IN CHAPTER 1 : WE LEARNED HOW TO CREATE TYPES

```
Type Point = {x:number; y:number}
```

Create a type

Use the type anywhere

```
function distance(Point p1, Point p2) : number {
  return Math.sqrt( (p2.x – p1.x)**2 + (p2.y – p1.y)**2)
}
```

1 Typed languages

5 Abstraction

OOP

2 Object/Class

4 Polymorphism

3 Encapsulation / Aggregation

SAME SAME

DATA	ACTIONS
ATTRIBUTES	FUNCTIONS
PROPERTIES	METHODS

ACTIVITY 1

ACTIVITY 2

We have coded a **BankAccount** using a **type**

>> Complete missing code to add (**credit**) or remove (**debit**) money from an account


```
type BankAccount = {
  balance: number;
  name: string;
};

function debit(account: BankAccount, valueDollars: number) {
  // Complete this code to add valueDollars to the bank account
}

function credit(account: BankAccount, valueDollars: number) {
  // Complete this code to remove valueDollars from the bank account
}
```

DEMO

Let’s move our BankAccount from Type to Class


```
class BankAccount {
  balance: number;
  name: string;

  constructor(name: string) {
    this.name = name;
    this.balance = 0;
  }

  debit(valueDollars: number) {
    this.balance -= valueDollars;
  }

  creadit(valueDollars: number) {
    this.balance += valueDollars;
  }
}
```

A class defines the **attributes** of a specific object
And the **actions** to perform on this object

```
color
fuel
speed

drive()
addFuel()
stop()
```

Attributes of the car

Methods of the car

The main keywords of a Class !!

CLASS keyword to define a class

```
class BankAccount {
  balance: number;
  name: string;

  constructor(name: string) {
    this.name = name;
    this.balance = 0;
  }

  debit(valueDollars: number) {
    this.balance -= valueDollars;
  }
}
```

CONSTRUCTOR

The main keywords of a Class !!

CONSTRUCTOR keyword
As the entry function when
Objects of this class are created

It's like a function, but called when Objects from this model are Created !!

```
class BankAccount {
  balance: number;
  name: string;

  constructor(name: string) {
    this.name = name;
    this.balance = 0;
  }

  debit(valueDollars: number) {
    this.balance -= valueDollars;
  }
}
```

The main keywords of a Class !!

```
class BankAccount {
  balance: number;
  name: string;

  constructor(name: string) {
    this.name = name;
    this.balance = 0;
  }

  debit(valueDollars: number) {
    this.balance -= valueDollars;
  }
}
```

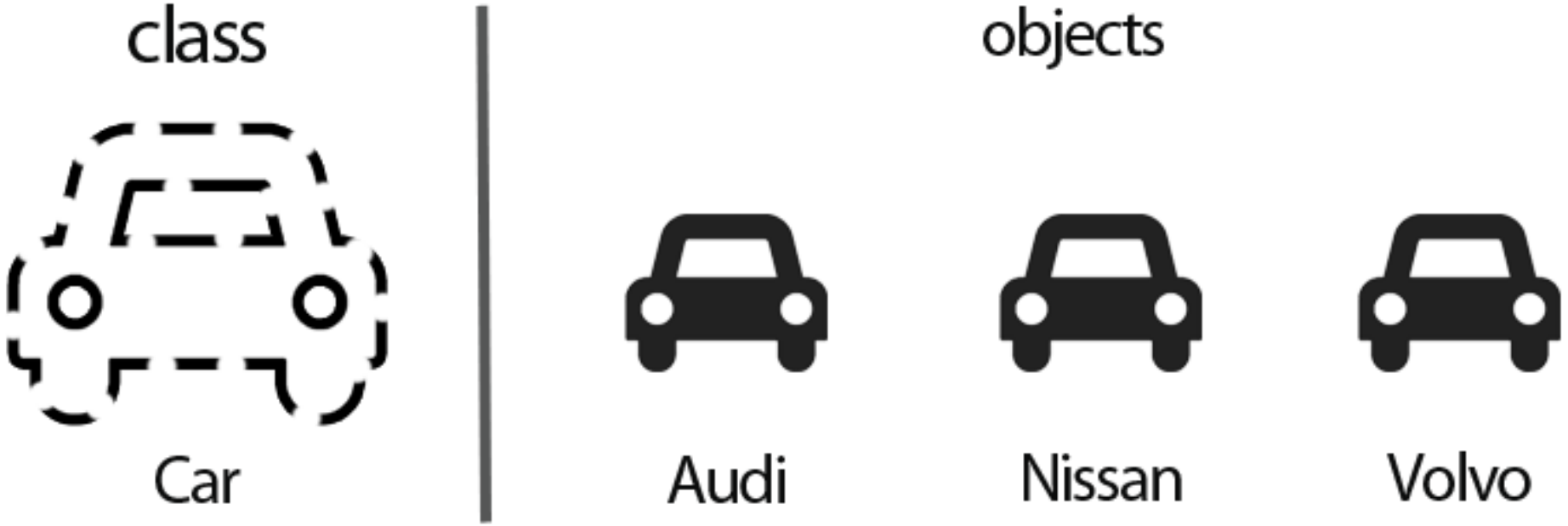
Attributes

THIS keyword is used to access
To the attribute of the object

“The best functions are those without parameters”

UNCLE BOB

Form 1 model (the class) you can create many variation (objects)



How to create an object from a class ?

```
Let myPoint = new Point(40, 30);
```

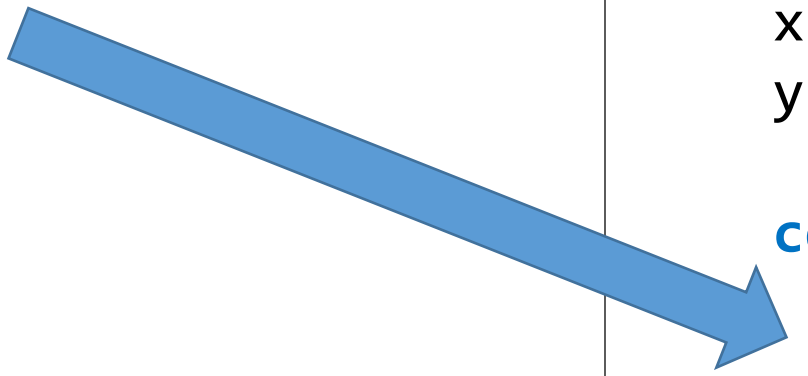
1 Call the constructor of Point
By using the keyword NEW

```
class Point {
  x: number;
  y: number;

  constructor(x: number, y: number)
    this.x = x;
    this.y = y;
  }
}
```

How to create an object from a class ?

```
Let myPoint = new Point(40, 30);
```



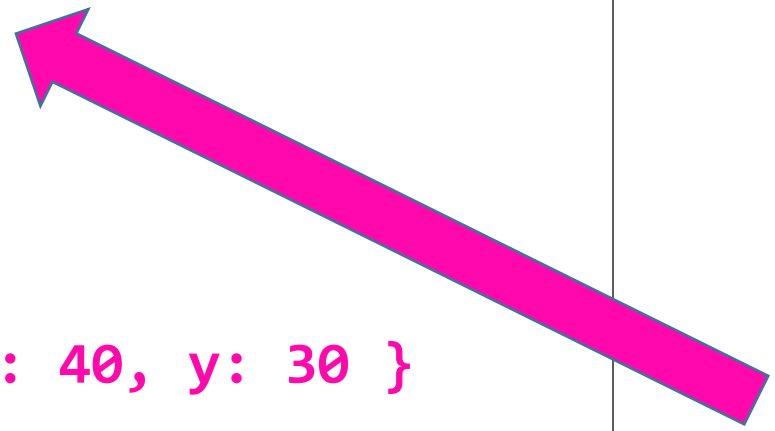
```
class Point {
  x: number;
  y: number;

  constructor(x: number, y: number)
    this.x = x;
    this.y = y;
  }
}
```

2 The constructor will return
A new object Point
With the X and Y values assigned
With the parameters (40 and 30

How to create an object from a class ?

```
Let myPoint = new Point(40, 30);
```



```
{x: 40, y: 30 }
```

3 The object is returned

```
class Point {
  x: number;
  y: number;

  constructor(x: number, y: number)
    this.x = x;
    this.y = y;
  }
}
```

new is the keyword to create objects from Class definitions

```
Let myCar = new Car("audi");
```

DEMO

DEBUGGER

STEP
BY STEP !!

GAME !!!

ACTIVITY 3

the “functional” word

```
type Point = {
  x: number;
  y: number;
};



function isEqualTo(p1: Point, p2: Point): boolean {
  return p1.x == p2.x && p1.y == p2.y;
}



let point1: Point = { x: 40, y: 40 };
let point2: Point = { x: 40, y: 40 };

console.log(isEqualTo(point1, point2));
```

the “Object” word

```
class Point {
  x: number;
  y: number;

  constructor(x: number, y: number) {
    this.x = x;
    this.y = y;
  }

  isEqualTo(other: Point): boolean {
    return this.x == other.x && this.y == other.y;
  }
}



let point1 = new Point(40, 30);
let point2 = new Point(40, 30);

console.log(point1.isEqualTo(point2));
```

HOMEWORK : ASNWER QUESTIONS

- Q1 : what is the difference between a **class** and an **object** ?
- Q2 : what is the difference between a **class** and a **type** ?
- Q3 : when do we use the keyword '**this**' ?
- Q4 : when do we use the keyword '**new**' ?
- Q5 : how many **objects** can we create from a **class** ?

WANT TO GO FURTHER ?

CLASSES & OBJECTS IN TYPESCRIPT

https://www.typescriptlang.org/docs/handbook/classes.html