Prototype

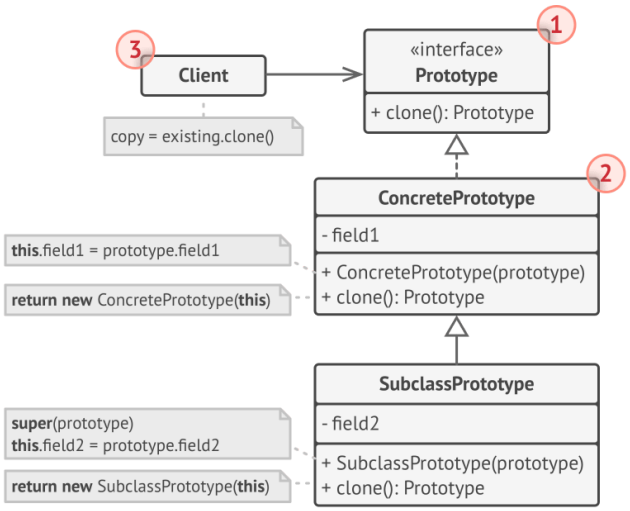
Prototype è un modello di progettazione creazionale che consente di copiare oggetti esistenti senza rendere il codice dipendente dalle loro classi.

Il pattern prototype delega il processo di clonazione agli oggetti effettivi che vengono clonati. Il pattern dichiara un'interfaccia comune per tutti gli oggetti che supportano la clonazione. Questa interfaccia consente di clonare un oggetto senza dover accoppiare il codice alla classe di quell'oggetto. Solitamente, un'interfaccia di questo tipo contiene un solo clone metodo.

L'implementazione del metodo clone è molto simile in tutte le classi. Il metodo crea un oggetto della classe corrente e trasferisci tutti i valori dei campi del vecchio oggetto in quello nuovo (è possibile copiare anche campi privati).

Un oggetto che supporta la clonazione è chiamato prototipo.

Struttura



1. L'interfaccia Prototype dichiara i metodi di clonazione. Quasi sempre si tratta di un singolo metodo clone();
2. La classe concretePrototype implementa il metodo di clonazione. Oltre a copiare i dati dell'oggetto originale nel clone, questo metodo può anche

gestire alcuni casi limite del processo di clonazione (clonazione di oggetti collegati, risoluzione di dipendenze ricorsive ecc...);

3. Il cliente può produrre una copia di qualsiasi oggetto che segua l'interfaccia del prototipo;

Applicabilità

- Usare tale pattern quando il codice non deve dipendere dalle classi concrete degli oggetti che bisogna copiare;
- Usare tale pattern quando si vuole ridurre il numero di sottoclassi che differiscono solo nel modo in cui inizializzano i rispettivi oggetti;

Implementazione

- Creare l'interfaccia prototipo e dichiarare il metodo clone al suo interno;
- Una classe prototipo deve definire il costruttore alternativo che accetta un oggetto di quella classe come argomento.Il costruttore deve copiare i valori di tutti i campi definiti nella classe dall'oggetto passato alla nuova istanza creata. Se si modifica una sottoclasse, è necessario chiamare il costruttore padre per consentire alla superclasse di gestire la clonazione dei suoi campi privati;
- Il metodo di clonazione consiste solitamente in una sola riga: l'esecuzione di un `new` operatore con la versione prototipica del costruttore;
- Facoltativamente, creare un registro centralizzato dei prototipi per archiviare un catalogo dei prototipi utilizzati di frequente.

Vantaggi

- E' possibile clonare oggetti senza accorparli alle loro classi concrete;
- E' possibile eliminare il codice di inizializzazione ripetuto a favore della clonazione di prototipi predefiniti;
- E' possibile realizzare oggetti complessi in modo più pratico;

- Si ottiene un’alternativa all’ereditarietà quando di gestiscono preimpostazioni di configurazione per oggetti complessi.

Contro

- Clonare oggetti complessi con riferimenti circolari può essere molto complicato.

Relazione con altri modelli

- Molti progetti iniziano usando il Factory Method e si evolvono verso Abstract Factory, Prototype o Builder;
- Le classi Abstract Factory sono spesse basate su un set di Factory Method, ma è anche possibile usare Prototype per comporre i metodi su queste classi;
- Prototype può essere utile quando è necessario salvare copie dei comandi nella cronologia;
- I progetti che fanno ampio uso di Composite e Decorator possono spesso trarre vantaggio dall’uso di Prototype (clonare struttre complesse anzichè ricrearle da capo);
- A volte Prototype può essere un’alternativa più semplice a Memento. Questa soluzione funziona se l’oggetto, di cui si desidera memorizzare lo stato nella cronologia, è abbastanza semplice e non ha collegamenti a risorse esterne, oppure se i collegamenti sono facili da ristabilire;
- Abstract Factory, Builder e Prototype possono essere tutti implementati come Singleton.