

## Todo IT – part 1

### Topics:

- Object Oriented Programming
- Encapsulation
- Unit testing
- UML diagram

### Overview:

This is the first part of three where you are going to build a Todo application.

This first part consists of building the first models (classes), where we put focus on **encapsulation** and **abstraction**. We will continue with this project in the following week so **when you are done** with this first step **do not add anything further**.

### Requirements:

- Need to be a maven project.
- **Person.class**, **TodoItem.class** and **TodoItemTask.class** fully implemented according to specific requirements.
- **Person.class**, **TodoItem.class** and **TodoItemTask.class** tested with **Junit4** or **Junit5**

Good luck!

## Person.class:

### Fields:

- **id** (private) is an **int** representing each Person object.
- **firstName** (private) represents each person's first name. **Not allowed to be null**
- **lastName** (private) represents each person's last name. **Not allowed to be null**
- **email** (private) represents each person's email. **Not allowed to be null**

### Constructor:

Up to you

### Methods:

- Common getters and setters.
- *getSummary()* should return a **description of the object**. Like {id: 4, name: Nisse Olsson, email: nisse@gmail.com}

| Person   |
|--|
| id : int<br>firstName: String<br>lastName: String<br>email: String   |
| getId() : int<br>getFirstName() : String<br>setFirstName(firstName) : void<br>getLastName() : String<br>setLastName(lastName) : void<br>getEmail() : String<br>setEmail(email) : void<br>getSummary() : String |

## TodoItem.class:

### Fields:

- **id** (private) is an **int** representing each TodoItem object.
- **title** representing a title like 'Change tires.' **Not** allowed to be **null or empty**
- **description** is used to hold further information
- **deadLine** TodoItem is overdue if current date > deadline. **Not** allowed to be **null**
- **done** represent if task is finished
- **creator** represent who created this task.

### Constructor:

Up to you

### Methods:

- Common getters and setters
- *getSummary()* (see Person)
- *isOverdue()* should return **true** if current date has passed deadLine.

| <i>TodoItem</i>   |
|---|
| <i>id</i> : int<br><i>title</i> : String<br><i>taskDescription</i> : String<br><i>deadLine</i> : LocalDate<br><i>done</i> : boolean<br><i>creator</i> : Person  |
| <i>getId()</i> : int<br><i>getTitle()</i> : String<br><i>setTitle(title)</i> : void<br><i>getTaskDescription()</i> : String<br><i>setTaskDescription(description)</i> : void<br><i>getDeadLine()</i> : LocalDate<br><i>setDeadLine(deadLine)</i> : void<br><i>setDone(done)</i> : void<br><i>isDone()</i> : boolean<br><i>getCreator()</i> : Person<br><i>setCreator(person)</i> : void<br><i>isOverdue()</i> : boolean<br><i>getSummary()</i> : String |

## TodoItemTask.class

Fields:

- **id** (private) is an **int** representing each TodoItemTask object
- **assigned** (private) set to true if assignee is not null
- **todoItem** (private) represent the details what assignee should do before deadline. **Not null**
- **assignee** (private) is the Person being assigned to do the Task.

Constructor:

Up to you

Methods:

- Common getters and setters
- *getSummary()* (see Person)

Tip: make sure boolean assigned is **encapsulated**

| <i>TodoItemTask</i>  |
|--|
| <i>id : int</i><br><i>assigned: boolean</i><br><i>todoItem: TodoItem</i><br><i>assignee: Person</i>  |
| <i>getId() : int</i><br><i>isAssigned() : boolean</i><br><i>setAssigned(assigned) : void</i><br><i>getTodoItem() : TodoItem</i><br><i>setTodoItem(todoItem) : void</i><br><i>getAssignee() : Person</i><br><i>setAssignee(person) : void</i><br><i>getSummary() : String</i> |

Full diagram:

