styleguide

*Examen Volg Systeem (EVS)*

### Opdrachtnemers

|  |  |
| --- | --- |
| Naam, Studentnummer, Klas | Mike ten Klooster, 226751, DEV3 |

|  |  |
| --- | --- |
| Cohort | 2016-2017, DEV3 |
| Crebonummer | **25187** |
| Opleiding naam | **Applicatie- en mediaontwikkelaar** |
| Onderwijsinstelling | **Friesland College, Heerenveen** |

### Cliënten

|  |  |
| --- | --- |
| Naam | Luke ten Klooster |
| Naam | **Mike ten Klooster** |

### Document

|  |  |
| --- | --- |
| Datum publicatie | 08-09-2020 |
| Documentversie | **1.0** |

# Inhoudsopgave

[1. Inhoudsopgave 2](#_Toc50459777)

[2. Inleiding / Management samenvatting 3](#_Toc50459778)

[2.1 Versie geschiedenis 3](#_Toc50459779)

[2.2 Distributie 4](#_Toc50459780)

[3. Globale guidelines 4](#_Toc50459781)

[4. PHP: Inline-documentatie 5](#_Toc50459782)

[4.1 Individuele Class 5](#_Toc50459783)

[4.2 Afgetakte Class 5](#_Toc50459784)

[4.3 Methodes 6](#_Toc50459785)

[4.4 Overige 6](#_Toc50459786)

[5. Javascript: Inline-documentatie 7](#_Toc50459787)

[5.1 Individuele Class 7](#_Toc50459788)

[5.2 Afgetakte Class 7](#_Toc50459789)

[5.3 Functies 8](#_Toc50459790)

[6. CSS: Inline-documentatie 9](#_Toc50459791)

[6.1 Bestand 9](#_Toc50459792)

[6.2 Color 9](#_Toc50459793)

[7. Class declaratie 10](#_Toc50459794)

[8. PHP: Methodes declaratie 10](#_Toc50459795)

[9. Javascript: Methodes declaratie 10](#_Toc50459796)

[10. PHP: Variabele declaratie 11](#_Toc50459797)

[11. HTML/CSS: Id/Classname declaratie 11](#_Toc50459798)

[12. Statements 11](#_Toc50459799)

[12.1 Algemeen 11](#_Toc50459800)

[12.2 If-Else Statements 12](#_Toc50459801)

[12.3 Return Statements 13](#_Toc50459802)

[13. Gebruikte documenten/ Bronnen 14](#_Toc50459803)

# Inleiding / Management samenvatting

In dit document wordt beschreven hoe de code geschreven moet worden.

## Versie geschiedenis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versie | Datum | Status | Deelnemers | Omschrijving |
| 1.0 | 08-09-2020 | Opzet | Mike ten Klooster | Eerste opzet van het document. |

**Opzet =** Ongepubliceerde, onvoltooide versie.  
**Aangeboden =** Aangeboden aan cliënt, wacht op acceptatie of revisie.  
**Geweigerd =** Geweigerd door cliënt, vereist nieuwe revisie.  
**Geaccepteerd =** Geaccepteerd door cliënt, basis voor Design Document.

## Distributie

|  |  |  |
| --- | --- | --- |
| Naam | Rol | Email |
| Mike ten Klooster | Opdrachtnemer | [mike.tenklooster001@fclive.nl](mailto:mike.tenklooster001@fclive.nl) |

Alle opdrachtnemers worden direct op de hoogte gesteld van veranderingen in dit document, omdat dit invloed heeft op de codebase.

# Globale guidelines

* Maximaal één class per bestand!
* De class naam moet gelijk zijn aan de bestandsnaam. Bijvoorbeeld een class genaamd ‘*Document*’ heeft dan ook de bestandsnaam ‘*Document.php*’.
* We maken gebruik van spaties. Eén tab staat gelijk aan vier spaties.
* Alles wat in de code geschreven wordt, zowel variabelen als documentatie, moet geschreven worden in het *Engels*.
* Er mag gebruik worden gemaakt van ***CSS Variabelen***.

# PHP: Inline-documentatie

## Individuele Class

Een op zichzelf staande klasse moet op de volgende wijze gedocumenteerd worden.

/\*\*----------------------------------------------  
 \* <class name="ExampleClass"/>

\*

\* <summary>

\* This is the class summary.

\* </summary>  
 \*   
 \* <code>  
 \* // Tells how to interact with the class.  
 \* ExampleClass ec = new ExampleClass();  
 \* </code>

\*/----------------------------------------------

Class ExampleClass  
{  
 // Class contents…  
}

## Afgetakte Class

Een klasse die afgetakt is van een andere klasse, moet op de volgende manier gedocumenteerd worden.

/\*\*----------------------------------------------  
 \* <class implements="InterfaceClass" name="ExampleClass"/>

\*

\* <summary>

\* This is the class summary.

\* </summary>  
 \*   
 \* <code>  
 \* // Tells how to interact with the class.  
 \* ExampleClass ec = new ExampleClass();  
 \* </code>

\*/----------------------------------------------

Class ExampleClass extends InterfaceClass  
{  
 // Class contents…  
}

## Methodes

Een methode moet op de volgende manier gedocumenteerd worden.

/\*\*--------------------------------------------------------------------------  
 \* <method name="MethodName"/>

\*

\* <summary>

\* This is the summary about what the method does.

\* </summary>  
 \*

\* <param type="string" name="parameterName">Parameter description.<param/>  
 \*

\* <returns type="string">  
 \* Tells something about what the return should be and when..  
 \* <returns/>

\*/--------------------------------------------------------------------------  
public function MethodName($parameterName)  
{  
 // Method contents…  
}

## Overige

Voor alle overige inline-documentatie wordt gestart met twee forward slashes.  
//

En dus niet met één slash en twee sterretjes.  
/\*\* \*/

# Javascript: Inline-documentatie

## Individuele Class

Een op zichzelf staande klasse moet op de volgende wijze gedocumenteerd worden.  
   
/\*\*  
 \* Class ExampleClass  
 \*  
 \* This is the class summary.  
 \*  
 \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Tells how to interact with the class.  
 \* var ec = new ExampleClass();  
 \*/

class ExampleClass  
{  
 // Class contents…  
}

## Afgetakte Class

Een klasse die afgetakt is van een andere klasse, moet op de volgende manier gedocumenteerd worden.  
  
/\*\*  
 \* Class ExampleClass  
 \*  
 \* This is the class summary.  
 \*  
 \* @extends BaseClass  
 \*  
 \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 \* Tells how to interact with the class.  
 \* var ec = new ExampleClass();  
 \*/  
class ExampleClass extends BaseClass  
{  
 // Class contents…  
}

## Functies

Een functie moet op de volgende manier gedocumenteerd worden.  
  
/\*\*  
 \* Function functionName  
 \*  
 \* This is the function description  
 \*  
 \* @param {type} parameterName Parameter description.  
 \*   
 \* @return {type} Description about the return.  
 \*/  
function functionName(parameterName)  
{  
 // Function contents…  
}

# CSS: Inline-documentatie

## Bestand

Een CSS bestand moet op de volgende wijze gedocumenteerd worden.  
   
/\*--------------------------------------------------------\*\  
 file.css  
  
 Description of what this CSS File contains for styling.  
\\*--------------------------------------------------------\*/

## Color

Een kleur CSS Styling moet op de volgende manier beschreven worden.  
   
.label-danger {  
 /\* Light Grey \*/  
 background-color: #8c8d8f;

}

# Class declaratie

* Voor de class namen wordt er gebruik gemaakt van *PascalCase*.

# PHP: Methodes declaratie

* Voor de methode namen wordt er gebruik gemaakt van *PamelCase*.
* Wanneer er twee of meer parameters aan een methode worden meegegeven, worden deze beide op een aparte regel gedefinieerd, als volgt.  
  /\*\* …  
  public function MethodName(  
   $firstParameterName,  
   $secondParameterName  
  ) {  
   // Method contents…  
  }
* Wanneer je een nieuwe functie aanroept die meerdere *Monad’s* bevat, noteer je deze na de eerste op een nieuwe lijn. Zoals volgt.   
  int userId = user->getid();

# Javascript: Methodes declaratie

* Voor de methode namen wordt er gebruik gemaakt van *camelCase*.

De rest van de guide is hetzelfde als bij *PHP*

# PHP: Variabele declaratie

* Voor de variabele namen wordt er gebruik gemaakt van *camelCase*.
* Maximaal één variabele declaratie op één lijn! Dus niet dit:  
  $firstVariable = “test”, $secondVariable = “test2”;  
  Maar dit:  
  $firstVariable = “test”

$secondVariable = “test2”;

# HTML/CSS: Id/Classname declaratie

* Voor de *ID’s* en *Classnames* voor HTML en CSS wordt er gebruik gemaakt van *kebab-case*.

# Statements

## Algemeen

* Elke Curly Bracket in een statement, moet op een nieuwe regel gedefinieerd worden. Dus, zowel voor een if-statement, for-loop en een try-catch.

if (condition)

{

statement;  
}

else

{

statement;

}

* Voor elke declaratie van een statement, moet er een spatie zitten tussen de declaratie en de conditie. Dus:

**if (condition)**

{

statement;  
}

En dus niet:  
**if(condition)**{  
 statement;  
}

## If-Else Statements

* Wanneer een if-statement **meer dan 2** condities bevat, moeten deze op een nieuwe regel gescheiden te worden. Dus:

if (

(condition) &&

(condition2) ||

(condition3)

) {

statement;  
}

else

{

statement;

}

En dus niet:

if ((condition) && (condition2) || (condition3))  
{

statement;  
}

else

{

statement;

}

* Ook wordt er voorkeur gegeven aan short if-statements, als dit kan. Bijvoorbeeld:

(condition) ? statement : statement2;

## Return Statements

* Een methode moet zo snel mogelijk ge-returned worden. Wanneer een methode een bepaald type teruggeeft, ziet dit er bijvoorbeeld zo uit:

///  
public function MethodName()

{  
 if (condition)

{

return false;  
 }  
  
 // More conditions…

return true;   
}

Wanneer een methode van het type void is en je een early return kan doen, return je gewoon empty.  
///  
public function MethodName()  
{  
 if (condition)

{  
 return;

}  
  
 // More conditions…

}

* Wanneer een methode een return true of false doet aan de hand van één conditie, dan doe je dat in één regel. Dus:

///  
public function MethodName()

{

return (condition) ? true : false;   
}

# Gebruikte documenten/ Bronnen

* <https://stackoverflow.com/questions/11273282/whats-the-name-for-hyphen-separated-case>