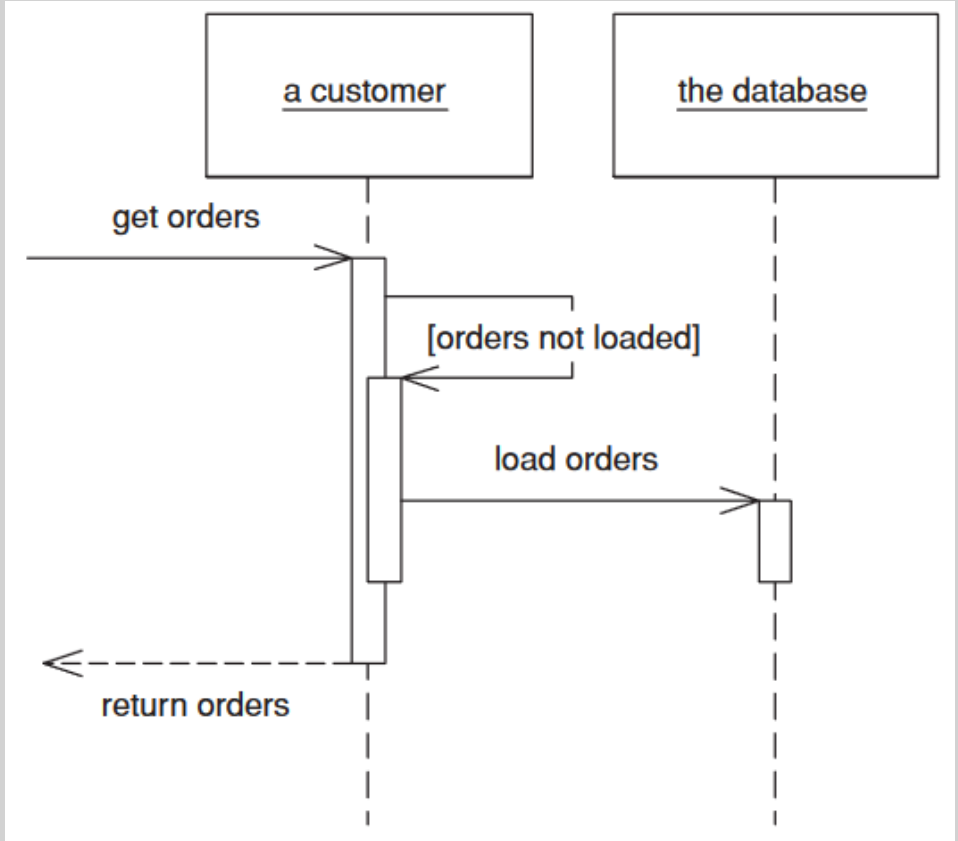
Lazy Load
Architectural Pattern

Francesco Refolli 865955

"An object that doesn’t contain all of the data you need but knows how to get it"
Martin Fowler

- Se possibile si rimanda il caricamento di dati non necessari alla prima occorrenza di utilizzo
- Deve poter ottenere al momento giusto le informazioni che servono.



Vantaggi del Lazy Load

Tempo	• Tempo di caricamento inferiore
Spazio	• Minor uso di memoria • Meno accessi quando non necessario
Prestazioni	• Operazioni lente solo se necessario

Svantaggi del Lazy Load

Single Responsability Principle

- La classe si deve occupare di manipolare i dati e recuperarli

Accoppiamento

- La classe puo' essere accoppiata con le classi che gestiscono il recupero dei dati

Motivi

Si vuole ritardare le operazioni onerose quando i loro frutti non sono immediatamente utili

Si puo' adoperare piu' Lazy Load nella stessa classe per diverse operazioni pesanti

Si vuole ottimizzare l'uso della memoria in Sistemi Applicativi di dimensioni notevoli

Si vuole gestire dall'interno le operazioni di popolamento dei dati

Lazy Load

Applicazioni

Lazy Initialization

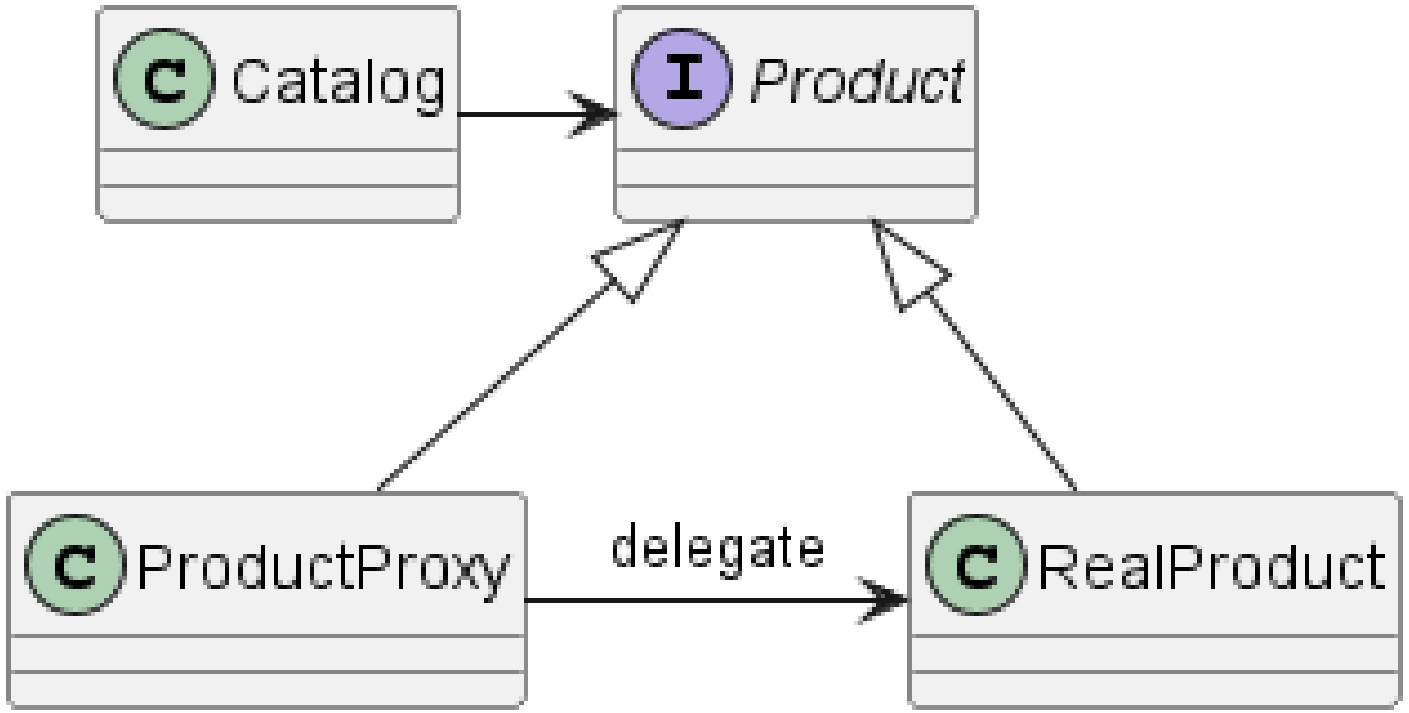
```java
public class Logger {
    private static Logger loggerInstance = null;

    private Logger() {
        /* ... constructor ... */
    }

    public static Logger getLogger() {
        if (loggerInstance == null)
            loggerInstance = new Logger();
        return loggerInstance;
    }
}
```

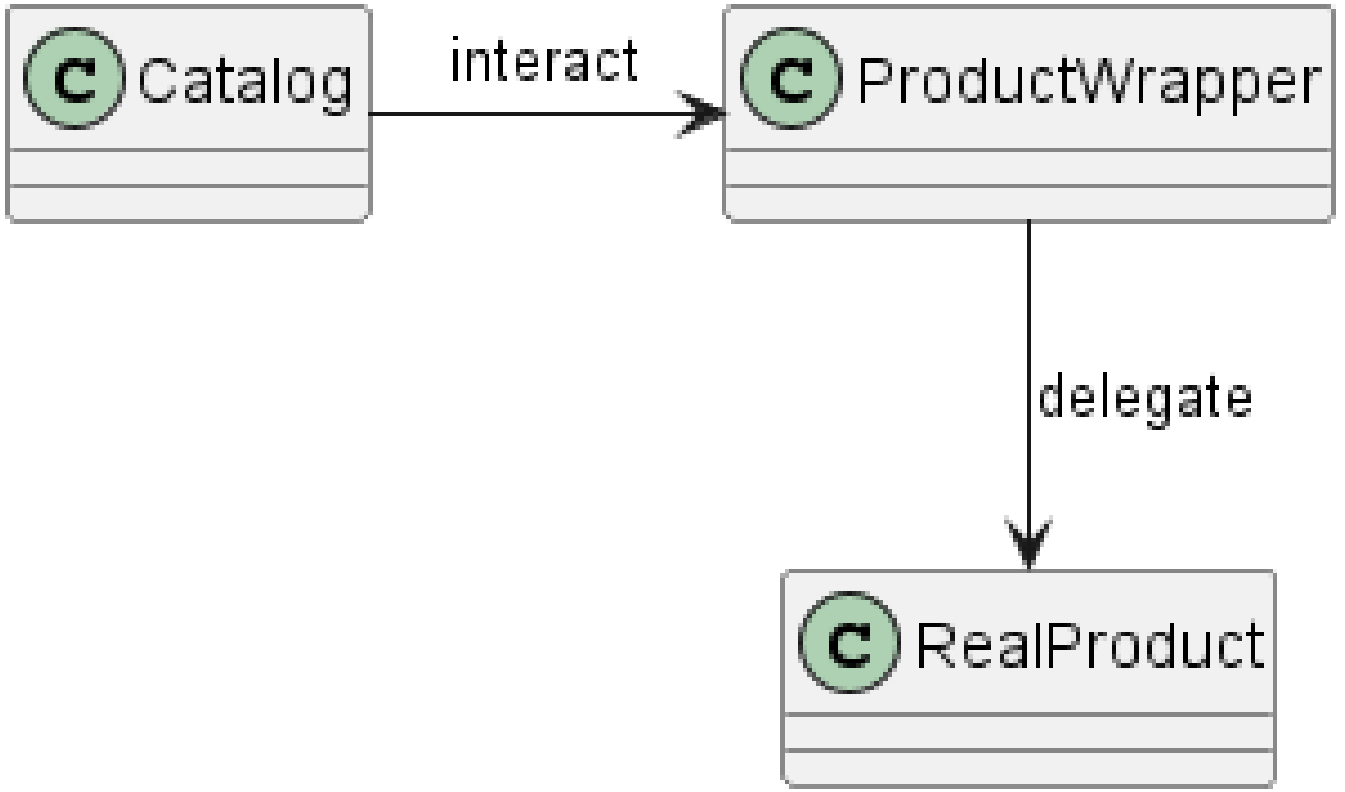
Virtual Proxy

- Astrazione di Product
- Proxy delega a RealProduct
- Proxy provvede all'istanziamento di RealProduct

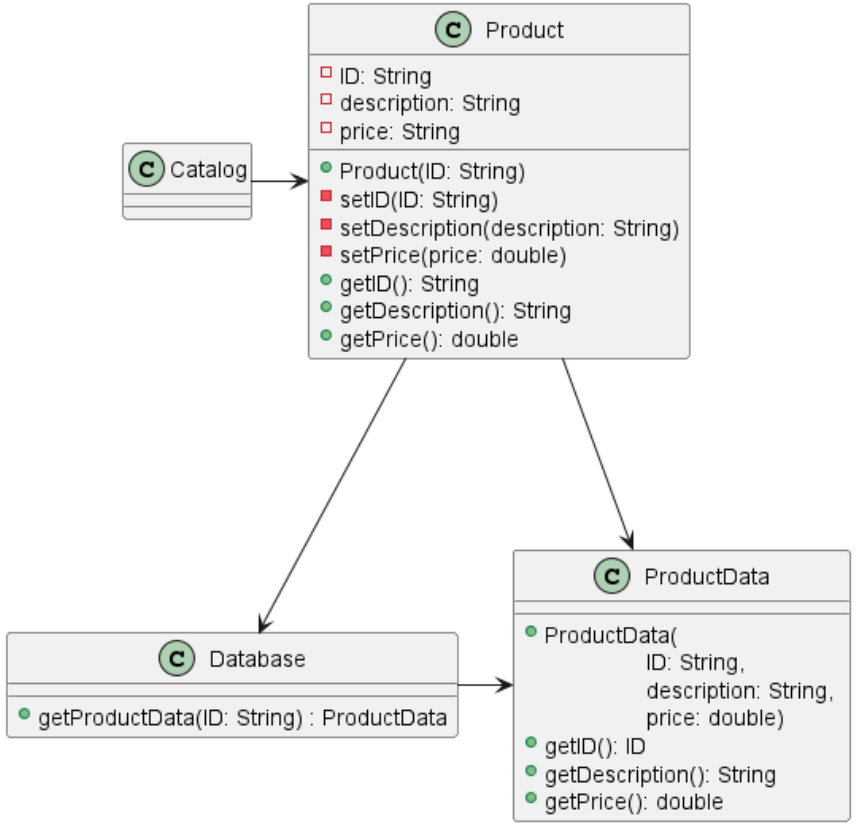


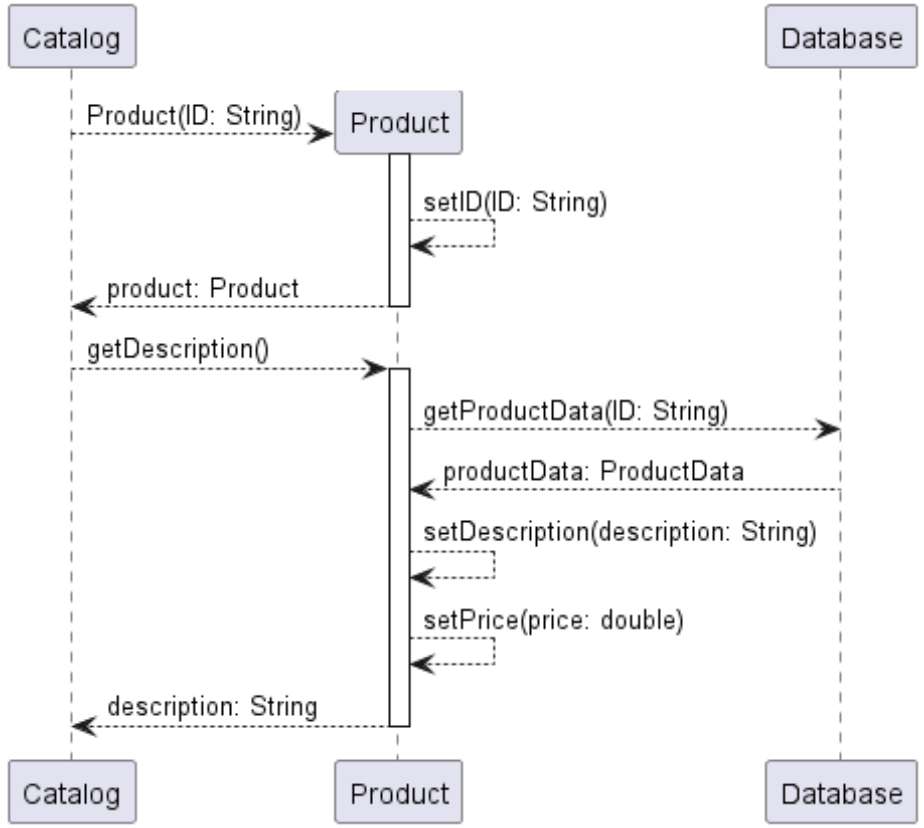
Value Holder

- un ProductWrapper assume tutte le responsabilita' che aveva ProductProxy
- Catalog e' cosciente di stare interagendo con il ProductWrapper



Ghost


Product
ID: String
description: String
price: String
Product(ID: String)
setID(ID: String)
setDescription(description: String)
setPrice(price: double)
getID(): String
getDescription(): String
getPrice(): double
Catalog
Database
getProductData(ID: String) : ProductData
ProductData
ProductData(
ID: String,
description: String,
price: double)
getID(): ID
getDescription(): String
getPrice(): double


Catalog
Database
Product(ID: String)
Product
setID(ID: String)
product: Product
getDescription()
getProductData(ID: String)
productData: ProductData
setDescription(description: String)
setPrice(price: double)
description: String
Catalog
Product
Database