DOCUMENTATION WORKSHOP-2 PROJECT OBJECT-ORIENTED PROGRAMMING: NOTION (NOTES APP)

Juan Carlos Córdoba Asprilla – 20242020047

Sebastián Camilo Sánchez Cárdenas- 20242020086

Marzo 01 de 2025

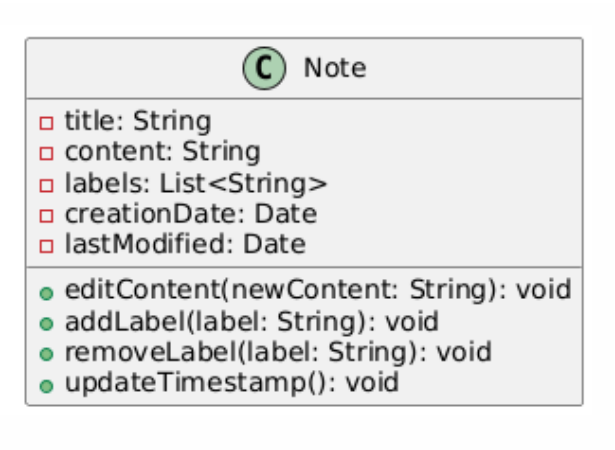
UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Object-oriented Programming

Ing. Carlos Andrés Sierra

CONCEPTUAL DESING UPDATES:

1. Addition of the Note Class



Reason:

Although the Notes class referenced a list of Note objects (notesList: List<Note>), the Note class itself was not explicitly defined. It has been created to represent a single note, fulfilling key user stories and functional requirements related to note creation, editing, tagging, and tracking modification times.

Attributes:

- title: String
- content: String
- labels: List<String>
- creationDate: Date
- lastModified: Date

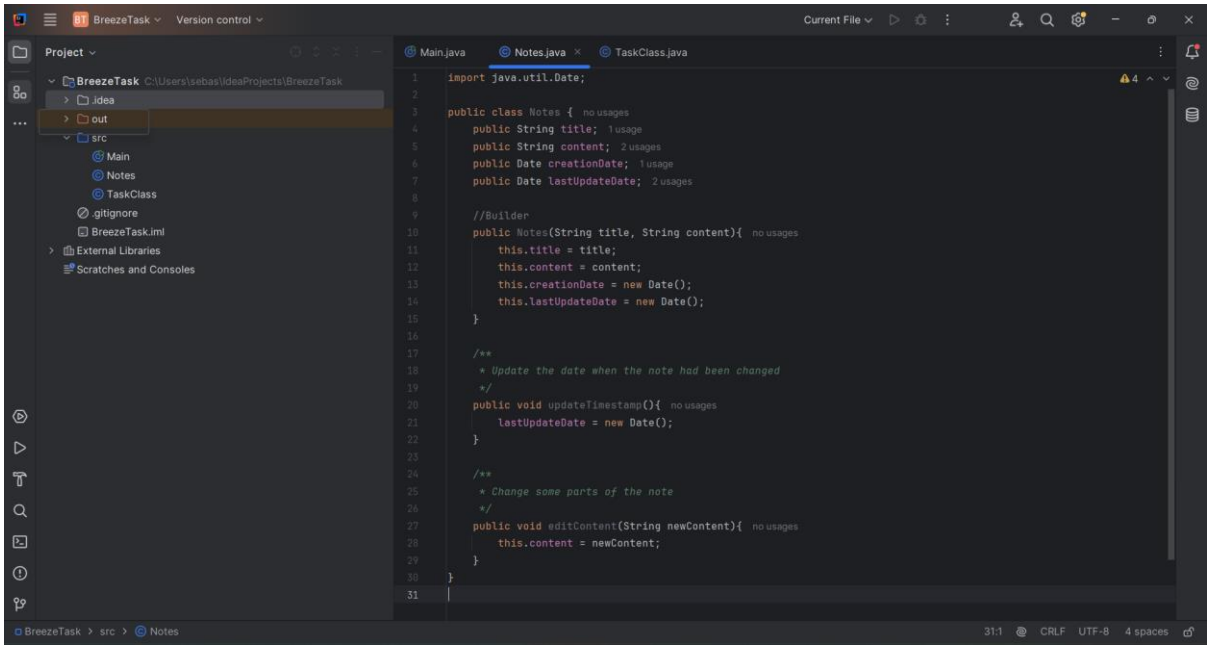
Methods:

- editContent(newContent: String): void
- addLabel(label: String): void
- removeLabel(label: String): void
- updateTimestamp(): void

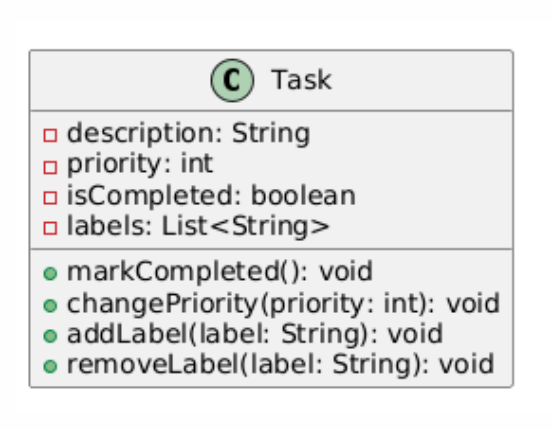
Relationships:

Notes --> Note

Note --> Storage



2. Addition of the Task Class



Reason:
The Tasks class was already listed as managing multiple tasks, but no definition existed for the individual Task entity. This class was added to model a single task with its priority, completion status, and labels—supporting user stories that involve goal tracking, task organization, and filtering.

Attributes:

- description: String
- priority: int (ejem., 1 = high, 2 = medium, 3 = low)
- isCompleted: boolean
- labels: List<String>

Methods:

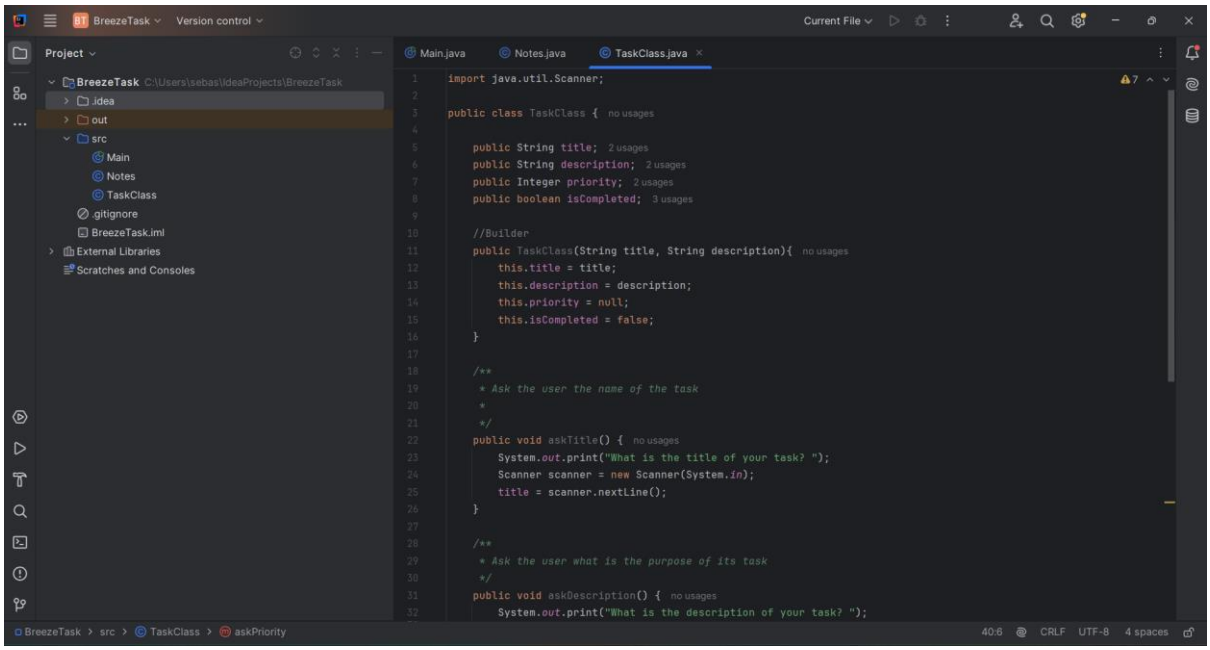
- markCompleted(): void

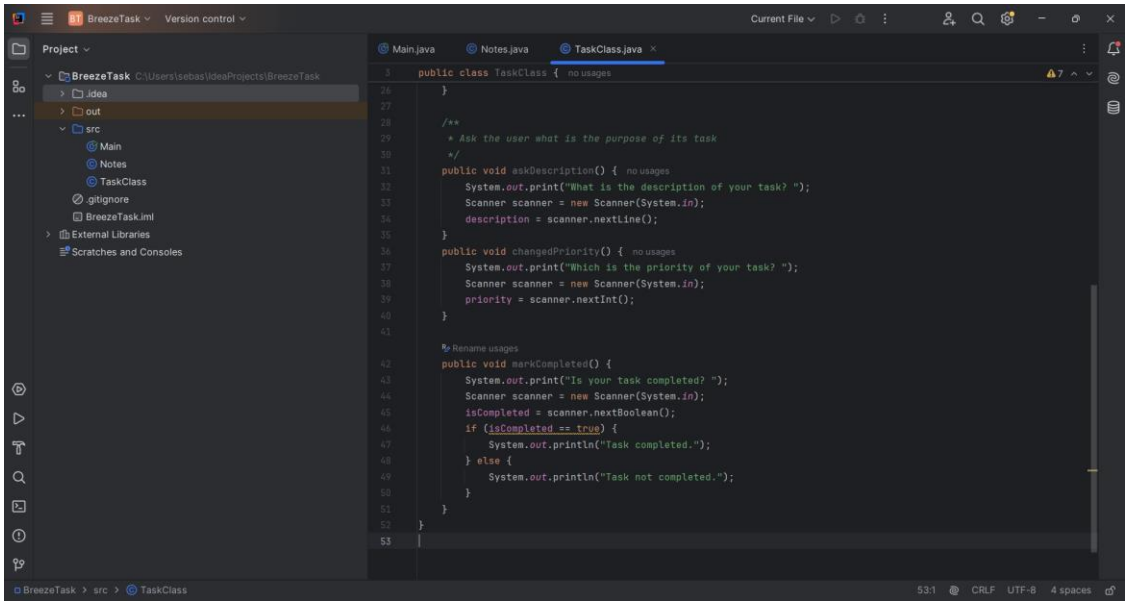
- changePriority(priority: int): void
- addLabel(label: String): void
- removeLabel(label: String): void

Relationships:

Tasks --> Task

Task --> Storage





3. Actualized and new CRC cards for :

CLASS (Notes)	
RESPÓNSABILITY	COLABORATOR
• Handle note creation, filtering and deletion,	• Storage • Application • note

CLASS (Note) - new crc card	
RESPÓNSABILITY	COLABORATOR
• Manage note content with functions such as edit and delete content	• notes

CLASS (Tasks)	
RESPÓNSABILITY	COLABORATOR
• Handle task creation, filtering and deletion	• Storage • Application • task

CLASS (Task) - new crc card	
RESPÓNSABILITY	COLABORATOR
• Manage task content with features like marking as complete, choosing a task name, and more.	• tasks

DIAGRAMS

ACTIVITY DIAGRAM - UML CLASS DIAGRAM - SECUENCE DIAGRAM

Note: The diagrams shown below use classes with conceptual (yet to be defined) attributes and methods to establish their respective connections. In the future, the structure of these methods and the necessary attributes to include or exclude will be reviewed to ensure the application functions are as intended.

APPLICATION OF OOP CONCEPTS

At this stage of the application's development, the use of object-oriented programming principles is not yet fully defined or implemented, as many functionalities are focused on direct user interaction with specific modules.

Regarding **encapsulation**, protecting the user's information has been considered essential, especially since this is a local application — without online connectivity — meaning all data will be stored directly on the user's device. Therefore, appropriate access modifiers are planned to restrict direct access to attributes and ensure that any modifications are handled through well-defined public methods.

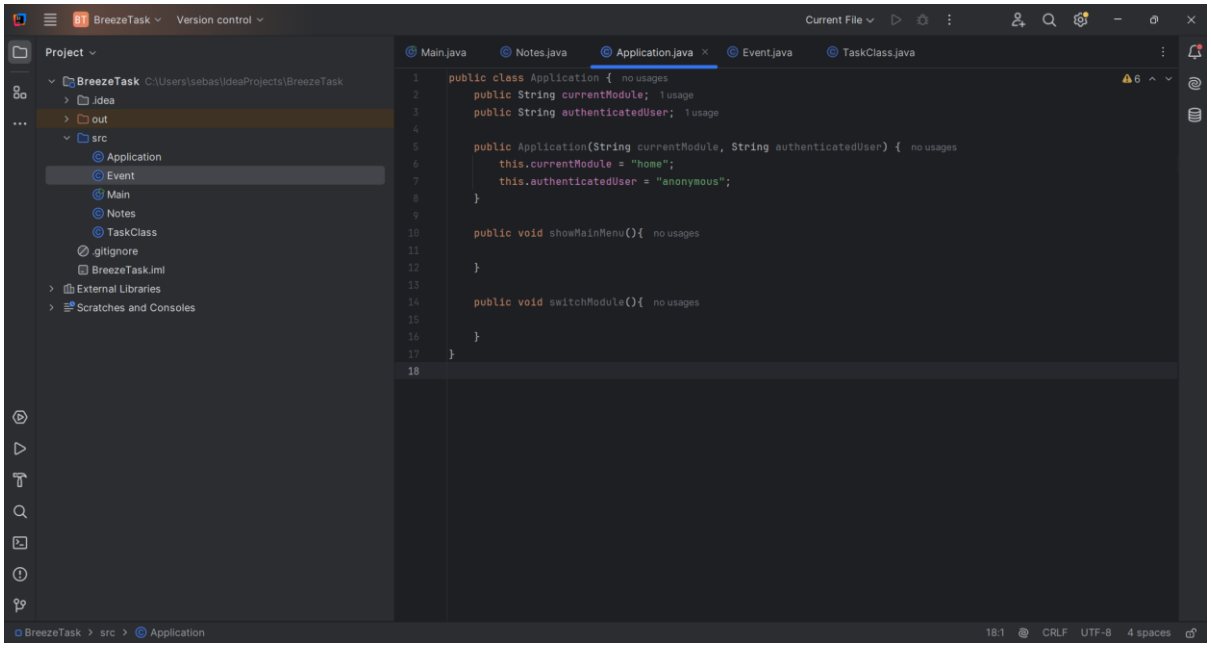
Concerning **inheritance**, the possibility of creating a base class called Module has been considered, from which components such as calendar, notes, tasks, and reminders could be derived. However, this decision is still under evaluation, as it must be determined whether it truly adds value or introduces unnecessary complexity into the design.

Lastly, regarding **polymorphism**, its implementation is not currently anticipated, as each class's responsibilities are clearly defined and do not require flexible or alternative behaviors. Each module operates independently and fulfills specific tasks, making the use of this concept currently unnecessary.

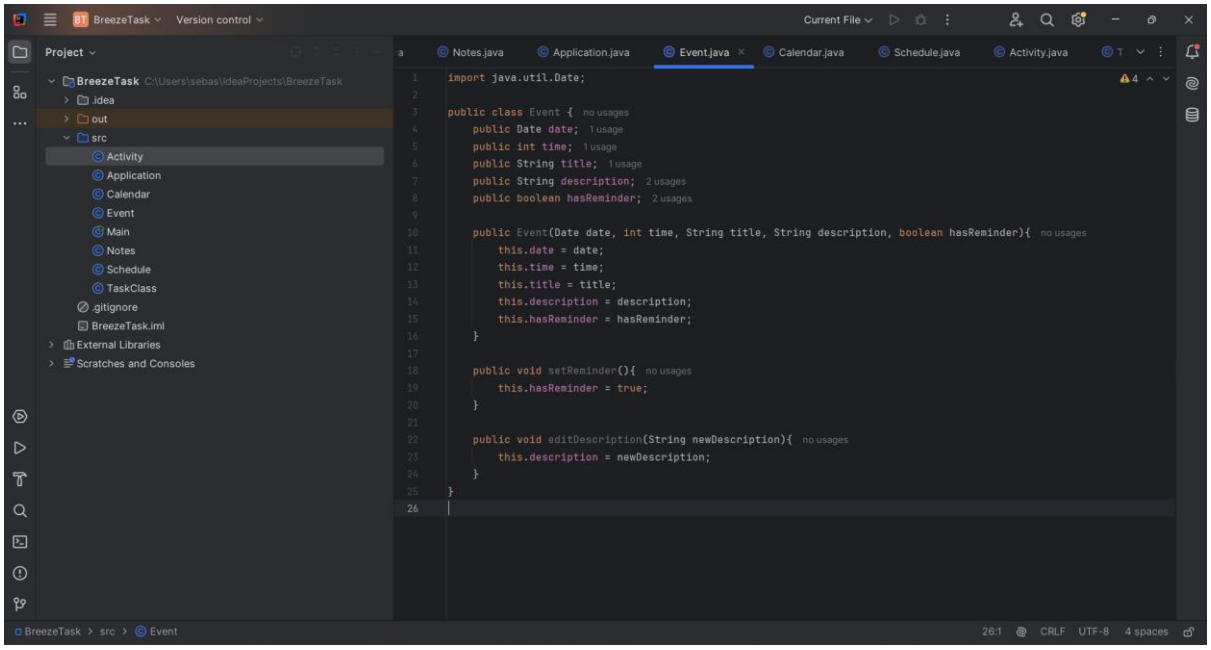
PACKAGE ESTRUCTURE

WORK IN PROGRESS

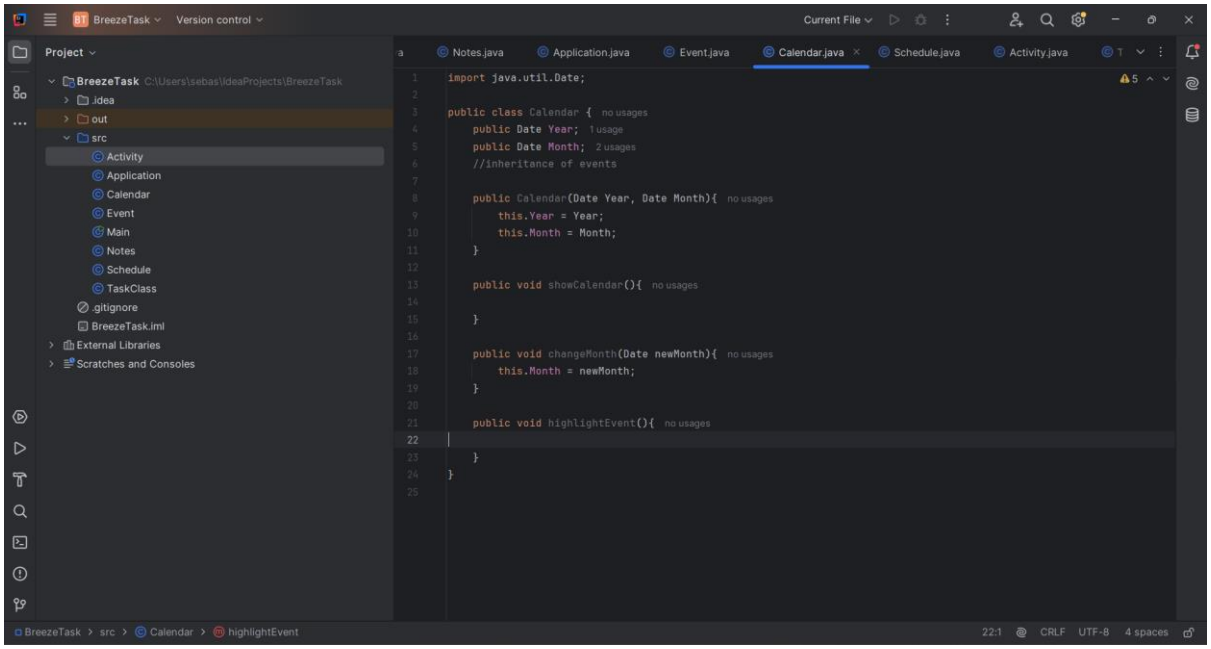
Application



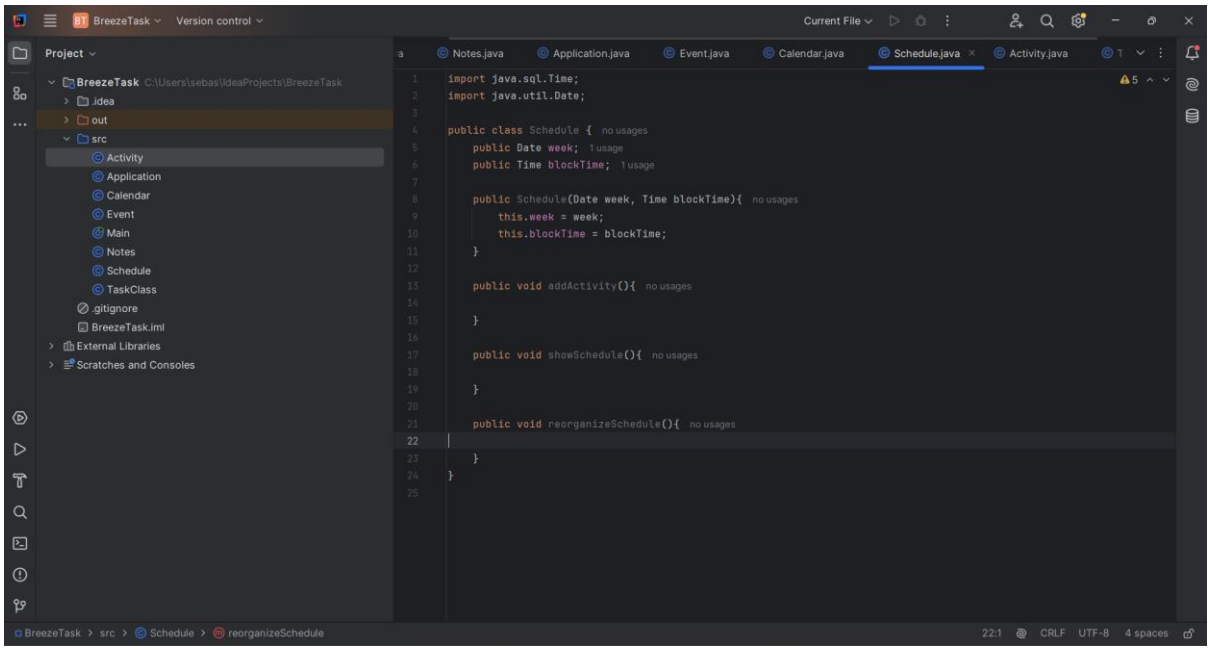
Event



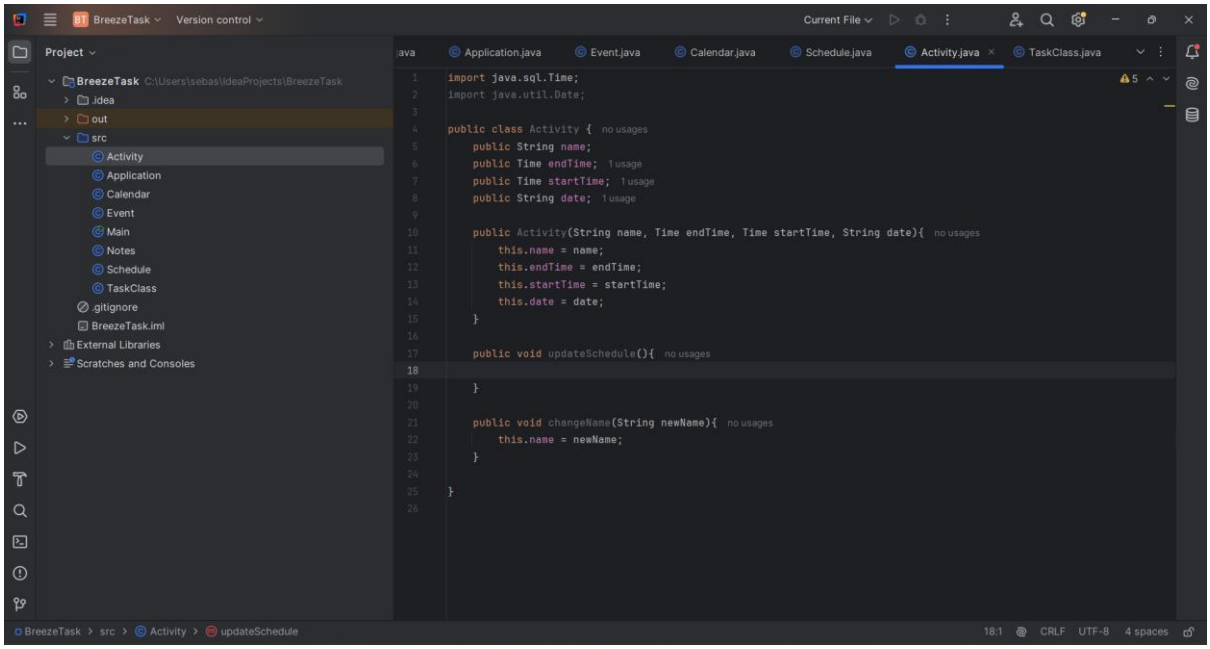
Calendar



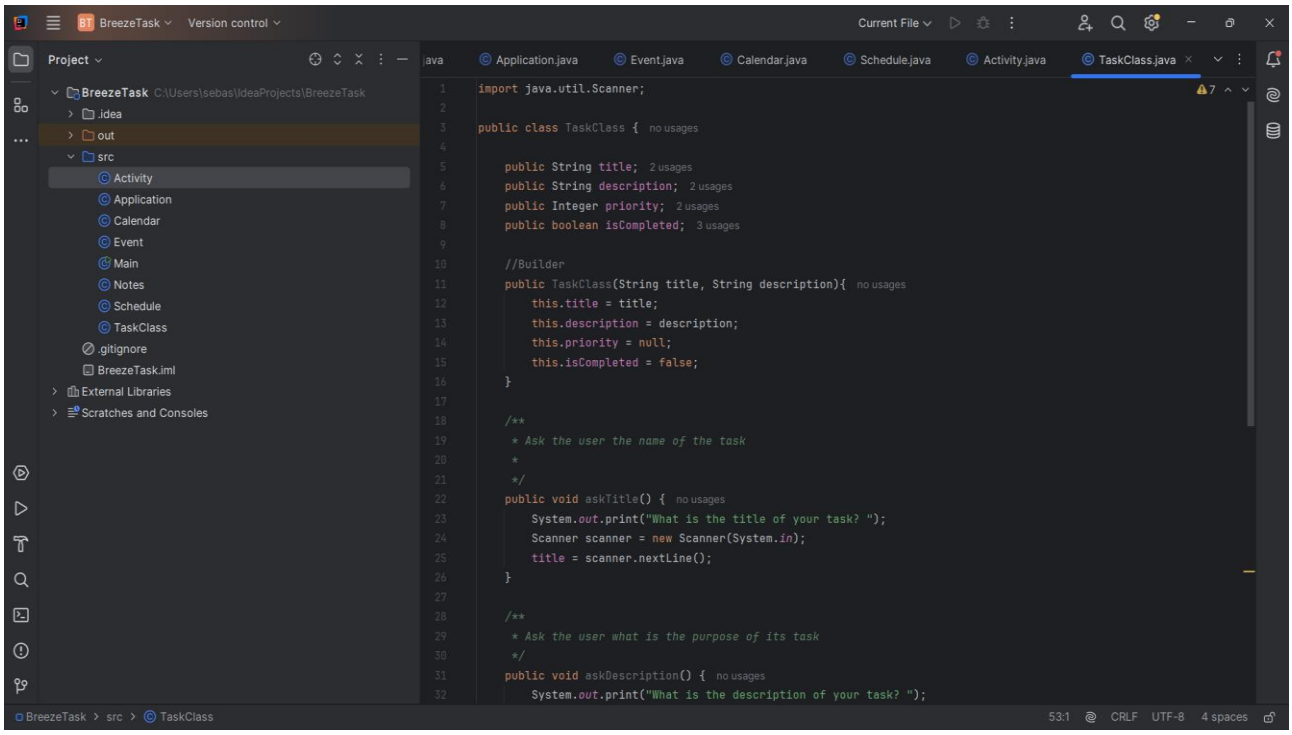
Schedule

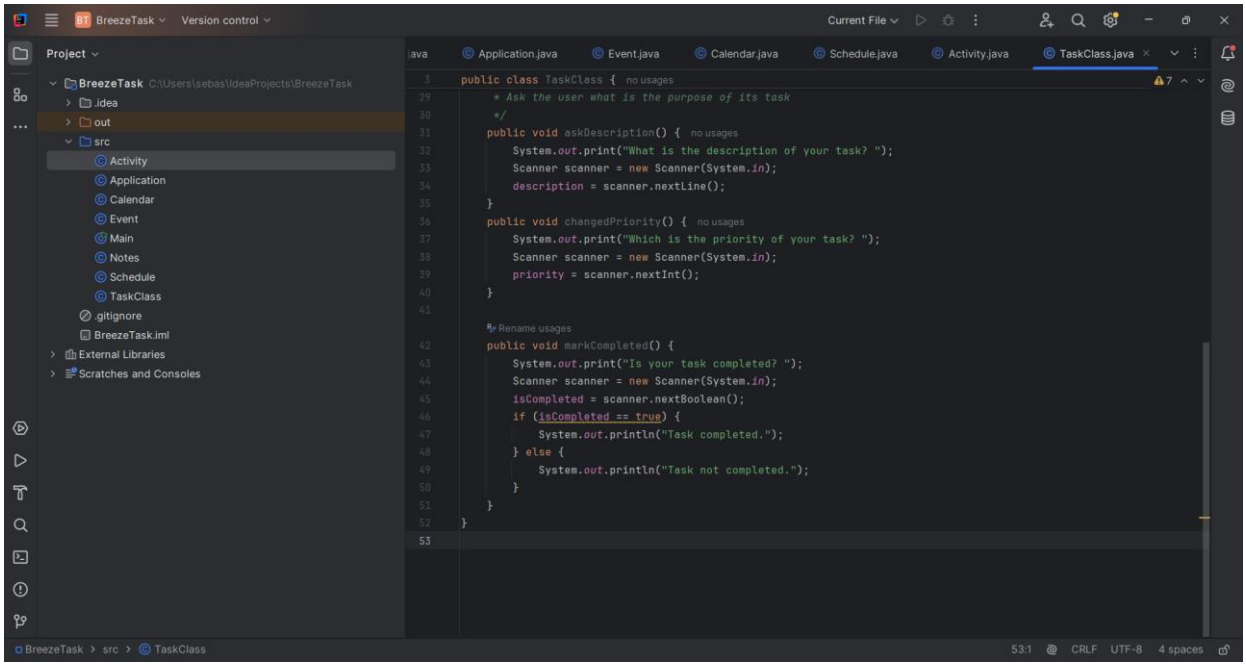


Activity


BreezeTask
Version control
Current File
Project
Application.java
Event.java
Calendar.java
Schedule.java
Activity.java
TaskClass.java
BreezeTask C:\Users\sebas\IdeaProjects\BreezeTask
.idea
out
src
Activity
Application
Calendar
Event
Main
Notes
Schedule
TaskClass
.gitignore
BreezeTask.iml
External Libraries
Scratches and Consoles
import java.sql.Time;
import java.util.Date;
public class Activity { no usages
public String name;
public Time endTime; 1 usage
public Time startTime; 1 usage
public String date; 1 usage
public Activity(String name, Time endTime, Time startTime, String date){ no usages
this.name = name;
this.endTime = endTime;
this.startTime = startTime;
this.date = date;
}
public void updateSchedule(){ no usages
}
public void changeName(String newName){ no usages
this.name = newName;
}
}
BreezeTask › src › Activity › updateSchedule
18:1
CRLF
UTF-8
4 spaces

Task


BreezeTask
Version control
Current File
Project
Application.java
Event.java
Calendar.java
Schedule.java
Activity.java
TaskClass.java
BreezeTask C:\Users\sebas\IdeaProjects\BreezeTask
.idea
out
src
Activity
Application
Calendar
Event
Main
Notes
Schedule
TaskClass
.gitignore
BreezeTask.iml
External Libraries
Scratches and Consoles
import java.util.Scanner;
public class TaskClass { no usages
public String title; 2 usages
public String description; 2 usages
public Integer priority; 2 usages
public boolean isCompleted; 3 usages
//Builder
public TaskClass(String title, String description){ no usages
this.title = title;
this.description = description;
this.priority = null;
this.isCompleted = false;
}
/**
* Ask the user the name of the task
*
*/
public void askTitle() { no usages
System.out.print("What is the title of your task? ");
Scanner scanner = new Scanner(System.in);
title = scanner.nextLine();
}
/**
* Ask the user what is the purpose of its task
*/
public void askDescription() { no usages
System.out.print("What is the description of your task? ");
BreezeTask › src › TaskClass
53:1
CRLF
UTF-8
4 spaces


BreezeTask
Version control
Current File
Project
BreezeTask C:\Users\sebas\IdeaProjects\BreezeTask
.idea
out
src
Activity
Application
Calendar
Event
Main
Notes
Schedule
TaskClass
.gitignore
BreezeTask.iml
External Libraries
Scratches and Consoles
Application.java
Event.java
Calendar.java
Schedule.java
Activity.java
TaskClass.java
public class TaskClass { no usages
* Ask the user what is the purpose of its task
*/
public void askDescription() { no usages
System.out.print("What is the description of your task? ");
Scanner scanner = new Scanner(System.in);
description = scanner.nextLine();
}
public void changedPriority() { no usages
System.out.print("Which is the priority of your task? ");
Scanner scanner = new Scanner(System.in);
priority = scanner.nextInt();
}
Rename usages
public void markCompleted() {
System.out.print("Is your task completed? ");
Scanner scanner = new Scanner(System.in);
isCompleted = scanner.nextBoolean();
if (isCompleted == true) {
System.out.println("Task completed.");
} else {
System.out.println("Task not completed.");
}
}
}
BreezeTask > src > TaskClass
53:1
CRLF
UTF-8
4 spaces