Завдання 4. Клас Image. Частина 2

Версія 16 квітня 2023 р.

В цьому завданні ми будемо розширювати клас `Image`, дозволяючи йому працювати з різними форматами файлів за допомогою паттерну `Strategy`. Стратегія дозволяє нам мати екземпляр нашого класу, який може "перемикатися" між роботою в різних режимах, в нашому випадку, мова буде йти про різні формати зображень. Це завдання складається з двох під-завдань.

- Інтерфейс стратегії. Підготуйте ваш існуючий клас `Image` для роботи в режимі стратегії:
  - Створіть інтерфейс (чистий абстрактний клас) з обов'язковими спільними методами для всіх конкретних стратегій.
  - Створіть клас, який втілює цю стратегію і перенесіть туди функції, написані раніше, для зчитування та запису PGMA файлів.
  - Інтегруйте цю стратегію в клас `Image` через вказівник на інтерфейс.
- Додаткові стратегії. Звичайно, використання цього паттерну має сенс за наявності декількох взаємозамінних стратегій. Створіть другу стратегію, для роботи з кольоровими файлами в PPM форматі. Довідку про PPM можна знайти, зокрема, тут. За великим рахунком, від PGMA він відрізняється тільки тим, що замість одного числа, що характеризує відтінок монохромного пікселя, їх там три – для червоного, зеленого та синього, відповідно.[1] Цей новий клас має мати той самий функціонал, як і старий, що диктується інтерфейсом. Вибір стратегії може здійснюватись вручну за допомогою setter-методів, або автоматично зважаючи на розширення файлу на моменті створення екземпляру `Image` (виклик конструктора), або використання одного з його методів (зокрема зчитування).

[1] Цю трійку має сенс зберігати разом у вкладеній структурі `Image::Pixel`.

Додатковий бал

- В окремому файлі, спробуйте переписати клас `Image`, так, щоб він використовував поліморфізм напряму, замість стратегії. Тобто, мав похідні класи `PGMAImage` та `PPMImage`. В чому різниця між цими підходами коли один може буде кращим за інший?

- Напишіть автоматизовані тести, для тестування абстрактної стратегії, які зможуть працювати з будь-якою конкретною стратегією.

- * Імплементуйте ще одну стратегію, яка дозволить нам працювати з PNG файлами. Для цього не варто намагатись розробити своє рішення. Натомість, треба використовувати наявні поширені та відтестовані бібліотеки. А саме `libpng`[2], та її обгорткою на C++ `png++`. В середі Ubuntu Linux файли для роботи з PNG зображеннями знаходяться в пакеті `libpng++`, що можна встановити наступним чином:

  ```
  sudo apt install libpng++-dev
  ```

  Встановлення залежностей та підключення їх через CMake залишається частиною і важливим педагогічним аспектом цього завдання.

[2]Зауважте, в неї є залежність, інша бібліотека `zlib`.