Bijlage 1.6 Coding conventions

Inhoud

[Inleiding 3](#_Toc21612265)

[Naamgeving van variabelen en functies 3](#_Toc21612266)

[Uitlijning 4](#_Toc21612267)

[Commentaarregels naar aanleiding van het doorvoeren van tickets 4](#_Toc21612268)

### Inleiding

De coding conventions betreffen naamgeving van variabelen en functies (zoals hoofdlettergebruik), uitlijning van de broncode en commentaarregels naar aanleiding van doorvoeren van tickets. Deze conventions zijn toepasbaar op onder meer C#, Delphi, PHP/Javascript en Java.

De voorbeelden zijn gecodeerd in PHP.

### Naamgeving van variabelen en functies

* namen van klassen - camelCase[[1]](#footnote-1)

class clsColumnHeader

{

private $columns;

private $connection;

* methoden – camelCase1

public function setColumnHeader($columnheader)

{ $this->columnheader = $columnheader;

}

* parameters van methoden – kleine letters

public function addColumn($column, $afterColumnName = false)

{

$column->setLookupValues($this->connection);

if ($afterColumnName)

{

$newcolumns = array();

* lokale variabelen – kleine letters

public function getWidthClass()

{ $colw = 0;

$widthclass = " class='col-xs-\* col-sm-\* col-md-\* col-lg-\*'";

$this->columns = array();

$this->connection = $connection;

}

#### Gebruik géén underscore (onderstreepte spatie) in namen.

* Gebruik zelfstandige naamwoorden of zelfstandig naamwoordzinnenvoor de naam van een klasse. Laat de klassenaam voorafgaan door cls.

class clsDateTimeColumn extends clsColumn

{

public function

### Uitlijning

* Lijn alle accolades verticaal uit: zet ze netjes onder elkaar. Spring binnen de accolades in met tabs van vijf spaties.

public function makeInput($name) // Overrides clsColumn->makeInput()

{ $readonly = " readonly='readonly'";

$readonlyclass = " form-control-plaintext";

$disabled = "disabled='true' ";

if (!$this->getReadOnly())

{ $readonly = "";

$readonlyclass = "";

$disabled = "";

}

$output

### Commentaarregels naar aanleiding van het doorvoeren van tickets

#### Toevoegen van regels code

// Toegevoegd n.a.v. ticketnummer [[1506]]

public function setFieldValueToDBFormat($getal)

{ if ((!$getal) || ($getal === ""))

{ return "NULL";

}

return str\_replace(",", ".", $getal);

}

// Einde Toegevoegd n.a.v. ticketnummer [[1506]]

#### Wijzigen van diverse regels code

// Gewijzigd n.a.v. ticketnummer [[1516]]

public function setFieldValueToDBFormat($getal)

{ if ((!$getal) || ($getal === ""))

{ return "NULL";

}

return str\_replace(",", ".", $getal);

}

// Einde Gewijzigd n.a.v. ticketnummer [[1516]]

#### Toevoegen/wijzigen van een enkele regel code

$column->setReadOnly(); // Toegevoegd n.a.v. ticketnummer [[1541]]

$this->ordersql = "ORDER BY a.Naam, i.Tijdstip"; // Gewijzigd n.a.v. ticketnummer [[1511]]

}

1. camelCase: één woord maken van afzonderlijke woorden; de eerste letter van het eerste woord is altijd een kleine letter. De overige woorden beginnen met een hoofdletter.

   Voorbeeld: Gekookte Eieren Eten wordt in camelCasing: gekookteEierenEten. [↑](#footnote-ref-1)