Object Calisthenics: Don’t use the ELSE Keyword

Mike Mugglin

Don’t use the ELSE keyword ???

Why avoiding the Else keyword ?

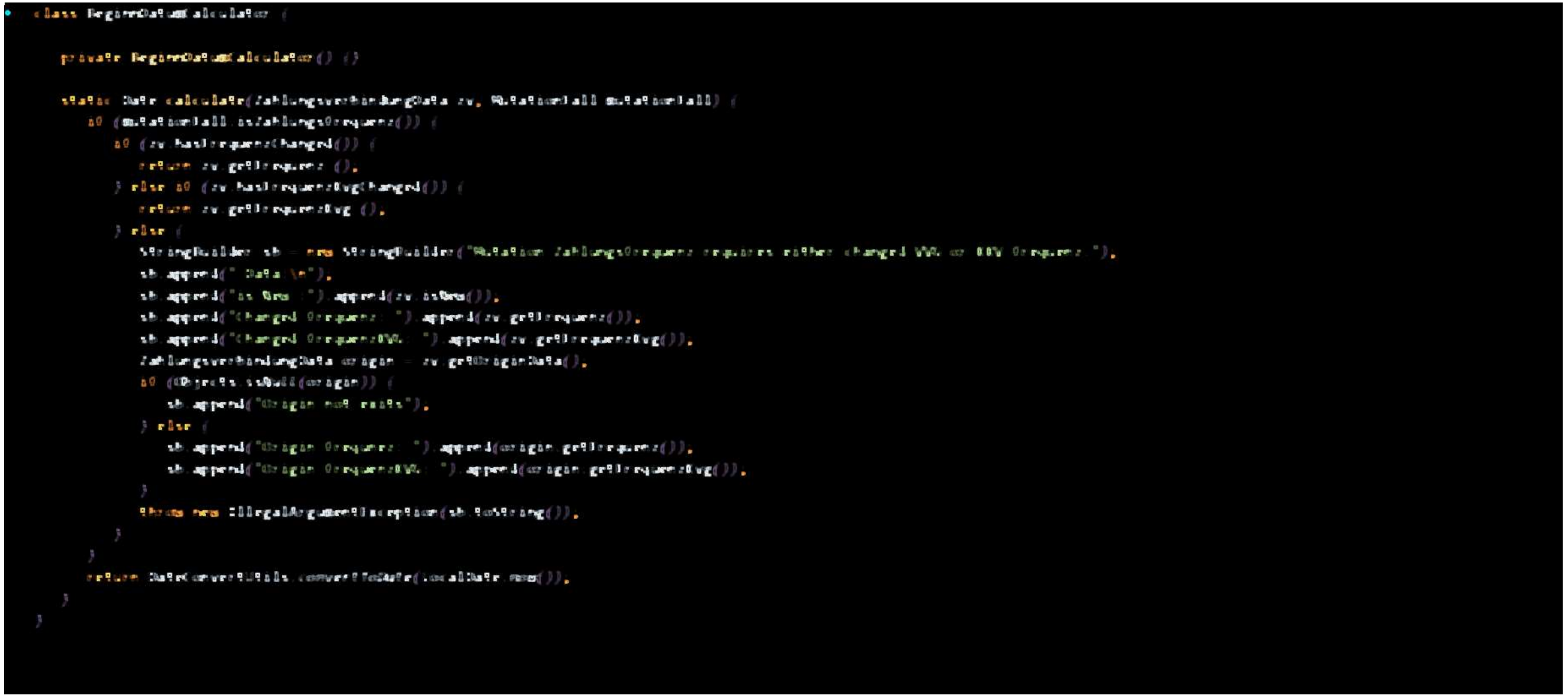
- Do you remember the last time you saw a nested conditional in the code? Did you enjoy reading it?
  No!

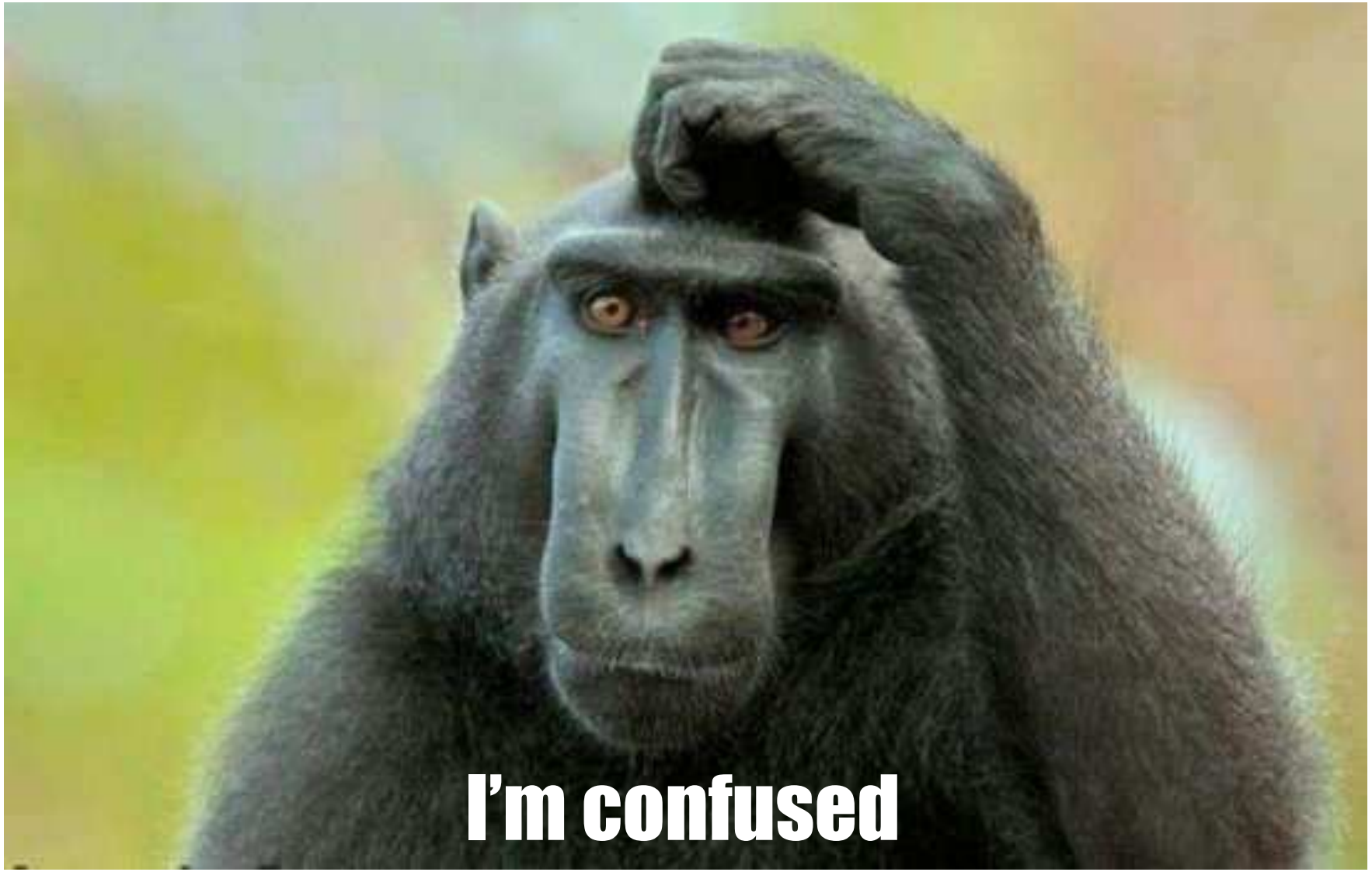
- *If/else* blocks decrease readability

- Using *if/else* can be seen as a code smell around breaking Single Responsibility:
  It normally means that you are doing more than a one behavior in your method

- The goal of this rule is to generate a cleaner and faster code because it has fewer execution flows and reduce complexity that’s why the code gets much more readable

Is this code easy to read ?




I’m confused

Early return

```
public void login(String username,String password){
    if(userRepository.isValid(username,password)){
        redirect("homepage");
    } else {
        redirect("login");
    }
}
```

```
public void login(String username,String password){
    if(userRepository.isValid(username,password)){
        return redirect("homepage");
    }
    redirect("login");
}
```

Introduce a variable

```
public void login(String username,String password){
    if(userRepository.isValid(username,password)){
        redirect("homepage");
    } else {
        redirect("login");
    }
}
```

```
public void login(String username, String password) {
    String redirectRoute = "homepage";
    if (!userRepository.isValid(username, password)) {
        redirectRoute = "login";
    }
    redirect(redirectRoute);
}
```

Polymorphism

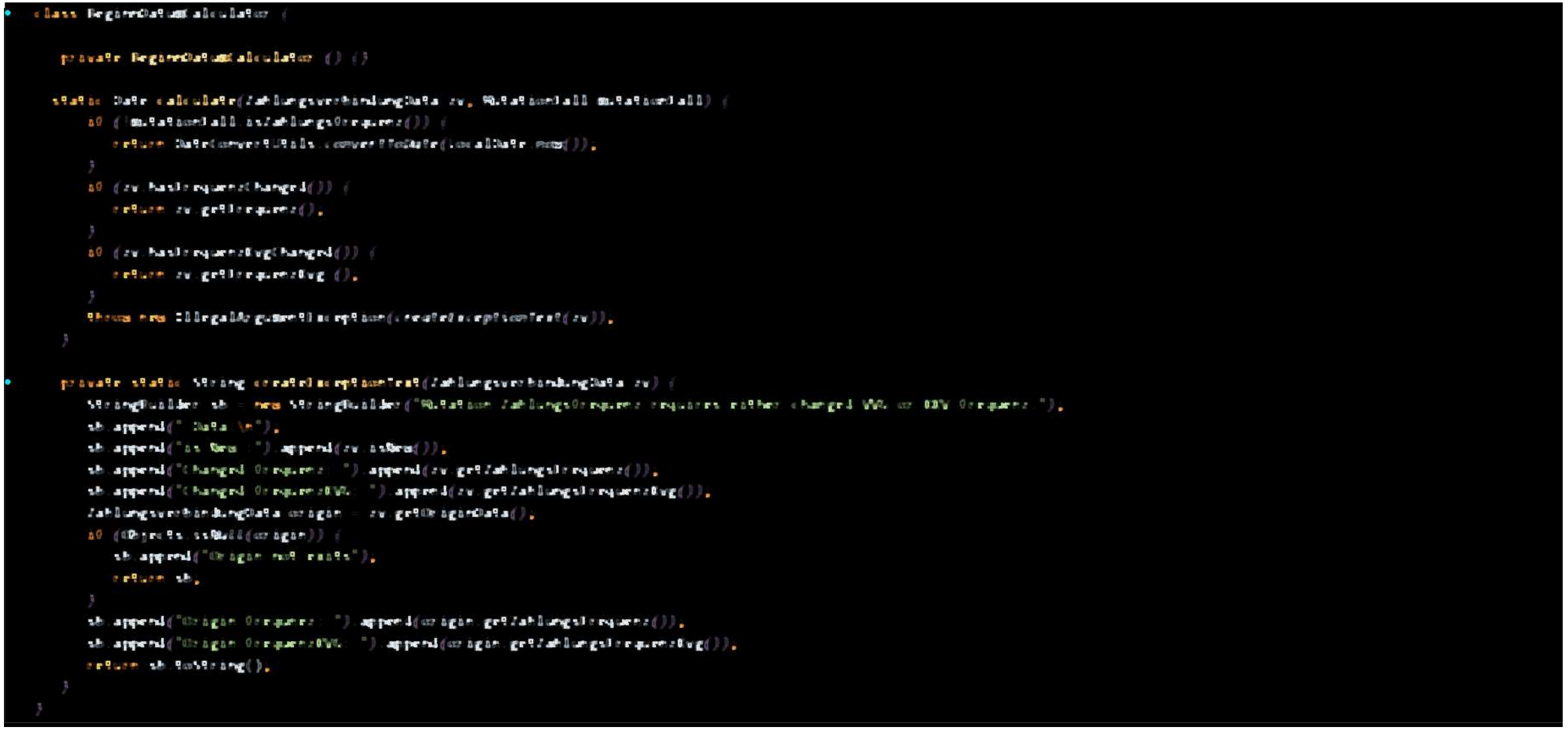
```
public class ProcessOrder {
    public int getOrderGrandTotal(Customer customer, int subTotal) {
        if (customer.type.equals("EMPLOYEE")) {
            return subTotal - 20;
        } else if (customer.type.equals("NON_EMPLOYEE")) {
            return subTotal - 10;
        }
        return subsTotal;
    }
}

////////////////////////////////////////////////////

public class Employee extends Customer {
    @Override
    public int getDiscount() {
        return 20;
    }
}
public class NonEmployee extends Customer {
    @Override
    public int getDiscount() {
        return 10;
    }
}

public class ProcessOrder {
    public int getOrderGrandTotal(Customer customer, int itemAmount) {
        return itemAmount - customer.getDiscount();
    }
}
```

How can you avoid Else blocks ?

- Early return
- Introduce a variable
- Polymorphism

- Null Object Pattern
- Strategy Pattern
- State Pattern

Refactored code




BRING IN
THE CHAMPAGNE
memegenerator.net