

# Prototype

Grupo atom: Antonio Carlos, Adson Dias, Breno Dinizio, David Felipe e Felipe Barreto

The background is a dark blue gradient. It features several abstract geometric elements: a network of thin white lines connecting small blue dots at the top; a large, multi-layered white triangle on the left side; and several smaller, 3D-style triangles in various shades of blue and white scattered on the right and bottom. 

# Atenção!

## Objetivo oficial do padrão

Especificar os tipos de objetos a serem criados usando uma instância-protótipo e criar novos objetos pela cópia desse protótipo, ou seja, é possível criar clones de objetos reais usando apenas o protótipo dela.

# Considerações

**01**

JS/TS

**03**

Shallow ou deep  
copy

**02**

Criar um protótipo

**04**

Consequências



01

**JS/TS**

São linguagens baseadas em protótipos



# JS/TS



TS

```
TS prototype1.ts > Person > clone
1  export interface Prototype {
2    |   clone(): Prototype
3  }
4
5  export class Person implements Prototype{
6    |   constructor(public name: string, public age:number) {}
7
8    |   clone(): this {
9    |     |   const newObj = Object.create(this);
10   |     |   return newObj;
11   |   }
12 }
13
14 const person1 = new Person('Adson', 17);
15 const person2 = person1.clone();
16
17 console.log(person2);
18 console.log(person2.name);
```

# JS/TS



JS

JS Prototype.js > ...

```
1  const person1 = {  
2    name: 'Adson',  
3    age: 17,  
4  };  
5  
6  const person2 = Object.create(person1);  
7  
8  console.log(person1.name); //Adson  
9  console.log(person2.name); //Adson  
10  
11 //person1 é o prototype de person2  
12 console.log(person1 === Object.getPrototypeOf(person2));
```



## **02 Criar um protótipo**

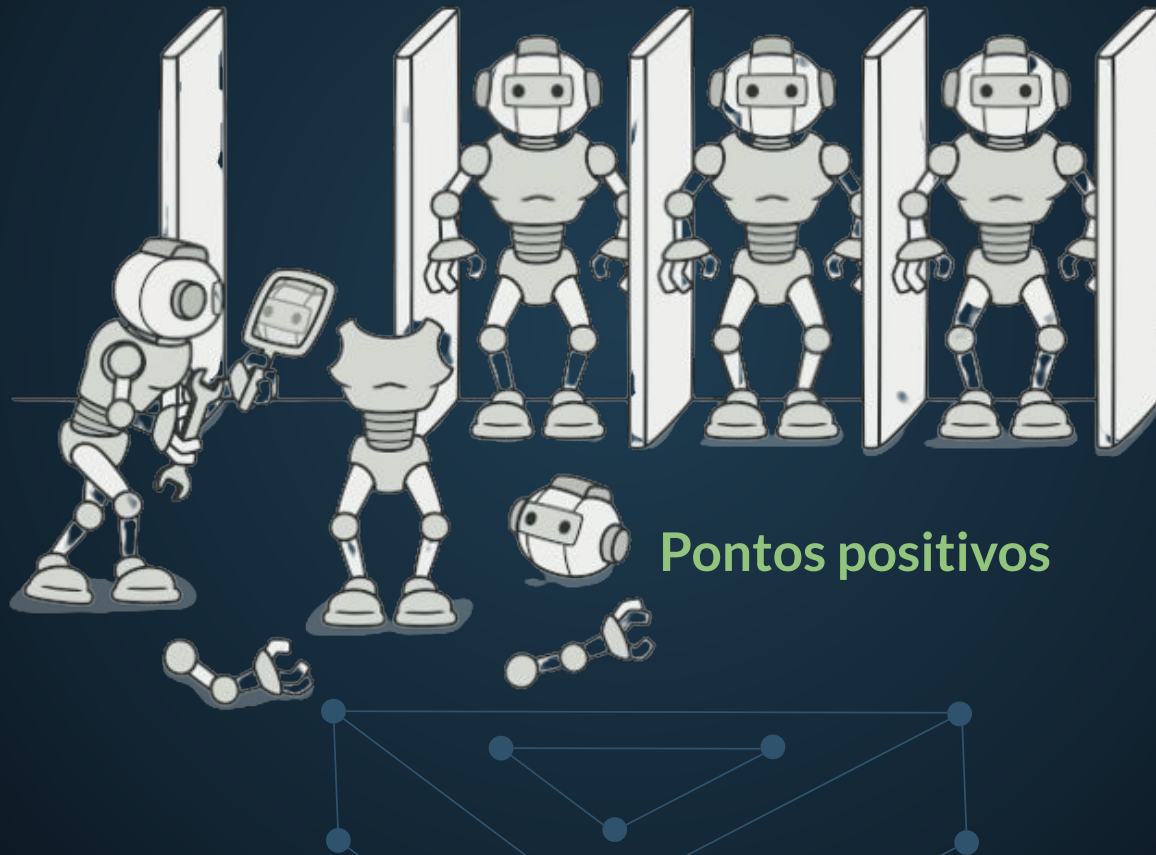
# Criar um protótipo

- O tipo objeto a ser criado é determinado pelo objeto protótipo
- É tipicamente usado para evitar a recriação de objetos "caros"
- Ajuda a evitar a explosão de subclasses
- Pode (ou não) manter um registro de objetos protótipo em um objeto separado
- Geralmente é criado apenas com um método "clone" dentro do objeto protótipo





# Criar um protótipo



Pontos positivos

# Criar um protótipo



Base concreta

A importância de criar um protótipo é justamente o fato de ter uma base para outros clones que não precisam ser criados do zero, uma vez que já possui um modelo pronto

# Criar um protótipo

Evita explosão de  
sub-classes



Esse protótipo poderá ser  
clonado e seus clones podem ser  
modificados, além disso evita  
que o cliente conheça as classes  
que criam os objetos



# Criar um protótipo

Evita a dependência de classe



Os clones são sombras do protótipo, o que não permite a visão aos atributos do protótipo, mantendo a integridade dos atributos privados e a independência com a classe

# Criar um protótipo



## Método clone

O padrão Prototype delega o processo de clonagem aos objetos reais que estão sendo clonados. O padrão declara uma interface comum para todos os objetos que suportam clonagem. Essa interface permite clonar um objeto sem acoplar seu código à classe desse objeto. Normalmente, essa interface contém apenas um único **clone** método.

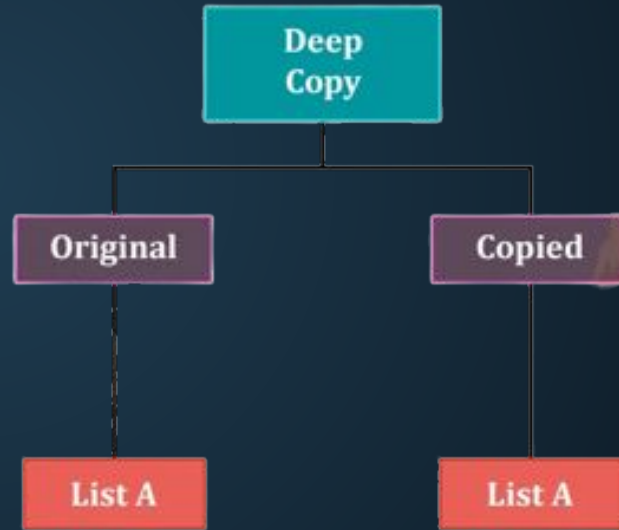
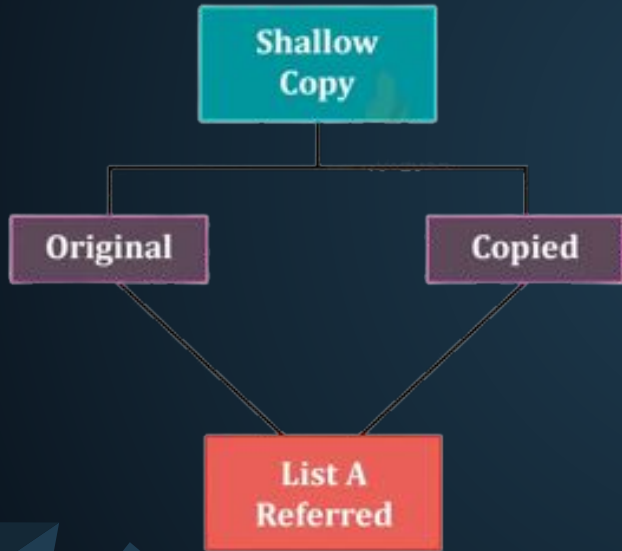
A cluster of overlapping, semi-transparent blue triangles and squares in the top-left corner.

**03**

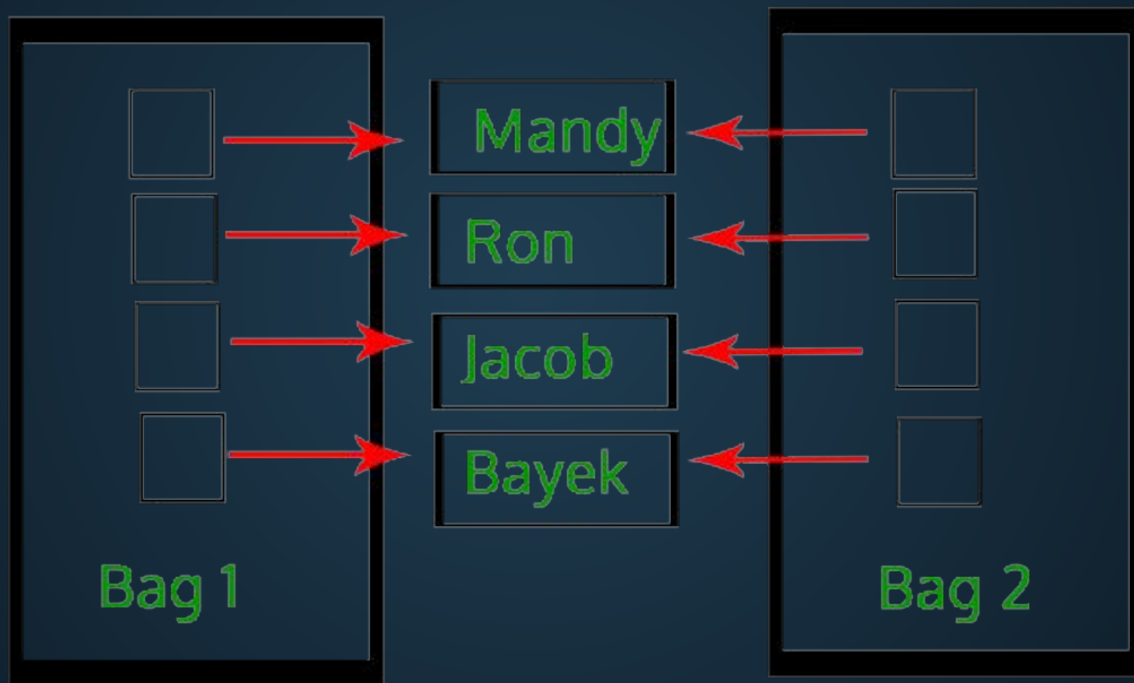
**Shallow ou deep copy**

A cluster of overlapping, semi-transparent blue triangles and squares in the bottom-right corner.

# Shallow ou deep copy

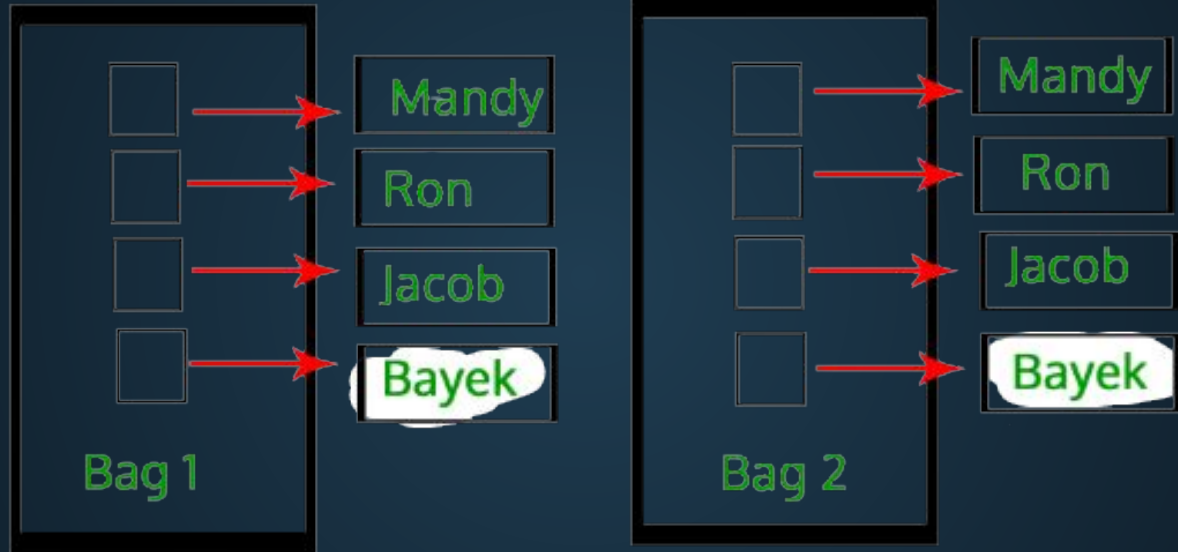


## Shallow Copy





## Deep Copy





# **04** Consequências

# CONSEQUÊNCIAS

**25%**



Ocultas classes  
concretas do  
código cliente

**25%**



Ajuda na  
criação de  
objetos caros  
ou complexos

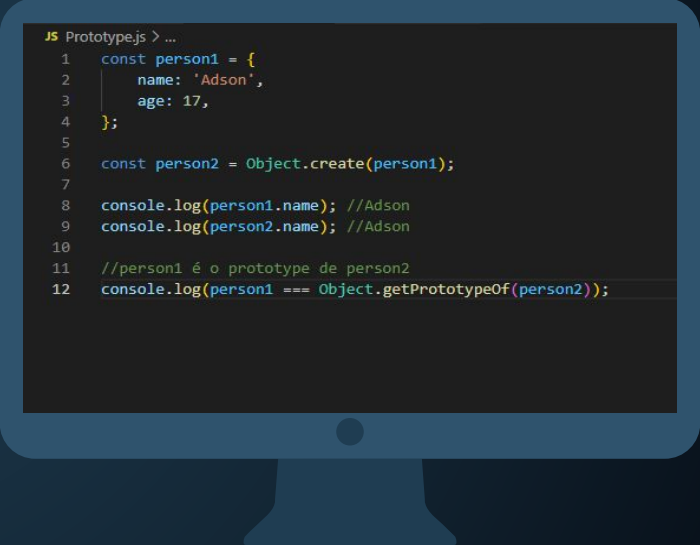
**50%**



Evita a explosão de  
subclasses

# Aplicabilidade

- °Use o padrão prototype quando precisar que seu código não dependa de classes concretas para a criação de novos objetos
- °Use o padrão prototype quando quiser evitar explosão de subclasses para objetos muito similares
- °Use o padrão prototype para evitar a recriação de objetos "caros"



```
JS Prototype.js > ...
1  const person1 = {
2    name: 'Adson',
3    age: 17,
4  };
5
6  const person2 = Object.create(person1);
7
8  console.log(person1.name); //Adson
9  console.log(person2.name); //Adson
10
11 //person1 é o prototype de person2
12 console.log(person1 === Object.getPrototypeOf(person2));
```

The background is a dark blue gradient. It features several abstract geometric elements: in the top left, a series of nested, slightly offset triangles; in the top right, a 3D pyramid-like shape; in the bottom left, a network of thin lines connecting small circular nodes; and in the bottom right, a series of concentric, slightly offset rectangular frames.

**OBRIGADO!**

**fim**