

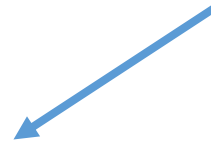
IN CHAPTER 1 : WE LEARNED HOW TO CREATE TYPES

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

2 Object/Class

OOP

4 Polymorphism

3 Encapsulation / Aggregation

SAME SAME



DATA

ATTRIBUTES

PROPERTIES

ACTIONS

FUNCTIONS

METHODS



15 MIN



TEAM 3

ACTIVITY

PROPERTIES **ACTIONS**



10 MIN



INDIV

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  
}
```

Let's move our BankAccount from **Type** to **Class**

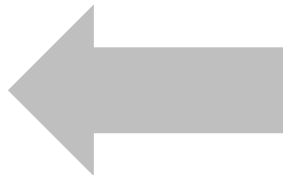


```
class BankAccount {  
    balance: number;  
    name: string;  
  
    constructor(name: string) {  
        this.name = name;  
        this.balance = 0;  
    }  
  
    debit(valueDollars: number) {  
        this.balance -= valueDollars;  
    }  
  
    credit(valueDollars: number) {  
        this.balance += valueDollars;  
    }  
}
```

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

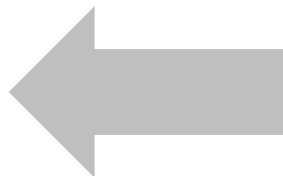


color
fuel
speed



Attributes of the car

drive()
addFuel()
stop()



Methods of the car

CLASS

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;  
    }  
}
```

THIS

The main keywords of a Class !!

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

Attributes

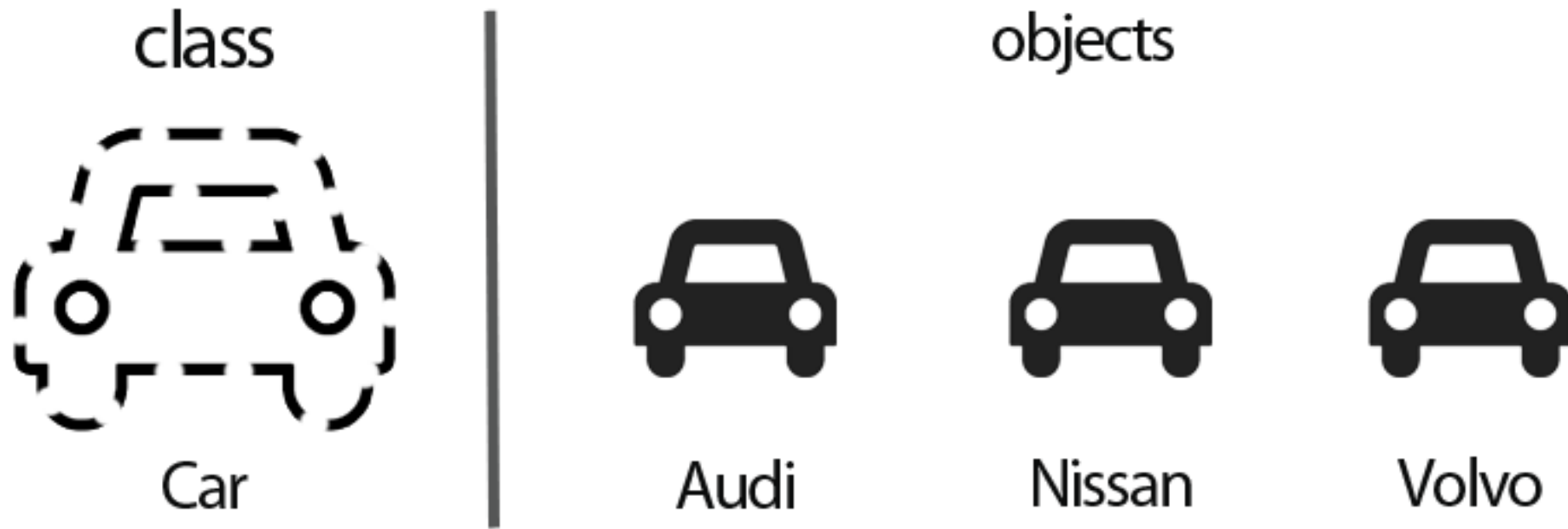
THIS keyword is used to
access
To the attribute of the
object

```
    constructor(name: string) {  
        this.name = name;  
        this.balance = 0;  
    }  
  
    debit(valueDollars: number) {  
        this.balance -= valueDollars;  
    }  
}
```

“The best functions
are those without
parameters”

UNCLE BOB

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



A close-up photograph of a silver metal cookie cutter with a serrated edge. The word "Class" is printed in a bold, black, sans-serif font in the center of the cutter's opening. Surrounding the cutter are several round, golden-brown cookies, some of which also have the word "Objects" printed on them. The background is a plain, light-colored surface.

Class

Objects

How to create an object from a class ?

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

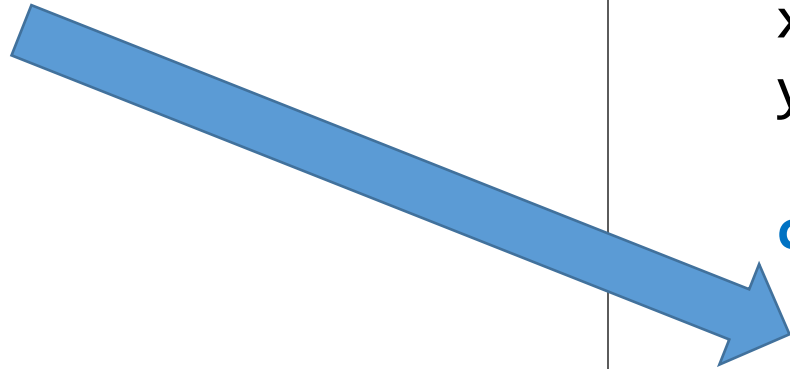


- 1 Call the constructor of Point
By using the keyword NEW

```
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 }

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

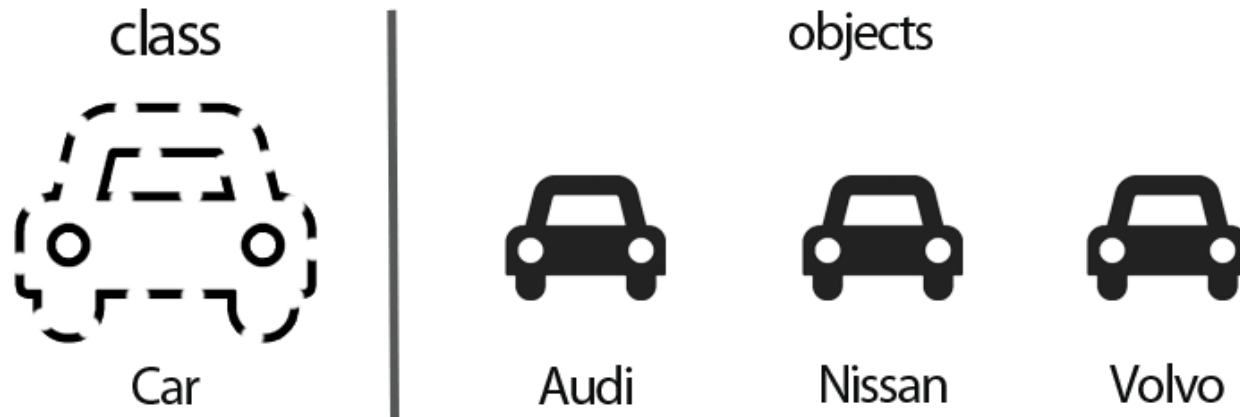
```
constructor(x: number, y: number){  
  this.x = x;  
  this.y = y;  
}
```

3

The object is returned

new is the keyword to create objects from
Class definitions

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





30 MIN



TEAM 2

ACTIVITY

HOUSE

GAME





15 MIN



TEAM 4

ACTIVITY

Step 1 : Each member draw their **dream house** o paper





15 MIN



TEAM 4

ACTIVITY

Step 2 :

- Look at the 4 plans of dream houses :
 - What do they have in **common** ?
 - What are their **differences** ?
- Find the **model** (the CLASSES) that can **implement** the 4 dream houses

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 : ANSWER 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>