Aalto University
School of Arts, Design
and Architecture

Clean Code

The next 45min

- **Clean Code**
  - What, why, how

- **This is mostly based on**

  Clean Code: A Handbook of Agile Software Craftsmanship

  By Robert C. Martin

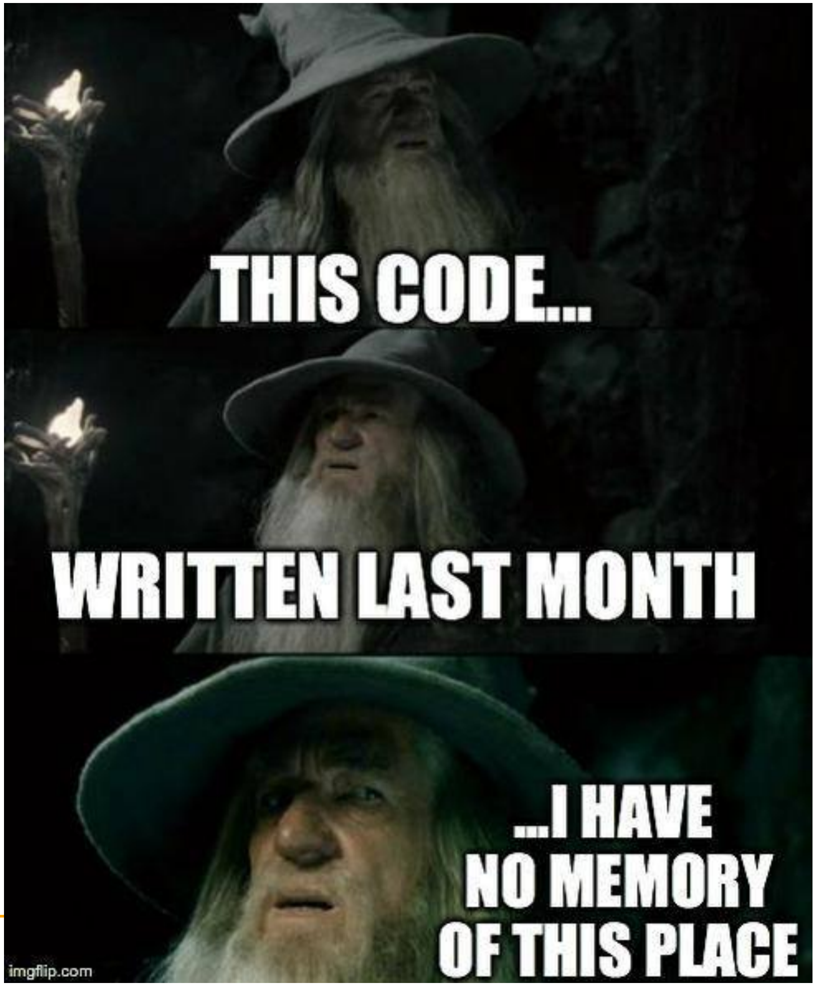
- **The book is available via Aalto Accounts from:**

  https://learning.oreilly.com/library/view/clean-code-a/9780136083238/

(There is also a Video Series which… exists https://learning.oreilly.com/videos/clean-code/9780134661742/9780134661742-CODE_01_00_00)

What is Clean Code?

- **Programmers are authors**
  - -> You write something and other people have to read it
  - <u>You are communicating</u>
- **Sooner or later you will have to read code from other people**
  - … or from yourself 2 weeks ago.



Let's have a look at Jan's project that is almost finished but he just can't get motivated to spend 3 hours reading scripts to understand where he left off and so it is slowly rotting on his hard-drive, doomed to become another project he never finishes.

So what is Clean Code?

Clean Code allows us to understand what code does without having to decipher it.

→ Good Code is readable like a book.

- **It flows, has internal logic and communicates the reason why it is there.**

What is included in clean code?

- **Human-Readable names for**
  - Variables
  - Functions
  - Arguments
  - Classes
  - Directories
- **Good Comments**
- **Clear Formatting**

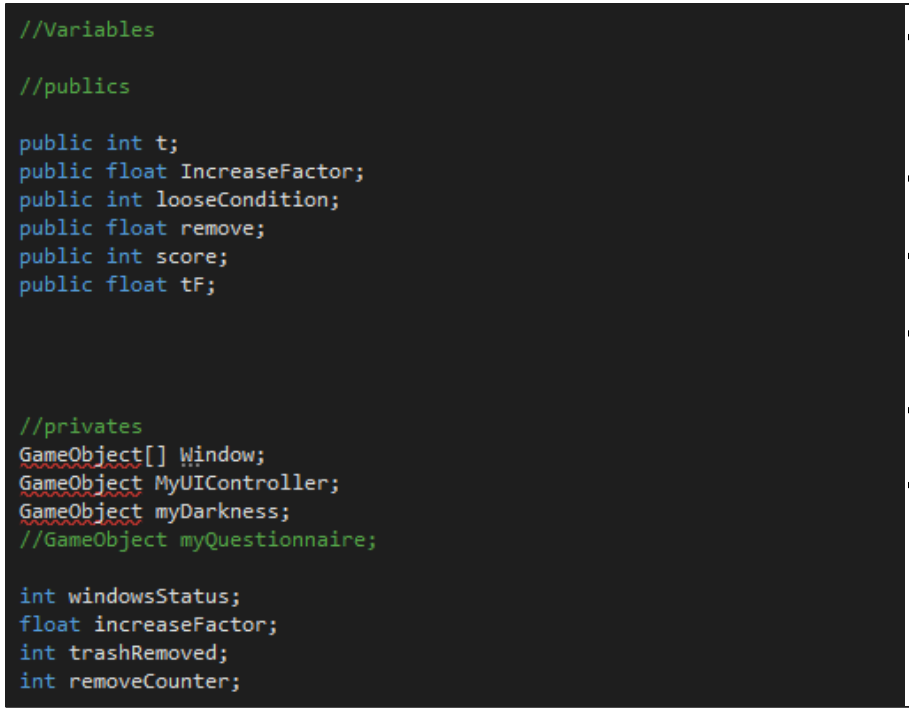
- **Is there one single rule for clean code?**
  - Short Answer: No

Nomen est Omen

Naming things

- Name things as what they are.
- Avoid Disinformation
- Make Meaningful Distinctions
- Use Pronouncable Names
- Use Searchable Names
- Functions should include a Verb

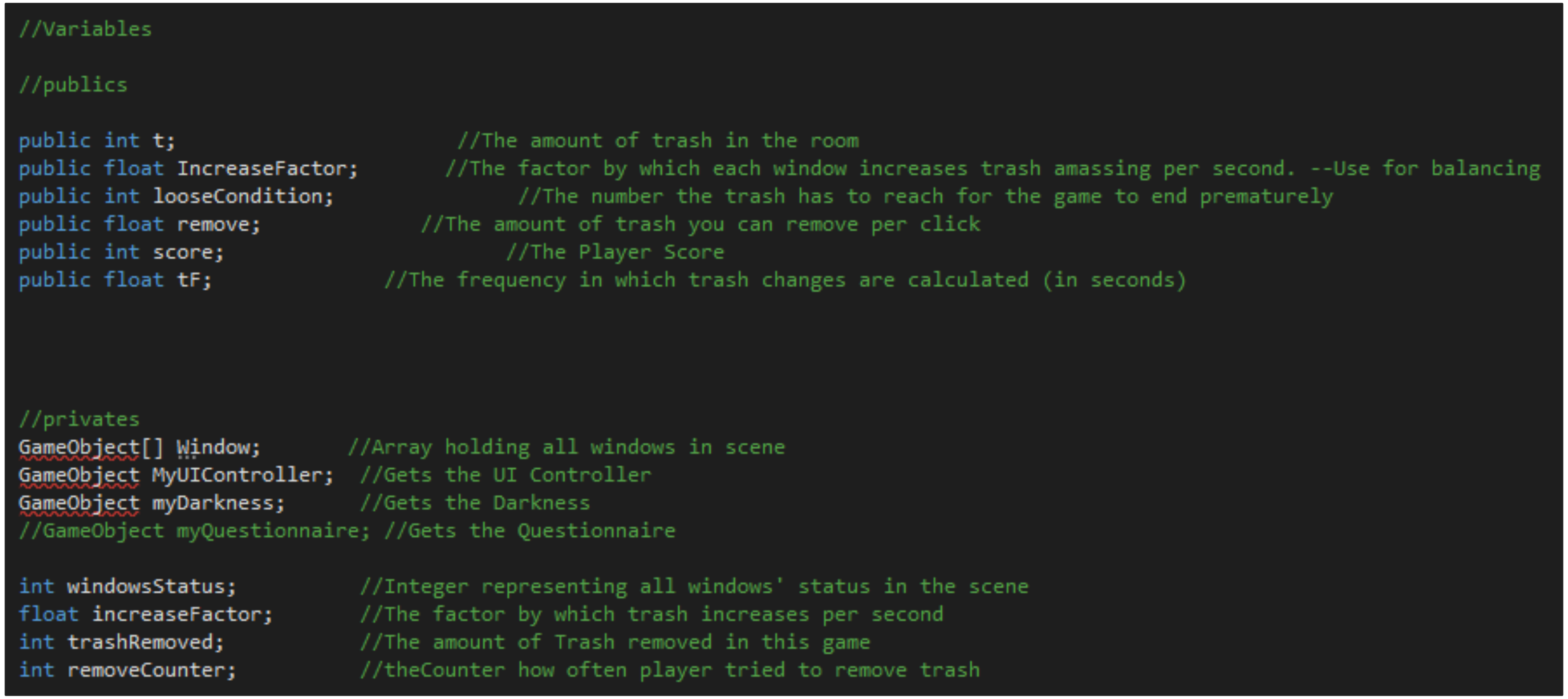
The Good, the Bad, and the Ugly



```
//Variables

//publics

public int t;
public float IncreaseFactor;
public int looseCondition;
public float remove;
public int score;
public float tF;



//privates
GameObject[] Window;
GameObject MyUIController;
GameObject myDarkness;
//GameObject myQuestionnaire;

int windowsStatus;
float increaseFactor;
int trashRemoved;
int removeCounter;
```

- **Name things as what they are.**
- **Avoid Disinformation**
- **Make Meaningful Distinctions**
- **Use Pronouncable Names**
- **Use Searchable Names**
- **Functions should include a Verb**

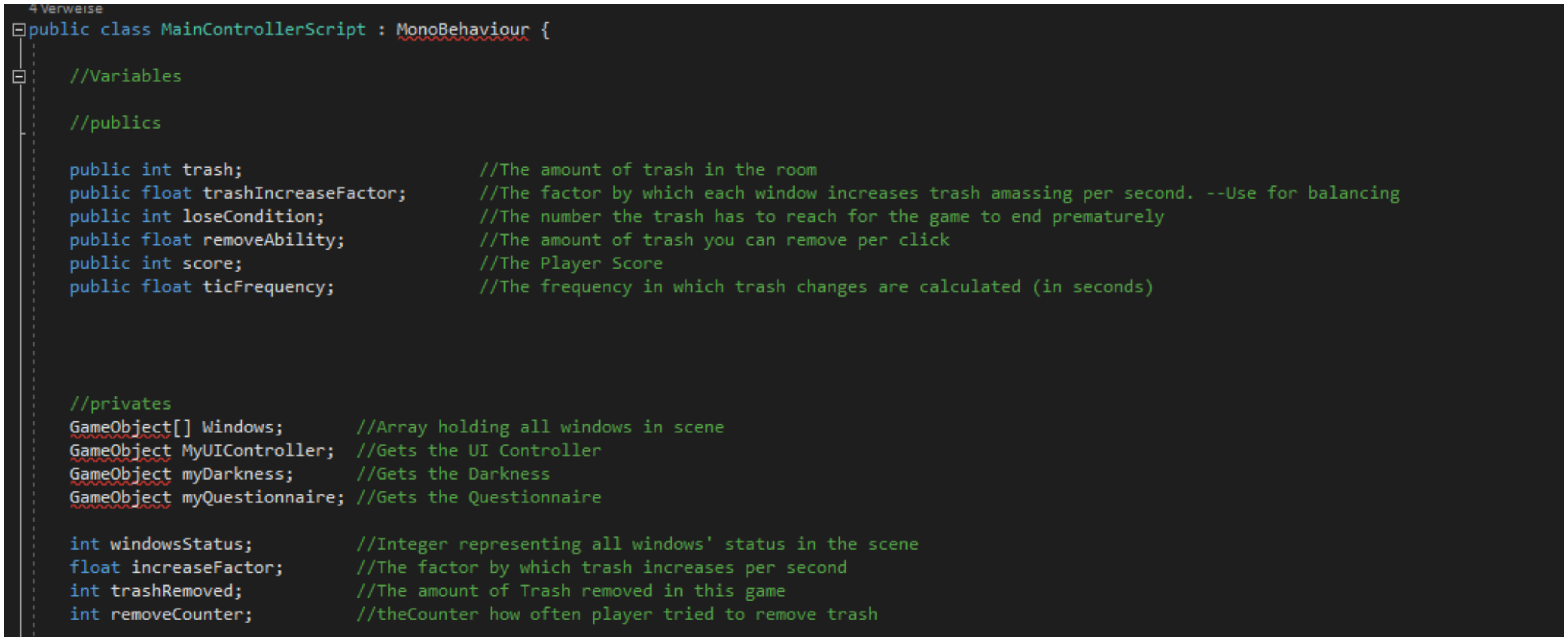
The Good, the Bad, and the Ugly



```
//Variables

//publics

public int t;                          //The amount of trash in the room
public float IncreaseFactor;           //The factor by which each window increases trash amassing per second. --Use for balancing
public int looseCondition;                   //The number the trash has to reach for the game to end prematurely
public float remove;                  //The amount of trash you can remove per click
public int score;                            //The Player Score
public float tF;                   //The frequency in which trash changes are calculated (in seconds)




//privates
GameObject[] Window;       //Array holding all windows in scene
GameObject MyUIController;  //Gets the UI Controller
GameObject myDarkness;      //Gets the Darkness
//GameObject myQuestionnaire; //Gets the Questionnaire

int windowsStatus;          //Integer representing all windows' status in the scene
float increaseFactor;       //The factor by which trash increases per second
int trashRemoved;           //The amount of Trash removed in this game
int removeCounter;          //theCounter how often player tried to remove trash
```

Aalto University
School of Arts, Design
and Architecture

The Good, the Bad, and the Ugly



```
4 Verweise
public class MainControllerScript : MonoBehaviour {

    //Variables

    //publics

    public int trash;                   //The amount of trash in the room
    public float trashIncreaseFactor;   //The factor by which each window increases trash amassing per second. --Use for balancing
    public int loseCondition;           //The number the trash has to reach for the game to end prematurely
    public float removeAbility;         //The amount of trash you can remove per click
    public int score;                   //The Player Score
    public float ticFrequency;          //The frequency in which trash changes are calculated (in seconds)




    //privates
    GameObject[] Windows;        //Array holding all windows in scene
    GameObject MyUIController;   //Gets the UI Controller
    GameObject myDarkness;       //Gets the Darkness
    GameObject myQuestionnaire;  //Gets the Questionnaire

    int windowsStatus;           //Integer representing all windows' status in the scene
    float increaseFactor;        //The factor by which trash increases per second
    int trashRemoved;            //The amount of Trash removed in this game
    int removeCounter;           //theCounter how often player tried to remove trash
```

Functions

Functions should be

- **…small**
- **…doing one thing**
- **…ordered from Top to Bottom**
- **…have as little arguments as possible**

Let‘s write a short script

Comments

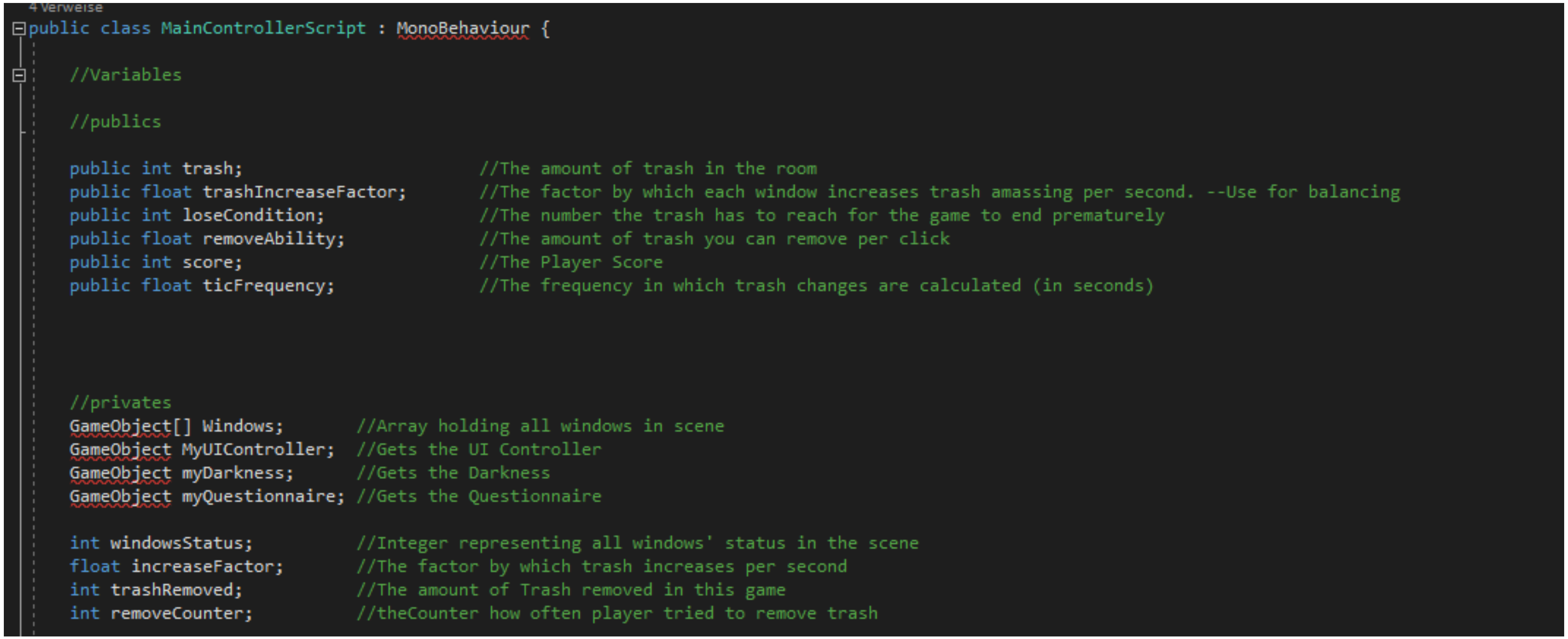
Technically: You shouldn’t need them

However:

- **Some Comments are needed (e.g., legal comments)**
- **Sometimes you need to communicate more**
- **Comments can be used as headers**
  - (though again: Your code shouldn’t be that long)

But: Comments can easily become outdated or false

Comments



```
4 Verweise
public class MainControllerScript : MonoBehaviour {

    //Variables

    //publics

    public int trash;                   //The amount of trash in the room
    public float trashIncreaseFactor;   //The factor by which each window increases trash amassing per second. --Use for balancing
    public int loseCondition;           //The number the trash has to reach for the game to end prematurely
    public float removeAbility;         //The amount of trash you can remove per click
    public int score;                   //The Player Score
    public float ticFrequency;          //The frequency in which trash changes are calculated (in seconds)




    //privates
    GameObject[] Windows;        //Array holding all windows in scene
    GameObject MyUIController;   //Gets the UI Controller
    GameObject myDarkness;       //Gets the Darkness
    GameObject myQuestionnaire; //Gets the Questionnaire

    int windowsStatus;           //Integer representing all windows' status in the scene
    float increaseFactor;        //The factor by which trash increases per second
    int trashRemoved;            //The amount of Trash removed in this game
    int removeCounter;           //theCounter how often player tried to remove trash
```

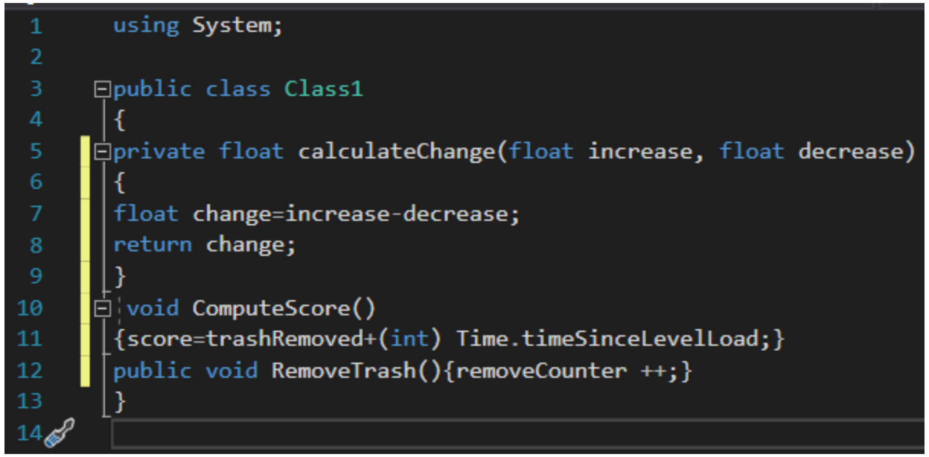
Formatting

- **Luckily: Formatting is done *for the most part* by your editor**
  - Still: The compiler does not care for spaces, tabs or returns, so you can use them at will.
- **Also: Space is cheap!**

Rules for Space

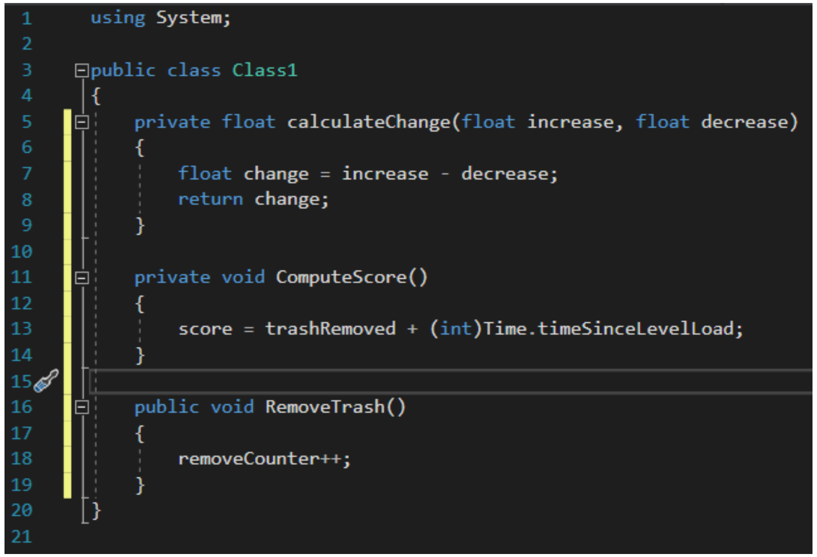
1. Density indicates relatedness
2. Indents indicate levels
3. Lines should be short

For example



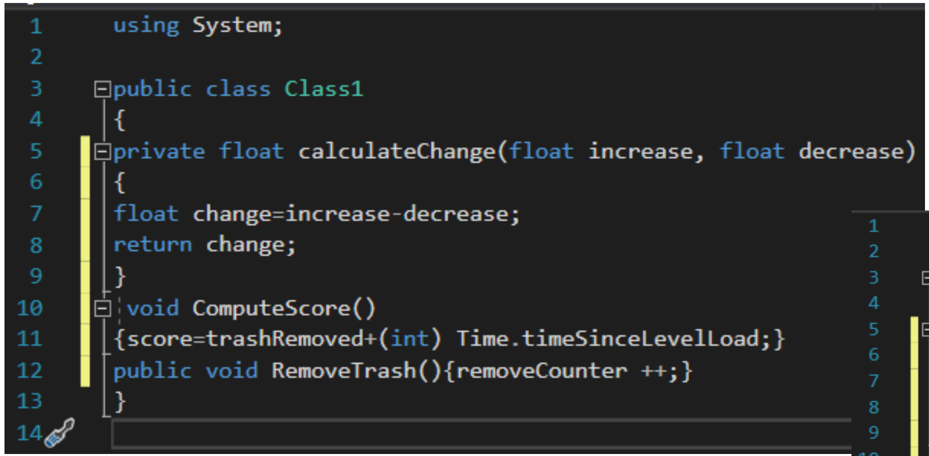
```
using System;

public class Class1
{
private float calculateChange(float increase, float decrease)
{
float change=increase-decrease;
return change;
}
 void ComputeScore()
{score=trashRemoved+(int) Time.timeSinceLevelLoad;}
public void RemoveTrash(){removeCounter ++;}
}
```

For example

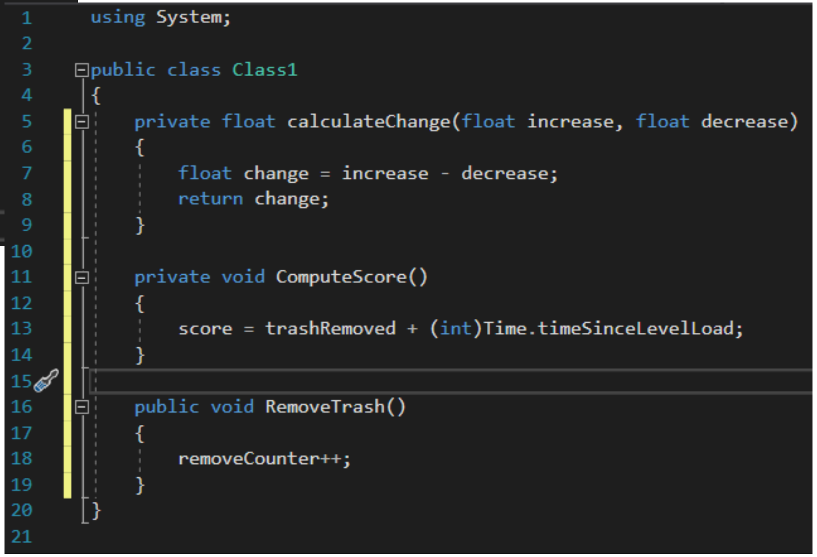


```
using System;

public class Class1
{
    private float calculateChange(float increase, float decrease)
    {
        float change = increase - decrease;
        return change;
    }

    private void ComputeScore()
    {
        score = trashRemoved + (int)Time.timeSinceLevelLoad;
    }

    public void RemoveTrash()
    {
        removeCounter++;
    }
}
```

For example



```
using System;

public class Class1
{
private float calculateChange(float increase, float decrease)
{
float change=increase-decrease;
return change;
}
 void ComputeScore()
{score=trashRemoved+(int) Time.timeSinceLevelLoad;}
public void RemoveTrash(){removeCounter ++;}
}
```



```
using System;

public class Class1
{
    private float calculateChange(float increase, float decrease)
    {
        float change = increase - decrease;
        return change;
    }

    private void ComputeScore()
    {
        score = trashRemoved + (int)Time.timeSinceLevelLoad;
    }

    public void RemoveTrash()
    {
        removeCounter++;
    }
}

```

In Conclusion

Aalto University
School of Arts, Design
and Architecture

If code is a language
Clean Code is your handwriting

In Conclusion

- **Clean Code is a process**
- **Try to be as clear as possible when you write your code**
  - Both for others and for yourself

- **What is considered clean code changes but the idea stays the same.**

- **Clean Code won‘t make or break your program**
  - Case in point: VVVVVV Source Code