

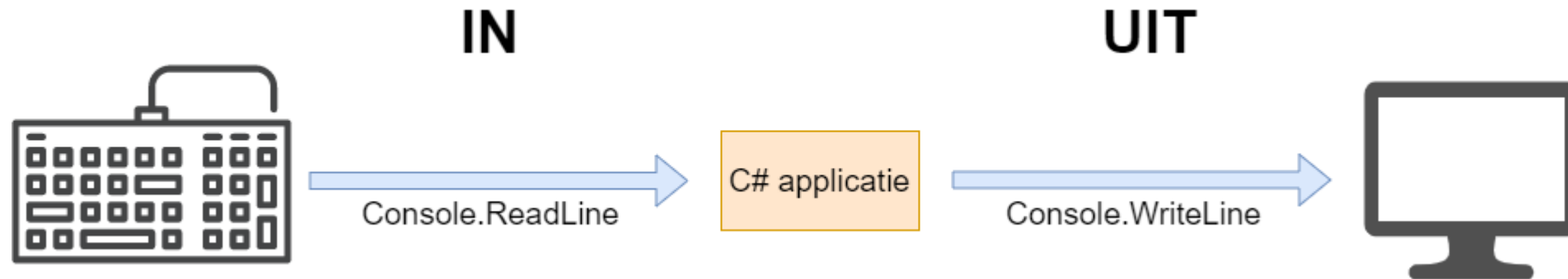
# 4. Input verwerken met ReadLine

H1: De eerste stappen



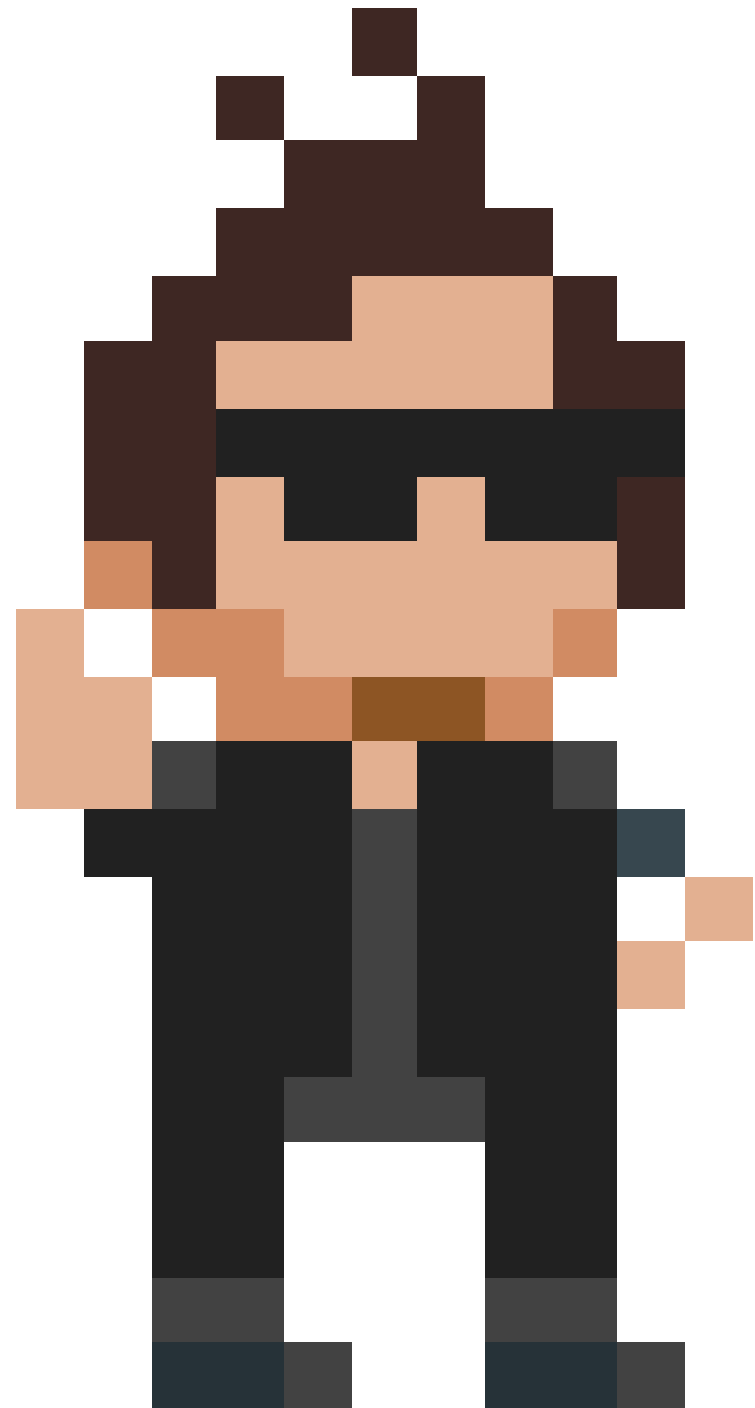
# WriteLine() & ReadLine()

- WriteLine: Basis methode om tekst op het scherm te plaatsen
- ReadLine: Tekst van gebruiker inlezen via toetsenbord



# Demo time

- Gebruik ReadLine
- Gebruik resultaat van ReadLine

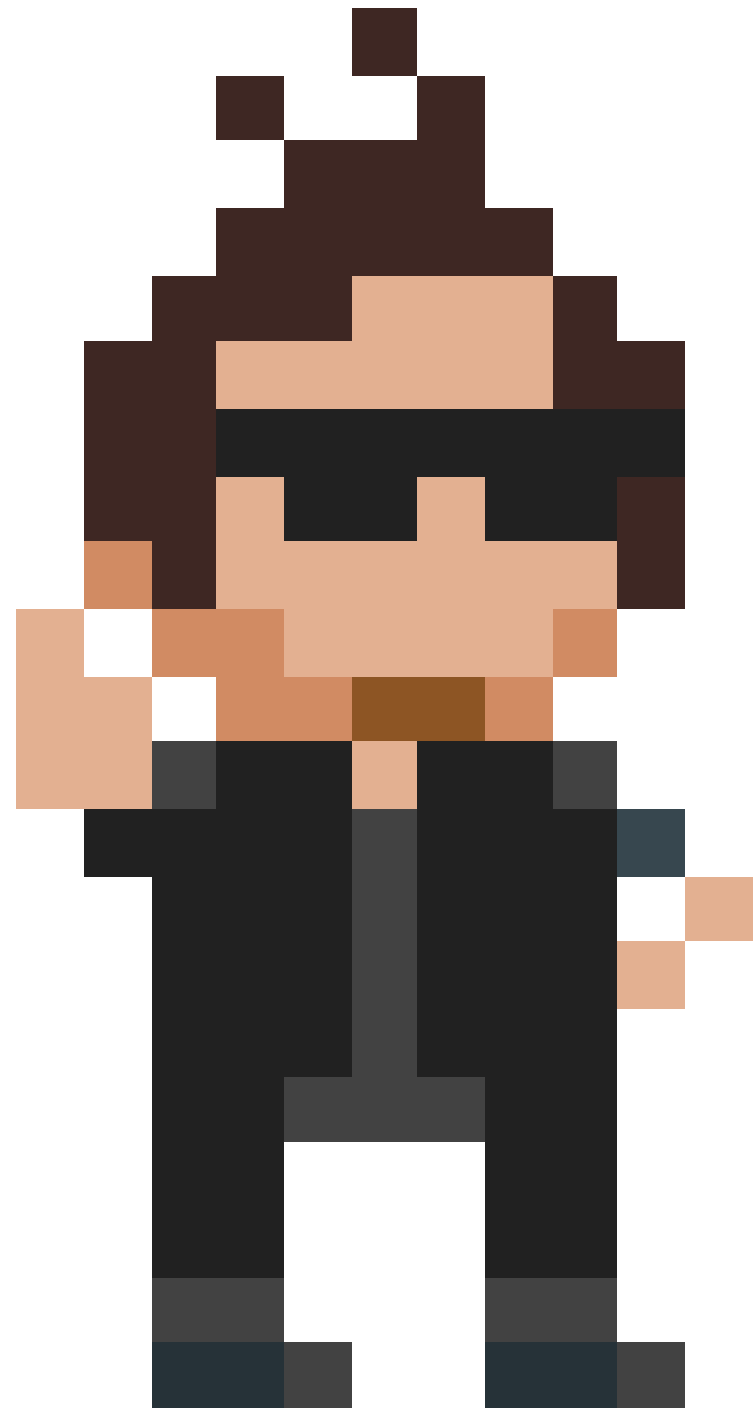


# Write en WriteLine

- Console.Write(): **geen enter** op einde
- Console.WriteLine(): wel **enter** op einde

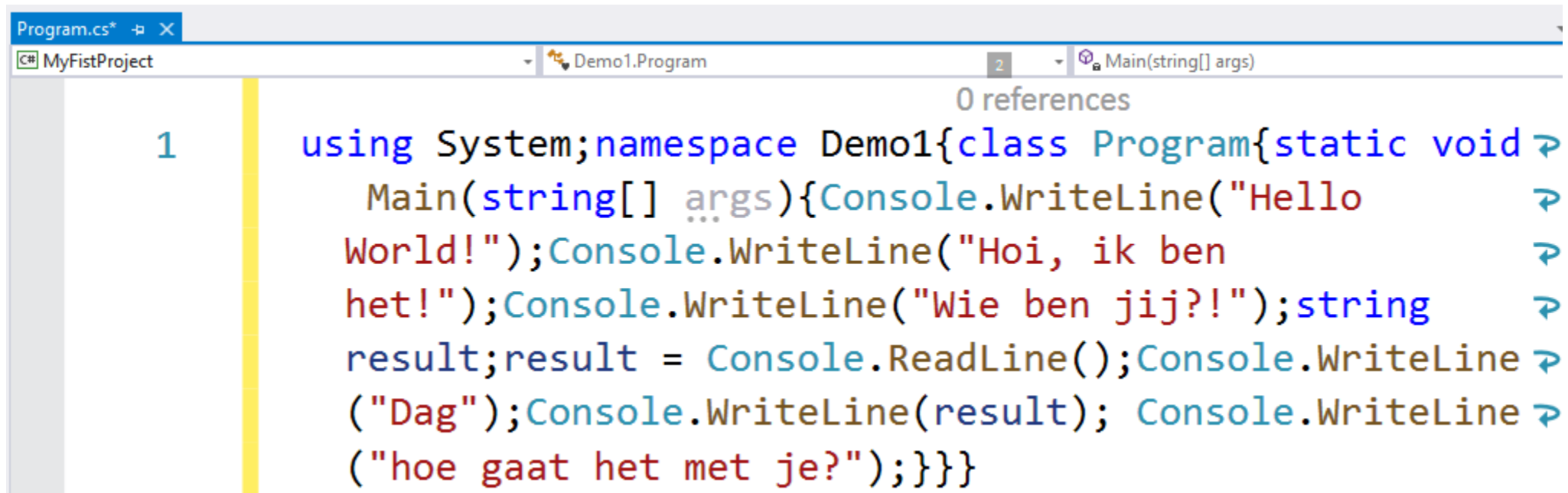
# Demo time

- Gebruik Write vs WriteLine



# Witregels in C#

- Witregels in code doen er niet toe.
- Enkel tussen “ “ om naar scherm te brengen

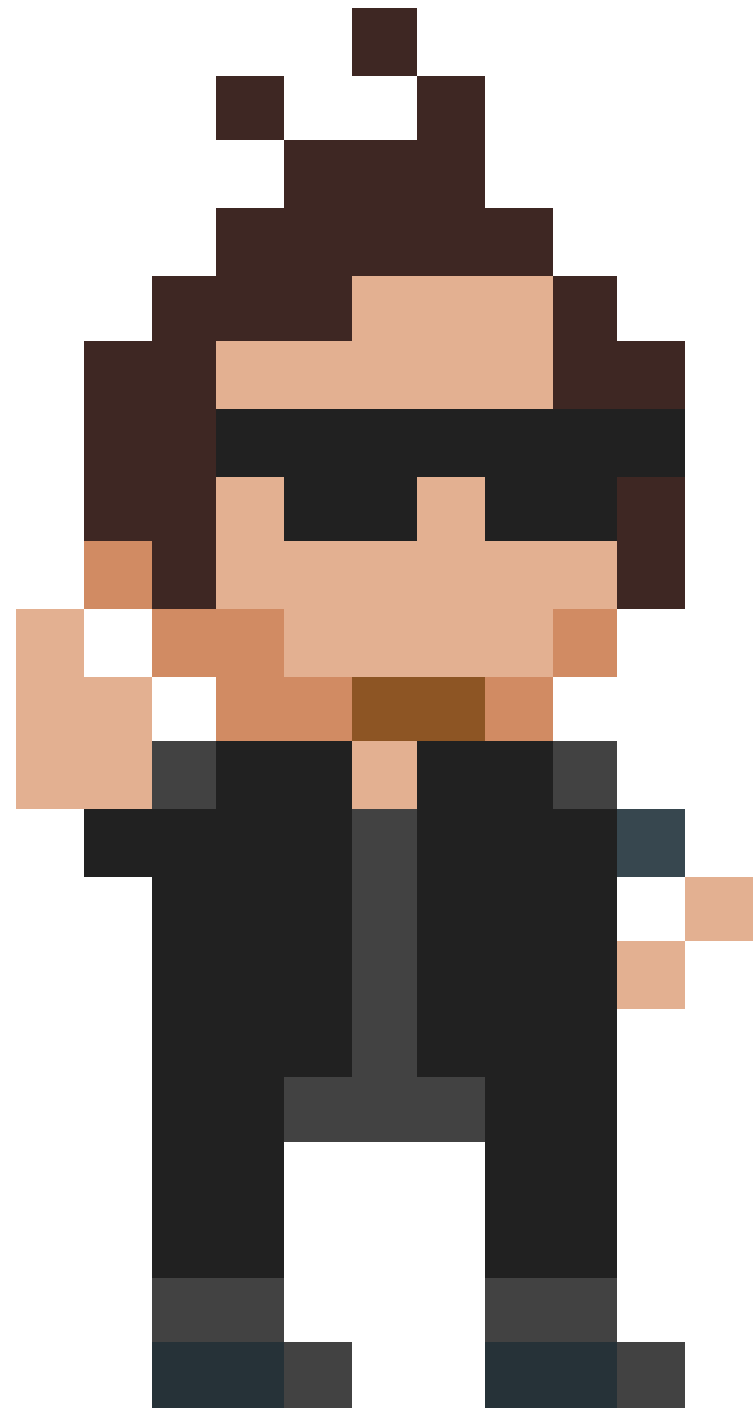


The screenshot shows a code editor window titled 'Program.cs\*' with a tab for 'MyFistProject'. The code is for a class 'Program' in the 'Demo1' namespace. It contains a 'Main' method that prints several lines of text. The code is formatted with multiple blank lines to improve readability, demonstrating that white space in C# code is ignored by the compiler but useful for humans.

```
1 using System; namespace Demo1 { class Program { static void  
    Main(string[] args) { Console.WriteLine("Hello  
World!"); Console.WriteLine("Hoi, ik ben  
het!"); Console.WriteLine("Wie ben jij?!"); string  
result; result = Console.ReadLine(); Console.WriteLine  
("Dag"); Console.WriteLine(result); Console.WriteLine  
("hoe gaat het met je?"); } } }
```

# Demo time

- Witregels en spaties



# Zinnen aan elkaar plakken

- Met behulp van +-operator

```
Console.WriteLine("Dag " + result + " hoe gaat het met je?");
```



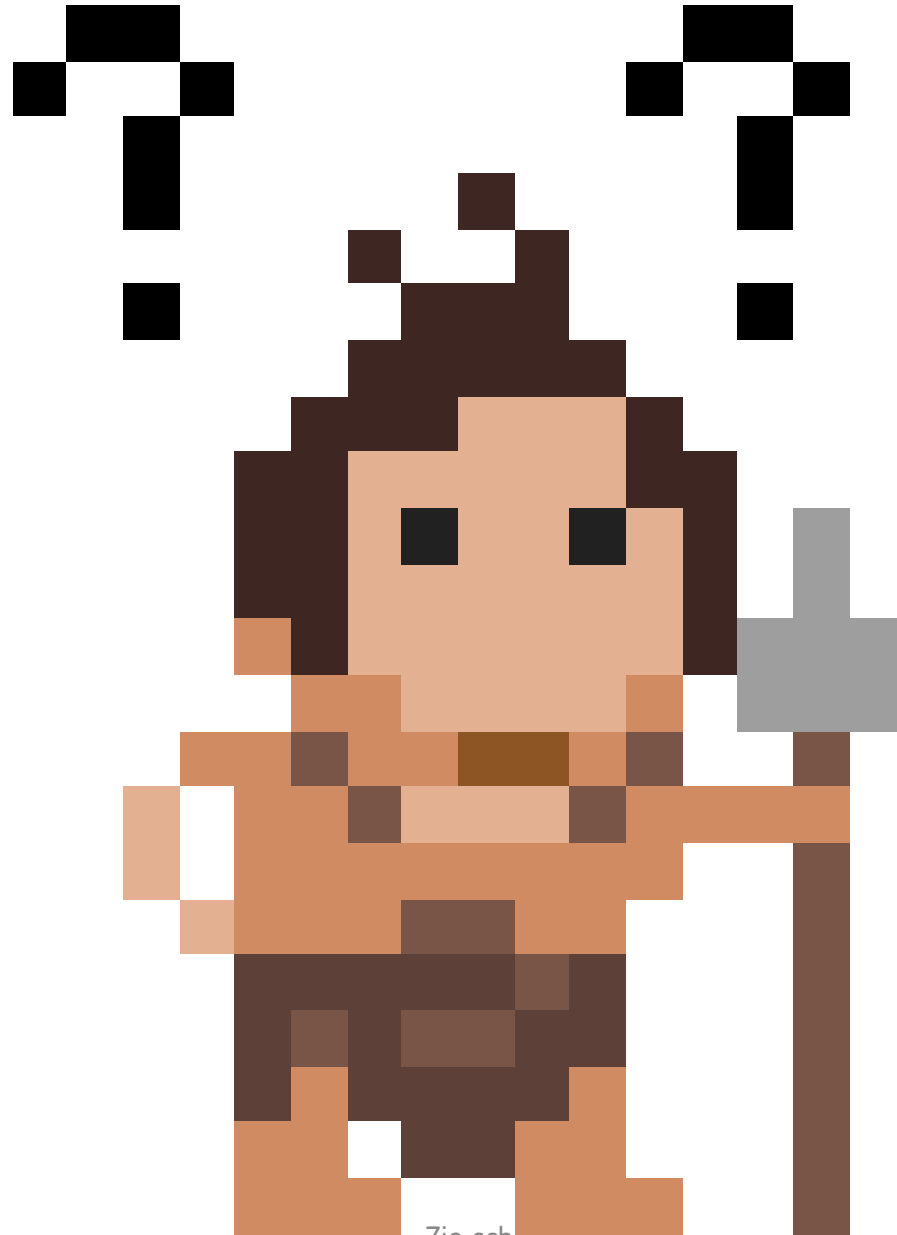
# Meer input vragen

- Per input van gebruiker nodig:
  - `Console.ReadLine();`
  - Resultaat in nieuwe variabele plaatsen

# Demo time

- Zinnen plakken met +
- Meer input vragen





Zie scherp