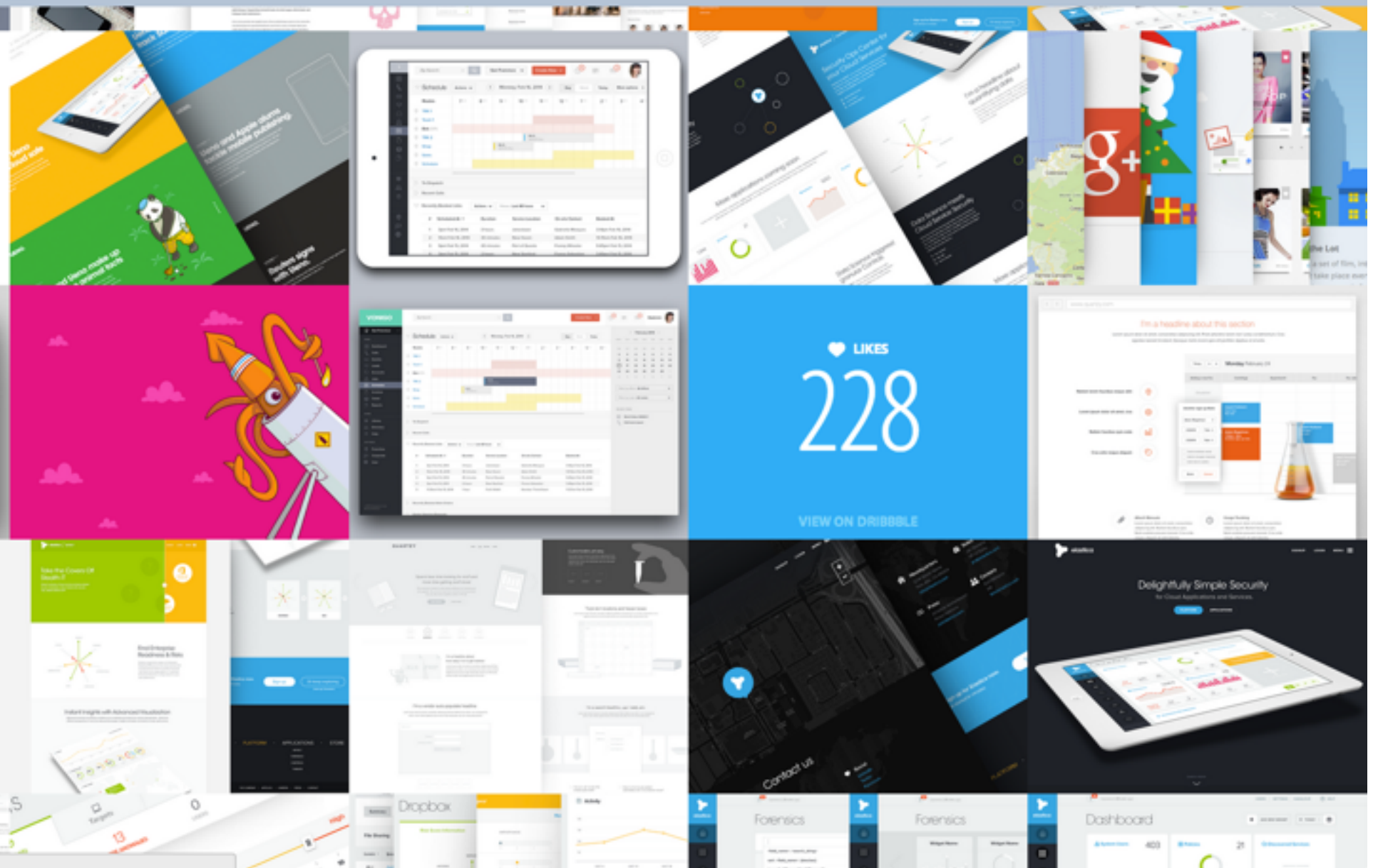
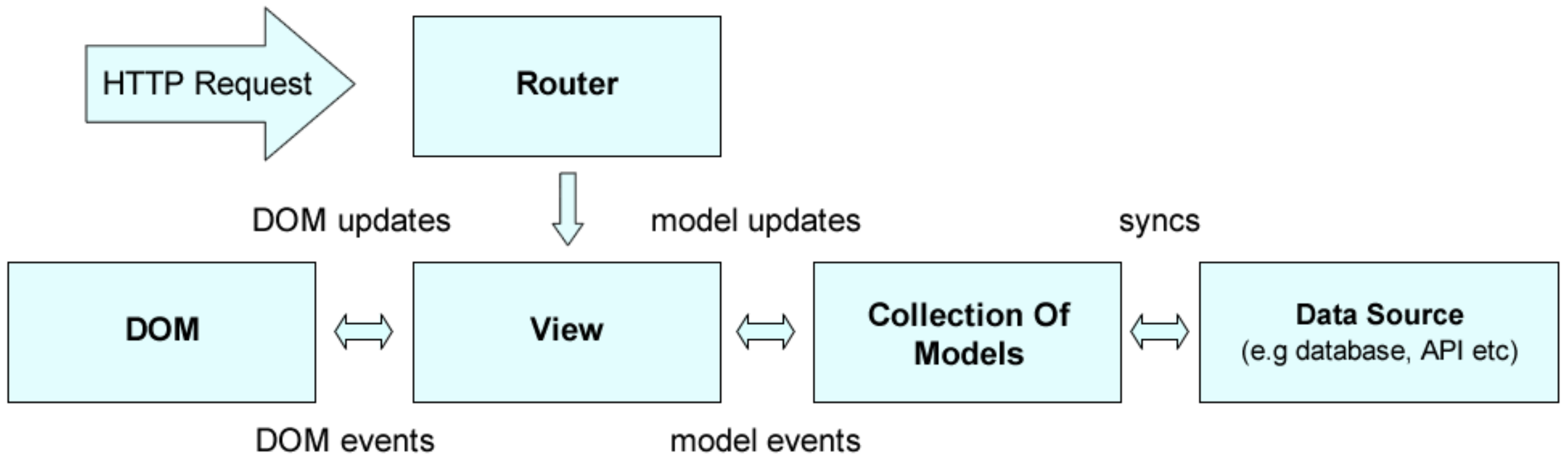


serious **#js**
to **infinity** and beyond!











ECMA6

<http://code.tutsplus.com/articles/use-ecmascript-6-today--net-31582>

code complexity

[http://www.tomdalling.com/blog/software-design/
fizzbuzz-in-too-much-detail/](http://www.tomdalling.com/blog/software-design/fizzbuzz-in-too-much-detail/)

Leerdoelen

Kennisgestuurde leerdoelen:

- De student begrijpt het OO paradigma in JavaScript
- De student begrijpt de principes van functional programming.
- De student kent de volgende JS patterns: module, factory en MV*.
- De student kent de inzet van diverse professionele JS library's.

Praktijkgestuurde leerdoelen:

- De student kan object georiënteerde JavaScript applicaties ontwikkelen.
- De student kan functional programming principes op een goede manier toepassen.
- De student kan de module, factory en de MV* pattern zelf toepassen.
- De student kan een professionele werkomgeving voor frontend development opzetten (JSLint, Minify, Coffescript, Grunt).
- De student kan Backbone als professionele library for frontend development goed toepassen.

Programma

Week	Inhoud	Deadline
1	JS functional programming	
2	JS OOP objects	
3	JS OOP inheritance	Summatief Voor vrijdag 17.00 uur inleveren
4	JavaScript views en underscore	
5	Backbone	
6	Backbone	
7		
8		
9	Toetsweek	Formatief Vrijdag voor 17.00 uur inleveren

Literatuur

Facultatief

Abonnement op <https://tutsplus.com>.

Verplicht

Zakas, N.C. (2014) The principles of object-oriented JavaScript.

GIT repository

<https://github.com/rimmertzelle/professional-javascript-fed01.git>

Opdrachten

Tussenopdracht

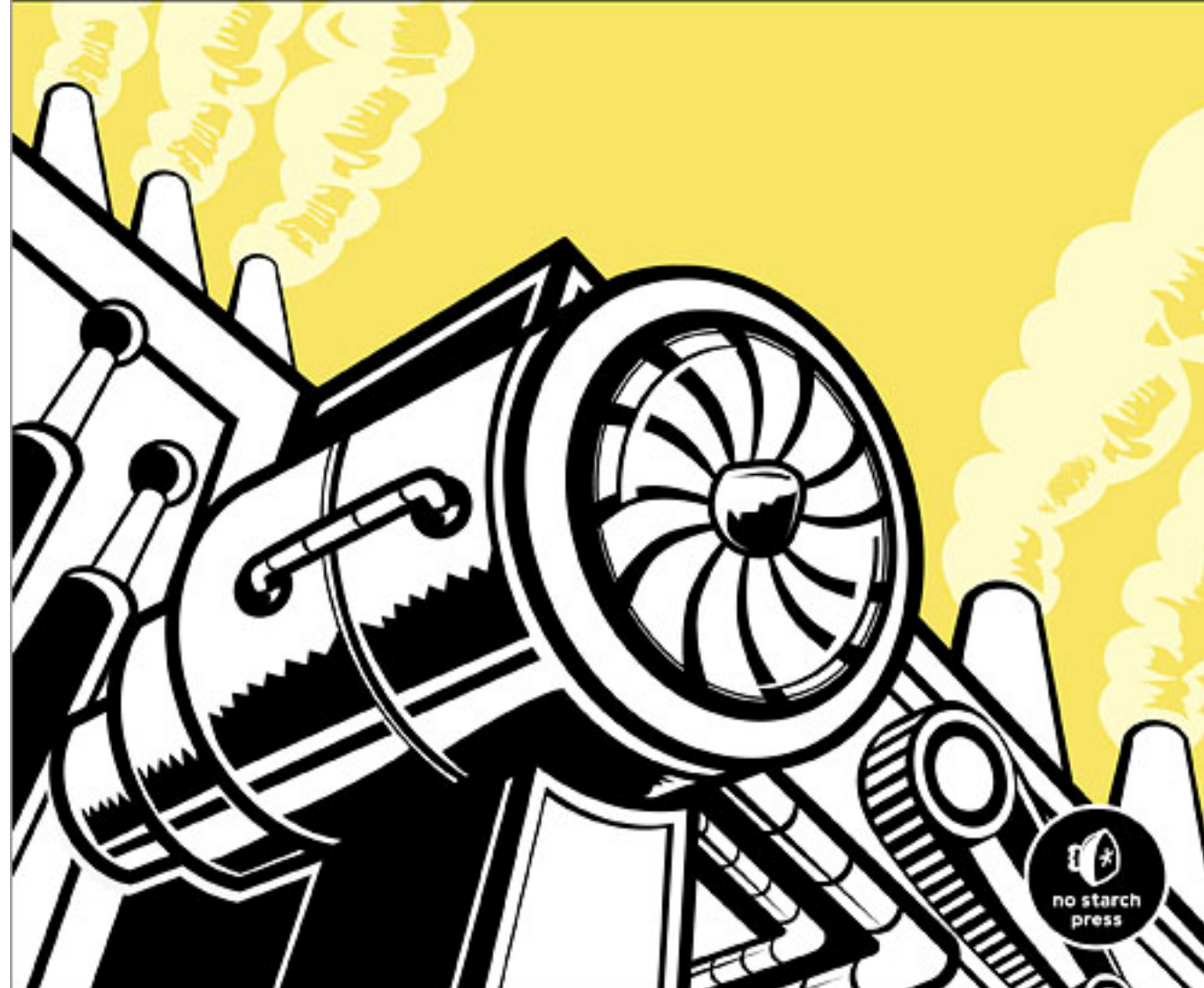
Feedback

Eindopdracht

Cijfer is eindcijfer

THE PRINCIPLES OF **OBJECT-ORIENTED** **JAVASCRIPT**

NICHOLAS C. ZAKAS



Learn Faster

tuts+™



Source Files, Tutorials, Resources

JOIN FOR JUST \$9 P/M FOR ALL TUTS SITES

primitive and referenced types

assignment

1. Probeer eens een primitieve variabele (bv een number) te kopiëren naar een andere variabele.
2. Tel bij de eerste var 20 op. Wat gebeurt er met de 2e var?

assignment

1. Probeer eens een object (met bijvoorbeeld een waarde amount) te kopiëren naar een ander object.
2. Tel bij de eerste amount 20 op. Wat gebeurt er met met de waarde amount in het gekopieerde object?

Primitive types

Boolean

Number

String

Null

Undefined

Primitive types

```
var color1 = 'red'  
var color2 = color1
```

Primitive types

color1	"red"
color2	"red"

Identifying primitive types

```
console.log(typeof "my-string"); // "string"
```

```
console.log(typeof 10); // "number"
```

```
console.log(typeof null); // "object"
```

Comparing

```
console.log("5" == 5);           //true
console.log("5" === 5);          //false
console.log(value === null);      //true or false
console.log(undefined == null);   //true
console.log(undefined === null);  //false
```


Referenced types

Object
Function
Array

Object

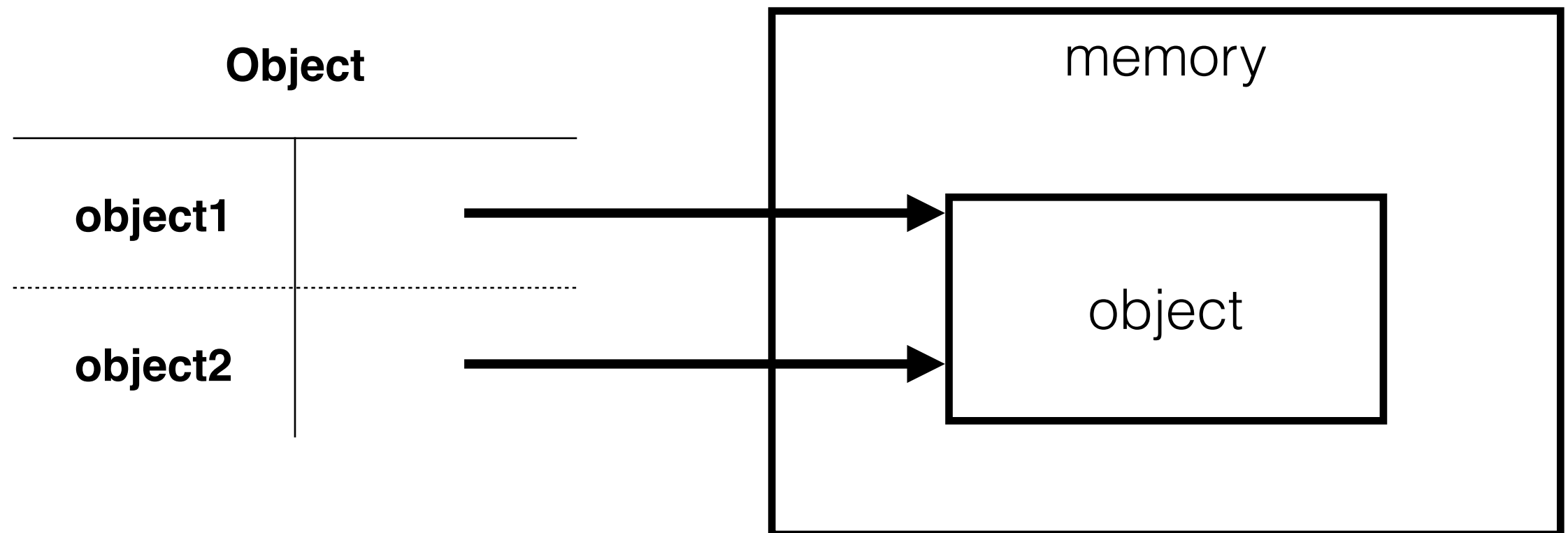
Object

Object	
name	value
name	value

Object

```
var object1 = new Object();  
var object2 = object1;
```

Object



Identifying Reference types

```
var items = [];  
var object = {};
```

```
//function literal  
function reflect(value) {  
    return value;  
}
```

```
console.log(item instanceof Array);  
console.log(object instanceof Object);  
console.log(reflect instanceof Function);
```

functions

assignment

Maak een functie `amount` die afhankelijk van het aantal parameters die je meegeeft de som teruggeeft

```
amount(20)           // outputs 20  
amount(1, 5, 7, 9)   // outputs 22
```

Function Declaration Expression

```
function add(num1, num2)  
{ return num1+ num2 }
```

```
var add = function(num1, num2)  
{ return num1 + num2 }
```


Function Arguments

```
function amount() {  
    var result = 0,  
        i = 0,  
        len = arguments.length;  
    while (i < len) {  
        result += arguments[i];  
        i++;  
    }  
  
    return result;  
}  
  
console.log(sum(1, 2)); // 3  
console.log(sum(3, 4, 5, 6)); // 18  
console.log(sum(50)); // 50  
console.log(sum()); // 0
```

Functions

This

```
function sayNameForAll() {  
    console.log(this.name);  
}  
  
var person1 = {  
    name: "Nicholas",  
    sayName: sayNameForAll  
};  
  
var person2 = {  
    name: "Greg",  
    sayName: sayNameForAll  
};  
  
var name = "Michael";  
person1.sayName();    // outputs "Nicholas"  
person2.sayName();    // outputs "Greg"  
sayNameForAll();      // outputs "Michael"
```

this and that

This scope call()

```
function sayNameForAll(label) {  
    console.log(label + ":" + this.name);  
}  
  
var person1 = {  
    name: "Nicholas"  
};  
  
var person2 = {  
    name: "Greg"  
};  
  
var name = "Michael";  
  
sayNameForAll.call(this, "global");           // .apply(this, ["global"]);  
sayNameForAll.call(person1, "person1");  
sayNameForAll.call(person2, "person2");
```

This scope bind()

```
function sayNameForAll(label) {  
    console.log(label + ":" + this.name);  
}  
  
var person1 = {  
    name: "Nicholas"  
};  
  
var person2 = {  
    name: "Greg"  
};  
  
// create a function just for person1  
var sayNameForPerson1 = sayNameForAll.bind(person1);  
sayNameForPerson1("person1");  
  
// create a function just for person2  
var sayNameForPerson2 = sayNameForAll.bind(person2, "person2");  
sayNameForPerson2();
```

functional programming

versimpeld

http://www.hunlock.com/blogs/Functional_Javascript

assignment

1. Maak een functie die een array als parameter mee krijgt en de inhoud van deze array alert
2. Maak de functie slimmer door de gebruiker zelf de actie (in dit geval alert) te laten meegeven.

Functional programming

```
function alertArray(array) {  
    for (var i = 0; i < array.length; i++)  
        alert(array[i]);  
}
```


Functional programming

```
function forEach(array, action) {  
    for (var i = 0; i < array.length; i++) {  
        action(array[i]);  
    }  
}
```

```
forEach(["Wampeter", "Foma", "Granfalloon"], alert);
```

assignment

Maak in 1 uur een (bijvoorbeeld todo) app met je beste JavaScript. De app moet aan de volgende eisen voldoen.

1. De app bevat een lijst aan gegevens,
2. Er is sprake van event handling.
3. De app heeft een formulier die moet worden verwerkt.
4. Er moet data worden onthouden in tijdelijk geheugen,
5. Er moet data naar de DOM worden geschreven.
6. Heeft een volledige GUI (caliber todolist).
7. Publiceer de opdracht op github.

assignment

Deel in groepjes van 2 studenten de uitwerking. Schrijf op wat je interessant vindt / lijkt en formuleer individueel een leerdoel.

Kijken

URL(<https://tutsplus.com/course/object-oriented-javascript/>)

Bekijk de volgende video's

- Primitives and objects
- Creating objects and factory functions
- The 'this' keyword
- Data and Accessor properties
- Excercise 01: building a simple toolbar

Lezen

- Principles of Object Oriented JavaScript: h1 en h2
- <http://eloquentjavascript.net/chapter6.html>
- http://www.hunlock.com/blogs/Functional_Javascript
- <http://www.tomdalling.com/blog/software-design/fizzbuzz-in-too-much-detail/>

Maken

Gegevens is het volgende object '**athlete**' met de volgende attributen: *dayOfBirth*, *name*, *surname*, *stamina*, *strength* en *length*.

Schrijf tenminste 2 manieren om het object te declareren.

Schrijf een factory function om ten minste 4 instanties van het object te creëren.