

S1 Z4



Arrays and Objects

Iffe (question 😊)

```
let dataObject = {
  id:1,
  data: 'example data'
}

var proxy = (function(foo){
  return {
    getData: function(){
      return foo;
    },
    setData: function(val) {
      foo.data = val;
    }
  }
})(dataObject);

console.log(proxy.getData());
proxy.setData('changed data');
console.log(proxy.getData());

console.log(dataObject);
```

Iffe (question 😊)

```
let dataObject = {
  id:1,
  data: 'example data'
}

var proxy = (function(foo){
  return {
    getData: function(){
      return foo;
    },
    setData: function(val) {
      foo.data = val;
    }
  }
})(Object.assign({},dataObject));

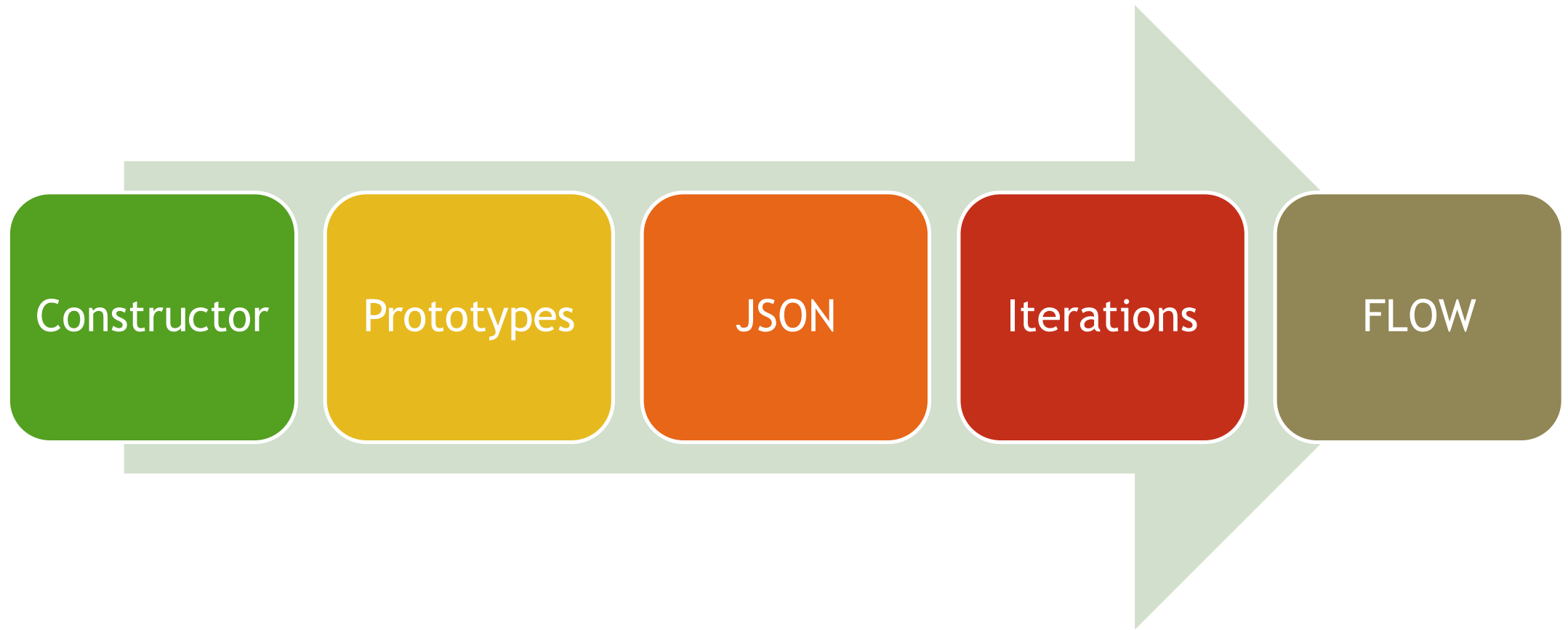
console.log(proxy.getData());
proxy.setData('changed data');
console.log(proxy.getData());

console.log(dataObject);
```

assign

```
let obj1 = { x: 1, y: 2, z: 3 };  
let obj2 = { z: 4 };  
let result;  
result = Object.assign({}, obj1, obj2);  
result = Object.assign({}, obj2, obj1);  
result = Object.assign({ x: 0 }, obj2, obj1);  
console.log(result);
```

Arrays and Objects





Person



NEW



Constructor

```
function Person() {}
```

```
let karol = new Person();
```

```
console.log(karol);
```

```
console.log(typeof karol);
```

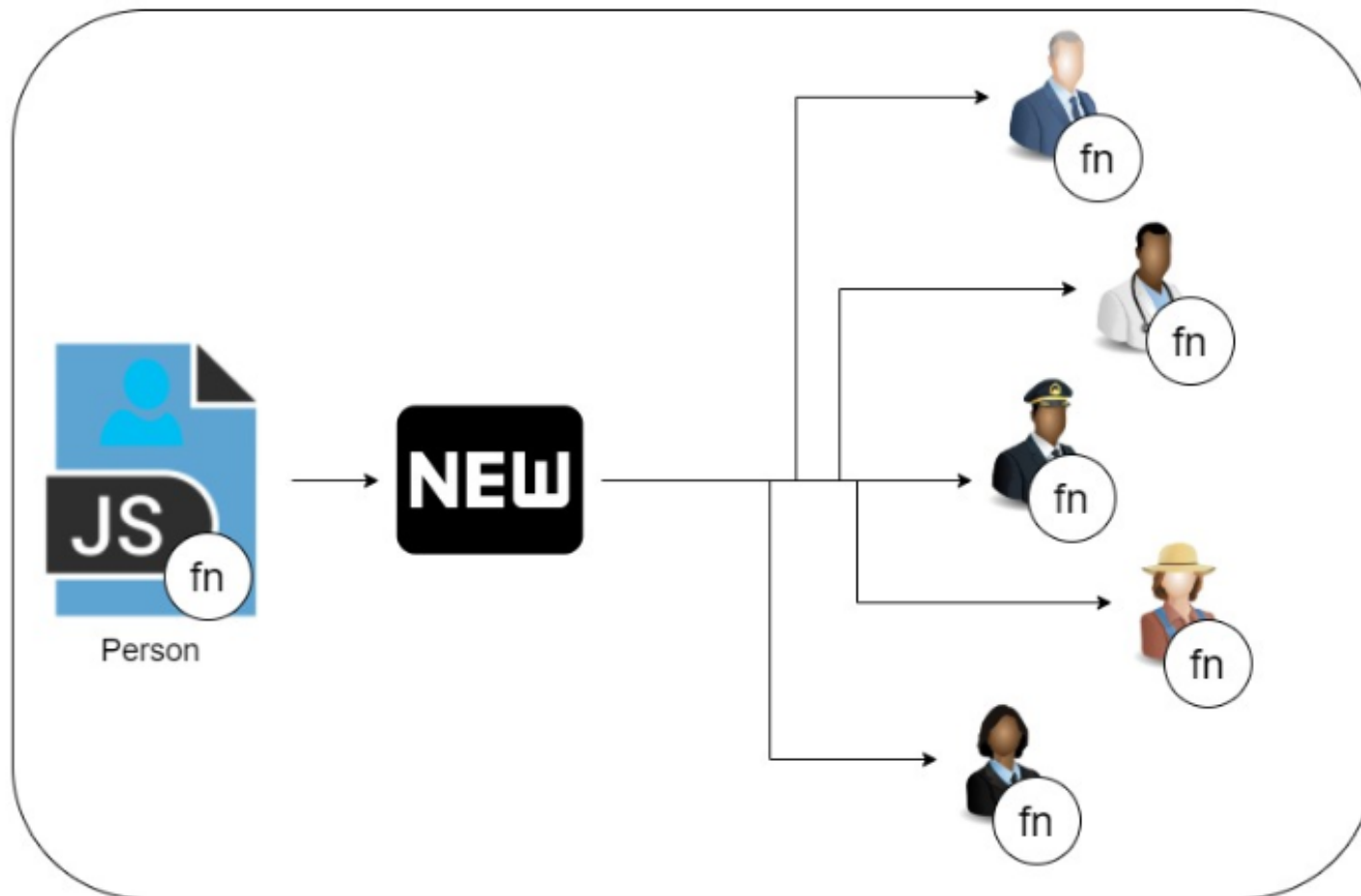

Constructor

```
function Person(){  
    console.log(this);  
}
```

```
let karol = new Person();  
let adam = Person();
```

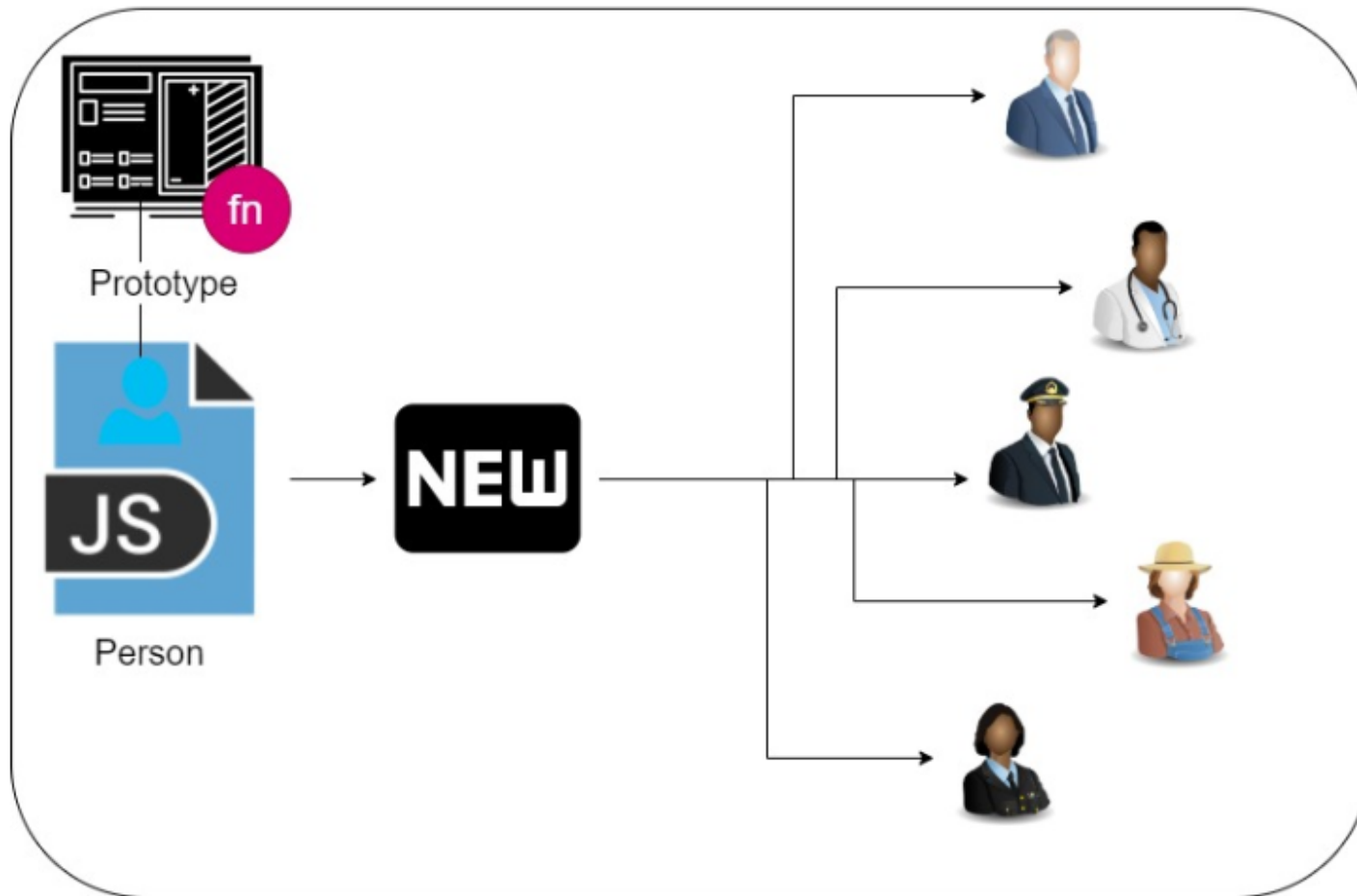
Constructors

```
function Person(firstName, lastName){  
    this.firstName = firstName;  
    this.lastName = lastName;  
}  
  
let karol = new Person('Karol', 'Rogowski');  
  
console.log(karol);
```



Constructor

```
function Person(firstName, lastName){  
  this.firstName = firstName;  
  this.lastName = lastName;  
  this.sayHello = () => console.log( 'Hello from ' + this.firstName +  
                                     ' ' + this.lastName);  
}  
  
let karol = new Person('Karol', 'Rogowski');  
  
console.log(karol);  
karol.sayHello();
```



Prototype

```
function Person(firstName, lastName){  
  this.firstName = firstName;  
  this.lastName = lastName;  
}
```

```
Person.prototype.sayHello = function() {  
  console.log( 'Hello from ' + this.firstName + ' ' + this.lastName);  
}
```

```
let karol = new Person('Karol', 'Rogowski');
```

```
console.log(karol);  
karol.sayHello();
```

Prototype

```
function Person(firstName, lastName){  
    this.firstName = firstName;  
    this.lastName = lastName;  
}
```

```
let karol = new Person('Karol', 'Rogowski');
```

```
Person.prototype.sayHello = function() {  
    console.log( 'Hello from ' + this.firstName + ' ' + this.lastName);  
}
```

```
console.log(karol);  
karol.sayHello();
```

Prototype

```
function Person(firstName, lastName){  
    this.firstName = firstName;  
    this.lastName = lastName;  
}
```

```
let karol = new Person('Karol', 'Rogowski');
```

```
console.log(karol);  
karol.sayHello();
```

```
Person.prototype.sayHello = function() {  
    console.log( 'Hello from ' + this.firstName + ' ' + this.lastName);  
}
```


Prototype

```
function Person(firstName, lastName){  
  this.firstName = firstName;  
  this.lastName = lastName;  
}
```

```
let karol = new Person('Karol', 'Rogowski');
```

```
Person.prototype.sayHello = () => {  
  console.log( 'Hello from ' + this.firstName + ' ' + this.lastName);  
  console.log(this);  
}
```

```
console.log(karol);  
karol.sayHello();
```

Prototype

```
String.prototype.showMe = function(){  
    console.log('Hello world from '+this);  
}
```

```
'Karol Rogowski'.showMe();
```

Prototype

```
Number.prototype.getValueDescription = function(){  
    return "My value is: " + this  
}
```

```
console.log((4).getValueDescription());
```

Prototype

```
function Demo(){  
  console.log('Demo function result');  
}
```

```
Function.prototype.customRun = function(){  
  console.log('Custom run begin');  
  this();  
  console.log('Custom run end');  
}
```

```
Demo.customRun();
```

JSON

```
let person = {  
  id: 1,  
  name: 'Karol Rogowski'  
}  
  
console.log(person);  
console.log(JSON.stringify(person));
```

JSON

```
let people = [{  
  id: 1,  
  name: 'Karol Rogowski'  
}, {  
  id: 2,  
  name: 'Jan Kowalski'  
}, {  
  id: 3,  
  name: 'Robert Lewandowski'  
}]  
  
console.log(people);  
console.log(JSON.stringify(people));
```

JSON

```
let personJSON = `{  
  "id":1,  
  "name":"Karol Rogowski"  
}`;
```

```
let person = JSON.parse(personJSON);  
console.log(person);
```

JSON

```
let peopleJSON = `[
  {
    "id":1,
    "name":"Karol Rogowski"
  },
  {
    "id":2,
    "name":"Jan Kowalski"
  },
  {
    "id":3,
    "name":"Robert Lewandowski"
  }
]`
```

```
let people = JSON.parse(peopleJSON);
console.log(people);
```


Array Iteration

```
let people = [  
  {  
    innerId: 'dfr458hj',  
    name: 'Karol Rogowski',  
    birthYear: 1985,  
    sayHello: function(){console.log(this.name+ ' says hello')}}  
  },  
  {  
    innerId: 'plo745as',  
    name: 'Jan Kowalski',  
    birthYear: 1980,  
    sayHello: function(){console.log(this.name+ ' says hello')}}  
  },  
  {  
    innerId: 'qaz390pl',  
    name: 'Robert Lewandowski',  
    birthYear: 1988,  
    sayHello: function(){console.log(this.name+ ' says hello')}}  
  }  
]
```

Array Iteration

```
people.forEach(p => console.log(p));
```

```
people.forEach((p,i)=>console.log(i+':' + p.name));
```

```
people.forEach(p => p.sayHello());
```

Array Iteration

```
console.log(people.filter(p=> p.birthYear > 1980));
```

```
console.log(people.every(p=> p.birthYear > 1980));
```

```
console.log(people.every(p=> p.birthYear >= 1980));
```

Array Iteration

```
console.log(people.map((p,i)=>i + ':' + p.name));
```

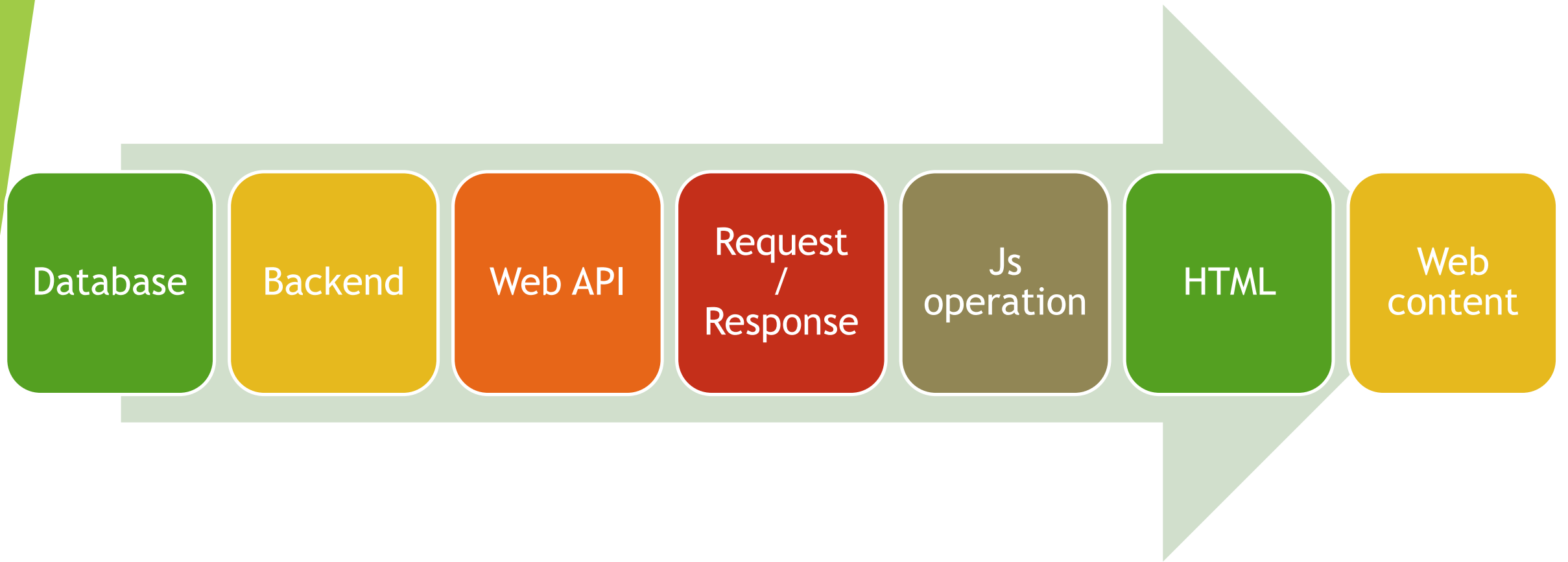
```
console.log(people.find(p=>p.birthYear !== 1985));
```

Array Iteration

```
function Person(firstName, lastName, id){  
  this.id = id;  
  this.firstName = firstName;  
  this.lastName = lastName;  
}
```

```
console.log(people.map((p,i)=> new Person(p.name.split(' ')[0],  
                                           p.name.split(' ')[1],  
                                           i)));
```

Flow - big picture



Flow “API”

```
let apiObject = {  
  getPeople: ()=>` [  
    {  
      "id":1231,  
      "name":"Karol Rogowski"  
    },  
    {  
      "id":2123,  
      "name":"Jan Kowalski"  
    },  
    {  
      "id":3111,  
      "name":"Robert Lewandowski"  
    }  
  ]`  
}
```

FLOW “mappings”

[illegible]