CLASSES

Java Clean code

Outline

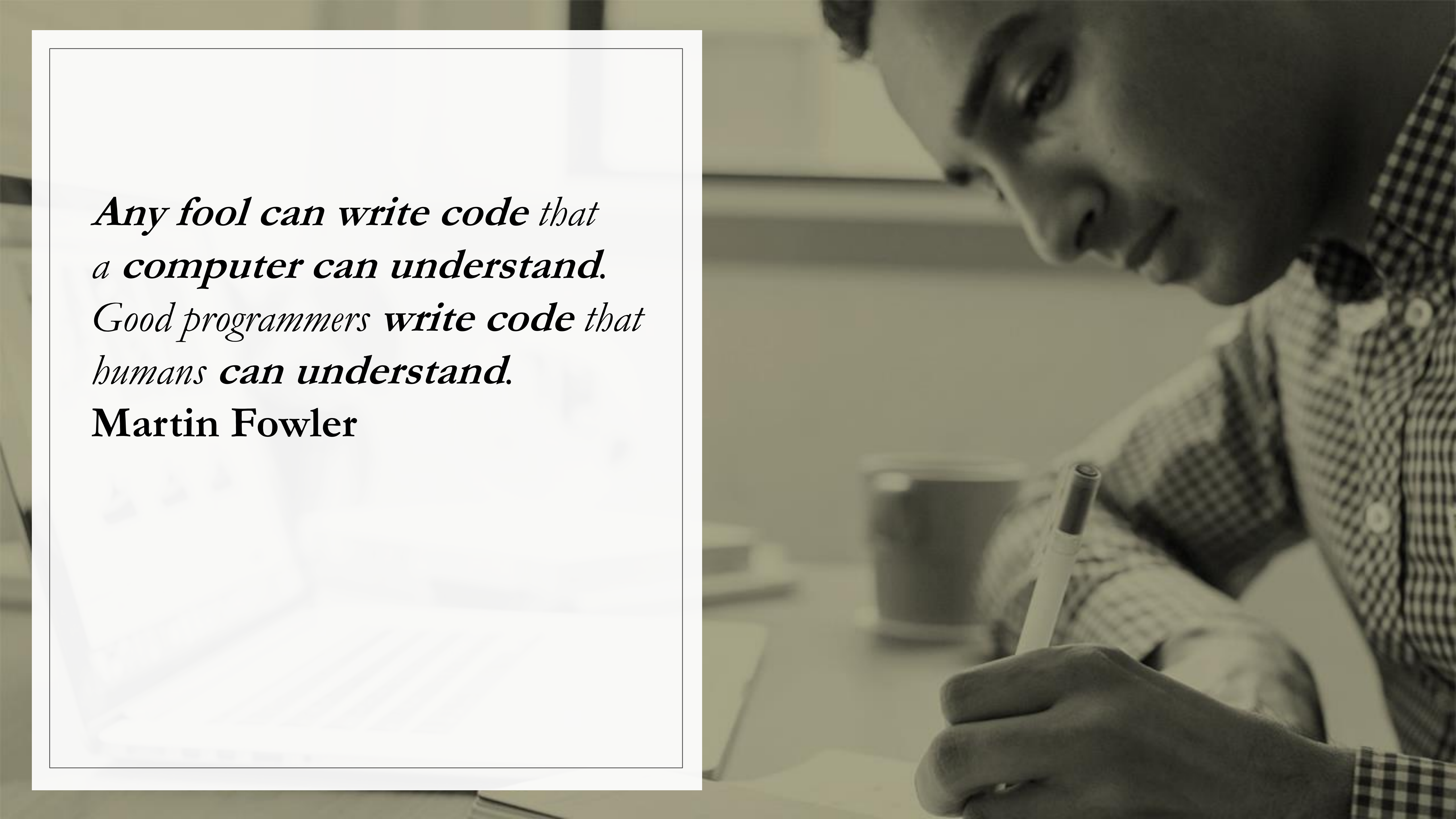
Lesson 1.
Clean code

Lesson 2.
Names

Lesson 3.
Methods

Lesson 4.
Classes

Lesson 5.
Comments



Any fool can write code *that*
a ***computer can understand****.*
Good programmers ***write code*** *that*
humans ***can understand****.*
Martin Fowler

What is a class

Class is like a container

- Class should do one thing only.
- Methods are important
- Use generalization for abstract classes and interfaces
- Opposite to data structures

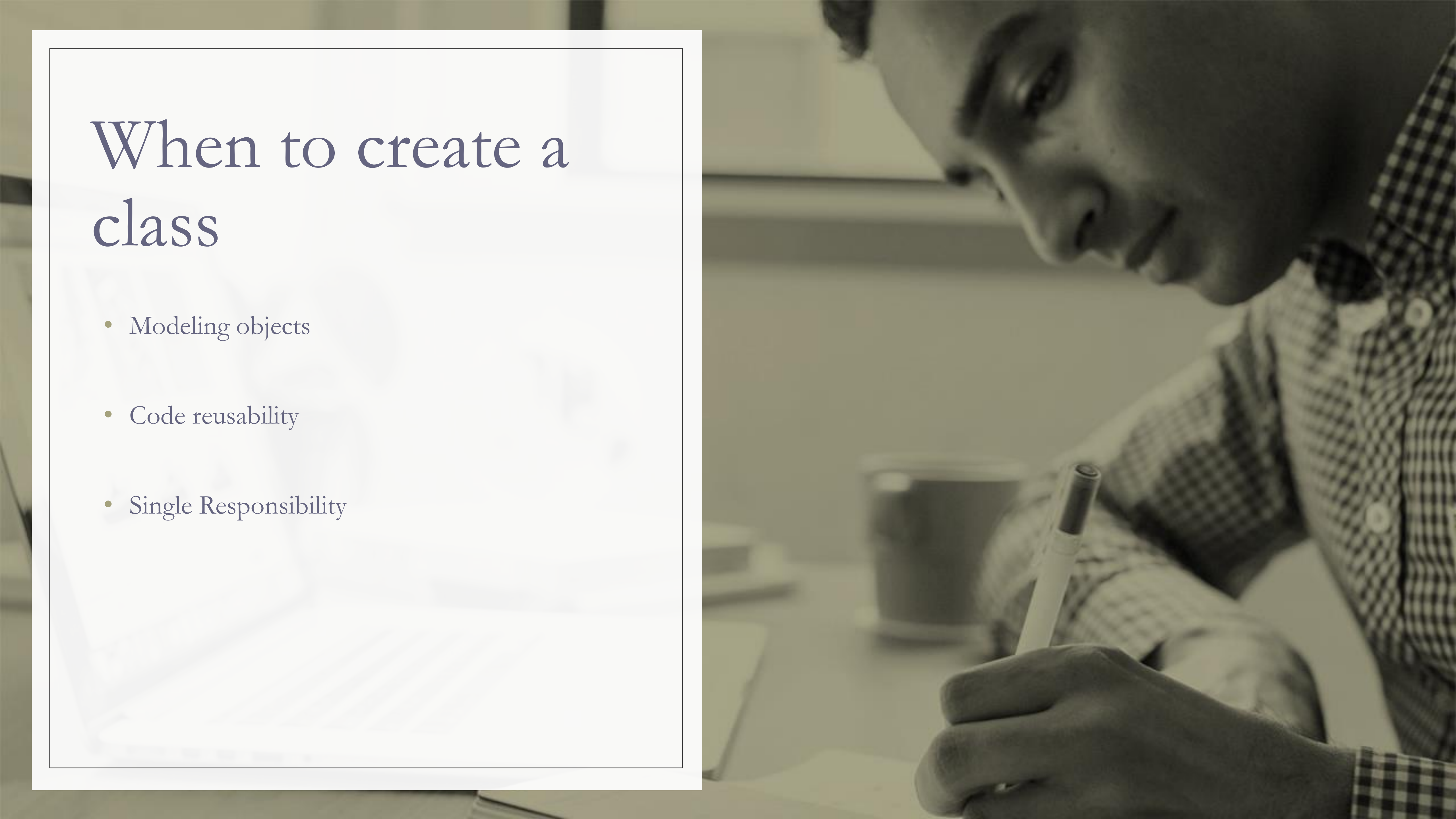
Class naming

Camel Case

- Nouns
- Simple
- Use whole word for the name

Classes name length

- Private and inner classes: long and descriptive names.
- Public classes should have small names



When to create a class

- Modeling objects
- Code reusability
- Single Responsibility

Cohesion

Strong responsibility

Reusable and easier to debug

Dead methods

Bad Example

Bus

- Edit bus
- Update ticket price
- Schedule maintenance
- Monthly payroll drivers
- Select payroll types

Good Example

Bus

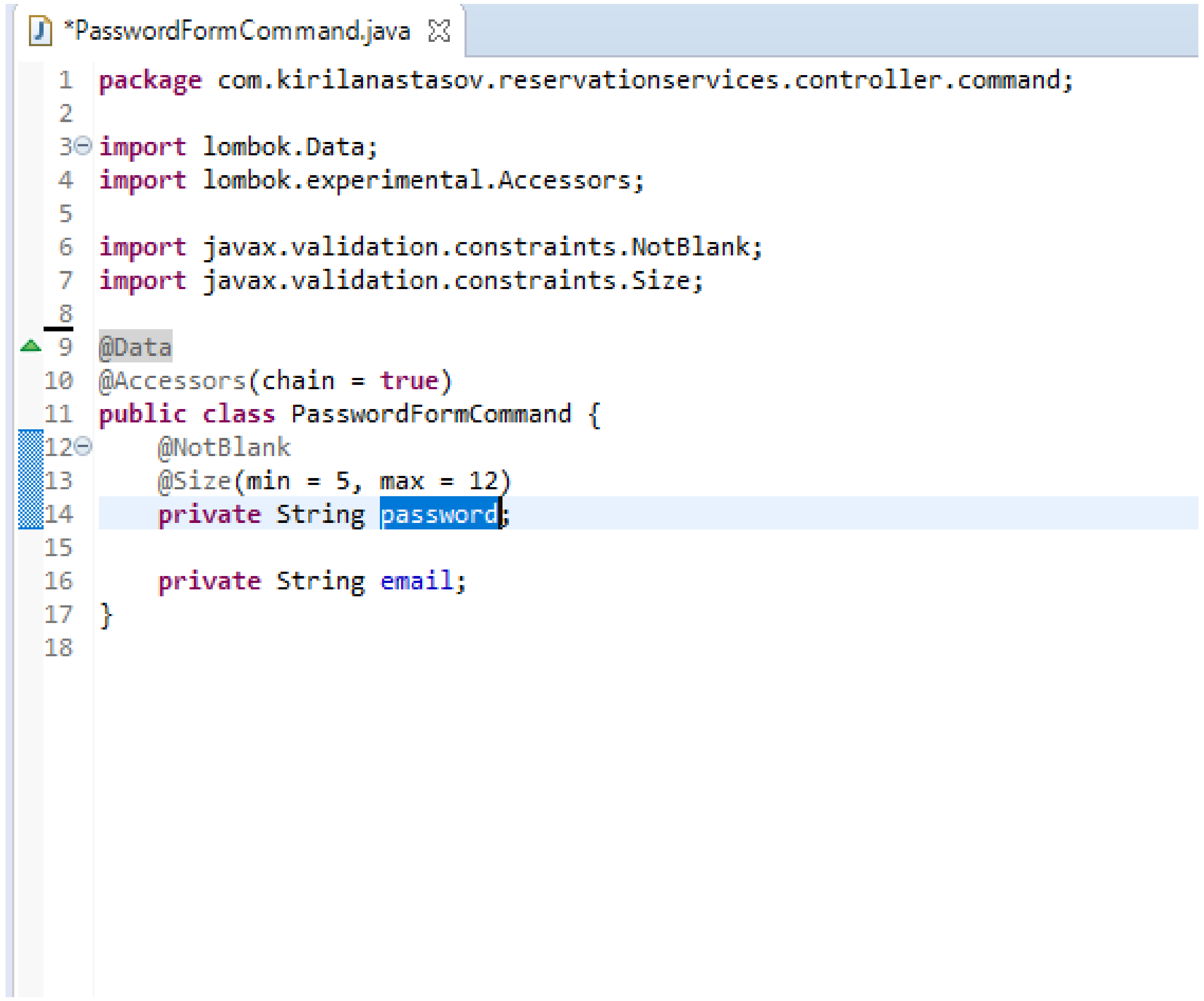
- Edit bus
- Update ticket price

BusRepairment

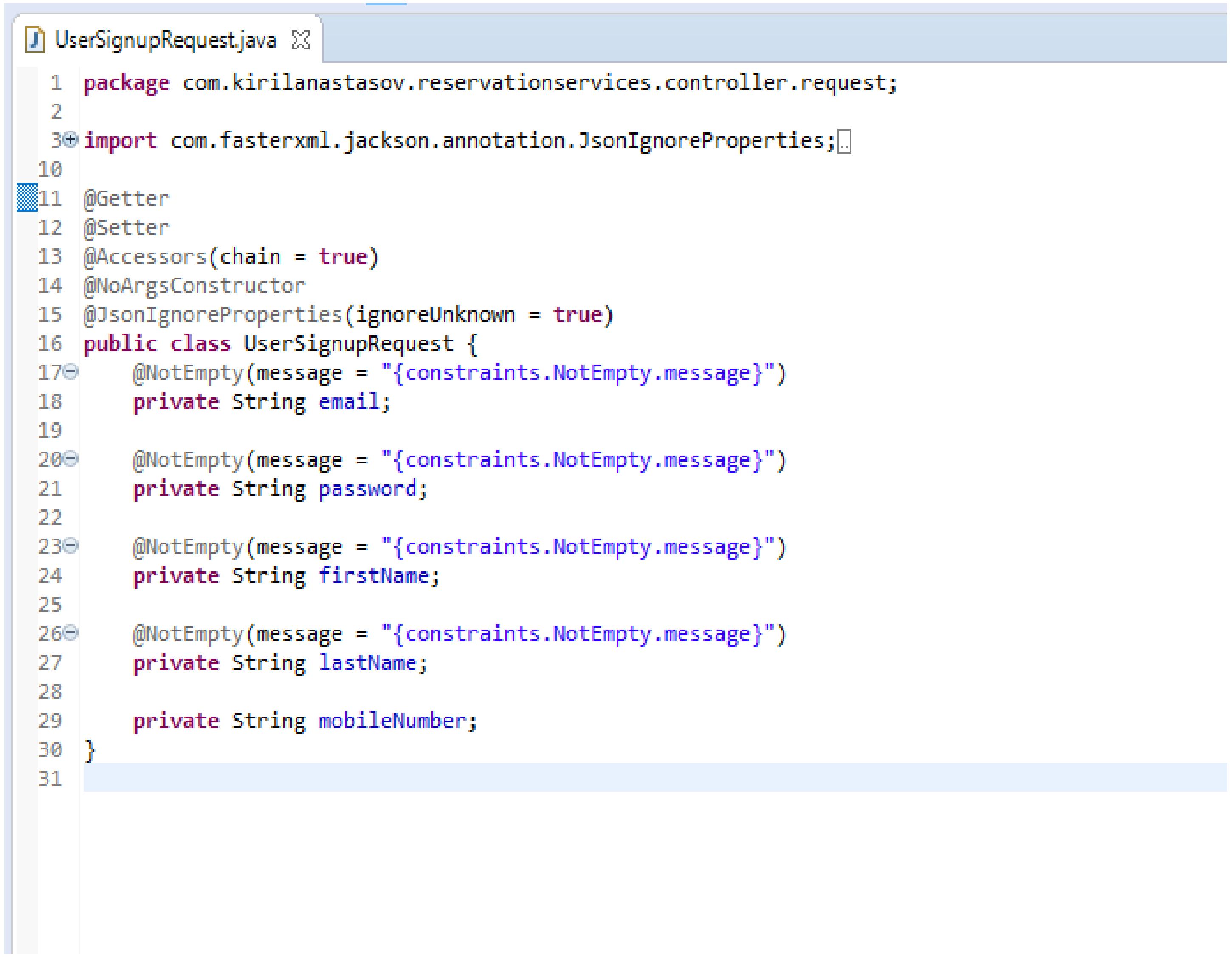
- Schedule maintenance

DriverPayroll

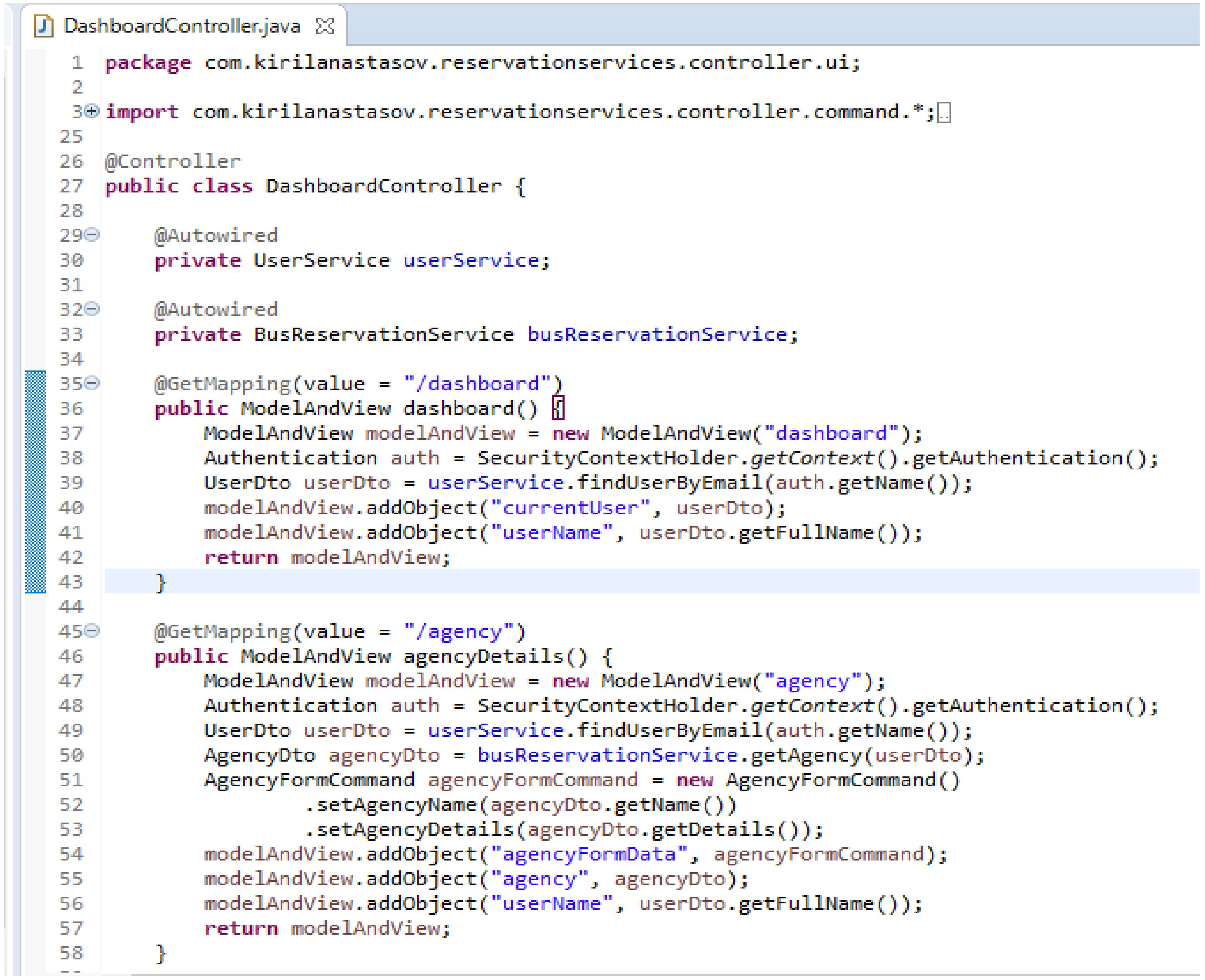
- Moonthly payroll drivers
- Select payroll types


*PasswordFormCommand.java

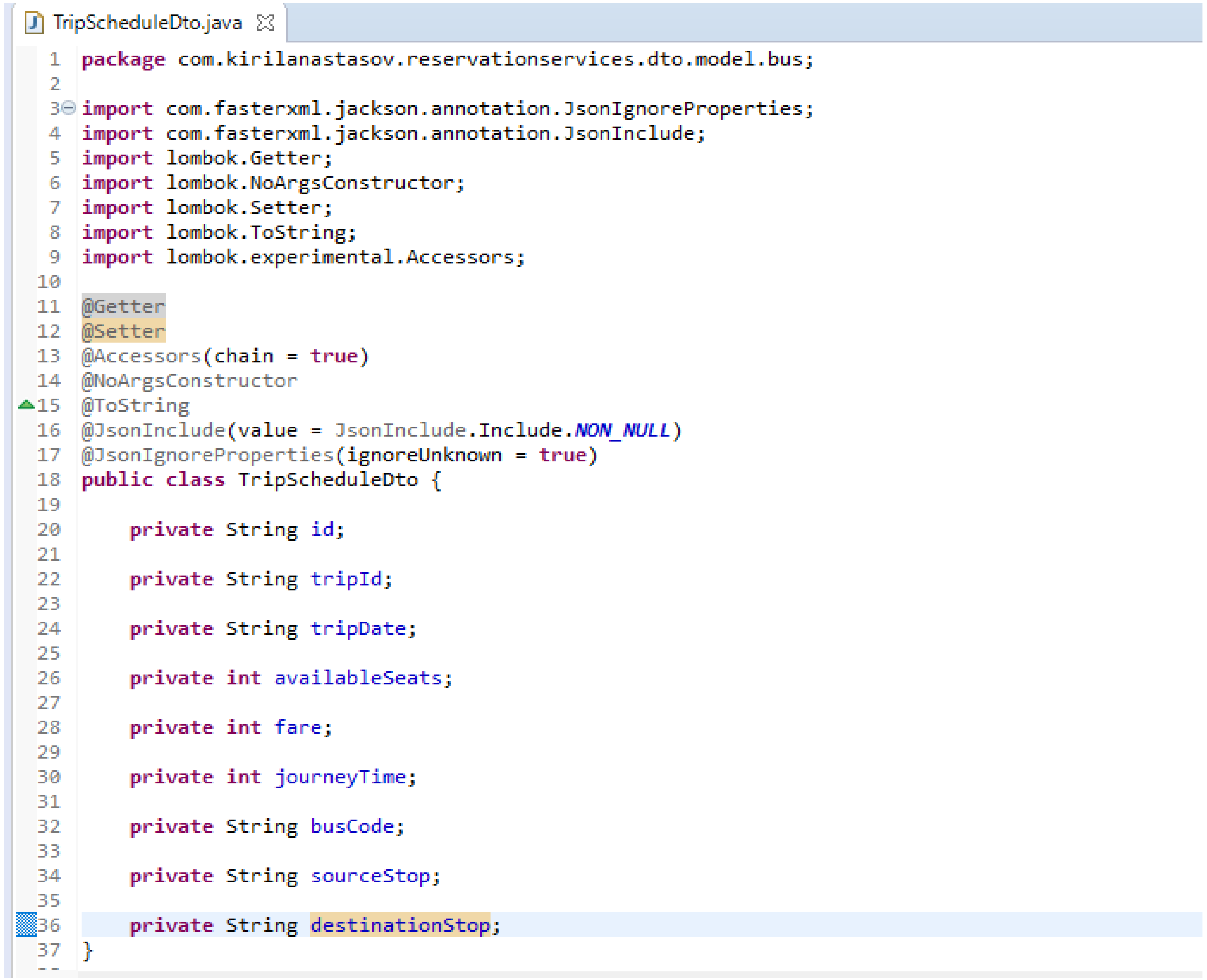
```java
package com.kirilanastasov.reservationservices.controller.command;

import lombok.Data;
import lombok.experimental.Accessors;

import javax.validation.constraints.NotBlank;
import javax.validation.constraints.Size;

@Data
@Accessors(chain = true)
public class PasswordFormCommand {
    @NotBlank
    @Size(min = 5, max = 12)
    private String password;

    private String email;
}

```


UserSignupRequest.java

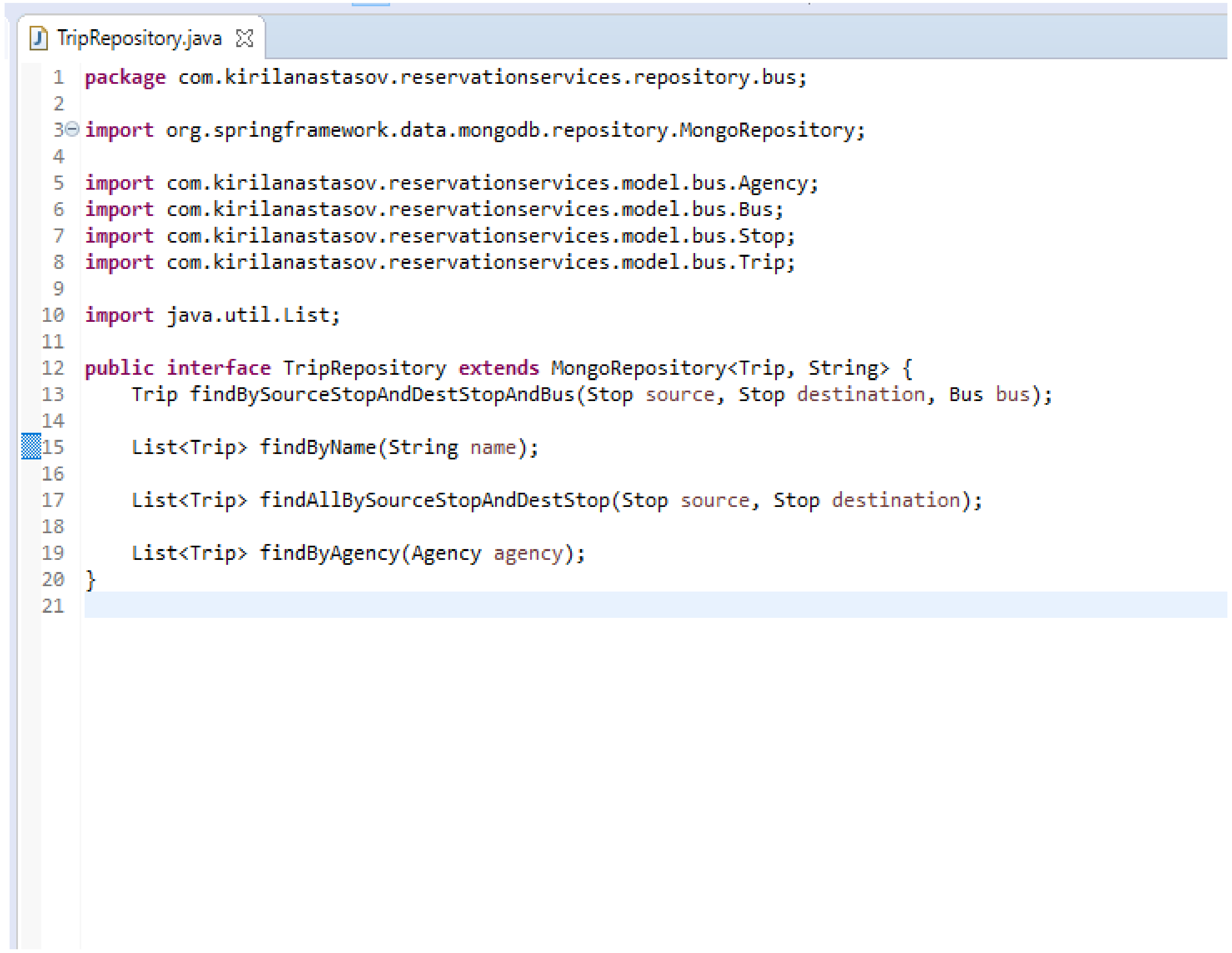
```java
package com.kirilanastasov.reservationservices.controller.request;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@Getter
@Setter
@Accessors(chain = true)
@NoArgsConstructor
@JsonIgnoreProperties(ignoreUnknown = true)
public class UserSignupRequest {
    @NotEmpty(message = "{constraints.NotEmpty.message}")
    private String email;

    @NotEmpty(message = "{constraints.NotEmpty.message}")
    private String password;

    @NotEmpty(message = "{constraints.NotEmpty.message}")
    private String firstName;

    @NotEmpty(message = "{constraints.NotEmpty.message}")
    private String lastName;

    private String mobileNumber;
}
```


DashboardController.java

```java
package com.kirilanastasov.reservationservices.controller.ui;

import com.kirilanastasov.reservationservices.controller.command.*;

@Controller
public class DashboardController {

    @Autowired
    private UserService userService;

    @Autowired
    private BusReservationService busReservationService;

    @GetMapping(value = "/dashboard")
    public ModelAndView dashboard() {
        ModelAndView modelAndView = new ModelAndView("dashboard");
        Authentication auth = SecurityContextHolder.getContext().getAuthentication();
        UserDto userDto = userService.findUserByEmail(auth.getName());
        modelAndView.addObject("currentUser", userDto);
        modelAndView.addObject("userName", userDto.getFullName());
        return modelAndView;
    }

    @GetMapping(value = "/agency")
    public ModelAndView agencyDetails() {
        ModelAndView modelAndView = new ModelAndView("agency");
        Authentication auth = SecurityContextHolder.getContext().getAuthentication();
        UserDto userDto = userService.findUserByEmail(auth.getName());
        AgencyDto agencyDto = busReservationService.getAgency(userDto);
        AgencyFormCommand agencyFormCommand = new AgencyFormCommand()
                .setAgencyName(agencyDto.getName())
                .setAgencyDetails(agencyDto.getDetails());
        modelAndView.addObject("agencyFormData", agencyFormCommand);
        modelAndView.addObject("agency", agencyDto);
        modelAndView.addObject("userName", userDto.getFullName());
        return modelAndView;
    }
```


TripScheduleDto.java

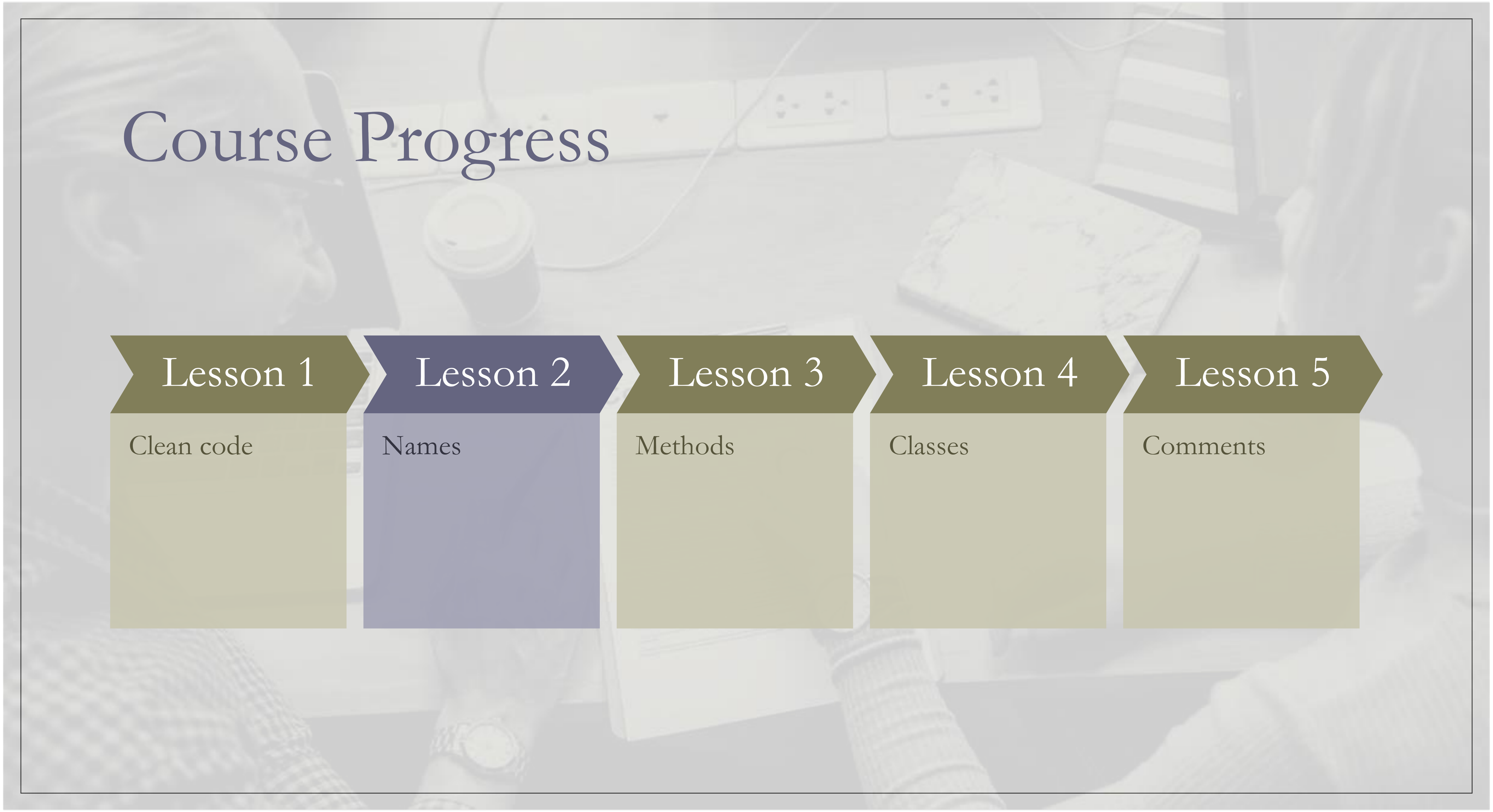
```
package com.kirilanastasov.reservationservices.dto.model.bus;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
import com.fasterxml.jackson.annotation.JsonInclude;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;
import lombok.ToString;
import lombok.experimental.Accessors;

@Getter
@Setter
@Accessors(chain = true)
@NoArgsConstructor
@ToString
@JsonInclude(value = JsonInclude.Include.NON_NULL)
@JsonIgnoreProperties(ignoreUnknown = true)
public class TripScheduleDto {

    private String id;

    private String tripId;

    private String tripDate;

    private int availableSeats;

    private int fare;

    private int journeyTime;

    private String busCode;

    private String sourceStop;

    private String destinationStop;
}
```


TripRepository.java

```java
package com.kirilanastasov.reservationservices.repository.bus;

import org.springframework.data.mongodb.repository.MongoRepository;

import com.kirilanastasov.reservationservices.model.bus.Agency;
import com.kirilanastasov.reservationservices.model.bus.Bus;
import com.kirilanastasov.reservationservices.model.bus.Stop;
import com.kirilanastasov.reservationservices.model.bus.Trip;

import java.util.List;

public interface TripRepository extends MongoRepository<Trip, String> {
    Trip findBySourceStopAndDestStopAndBus(Stop source, Stop destination, Bus bus);

    List<Trip> findByName(String name);

    List<Trip> findAllBySourceStopAndDestStop(Stop source, Stop destination);

    List<Trip> findByAgency(Agency agency);
}
```

Classes Lesson Summary

- Class should do one thing only
- Use generalization for abstract classes and interfaces
- Pubic classes should have small but descriptive names
- Cohesion



Course Progress

Lesson 1	Lesson 2	Lesson 3	Lesson 4	Lesson 5
Clean code	Names	Methods	Classes	Comments

Oracle naming convention

- https://www.oracle.com/java/technologies/javase/codeconventions-namingconventions.html

THANK YOU!