WEEK 1 - 2
Javascript Deel 2

DE HOGESCHOOL MET HET NETWERK

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be

Statements

Wat zijn statements

- Lijst van instructies die moeten worden uitgevoerd door een computer

- Worden opgebouwd uit:
  - Waardes
  - Operatoren
  - Expressies
  - Sleutelwoorden
  - Commentaren

Opbouw statements

- De lijst met instructies mag over meerdere regels worden verdeeld. Regelovergangen en inspringen worden genegeerd.
- Statements worden afgesloten met een puntkomma.
- De woorden in de statements zijn ***HOOFDLETTERGEVOELIG***!

Zoek de fouten…

```
document.getelementbyid(“inhoud”).InnerHTML = “Hallo”;
document.getelementbyid(“inhoud”).innerhtml = “Hallo”;
document.GetElementByID(“Inhoud”).innerhtml = “Hallo”;
document.getElementById(“inhoud”).innerHTML = “Hallo”;
```

```
document.getelementbyid(“inhoud”).InnerHTML = “Hallo”;
document.getelementbyid(“inhoud”).innerhtml = “Hallo”;
document.GetElementByID(“Inhoud”).innerhtml = “Hallo”;
document.getElementById(“inhoud”).innerHTML = “Hallo”;
```

Variabelen en constanten

Wat zijn variabelen

- Met variabelen kunnen we bepaalde gegevens in ons script onthouden om ze nadien opnieuw te gebruiken

- Variabelen hebben altijd een naam, ook wel ‘identifier’ genoemd

- *De waarde van een variabele kan doorheen de code gewijzigd worden, maar de identifier van de variabele blijft altijd hetzelfde!*

Variabelen declareren

- Nadat we een variabele *gedeclareerd* hebben, refereren we naar de waarde van die variabele door de naam te gebruiken

```
// Declaratie en initialisering in één regel schrijven
let aantal = 1;

// Declaratie en initialisering verspreiden
let aantal;
aantal = 1;
```

Identifier kiezen

Enkele belangrijke regels:

- Het eerste karakter moet een letter of underscore (_) zijn.
- De naam mag zowel hoofdletters als kleine letters bevatten.
- De naam mag letters, getallen, underscore en dollartekens bevatten.
- De naam mag geen spaties of andere lees- of speciale tekens bevatten.
- De naam mag geen gereserveerd woord zijn.

Gereserveerde woorden

abstract	arguments	await	boolean
break	byte	case	catch
char	class	const	continue
debugger	default	delete	do
double	else	enum	eval
export	extends	false	final
finally	float	for	function
goto	if	implements	import
in	instanceof	int	interface
let	long	native	new
null	package	private	protected
public	return	short	static
super	synchornized	switch	this
throw	throws	transient	true
try	typeof	var	void
volatile	while	with	yield

Zoek de fouten…

```
let 1cadeau;
let getal;
let aantal-repen;
let i;
let break;
let aantalKoekjes;
Let aantal_snoepjes;
let snoepjes&koekjes;
```

```
let 1cadeau;
let getal;
let aantal-repen;
let i;
let break;
let aantalKoekjes;
Let aantal_snoepjes;
let snoepjes&koekjes;
```

Wat zijn constanten

- Een constante is een variabele waarbij de waarde niet aangepast kan worden na toekenning.

```
// Declaratie en initialisering in één regel schrijven
const aantal = 1;

// Meerdere declaraties en initialisaties op één regel
const een = 1, twee = 2, drie = 3;
```

Gegevenstypen

Wat zijn gegevenstypes

Twee groepen gegevenstypen:

- Primitieve gegevenstypes
- Complexe- of objecttypen

Primitieve gegevenstypes

- Getallen (numbers)
- Tekenreeksen (strings)
- Logische waarden (booleans)
- Null
- NaN (not a number)
- Undefined

Complexe- of objecttypen

- Array
- Function
- Object

Komen in JS – Deel 3 diepgaander aan bod!

Getallen

- In Javascript kunnen we zowel gehele als gebroken getallen gebruiken.

```
let number = 13;
let decimal = 3.14;
let avagadro = 6.0221e23;
```

Getallen

- Een browser geeft een ingetypte waarde altijd als een string terug. Alvorens we het getal kunnen gebruiken in onze javascript-code, moeten we het omzetten naar een getal.

```
// String omzetten naar een geheel getal (integer)
parseInt(getal1)

// String omzetten naar een gebroken getal (floating point)
parseFloat(getal2)
```

Tekenreeksen - Strings

- Strings zijn reeksen van tekens (ASCII-karakters).
- In Javascript worden strings tussen enkele of dubbele aanhalingstekens geplaatst.

```
let zin = “Deze zin is een string”;
```

Tekenreeksen - Strings

- Speciale tekens, regelteruglopen, tabs en andere niet-afdrukbare tekens moeten we in een string vooraf laten gaan door een backslash (\)

```
alert(“Dit is de eerste regel\nDit is de tweede regel”);
alert(“Dit stukje staat voor een tabstop\tDit stukje na een tabstop);
alert(“Mijn favoriete quote is: \”Hakuna Matata\”.”);
alert(“Installeer de juiste driver in C:\\Windows\\System.”);
```

Tekenreeksen - Strings

- Via het gebruik van stringfuncties kunnen we informatie verkrijgen over een stringwaarde.

- Enkele stringfuncties:
  - .length()
  - .charArt()
  - .substring()
  - .indexOf()
  - …

Logische waarden – Booleaanse waarden

- Vergelijkingen, functies of lussen kunnen in Javascript een booleaanse waarde teruggeven.

- Een booleaanse waarde is waar (true) of onwaar (false).

True of false…

```
let getal = 30;


getal == 30;
getal < 31;
getal > 31;
Getal == 31;
```

```
let getal = 30;
```

```
getal == 30;          → True
getal < 31;           → True
getal > 31;           → False
Getal == 31;          → False
```

Null

- De waarde ‘Null’ staat letterlijk voor ‘niets’.
- ‘Null’ is de waarde van een niet-gedeclareerde variabele.

NaN

- De waarde NaN staat voor ‘not a number’.

- NaN krijgen we terug indien het programma een nummer had verwacht, maar deze niet heeft gekregen.

Undefined

- De waarde ‘undefined’ geeft aan dat een variabele nog niet geïnitialiseerd is of een functie niets retourneert.

```
let getal;
alert(getal);
```

Operatoren

Wat zijn operatoren

Met operatoren kunnen we de variabelen in een script:

- optellen
- Aftrekken
- Vergelijken
- … en overige handelingen mee laten uitvoeren.

Soorten operatoren

- Toewijzingsoperatoren (assignment operators)
- Wiskundige operatoren (arithmic operators)
- Stringoperatoren (string operators)
- Logische operatoren (logical operators)
- Bitoperatoren (bitwise operators)
- Vergelijkingsoperatoren (comparison operators)
- Speciale (overige) operatoren

Toewijzingsoperatoren

Het isgelijkteken (=) is een toewijzingsoperatoren en gebruiken we voor:

- het toewijzen van waarden aan variabelen;
- het wijzigen van waarden van variabelen.

```
// Kent de waarde 1 toe aan variabele getal1
getal1 = 1;
```

Wiskundige operatoren

+	Optellen
-	Aftrekken
*	Vermenigvuldigen
/	Delen
%	Restwaarde berekenen
++	Getal plus 1
--	Getal min 1

Wiskundige operatoren

```
let uitkomst = 10 + 5;
let uitkomst = 10 - 5;
let uitkomst = 10 * 5;
let uitkomst = 10 / 5;
let uitkomst = 11 % 5;

let uitkomst = 1;
uitkomst++;

let uitkomst = 6;
uitkomst--;
```

String operatoren

- Het plusteken (+) is een concatenatieoperator. Met deze operator kunnen we twee of meer strings samenvoegen tot één string.

```
let deel1 = “Dit is”
let deel2 = “een zin”
let uitkomst = deel1 + “ ” + deel2
```

Logische operatoren

- Met logische operatoren kunnen we nagaan of een expressie, bestaande uit logische waarden, waar (true) of onwaar (false) is.

Operator	Uitleg
&&	Logische ‘en’ Deze operator geeft ‘true’ terug indien alle expressies ‘true’ zijn. Deze operator geeft ‘false’ terug indien één of alle expressies ‘false’ zijn.
||	Logische ‘of’ Deze operator geeft ‘true’ terug indien één of alle expressies ‘true’ zijn. Deze operator geeft ‘false’ terug indien alle expressies ‘false’ zijn.
!	Logische ‘niet’ Deze operator geeft ‘true’ indien de expressie ‘false’ is. Deze operator geeft ‘false’ indien de expressie ‘true’ is.

Vergelijkingsoperatoren

- Vergelijkingsoperatoren geven, net zoals logische operatoren, ‘true’ of ‘false’ terug als resultaat, maar bij vergelijkingsoperatoren kunnen we ook met andere gegevenstypes werken.

Operator	Uitleg
==	Is gelijk aan
===	Is strikt gelijk aan
!=	Is niet gelijk aan
!==	Is niet strikt gelijk aan
<	Is kleiner dan
<=	Is kleiner of gelijk aan
>	Is groter dan
>=	Is groter of gelijk aan

True of False…

```
10 == ‘10’
10 === ‘10’
‘koekje’ != ‘Koekje’
10 < 50
10 <= 10
10 > 50
10 >= 50
```

```
10 == '10'                  → True
10 === '10'                 → False
'koekje' != 'Koekje'        → True
10 < 50                     → True
10 <= 10                    → True
10 > 50                     → False
10 >= 50                    → False
```

Speciale operatoren – *Typof operator*

- De typeof operator retourneert het gegevenstype van de variabele.

```
let waarde = true;
console.log(“Het gegevenstype is: “ + typeof waarde);
```

Speciale operatoren – *Voorwaardelijke operator*

- De voorwaardelijke operator kunnen we gebruiken voor voorwaardelijke expressies. Dit type expressies werkt als volgt:
  - Is een voorwaardelijke expressie ‘waar’, dan worden de bijhorende statements achter het vraagteken uitgevoerd.
  - Is een voorwaardelijke expressie ‘onwaar’, dan worden de bijhorende statements achter de dubbelepunt uitgevoerd.

Speciale operatoren – *Voorwaardelijke operator*

```
let dag = “Maandag”;
let uitspraak;
uitspraak = (dag == “Zaterdag”) ? “Joepie, feest!” : “Geen feest”;
console.log(uitspraak);
```