**Code Style Guide**

Inhoud

[Clean coding principles 2](#_Toc118719154)

[Regels 2](#_Toc118719155)

[C# Style Guide 2](#_Toc118719156)

Clean coding principles

Deze principes helpen developers om schone, begrijpbare en kwalitatief goede code te schrijven. Dit zijn geen regels die 1 op 1 getoetst kunnen worden en dat is ook niet het doel hiervan. Het doel hiervan is om na het schrijven van code na te gaan of de code bijvoorbeeld eenvoudiger kan. Hieronder staan een aantal CLEAN code principles beschreven.

* KISS: Keep It Stupid Simple: Geschreven code/methodes moeten simpel zijn. Ze moeten geen onnodige complexiteit bevatten die niet nodig is voor het doel van de methode. Je moet je afvragen: Kan ik deze code eenvoudiger maken?
* DRY: Don't Repeat Yourself: Code die al is geschreven zou niet nog een keer geschreven moeten worden. Als een methode lijkt op een andere methode, dan moet er gekeken worden of deze generiek gemaakt kunnen worden.
* YAGNI: You aren't gonna need it: Schrijf geen code die niet gebruikt wordt. Dit zorgt voor onnodige tests en resources.
* Prefereer leesbaarheid: Schrijf de code zo dat hij goed leesbaar is voor anderen, niet zo kort mogelijk.
* Beoefen consistentie: Schrijf code zo consistent mogelijk.

Regels

1. Er mogen niet meerdere public classes in één bestand staan, elke klasse heeft een eigen bestand. Dit geldt ook voor interfaces en enums.
2. Als je een folder aanmaakt wordt dit gedaan met PascalCase.

C# Style Guide

1. Schrijf maar één statement per regel.
2. Schrijf maar één declaratie per regel.
3. Comments moeten geschreven worden op een aparte regel.
4. Voeg een spatie toe tussen '//' en de comment zelf.
5. Gebruik bij lokale variabelen alleen 'var' wanneer het type van een variabele overduidelijk is. In andere gevallen moet bijvoorbeeld Integer of String worden gebruikt. Bijvoorbeeld bij uitkomsten van een methode.
6. In het algemeen moet een Integer gebruikt worden voor getallen i.p.v. var. Dit zorgt voor bruikbaarheid in andere frameworks en libraries.
7. Gebruik try- and catch statement voor het afhandelen van (de meeste) Exceptions.
8. Gebruik logische classnamen.
9. Wij gebruiken geen inline if/for/while-statements voor de leesbaarheid, dus altijd brackets toevoegen aan if/for/while-statements.
10. Wij gebruiken geen 'do while' maar gewoon 'while'.
11. Constante variabelen moeten met ALL\_UPPER geschreven worden.
12. Gebruik zoveel mogelijk de C# primitieve types (bool, string en int ipv Boolean, String en Integer), tenzij het echt nodig is.
13. Brackets moeten op een nieuwe regel.
14. Bij alle variabelen moet een access modifier voor staan.
15. Klassenamen = PascalCase
16. Private variabelen = \_camelCase
17. Public variabelen = camelCase
18. Functienamen = PascalCase
19. Parameters = camelCase

Een uitgebreide versie van de regels kun je vinden op [C-Sharp corner naming guide](https://www.c-sharpcorner.com/UploadFile/8a67c0/C-Sharp-coding-standards-and-naming-conventions/). Microsoft biedt zelf ook een zeer uitgebreide design guideline, als je dit interessant vind kun je deze vinden op [Microsoft Guide lines](https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/).