Opgave 6:

- Find ud af hvordan man definerer lister i C#. Vink: Se her: https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.list-1?view=netframework-4.7.2
- Skriv en klasse MyList som indeholder en liste af tal. Oprindeligt er listen tom når man laver en ny liste, men så skal du have en Add(int number) metode til klassen som tilføjer et tal. Tilføje nogle tal ved brug af denne metode – du kan jo bare tilføje det til dit konsolprogram. (altså i din main)
- Implementer en metode, PrintNumbers() på denne klasse til at udskrive alle elementerne i listen – her kan en foreach løkke anvendes (se linket fra før – der er faktisk en foreach loop også som passende kan anvendes.

  Så fra dit consol program skal du kunne bruge din MyList klasse f.eks. således:

```
var list = new MyList();
list.AddNumber(1);
list.AddNumber(10);
list.AddNumber(100);
list.PrintNumbers();
```

Opgave 7:

Random numbers – I denne øvelse skal du genbruge MyList klassen, men bruge tilfældige tal i stedet for at manuelt tilføje nogle som før.

Lav en liste af 10 random number, hvor hver nummer er et tal mellem 1 og 100 (inklusiv begge). Skriv dem ud i consolen. Mere info om Random numbers kan findes her: https://docs.microsoft.com/en-us/dotnet/api/system.random?view=netframework-4.7.2