Memento(Знімок)

Шаблон поведінки

Мотивація Знімку

- Як зберегти стан об’єкта, наприклад, щоб у деякий момент розробки зробити відкат(undo) і при цьому не порушити інкапсуляцію?

private: не скопіювати
public: небезпечно

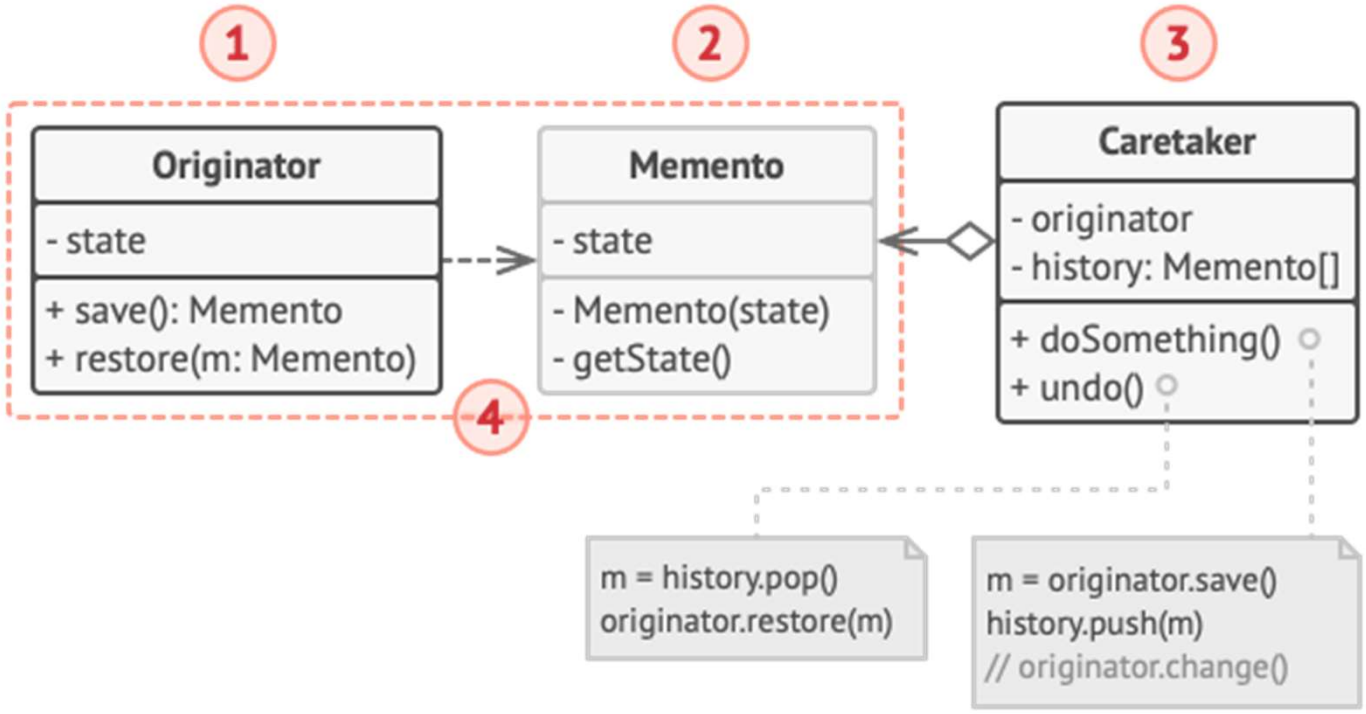
Що пропонує Знімок

- Створити клас Знімок, в якому тримати копію всіх полів.
- Довірити створення знімку об’єкта самому об’єкту(Originator або Творець). Додати методи:

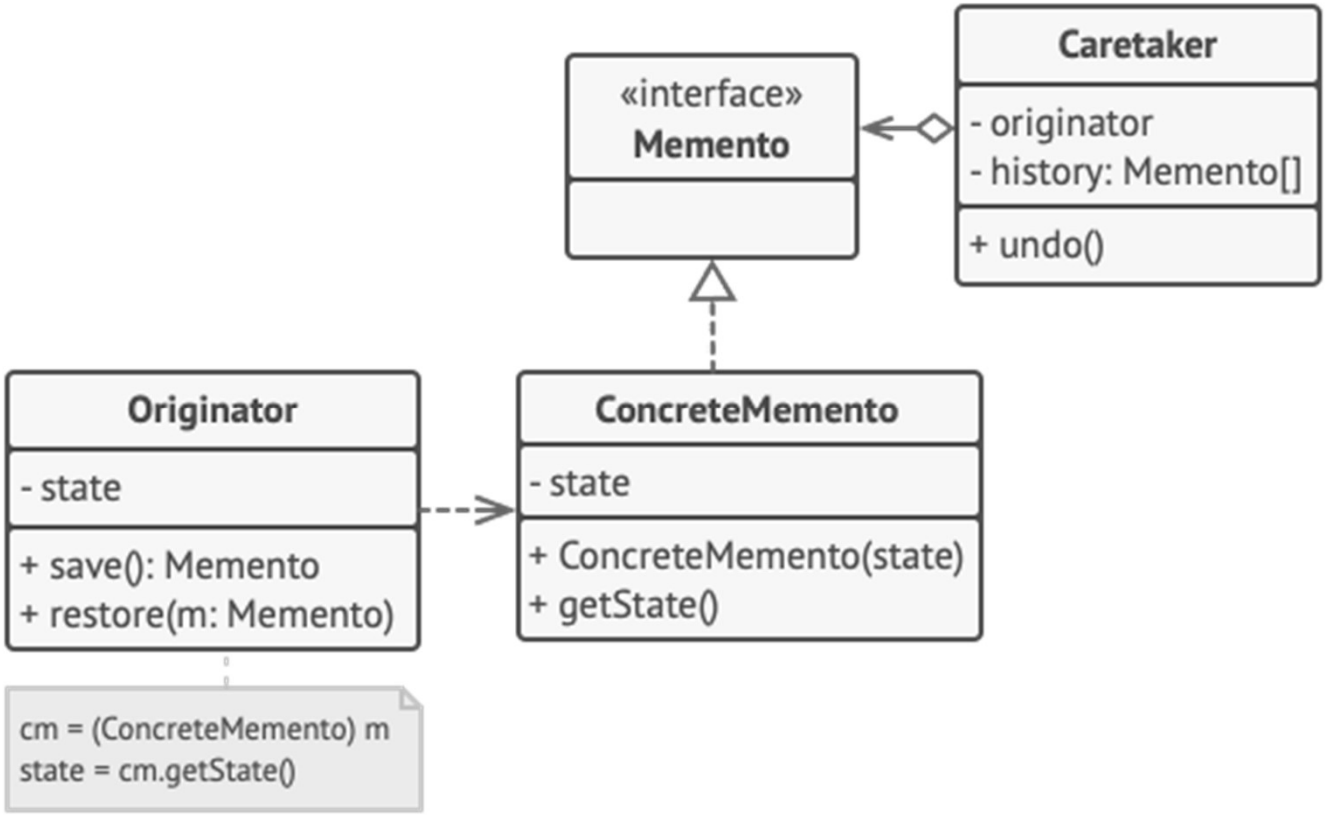
1. Збереження стану Memento Save()
2. Відкату до стану void Restore(Memento m)

- Створити клас Опікуна(Caretaker), у якому будуть зберігатися знімки. Також Опікун може попросити творця відновити свій стан, передавши йому відповідний знімок.

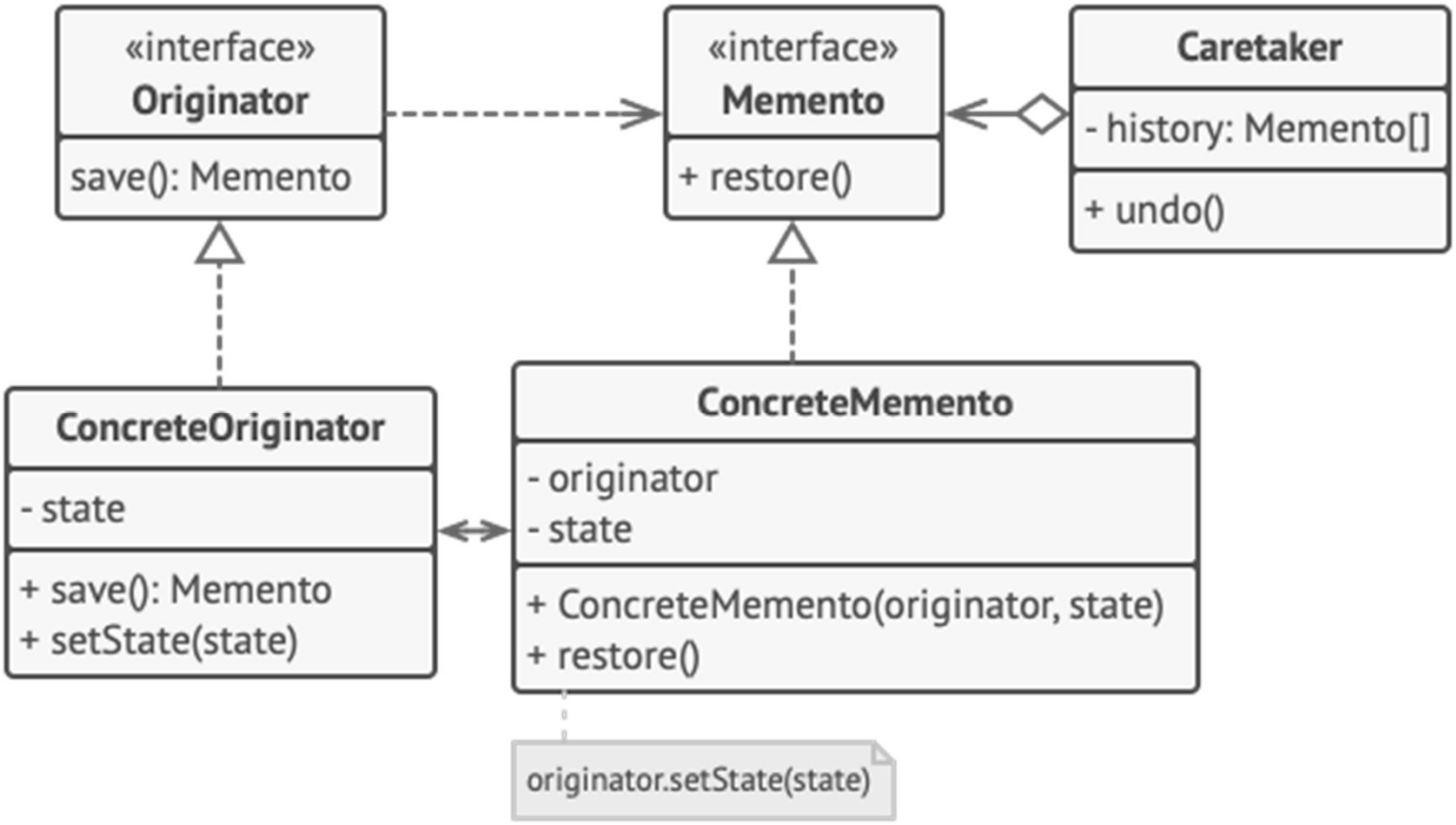
Класичний Знімок



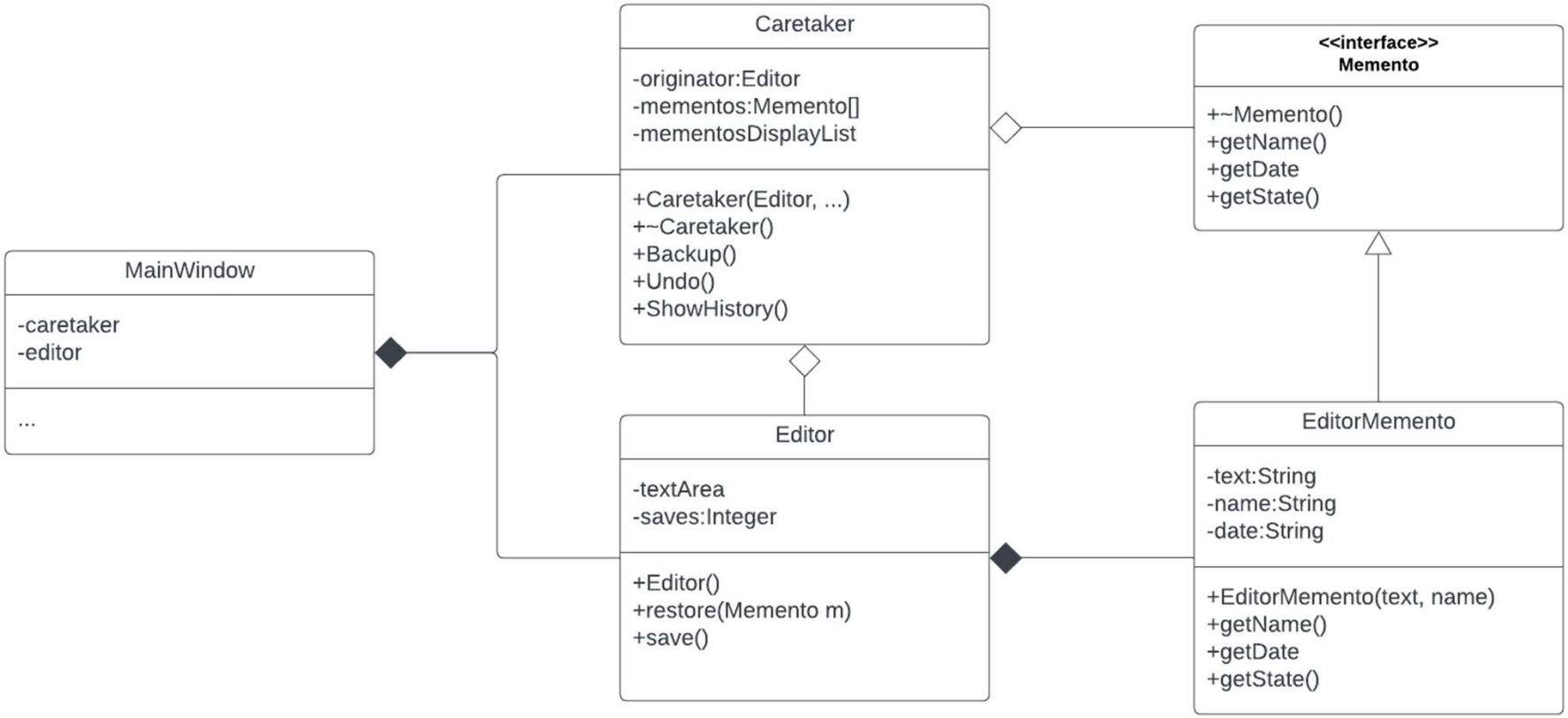
Знімок з проміжним порожнім інтерфейсом



Знімок з підвищеним захистом



Структура проєкту



Коли застосовувати Знімок?

- Коли потрібно зберігати стан об’єкта для подальшого відновлення.
- Коли не можемо зберегти стан без порушення інкапсуляції.

Зв’язок з іншими патернами

- Команду та Знімок можна використовувати спільно для реалізації скасування операцій.
- Знімок можна використовувати разом з Ітератором, щоб зберегти поточний стан обходу структури даних та повернутися до нього в майбутньому, якщо буде потрібно.
- Знімок іноді можна замінити Прототипом, якщо об’єкт, чий стан потрібно зберігати в історії, досить простий, не має посилань на зовнішні ресурси або їх можна легко відновити.

Плюси Знімку

- Не порушує інкапсуляцію Творця
- Спрощує структуру Творця: йому не потрібно зберігати історію свого стану.

Недоліки Знімку

- Вимагає багато пам’яті, якщо клієнти дуже часто створюють знімки.
- Може спричинити додаткові витрати пам’яті, якщо об’єкти, що зберігають історію, не звільняють ресурси, зайняті застарілими знімками.
- В деяких мовах (наприклад, PHP, Python, JavaScript) складно гарантувати, щоб лише вихідний об’єкт мав доступ до стану знімка.