**Личен календар (КП – ООП-1)**

*Проект изготвен от Богомил Стоев, СИТ, 2 Курс , Група 3а, ФН: 21621547*

Връзка към хранилището в GitHub: [www.github.com/bogomilstoev7](http://www.github.com/bogomilstoev7)

Структура на документацията:

1. Увод
2. Реализация
3. Заключение
4. **Увод**

Проекта представлява програма, реализираща информационна система, която поддържа личен календар, като го записва във файл. Програмата позволява различни операции, с които да помогне да менажирате вашето време. Тя включва обработка на календара, като отваряне, затваряне, запазване в файл, както и добавяне, премахване, търсене и други промени по срещите в календара. Главна цел на проекта е да се симулира обработване на персонализиран календар, записвайки в него дадени срещи, и действия с тях като някои от тях включват:   
-Извеждане на хронологичен списък с всички ангажименти за даден ден

-Промяна на дадена среща, по зададени параметри от потребителя

-Търсене на срещи по дата

-Отбелязване на даден ден като неработен (почивен)

-Извеждане на статистика за натовареност на дните в дадени граници

-Търсене и намиране на свободно място за вмъкване на среща в ден

-Извеждане на всички срещи в даденият календар

Главните концепции използвани в проекта са обектно-ориентирано програмиране, обработка на XML файлове, поддръжка на потенциални грешки.

Проблемите, които са решени представляват главно реализирането на отваряне, четене, записване на файлове в формат XML, както и правилно подреждане на кода спрямо изискванията за обектно-ориентиран стил.

1. **Реализация**

**Реализация на Мain класа:**

main метод-a се изпълнява при стартиране на програмата и съдържа главната логика на програмата.В метода има обект от клас "CalendarManager", който е отговорен за управлението на една календарна система.Извикването на методът "calendarManager.start()" представлява началната точка за изпълнение на календарната система.Ако възникнат грешки по време на изпълнението на програмата, те ще бъдатхванати от блока "try-catch", който обхваща изключения от типовете "IOException", "JAXBException", "CustomException" и "ParseException". Ако бъде хваната, ще се изведе информация за грешката.

**Реализация на CustomException класа:**

Класът CustomException създава персонализирано изключение. Конструкторът на CustomException приема един аргумент - съобщение за грешката, което се предава към конструктора на родителския клас (Exception) чрез използването на ключовата дума "super".

**Реализация на Appointment класа**:

Клас "Appointment" представлява една среща (appointment) в една календарна система.Класът има следните полета: "date", "startTime", "endTime", "name", "note" и "isHoliday".Конструктор за създаване на обект с параметри за всички атрибути на класа.*Getters и Setters* за всяко от полетата, тъй като ще ни бъдат нужни при обработката на срещи.Анотациите "@XmlRootElement" и "@XmlElement" идентифицират този клас като JAXB обект, който може да бъде сериализиран и десериализиран в XML формат.Анотацията "@XmlRootElement" задава името на кореновия елемент в XML файла, който съответства на обекта "Appointment".Анотациите "@XmlElement" задават името на елементите, които трябва да се генерират в XML файла,за да се съхранят стойностите на полетата на класа.Методът "toString" връща текстово представяне на обекта "Appointment". Сериализирането на обект в XML формат означава да се преобразува обекта от обектно-ориентиран вид във вид, който може да се съхранява във файл.В случая на класа "Appointment", той може да се сериализира в XML формат, който ще съдържа информацията за дата, начален и краен час, име на срещата, бележка и булева стойност дали е почивен ден.Десериализацията на обект от XML формат означава да се преобразува предварително сериализиран обект във формат на обект, който може да се използва в програмата. Така можем да заредим класа "Appointment" от XML файл, който съдържа данните за конкретна среща.JAXB е Java Architecture for XML Binding и предоставя възможността да се сериализират и десериализират Java обекти в XML формат, като автоматично генерира кода за сериализация и десериализация. В случая на класа "Appointment", анотациите "@XmlRootElement" и "@XmlElement" казват на JAXB как да генерира XML файл, който съдържа данните на обекта.

**Реализация на MyCalendar класа:**

Клас MyCalendar има списък от обекти от клас Appointment, представящи срещи в календара.Класът има няколко метода за манипулиране на списъка от срещи,включително добавяне, премахване и извеждане на срещи според определена дата.Той също така има методи за извеждане на всички записи в календара и търсене на запис по дата и начален час.Накрая, класът има метод за изчистване на списъка от срещи в календара.Анотацията @XmlRootElement се използва за маркиране на класа, който ще бъде сериализиран/десериализиран в XML формат. Това означава, че обектите от този клас могат да бъдат записани в XML файлове или представени като XML данни.Анотацията @XmlElement се използва за маркиране на полета на класа, които трябва да бъдат сериализирани/десериализирани като XML елементи. В този случай, тя се използва за маркиране на списъка от срещи (appointments) като XML елемент с име "Appointment".

public void addAppointment(String date, String startTime, String endTime, String name, String note)

-Добавя среща в списъка.

public void removeAppointment(String date, String startTime, String endTime)

-Премахва среща по дадени дата, начален час, краен час.

public void displayAppointmentsByDate(String date)

-Изписва среща по дадена дата. Ако няма такава се изписва съответно съобщение.

public void displayAll()

-Изписва всички срещи в списъка на календара.

public Appointment getAppointmentByDateAndStartTime(String date, String startTime)

-Връща среща по дадени дата и начален час.

public void clear()

-Изчиства всички срещи от календара, като задава стойност null на списъка с срещи.

**Реализация на XMLHelper класа:**

Това е помощен клас за четене и запис на обекти от тип MyCalendar в XML файлове.

Методът open() отваря файл и ако файлът не съществува, създава го и записва празен XML елемент, който ще съдържа календара.

Методът close() затваря файлът, като в случаят единствено е нужно да нулира променливата file.

Методът read() чете съдържанието на файла и го десериализира в обект от класа MyCalendar, използвайки JAXB.

Методът save() сериализира обекта от класа MyCalendar в XML формат и го записва в отворения файл.

Методът saveAs() записва обекта от тип MyCalendar в нов файл, който се задава в параметъра filePath.

Ако файлът не съществува, той се създава. Ако файлът вече съществува, старото съдържание се замества с новото.

public void open(String filePath)

-Методът open приема като параметър пътя до XML файл, който ще се отвори. Първоначално се опитва да се създаде нов файл на посочения път, ако такъв не съществува. Ако не успее да създаде файла, хвърля изключение от тип IOException.

В случай на успешно създаване на файла, се записва в него празен XML съдържание, създава се FileOutputStream и се записва във файла.

public void close()

-Методът close затваря отворения файл, като присвоява на полето file стойност null.

public MyCalendar read()

-Методът read прочита XML файла, като използва JAXB, за да превърне XML файла в обект от тип MyCalendar, който се връща като резултат. Aко не успее да прочете файла, хвърля изключение от тип JAXBException.

public void save(MyCalendar object)

-Методът save запазва обект от тип MyCalendar като XML файл във вече отворения файл. Използва JAXB за маршализацията на обекта към XML. Ако не успее да запише файла, хвърля изключение от тип IOException.

public void saveAs(MyCalendar object, String filePath)

-Методът saveAs запазва обект от тип MyCalendar като XML файл на ново място. Приема като параметър пътя до новия файл, който трябва да бъде създаден.Следва същата процедура като в метода save, но създава нов файл на посочения път и записва в него обекта от тип MyCalendar.Ако не успее да запише файла, хвърля изключение от тип IOException.

**Реализация на CalendarManager класа:**

Този клас представлява основата на приложението. Той инициализира няколко константи. В началото на метода се инициализират два обекта - MyCalendar и Scanner.MyCalendar е обект, който представлява календар и съдържа списък от срещи.Този обект е празен в началото, тъй като файлът, който го съдържа, не е отворен.Scanner се използва за четене на входния поток.Конструкторът е празен и няма параметри.По-голямата част от функционалността се извършва в метода start(), който приема няколко възможни изключения (JAXBException, IOException, CustomException, ParseException).Въвежда се цикъл, който чете входа от потребителя и извиква съответните методи за обработка на командите.Извикването на метода printDescription() изписва информация за приложението и неговата употреба.

Командите, които програмата приема, са:

open() - Зарежда съдържанието на даден файл. Ако такъв не съществува се създава нов с празно съдържание.

close() - Затваря календарния файл, който е бил отворен от потребителя.

save() - Запазва текущото състояние на календара в отворения файл.

saveAs() - Запазва текущото състояние на календара в нов файл, като потребителят избира мястото, където да се запази файла.

book() - Запазва час за среща с име <name> и коментар <note> на дата <date> с начален час <starttime> и краен час <endtime>.

unbook() - Отменя час за среща на дата <date> с начален час <starttime> и краен час <endtime>.

agenda() - Извежда хронологичен списък с всички ангажименти за деня.

displayAll() - Показва всички срещи в текущият календар.

change() - Променя детайли за дадена резервация. Потребителят избира дата и часове на резервацията,

която иска да промени, и въвежда новите данни за резервацията.

find() - Търсене на среща: извеждат се данните за всички срещи, в чието име или бележка се съдържа низът <string>.

holiday() - Датата <date> се отбелязва като неработна.

busydays() - Извеждане на статистика за натовареност: по дадени начална дата <from>

и крайна дата <to> се извежда списък с дните от седмицата.

findslot() - Намиране на свободно място за среща по дадена дата.

clear() - Изчиства всички резервации от текущият календар.

help() - Извежда кратка информация за поддържаните от програмата команди.

exit() - Излиза от програмата.

***Подробно описание на различните методи:***

open():

Методът проверява дали вече има отворен файл. Ако има, той просто предупреждава потребителя да избере друга опция. Ако няма отворен файл, методът изисква от потребителя да въведе пътя към желания файл. След това методът използва XMLHelper, за да отвори и прочете файла на зададения път и да зареди информацията в myCalendar обекта. Накрая методът извежда съобщение за успешното отваряне на файла.

close():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.След като файла е затворен, се отпечатва съобщение за успешно затваряне.

save():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.

Извиква метода save на xmlHelper, който записва информацията във файла, който е бил отворен. Накрая се изписва съобщение за успешното запазване на данните.

saveAs():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.Извиква метода saveAs на xmlHelper, който записва информацията във файл по зададен път. Накрая се изписва съобщение за успешното запазване на данните.

book():

Позволява на потребителя да резервира нова среща в календара.Методът изисква от потребителя да въведе дата, начален час, краен час, име на срещата и бележка. Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.При въвеждането на данни, методът проверява дали въведените данни са валидни, като използва няколко валидационни метода,които проверяват дали датата е в интервала от 01-01-2023 до 01-01-2030, дали часовете са в интервала от 00:00 до 23:59 и дали крайният час е след началният час. Ако потребителят въведе валидни данни, методът добавя новата среща в календара и извежда съобщение за успешно добавена среща. Ако някое от въведените данни е невалидно, методът хвърля CustomException със съобщението за грешка.

unbook():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.След това се изисква от потребителя да въведе дата (във формат YYYY-MM-DD), начален час и краен час на срещата, която желае да отмени.След като потребителят въведе датата и часовете, методът извиква метода removeAppointment() на обекта myCalendar.Този метод премахва срещата с указаната дата, начален час и краен час от календара.Накрая, методът извежда съобщение, че срещата е успешно отменена. Ако не съществува среща с такава дата, начален час и краен час в календара, методът няма да направи нищо.

agenda():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.Изисква от потребителя да въведе дата във формат "YYYY-MM-DD" и след това извиква метода displayAppointmentsByDate(date) на обекта myCalendar. Ако няма записи за тази дата, се извежда подходящо съобщение

displayAll():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.Извиква метода displayAll() на обекта myCalendar, който изписва всички срещи в текущият календар.

change():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.После се посочва датата и началния час на събитието, което искаме да променим. След това се посочва опцията, която искаме да променим (дата, начален час, краен час, име или бележка) и новата стойност за тази опция. Ако опцията е валидна, методът извършва съответната промяна в обекта за събитието и извежда съобщение за успешна промяна. Ако даденото събитие не е намерено, методът извежда съобщение, че събитието не е намерено. Ако посочената опция за промяна е невалидна, методът извежда съобщение за грешка.

find():

Oтново се проверява дали текущият файл е отворен и хвърля се грешка ако не е.След това се изисква от потребителя да въведе бележка (note), която търсим. След това се oбхождат всички записи в календара и сепроверява дали името или бележката на записа съдържат търсената дума. Ако да, то записът се извежда на екрана.

holiday():

Oтново се проверява дали текущият файл е отворен и хвърля се грешка ако не е.Намира се даденият ден и се отбелязва като почивен, задавайки стойност true на isHoliday.

busydays():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.След това иска от потребителя да въведе начална и крайна дата. След това той филтрира всички записи от календара, които са между тези дати и ги сортира по дата във възходящ ред.Накрая, той извежда всички намерени записи в конзолата.

findslot():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.Търси свободно време за нова среща, като първо се въвежда дата "fromDate" и продължителност на срещата в часове.След това, методът обхожда списъка с всички събития в календара, проверява дали има свободно време между тях и извежда събитията, за които това е изпълнено. Проверката за свободно време включва деня да не е празник, ограничения за началният и крайният час, да има свободно време.

clear():

Първоначално се проверява дали текущият файл е отворен. Ако не е отворен, хвърля се грешка.След това просто за изчиства целият текущ календар.

help():

Извиква помощна информация за командите

Помощни кратки функции, които имат точно едно предназначение, което е достатъчно ясно и се разбира от името им. :

checkIfMinTimeBeforeMaxTime(String minTime, String maxTime)

checkForValidHoursAndMinutes(String timeToBeChecked)

convertToLocalTime(String timeToBeChecked)

checkForValidData(String dateToBeChecked)

convertToLocalDate(String date)

isCurrentFileOpened()

printDescription()

1. **Заключение**

Като обобщение, това е добър пример за приложение на практика знанията от курсът ООП, създавайки приложението „личен календар“. Многото различни части от проекта ми помогнаха да разбера и усъвършенствам моите практични умения, като например работата с XML файлове.

Биха могли да се добавят в бъдеще някои подобрения към приложението, което ще го направи по-удобно и лесно за употреба. Като пример за подобрение аз съм добавил към проекта командите displayAll() и clear(). Може да включим към решението употреба и на други абстракции от ООП. Бихме могли да добавим и работа с друг вид файлове, както и още някои методи към календара за по-улеснено търсене на дадена среща.

Източници: <https://docs.oracle.com/javase/8/docs/api/javax/xml/bind/JAXB.html> , <https://docs.oracle.com/javase/8/docs/api/?java/io/File.html>