XDNATIS

Les prototypes

Michael
XDNATIS

XDNATIS

Compétence visée :

Comprendre la factorisation de code grâce à l'héritage

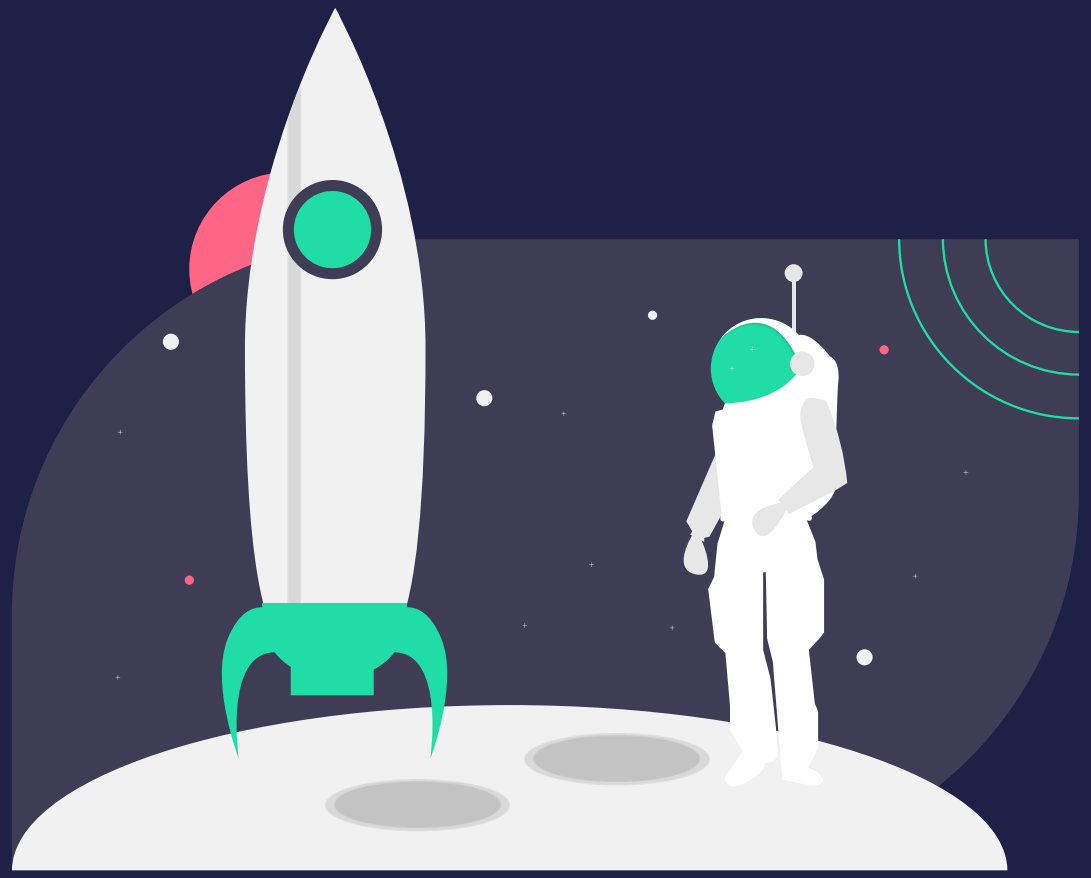
Mettre en œuvre le principe de l'héritage par les prototypes

XEBIA NATIS

1. Les prototypes
2. L'héritage par la programmation oritentée prototype (sous style de POO)
3. Dynamic dispatch

XWIKI SAS

XWIKI NATIS



XWIKI SAS

XWIKI NATIS

Les prototypes

XWIKI NATIS

QUESTION à 1 million
de faux dollars !

XDNATIS

Tout objet possède des membres (propriétés ou méthodes) que l'on ne voit pas par Object.getOwnPropertyNames.

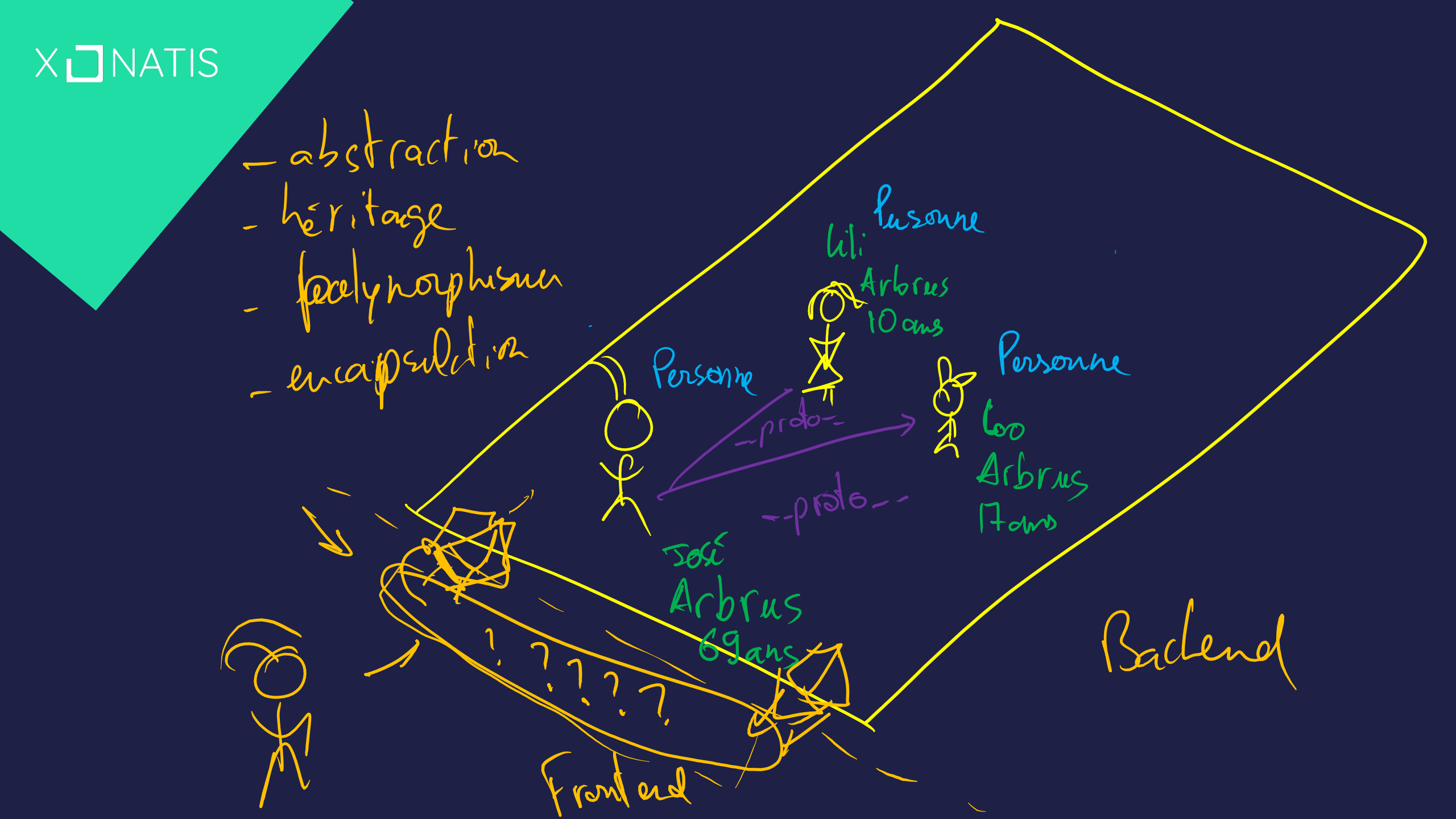
Alors où sont-elles ?

XWIKI NATIS

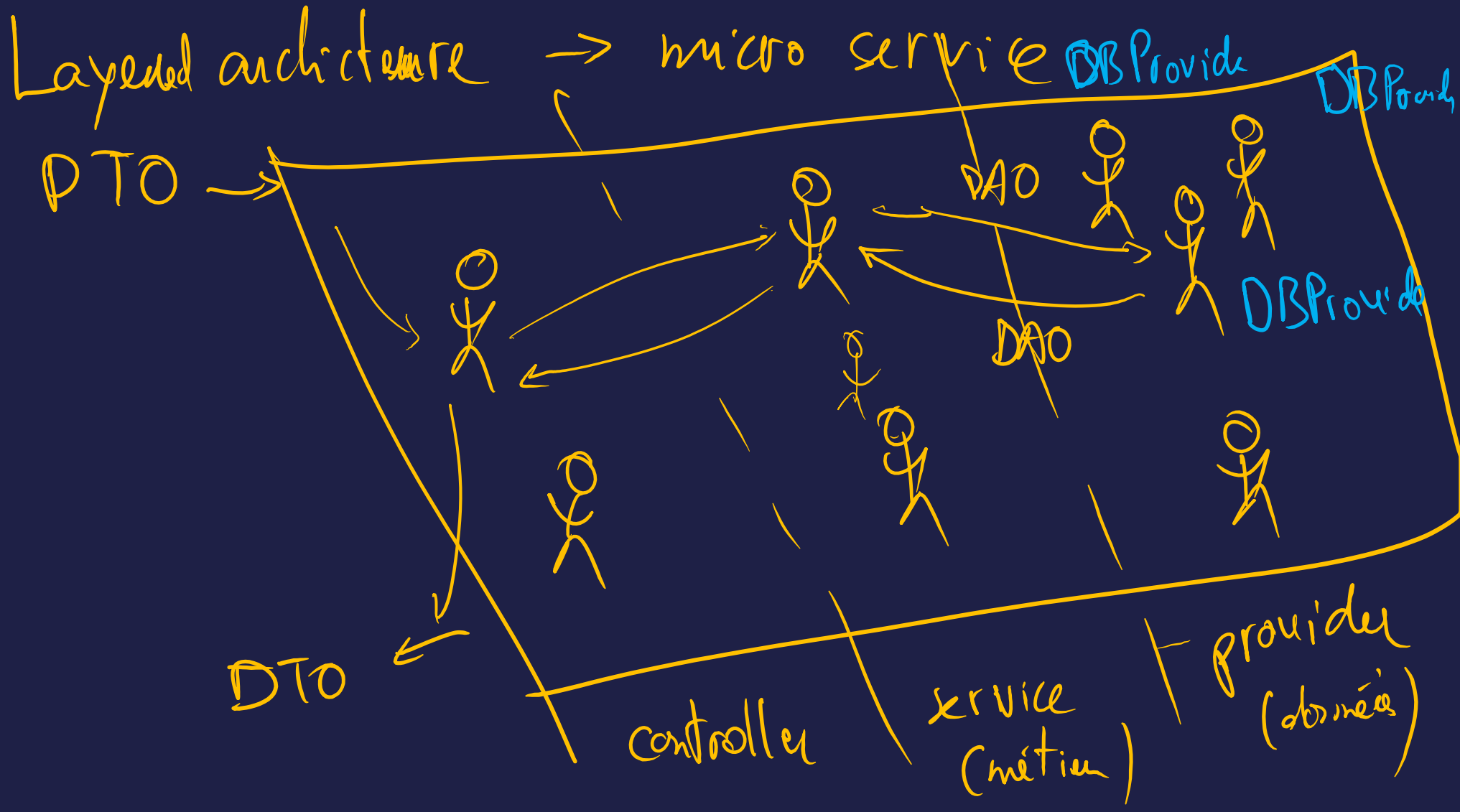
Soit … ce sont des membres :

- Internes : déclarés par des symbols et non accessibles sauf si on connait ces symbols
- Hérités : déclarés dans un autre objet connecté par la propriété __proto__

XDNATIS


- abstraction
- héritage
- polymorphisme
- encapsulation
Personne
Lili
Arbres
10 ans
Personne
--proto--
Personne
Loo
Arbres
17 ans
--proto--
José
Arbres
69 ans
Backend
Frontend

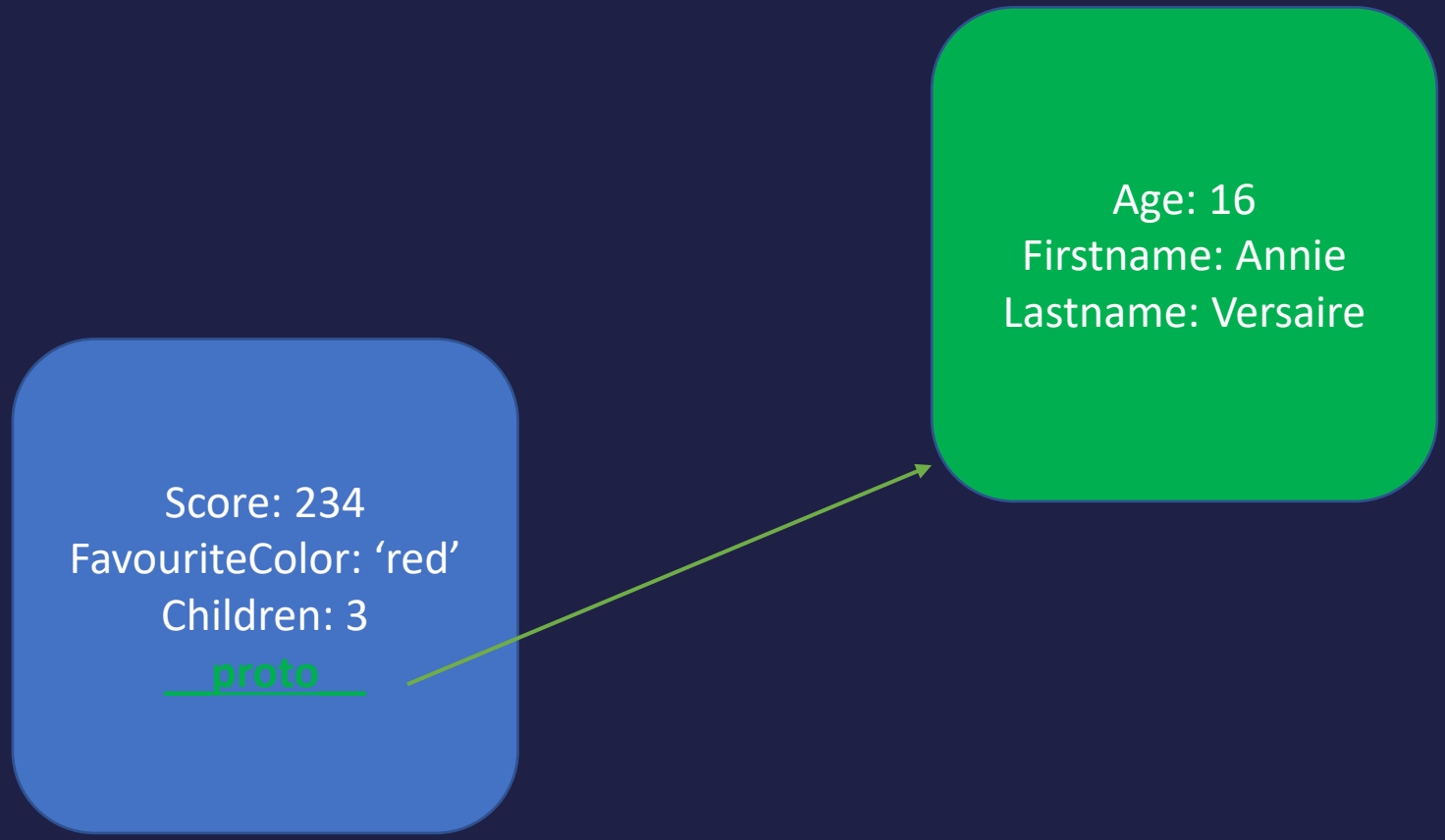
XWIKI SAS


Layered architecture → micro service
DTO
DAO
DAO
DTO
controller
service (métier)
provider (données)

XDNATIS

Pour chaque objet créé par une fonction constructeur, une association automatique se fait, entre la propriété __proto__ de l'objet et l'objet prototype de la fonction constructeur

XEBIA NATIS


Score: 234
FavouriteColor: ‘red’
Children: 3
__proto__
Age: 16
Firstname: Annie
Lastname: Versaire

XEBIA NATIS

```
function Person(age, firstname, lastname) {
    this.age = age;
    this.firstname = firstname;
    this.lastname = lastname;
}
const obj = new Person(16, 'Annie', 'Versaire');

console.log(obj.__proto__ === Person.prototype);
```

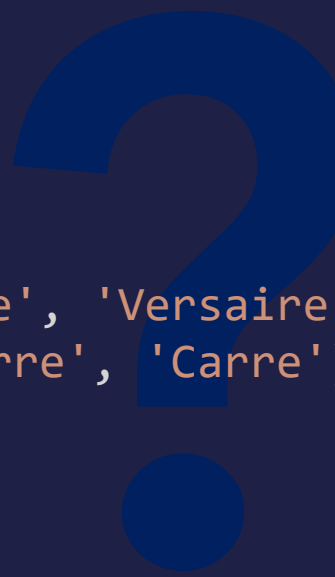
XEBIA NATIS

```
function Person(age, firstname, lastname) {
    this.age = age;
    this.firstname = firstname;
    this.lastname = lastname;
}

Person.prototype = {
    score: 234,
    favouriteColor: 'red'
};

const obj = new Person(16, 'Annie', 'Versaire');
const obj2 = new Person(21, 'Pierre', 'Carre');

console.log(obj.score);
console.log(obj2.score);
```



XWIKI SAS

XDNATIS

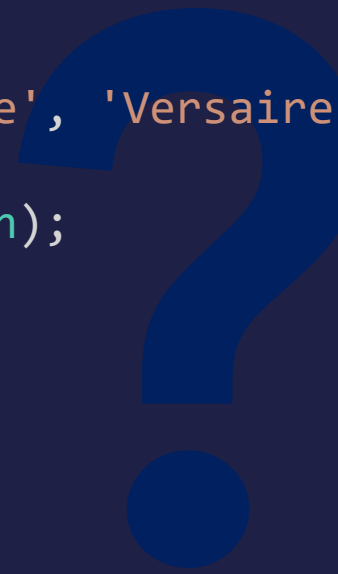
L'opérateur instanceof permet de tester si un objet possède, dans sa chaîne de prototype, la propriété prototype d'un certain constructeur

XEBIA NATIS

```
function Person(age, firstname, lastname) {
    this.age = age;
    this.firstname = firstname;
    this.lastname = lastname;
}

const obj = new Person(16, 'Annie', 'Versaire');

console.log(obj instanceof Person);
```



XEBIA NATIS

```
function Person(age, firstname, lastname) {
    this.age = age;
    this.firstname = firstname;
    this.lastname = lastname;
}

function Voiture() {

}

const obj = new Person(16, 'Annie', 'Versaire');

obj.__proto__ = Voiture.prototype;

console.log(obj instanceof Voiture);
```

XWIKI SAS

XWIKI SAS

Prototype-based programming

XEBIA NATIS

La programmation basée sur les prototypes est un style de programmation orientée objet dans lequel la réutilisation du comportement (appelée héritage) est effectuée via un processus de réutilisation d'objets existants qui servent de prototypes.

XWIKI NATIS

Ce modèle peut également être connu sous le nom de programmation prototypique, orientée prototype, sans classe ou basée sur des instances.

XWIKI NATIS

En somme :

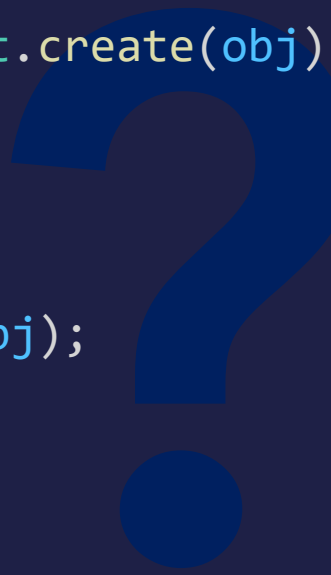
La programmation basée sur des prototypes utilise des objets généralisés, qui peuvent ensuite être clonés et étendus.

XEBIA NATIS

```
const obj = {
    age: 16,
    firstname: 'Annie',
    lastname: 'Versaire'
};
const obj2 = Object.assign(Object.create(obj), {
    score: 234,
    favouriteColor: 'red',
    children: 3
});
console.log(obj2.age);
```

XEBIA NATIS

```
const obj = {
    age: 16,
    firstname: 'Annie',
    lastname: 'Versaire'
};
const obj2 = Object.assign(Object.create(obj), {
    score: 234,
    favouriteColor: 'red',
    children: 3
});
console.log(obj2.__proto__ === obj);
```



XWIKI NATIS

Mais .. Quel est le prototype d'un objet de base ?

XEBIA NATIS

```
const obj = {
    age: 16,
    firstname: 'Annie',
    lastname: 'Versaire'
};
console.log(obj.__proto__);
```

?

XWIKI SAS

XONATIS

Dynamic dispatch

XEBIA NATIS

En informatique, le dynamic dispatch est le processus de sélection de l'implémentation d'une opération (méthode ou fonction) à appeler au moment de l'exécution.

XWIKI NATIS

L’opération a exécuter est donc ciblé au moment de l’exécution (runtime) en fonction de la chaine de prototypes.

XEBIANATIS

```
function Person(age, firstname, lastname) {
    this.age = age;
    this.firstname = firstname;
    this.lastname = lastname;
}
Person.prototype = {
    score: 432
}

function Voiture() {}
Voiture.prototype = {
    score: 234
}

const obj = new Person(16, 'Annie', 'Versaire');
obj.__proto__ = Voiture.prototype;

console.log(obj.score);
```

XWIKI NATIS

REFLEXION : Quels sont les dangers d’une telle implémentation ?

XWIKI SAS

XWIKI NATIS

Pour aller plus loin

https://developer.mozilla.org/fr/docs/Learn/JavaScript/Objects/Inheritance