Object Calisthenics

Esempio pratico

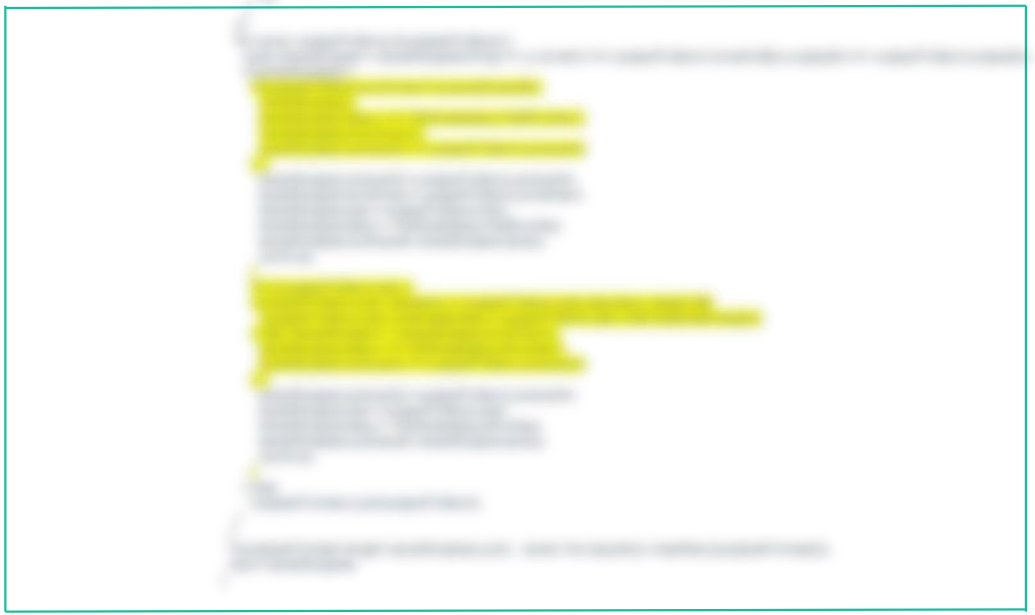
Re quisiti

- Scrivere una funzione che accetta in ingresso una lista di DbModel e li salvi a db
- Un DbModel al suo interno ha un riferimento a 1 owner, 1 riferimento a 1 Subject, uno stato e dei dati (enrollment e plan)
- La funzione deve effettuare un insert se per ogni owner il Subject non esiste
- La funzione deve effettuare un update se l'owner ha già il Subject, ma questo ha uno dei dati differente o se il suo stato è inconsistente. Lo stato deve essere aggiornato di conseguenza.

Refactoring (Object Calisthenics)

- Only one level of indentation per method
- Don’t use the ELSE keyword
- No abbreviations
- Keep all entities small

Creazione di utility



Vantaggi

- Codice più leggibile
- Riutilizzabile
- Modifiche e manutenzione più facili
- Più facile trovare bug

E adesso?

Continuare con gli esercizi:

- **Wrap all primitives and strings**
- **First class collections (wrap all collections)**
- **Only one dot per line**
- **No classes with more than two instance variables**
- **No public getters/setters/properties**
- **All classes must have state**

E adesso?

- Possiamo incapsulare la mappa contenuta in divideModelsByOwner?
- La dichiarazione dbModel.plan.pharmaceutical?.length è comprensibile?
- Nel metodo privato insertOrUpdate possiamo rifattorizzare ulteriormente il ciclo for?

- => Arrivare a un compromesso

Conclusioni

- Apparentemente aumenta la complessità del codice (più funzioni, più classi…)
- Aumenta leggibilità (auto-documentazione)
- Meno difficoltà nelle modifiche
- Riusabilità

Conclusioni

.Anche i nomi sono importanti (auto-documentazione)

.Prendersi il tempo necessario per il refactoring

. => Rifattorizzare anche durante la scrittura del codice!
(rules of three)

Grazie per l’attenzione