Builder

Rozšiřte aplikaci kavárny pomocí návrhového vzoru Builder, aby si zákazník mohl přidat ke svému nápoji některý z přídavků (např. mléko, cukr, apod.).

Vytvořte třídu `CustomDrink`. Třída by měla obsahovat privátní textový řetězec zastupující základní nápoj (pojmenujte jej např. `base`). Dále by třída měla obsahovat příznaky (`boolean`) pro jednotlivé dodatečné ingredience.

Uvnitř třídy `CustomDrink` vytvořte vnořenou třídu `Builder`, který bude opět obsahovat `base` a příznaky ingrediencí. Konstruktor `Builder(String base)` uloží do `base` název nápoje. Pro každou ingredienci bude vytvořena vlastní metoda (např. `milk()`), která nastaví příslušné příznaky na `true` a vrátí this (např. `this.milk = true; return this;`).

Třída `CustomDrink` bude mít privátní konstruktor `CustomDrink(Builder b)`, který převezme hodnoty `base` a příznaky z builderu `b` a hodnoty uloží do příznaků třídy.

Dále vytvořte Metodu `build()` ve třídě `Builder`, která vrátí novou instanci `CustomDrink`:

```
public CustomDrink build() {return new CustomDrink(this);}
```

Teď už je možné přepsat `toString()` v `CustomDrink` tak, aby vracela text servírování z třídy `DrinkFactory` společně se zvolenými přídavky např. následovně:

```
return DrinkFactory.createDrink(base).serve()
        + (milk ? ", milk" : "")
        + (sugar ? ", sugar" : "")
        + (caramel ? ", caramel" : "");
```

V hlavní třídě si následně vyzkoušejte funkčnost nového rozšíření:

```
CustomDrink customDrink =
new CustomDrink.Builder("coffee").milk().sugar().build();

System.out.println(customDrink
+" in "+CafeConfig.getInstance().getCafeName());
```

Pozn.: Hotová aplikace je taková, která není otestována pouze Vámi, ale především unit testy.