FRONT – END AKADEMIE AKADEMIE
ENGETO S.R.O.

Objektově Orientované Programování

Lekce 14: Objekty, základy OOP, konstruktor a atributy

Verze 4. prosince 2024

Šimon Jančа

Outline

Procedurální programování

- Doteď jsme pracovali s **Procedurami**.
- Příkazy se spouštěly jeden za druhým a upravovaly nějaká data, příp. plnily nějakou akci

Procedurální programování

- Doteď jsme pracovali s **Procedurami**.
- Příkazy se spouštěly jeden za druhým a upravovaly nějaká data, příp. plnily nějakou akci
- Vyzkoušejte si kód `alert(1); alert(2); prompt("zadejA");`

Objektově Orientované – co?

- *O*bjektově *O*rientované *P*rogramování je způsob vývoje aplikací
- Object Oriented Programming vs. Objektově Orientované Programování
- Program skládáme z tzv. objektů namísto funkcí a proměnných.

Objektově Orientované – co?

- *O*bjektově *O*rientované *P*rogramování je způsob vývoje aplikací
- Object Oriented Programming vs. Objektově Orientované Programování
- Program skládáme z tzv. objektů namísto funkcí a proměnných.
- Umožňují lépe znázornit objekty reálného světa a zjednodušují implementace větších projektů
- Další vlastnosti vyzkoušíme příště :)


The journey
of a thousand
sleepless nights
starts with
a single
'Hello World'

Třída

- **Třídy** představují jakousi šablonu/model pro tvorbu objektů.

Třída

- **Třídy** představují jakousi šablonu/model pro tvorbu objektů. Obsahují:

Třída

- **Třídy** představují jakousi šablonu/model pro tvorbu objektů. Obsahují:
  - **VLASTNOSTI/PROPERTY/PROPS** data obsahující informace o aktuálním stavu daného objektu
  - **METODY** představují chování objektu, obdoba **funkcí**

Obrázek: Vykrajovátko na přípravu perníku

Objekt

- **Objekt** je již Inicializovaná třída, která má svá konkrétní data a místo v pamětí.
- **Inicializace** třídy (tvorba NOVÉHO objektu) probíhá např. klíčovým slovem *new*, při tvorbě objektu se spustí metoda CONSTRUCT, v paměti se vytvoří místo pro vlastnosti třídy

Objekt

- **Objekt** je již Inicializovaná třída, která má svá konkrétní data a místo v pamětí.
- **Inicializace** třídy (tvorba NOVÉHO objektu) probíhá např. klíčovým slovem *new*, při tvorbě objektu se spustí metoda CONSTRUCT, v paměti se vytvoří místo pro vlastnosti třídy
- **Konstruktor** je metoda spouštějící se při vzniku objektu. Pokud obsahuje **parametry**, může např. zavádět data do objektu, měnit chování objektu ap.

Obrázek: Výsledný produkt perník z těsta

Zápis třídy a iniciace objektu

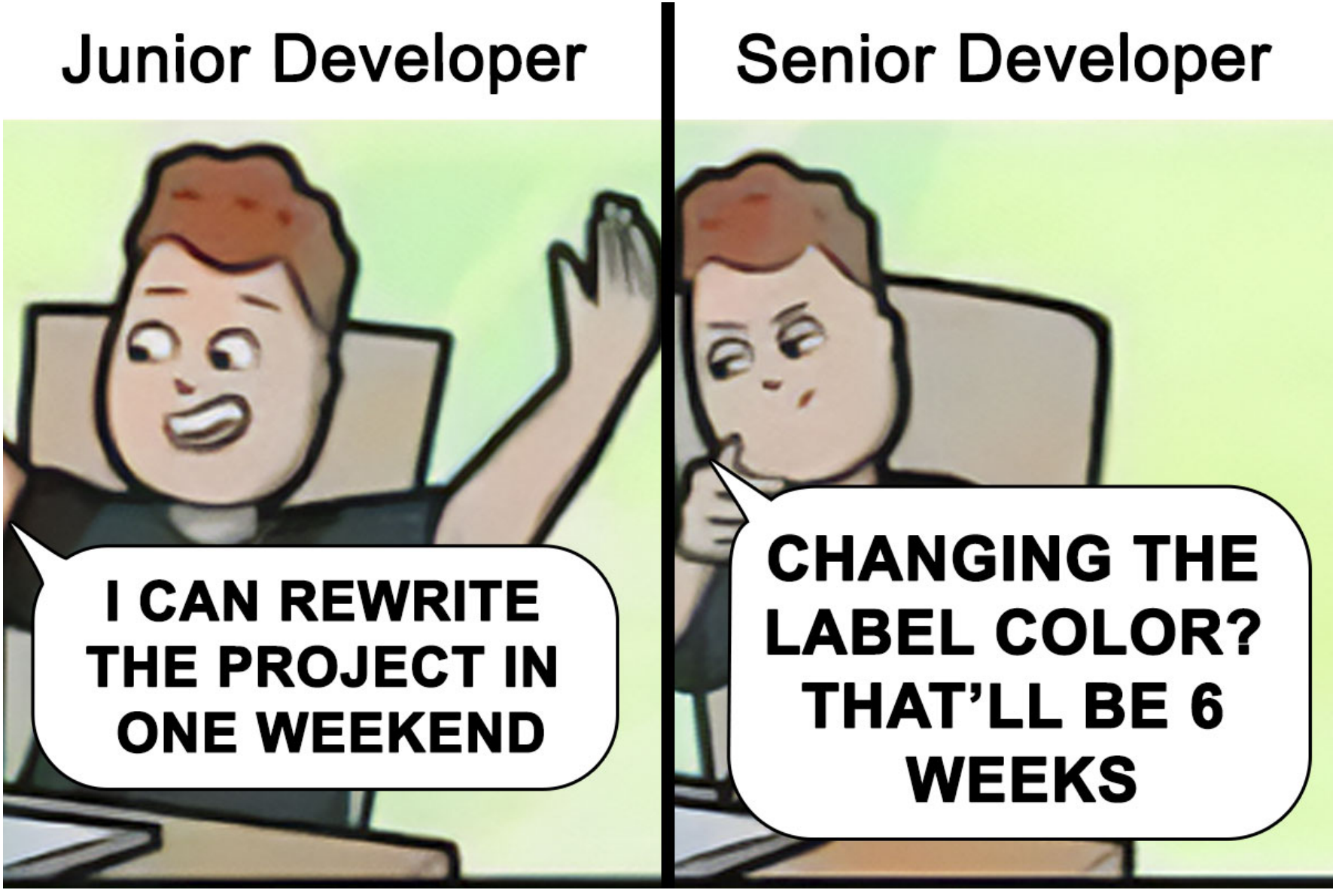
```
class Person {
    constructor(name, surname, codingYears) {
        this.name = name;
        this.surname = surname;
        this.codingYears = codingYears;
    }

    greet() {
        return `Ahoj! Jmenuji se ${this.name} ${this.surname}`
    }

    coding() {
        return `programuji už skoro ${this.codingYears} let`
    }
}

const person1 = new Person("Šimon", "Janča", 18);
// výstup: Ahoj! Jmenuji se Šimon Janča
console.log(person1.greet());
// výstup: programuji už téměř 18 let
console.log(person1.coding());
```

Constructor function

```
function createPerson(name, birthyear) {
    const ID = Math.ceil(Math.random() * 1000 % 1000) + 1;
    const age = (new Date).getFullYear() - birthyear

    return {
        ID,
        name: name,
        birthyear,
        greet: function() {
            return `Hello, my name is ${this.name}.`;
        },
        greet2() {
            return `Ahoj, jmenuji se ${this.name}.`;
        },
        // pozor na arrow FN!!!
        greet_arrow: () => {
            return `Hello, my name is ${this.name}.`;
        },
    };
}

const person1 = createPerson("Šimon", 1998);
console.log(person1.greet()); // Outputs: Hello, my name is Šimon.
console.log(person1.greet2()); // Outputs: Ahoj, jmenuji se Šimon.
```

Přímý zápis objektu

```
const Person = {
    name: "Helen",
    age: 42,
    // pozor na použití arrow function!!!
    greet: function() {
        return `Hello, my name is ${this.name}.`;
    }
};

console.log(Person.greet()); // Outputs: Hello, my name is Helen.
```


Junior Developer
Senior Developer
I CAN REWRITE THE PROJECT IN ONE WEEKEND
CHANGING THE LABEL COLOR? THAT'LL BE 6 WEEKS

Zápis pomocí třídy

```
class Car {
    constructor(make, model, year, maxSpeed) {
        this.make = make;
        this.model = model;
        this.year = year;
        this.maxSpeed = maxSpeed;
    }

    // Method to display car details
    displayInfo() {
        return `Car: ${this.make} ${this.model}, Year: ${this.year}`;
    }

    // Method to calculate the car's age
    calculateAge(currentYear) {
        return currentYear - this.year;
    }

    goMax() {
        return `Car is going by speed of ${this.maxSpeed}`;
    }
}
```

Inicializace (zavedení) objektu ze třídy

```
const car1 = new Car("Toyota", "Camry", 2015);
const car2 = new Car("Honda", "Civic", 2018);

// Outputs: Car: Toyota Camry, Year: 2015
console.log(car1.displayInfo());
// Outputs: Age of car1: 9 years
console.log(`Age of car1: ${car1.calculateAge(2024)} years`);

// Outputs: Car: Honda Civic, Year: 2018
console.log(car2.displayInfo());
// Outputs: Age of car2: 6 years
console.log(`Age of car2: ${car2.calculateAge(2024)} years`);
```

Kdo má rád kafe?

- Budeme měnit vnitřní stav objektu, čímž ovlivníme funkčnost (FSM – Stavový automat)
- Objekty spolu interagují (komunikují), navzájem se mohou ovlivňovat
- Implementujte pomocí Javascriptu třídu automatu na kafe
- Automat přijímá mince v hodnotách CZK mincí
- Umožněte volbu nápoje, pokud uživatel vloží dostatek peněz, vydejte mu kafe
- Pokud nedodá dostatek peněz do určitého limitu, vraťte případnou platbu a zruště objednávku

Kontrola mince

```
const app = document.getElementById('app');

class Coin{
    value;

    constructor(value){
        if (![1, 5, 10, 25, 50].includes(value)){
            throw new Error('Invalid coin inserted');
        }

        this.value = value;
    }
}
```

Kontrola mince

```
class CoffeeMachine{
    coffeePrice = 35;
    insertedValue = 0;

    constructor(){
        this.insertedValue = 0;
    }

    insertCoin(coin){
        this.insertedValue += coin.value;

        if (this.insertedValue < this.coffeePrice){
            return;
        }

        if (this.insertedValue > this.coffeePrice){
            const change = this.insertedValue - this.coffeePrice;
            console.log(`Returning you ${change} in change.`);
        }

        console.log("Coffee ready, yum yum. Thank you.");
    }
}
```

Vložení do automatu

```
const coinInput = document.createElement('input');
app.appendChild(coinInput);
coinInput.setAttribute('type', 'number');
coinInput.setAttribute('min', '1');
coinInput.setAttribute('max', '50');

coinInput.addEventListener('keydown', (e) => {
    if (e.key !== 'Enter'){
        return;
    }

    e.preventDefault();

    const coin = new Coin(parseInt(e.target.value));

    console.log(coin);

    if (!coin){
        alert('Invalid coin inserted');
    }
});
```