

# Singleton

- Intento
  - Assicurare che una classe abbia una solo istanza e fornire un punto di accesso globale all'istanza
- Motivazione
  - Alcune classi dovrebbero avere esattamente una istanza (uno spooler di stampa, un file system, un window manager)
  - Una variabile globale rende un oggetto accessibile ma non proibisce di istanziare più oggetti
  - La classe dovrebbe essere responsabile di tener traccia del suo unico punto di accesso

Ing. E. Tramontana - Pattern Singleton - 23-Mag-06 1

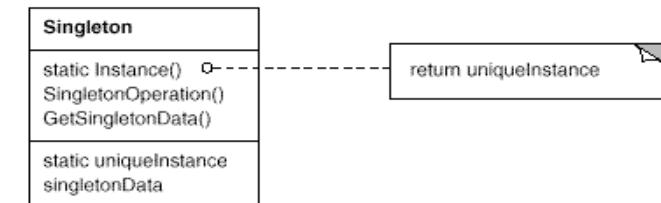
# Singleton

- Conseguenze
  - La classe ha pieno controllo di come e quando i client accedono
  - Evita che esistano variabili globali che tengono la sola istanza condivisa
  - Permette di controllare il numero di istanze create in un programma, facilmente ed in un solo punto
  - La soluzione è più flessibile rispetto a quella di usare static per tutte le operazioni e le variabili poiché si può cambiare facilmente il numero di istanze consentite
    - L'unico frammento di codice da variare quando si vuol variare il numero di istanze create è quello della classe Singleton, mentre usando static si dovrebbero variare tutte le invocazioni

Ing. E. Tramontana - Pattern Singleton - 23-Mag-06 3

# Singleton

- Soluzione
  - Singleton definisce una operazione `instance()` per la classe (in Java è un metodo static) che ritorna l'unica istanza creata
  - Singleton è responsabile per la creazione dell'istanza, il suo costruttore è privato, quindi la creazione con `new` è inaccessibile ad altre classi



Ing. E. Tramontana - Pattern Singleton - 23-Mag-06 2

# Codice Singleton

```

// Singleton design pattern che contiene
// un valore intero

class Singleton {
    // l'unica istanza è riferita da s
    private static Singleton
        s = new Singleton(47);
    private int i;

    private Singleton(int x) {
        i = x;
    }

    public static Singleton Instance() {
        return s;
    }

    public int getValue() {
        return i;
    }

    public void setValue(int x) {
        i = x;
    }
}

// Uso della classe Singleton

public class testSing {
    public static void main(String[] args) {
        // richiede una istanza
        Singleton s = Singleton.Instance();
        System.out.println("s contiene: " +
                           s.getValue());

        // richiede una nuova istanza
        Singleton s2 = Singleton.Instance();
        s2.setValue(9);

        int v = s.getValue();
        // quanto vale v?
        System.out.println("s contiene: "+v);

        // Si ha un errore a compile-time
        // Singleton s3 = (Singleton) s2.clone();
        // Singleton s4 = new Singleton(33);
    }
}
  
```

Ing. E. Tramontana - Pattern Singleton - 23-Mag-06 4