Міністерство освіти і науки України

Національний технічний університет України «КПІ ім. Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

**Звіт**

З лабораторної роботи №3

з курсу

«Основи розробки програмного забезпечення на платформі Microsoft.NET»

|  |
| --- |
| ІП-01 Галько Міла |
| (шифр, ПІБ) |

**Виконав студент**

|  |
| --- |
| Ліщук К.І. |
| (ПІБ) |

**Перевірила**

Київ 2022

**Комп‘ютерний практикум № 3. Шаблони проектування. Породжуючі шаблони**

**Мета:**

- ознайомитися з основними шаблонами проектування, навчитися застосовувати їх при проектуванні і розробці ПЗ.

**Постановка задачі комп‘ютерного практикуму № 3**

При виконанні комп‘ютерного практикуму необхідно виконати наступні дії:

1. Вивчити породжуючи патерни. Знати загальну характеристику та призначення кожного з них, особливості реалізації кожного з породжуючих патернів та випадки їх застосування.
2. Реалізувати задачу згідно варіанту №1. Розробити інтерфейси та класи з застосування одного або декількох патернів. Повністю реалізувати методи, пов‘язані з реалізацією обраного патерну.
3. Повністю описати архітектуру проекту (призначення методів та класів), особливості реалізації обраного патерну. Для кожного патерну необхідно вказати основні класи та їх призначення,
4. Навести UML-діаграму класів

**Варіант №1**

Реалізувати задачу «Електронний документообіг». Оброблюються документи декількох типів (наприклад, лист, службова записка, розпорядження, наказ, заявка на ресурс тощо). Кожен документ містить номер, дату та інформацію про документ. Крім того, в листи можуть бути як вхідними, так і вихідними та містять кореспондента, від кого надійшов лист або надсилається. Накази містять підрозділ, строк виконання та відповідального виконавця. Розпорядження – тільки підрозділ та строк виконання. Заявки на ресурси містять співробітника, котрому необхідно забезпечити доступ до ресурсів, перелік ресурсів.

**Програмний код:**

Посилання на github: <https://github.com/MilaHalko/C4_.NET/tree/Lab2/Lab2_XML/Lab2_XML>

Також код доданий в кінці документу.

**Опис проблематики та архітектури проекту:**

Оскільки стикаємося з проблемою, що заздалегідь невідомий кінцевий продукт; та за формулюванням задачі можуть бути і інші варіанти документів, що можуть бути додані до системи пізніше, то загалом зрозуміло, що код повинен бути більш універсальним для подальшого додання нових класів.

Отже, фабричний метод стане найкращим способом реалізації. Даний патерн надає змогу зробити код універсальним, завдяки тому, що немає прив’язки до конкретних класів. Для реалізації потребуємо загальний абстрактний клас – DocCreator. Від ного будуть унаслідуватись усі інші Cretor-и для створення кожного з файлів (спадкоємці Doc).

Єдина проблема – файли мають різні атрибути. В даній роботі для передачі усіх необхідних значень для створення файлів був створений абстрактний клас DocArgs та його спадкоємці (роль сховища). Об’єкт класу DocArgs буде переданий у конструктор спадкоємця DocCreator-а для подальшого вилучення значень. В залежності від переданого спадкоємця DocArgs буде створений певний спадкоємець DocCreator.

**Doc** – абстрактний клас узагальнюючий поняття «Документ». Має загальні атрибути: id, date, info.

**Memo, Letter, Order, Decree, ResourceRequest** – спадкоємці класу Doc, кожен є представником певного виду документу із своїми відповідними атрибутами.

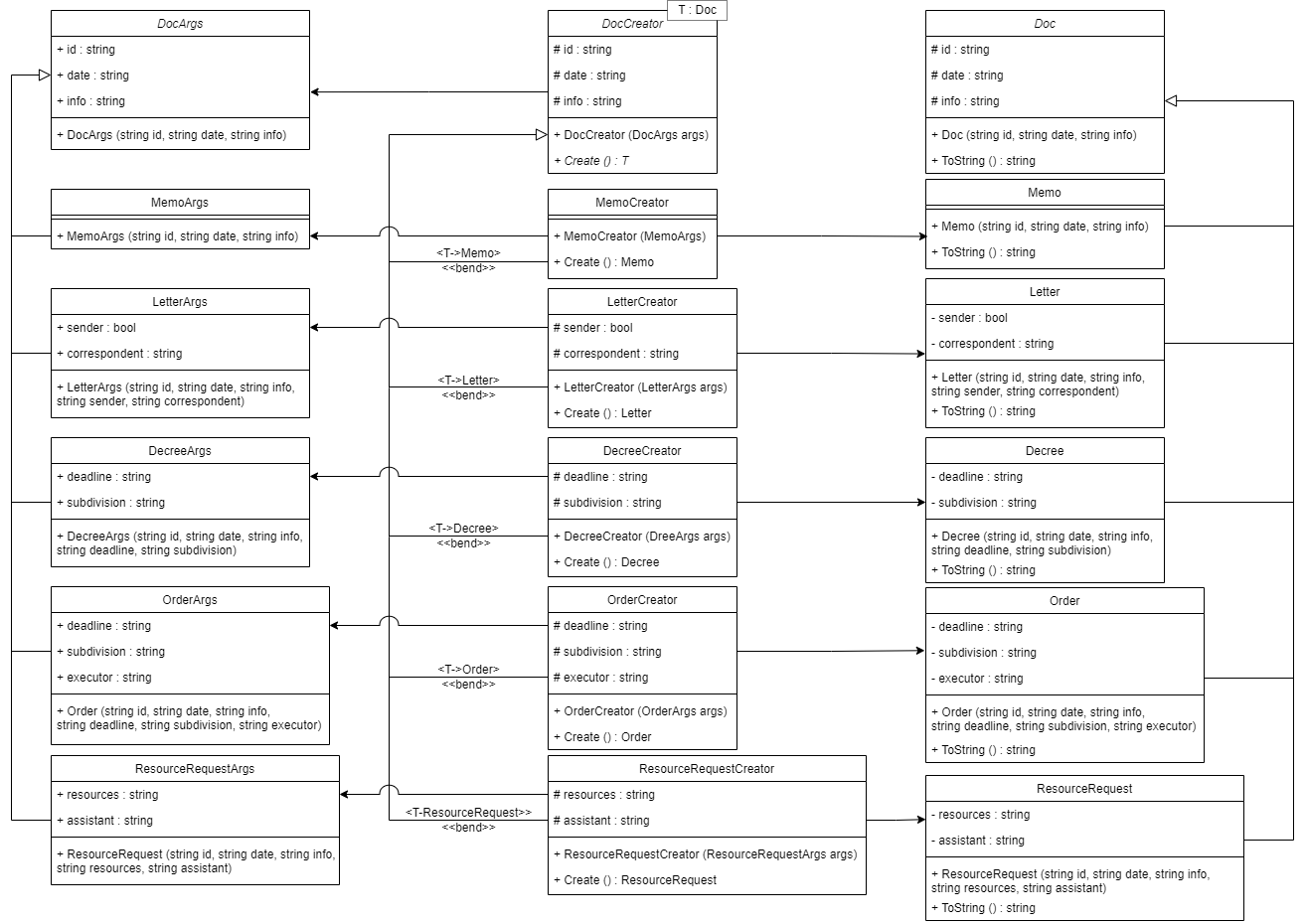
**DocCreator** – абстарактний клас, відповідає за повернення новостворених файлів через абстрактний метод Create().

**MemoCreator, LetterCreator, OrderCreator, DecreeCreator, ResourceRequestCreator** – конкретні створювачі; спадкоємці DocCreator; реалізують метод Create(); мають атрибути відповідно до файлу для зберігання значень після їх безпосереднього створення.

**T Create ()** – абстрактний метод класу DocCreator, що створює та повертає Т – один із спадкоємців класу Doc ().

**DocArgs** – абстрактний клас, відіграє роль сховища значень для створення файлу.

**MemoArgs, LetterArgs, OrderArgs, DecreeArgs, ResourceRequestArgs** – спадкоємці DocArgs; мають атрибути – параметри для майбутнього створення файлу; є необхідним параметром при створенні спадкоємця DocCreator, оскільки значення його атрибутів далі будуть зберігатися у даному класі.



**Маємо зв’язки типу:**

Generalization – звичайне наслідування класів;

Association – класи потребують об’єкти інших класів у методах лише для зібрання значень, але сама зміна переданих об’єктів не впливає на подальшу зміну залежного об’єкту.