Università degli Studi di Padova


Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica


# CrownSouls

## CrownSouls

Progetto di Programmazione ad Oggetti


Enrico Buratto
1142644


---

Anno Accademico 2019-2020

# Indice

# 1 Introduzione

## 1.1 Abstract

Si vuole realizzare un programma per la gestione di un inventario giocatore per il gioco *Action RPG* "Dark Souls". Il giocatore possiede svariati elementi di diversi tipi; questi tipi possono essere:

- **Armature**: oggetti indossati dal giocatore per aumentare la resistenza al danno;
- **Armi**: oggetti utilizzati dal giocatore per infliggere danno ai nemici;
- **Anelli**: oggetti utili al giocatore per l'aumento delle proprie statistiche;
- **Scudi**: oggetti utilizzati dal giocatore per ridurre il danno dai colpi nemici;
- **Guanti**: oggetti utilizzati sia come armatura, poiché aumentano la resistenza al danno, sia come arma, poiché permettono di infliggere danno;
- **Scudi d'attacco**: oggetti utilizzati sia come scudo, poiché riducono il danno dai colpi nemici, sia come arma, poiché permettono di infliggere danno.

Un inventario è composto da un insieme di oggetti appartenenti alle diverse tipologie; ogni oggetto presente nell'inventario possiede delle caratteristiche tecniche proprie della categoria di appartenenza.

## 1.2 Funzionalità

Per facilitare la visualizzazione degli oggetti dell'inventario, questi sono suddivisi all'interno del programma in quattro diverse schede; ogni scheda rappresenta una sottosezione dell'inventario, e mostra al suo interno solo gli elementi appartenenti alla categoria indicata dal titolo. Per la gestione degli oggetti dell'inventario sono presenti le seguenti funzionalità:

- Caricamento ed esportazione dell'intero inventario da e su file XML;
- Aggiunta di un nuovo oggetto all'inventario;
- Modifica di un elemento già presente nell'inventario;
- Rimozione di un elemento dell'inventario;
- Rimozione di tutti gli elementi presenti;
- Visualizzazione di oggetti dell'inventario divisi per categoria di appartenenza;
- Visualizzazione delle caratteristiche di ogni elemento dell'inventario, comprese alcune statistiche calcolate automaticamente dal programma;
- Visualizzazione di avvisi d'errore.

# 2 Progettazione

Lo sviluppo del progetto si è basato sul pattern **Model-View** di *Qt* e metodologia mista *top-down* e *bottom-up*.
Oltre alla gerarchia, è stato realizzato un Container templatizzato per il contenimento degli oggetti appartenenti alla gerarchia. Sono stati realizzati inoltre:

- Una GUI (Graphical User Interface), basata su classi preesistenti di Qt;
- Un Model, il quale si occupa della gestione dei dati del programma, basato anch'esso su classi preesistenti di Qt;
- Un filter proxy, che funge da intermediario tra model e view e permette di filtrare i dati per la visualizzazione corretta su ogni tab;
- Una classe di Input/Output su file XML.

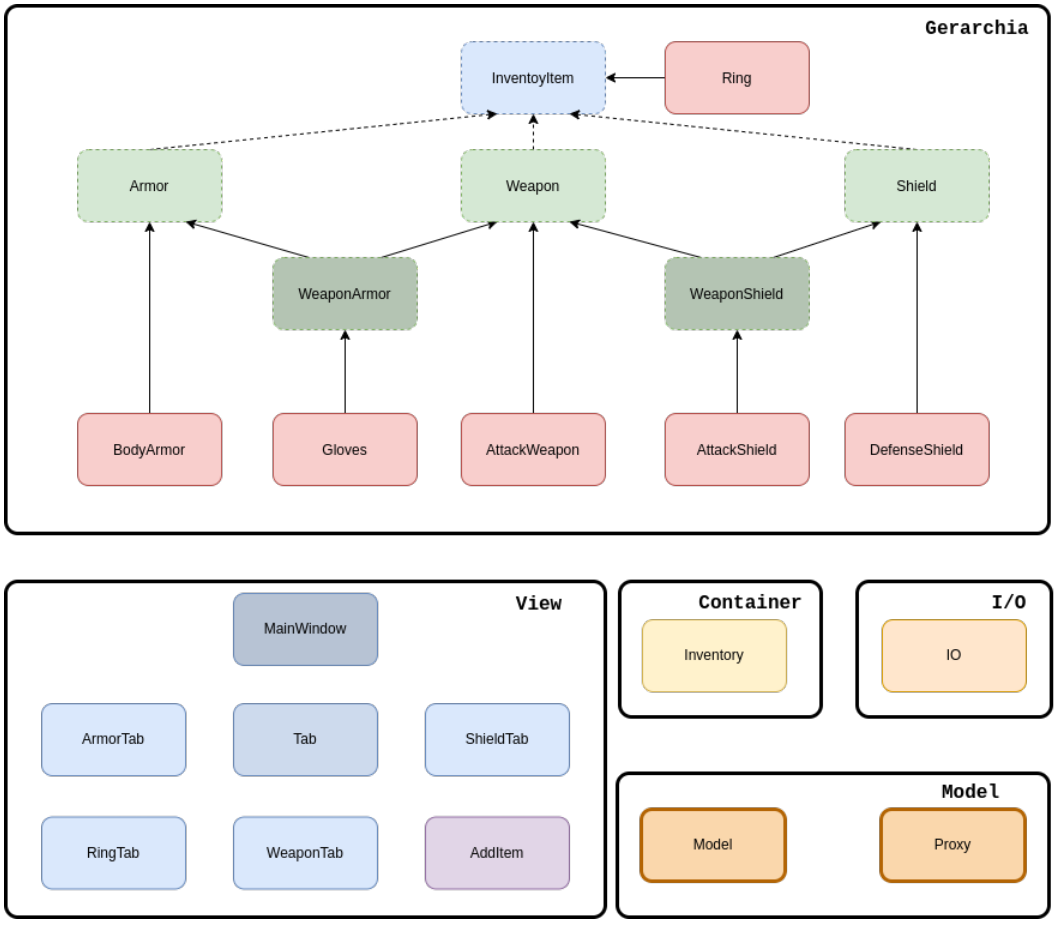


Figura 1: Diagramma delle classi di CrownSouls.

## 2.1 Gerarchia

La gerarchia è composta dalla classe base astratta *InventoryItem*, dalla quale deriva **direttamente** la classe concreta *Ring* e **virtualmente** le classi astratte *Armor*, *Weapon* e *Shield*. Da queste tre classi derivano **singolarmente** e **direttamente** le tre rispettive classi concrete *BodyArmor*, *AttackWeapon* e *DefenseShield*. Viene inoltre utilizzata l'*ereditarietà multipla* per la definizione di classi che rappresentano oggetti appartenenti a più tipi; nello specifico, la classe *WeaponArmor* deriva direttamente da *Armor* e *Weapon*, e la classe *WeaponShield* deriva direttamente da *Weapon* e *Shield*. Questa forma di ereditarietà multipla

è di tipo **is-a**, poiché un oggetto WeaponArmor è sia un oggetto Weapon che un oggetto Armor (e lo stesso vale per WeaponShield). Da queste due classi astratte derivano poi rispettivamente le classi concrete *Gloves* e *AttackShield*.
Ciascuna classe implementa dei metodi virtuali che riguardano l'impostazione e il recupero delle diverse proprietà degli elementi, e dei metodi virtuali specifici di ogni sottoclasse astratta per il calcolo e l'ottenimento di statistiche basate sulle proprietà. L'utilizzo del polimorfismo in tale contesto viene illustrato in seguito.

## 2.2 Container

Container

## 2.3 Modello

Modello

## 2.4 GUI

GUI

## 2.5 I/O

Input/Output

## 2.6 Polimorfismo

Polimorfismo

## 2.7 Scelte progettuali discutibili

Scelte progettuali

# 3 Gestione di Progetto

## 3.1 Suddivisione del lavoro progettuale

Il progetto è stato iniziato

## 3.2 Timeline individuale

## 3.3 Ambiente di lavoro

## 3.4 Istruzioni di compilazione ed esecuzione