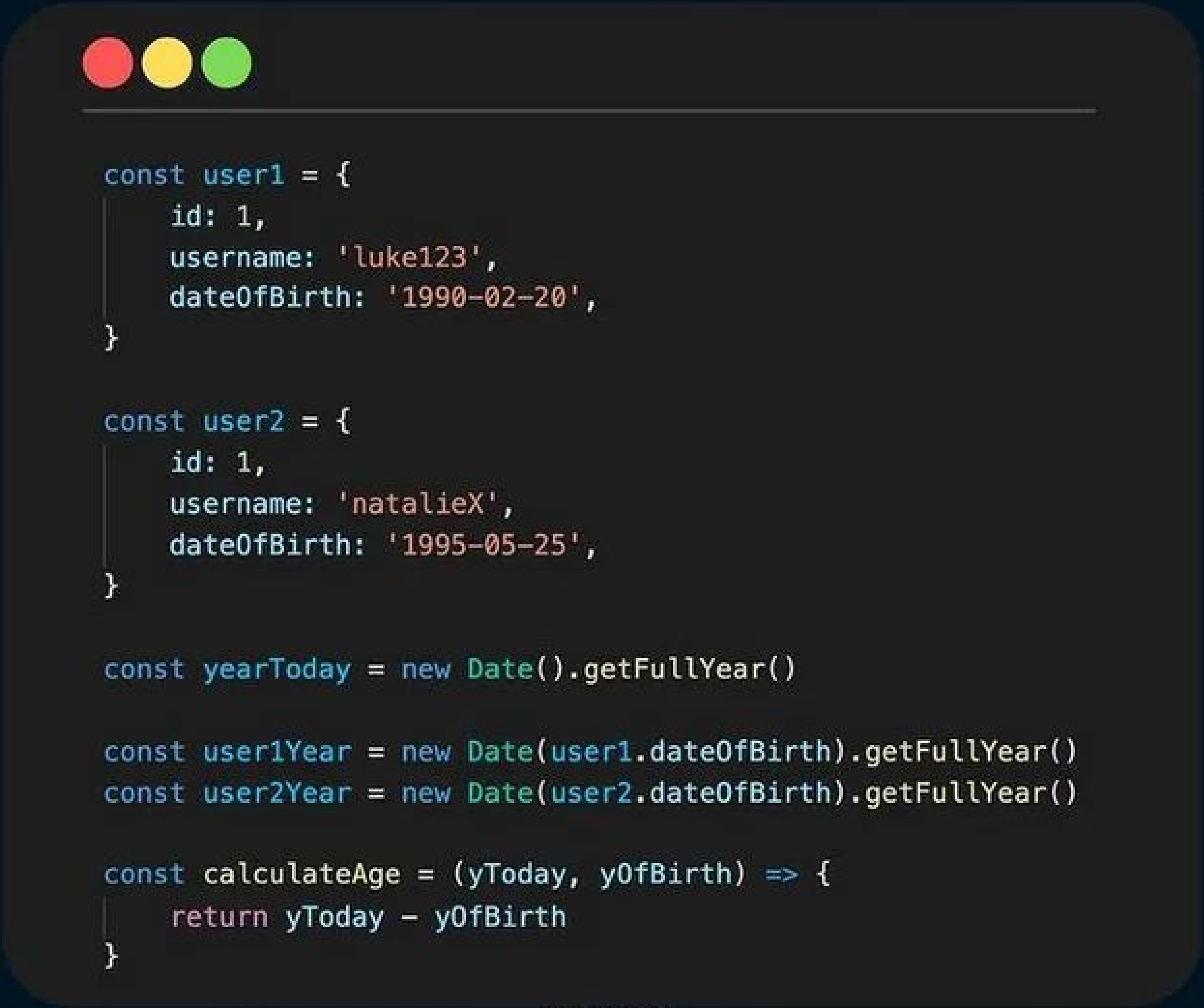
@pyplane_code

call vs apply vs bind

Introduction

In this tutorial we will learn how to use a little bit more advanced Javascript methods on a simple example - we will calculate the age of the user based on the current year and the date of birth. Starter code:



```
const user1 = {
    id: 1,
    username: 'luke123',
    dateOfBirth: '1990-02-20',
}

const user2 = {
    id: 1,
    username: 'natalieX',
    dateOfBirth: '1995-05-25',
}

const yearToday = new Date().getFullYear()

const user1Year = new Date(user1.dateOfBirth).getFullYear()
const user2Year = new Date(user2.dateOfBirth).getFullYear()

const calculateAge = (yToday, yOfBirth) => {
    return yToday - yOfBirth
}
```

@pyplane_code

call

The call() method can be used to invoke defined functions and methods to different objects

```
doSomething.call(object, arg1, arg2 ...)
```

Implementation:

```
console.log('CALL')
console.log(calculateAge.call(user1, yearToday, user1Year))
// 32
console.log(calculateAge.call(user2, yearToday, user2Year))
// 27
```

apply

The apply() method is very similar to the call() method, but instead of passing arguments one after another we pass them in an array

```
doSomething.apply(object, array)
```

Implementation:

```
console.log('APPLY')
console.log(calculateAge.apply(user1, [yearToday, user1Year]))
// 32
console.log(calculateAge.apply(user2, [yearToday, user2Year]))
// 27
```

bind

The bind method is a little bit different than the call and apply methods. It does not call the function but it returns it with a newly assigned "this" keyword. We can later execute it passing required arguments. It also enables "borrowing" methods from different objects.

```
const bound = doSomething.bind(object)
bound(arg1, arg2 ... )
```

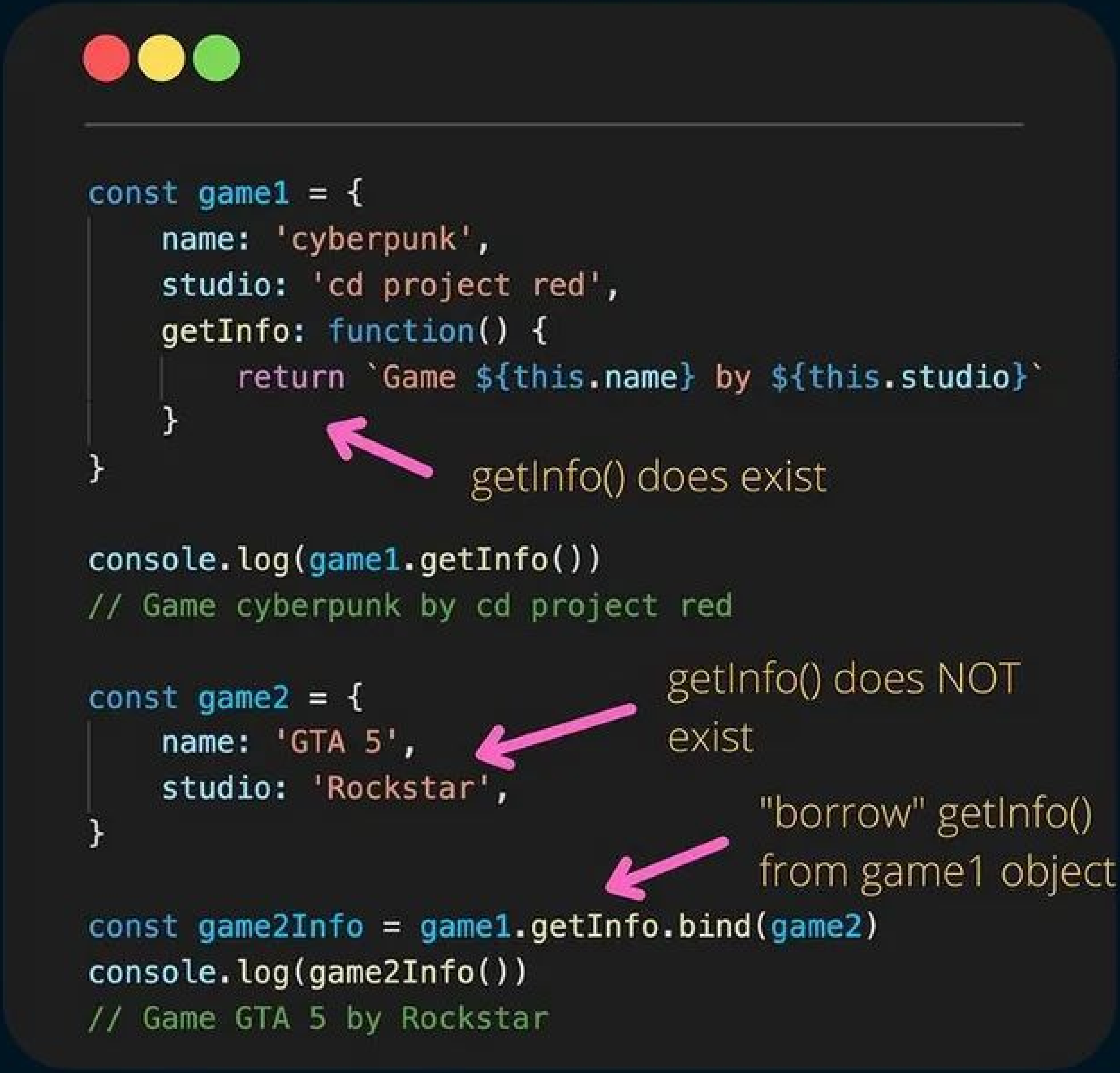
Implementation (example 1):

```
console.log('BIND')
const boundCalcAgeUser1 = calculateAge.bind(user1)
console.log(boundCalcAgeUser1(yearToday, user1Year))
// 32

const boundCalcAgeUser2 = calculateAge.bind(user2)
console.log(boundCalcAgeUser2(yearToday, user2Year))
// 27
```

bind

Implementation (example 2):



```
const game1 = {
    name: 'cyberpunk',
    studio: 'cd project red',
    getInfo: function() {
        return `Game ${this.name} by ${this.studio}`
    }
}

console.log(game1.getInfo())
// Game cyberpunk by cd project red

const game2 = {
    name: 'GTA 5',
    studio: 'Rockstar',
}

const game2Info = game1.getInfo.bind(game2)
console.log(game2Info())
// Game GTA 5 by Rockstar
```