|  |
| --- |
| http://www.fondazioneantoniodellamonica.com/images/logo_unisa.png  **Ingegneria del Software** |

****



**Docente:**

*Andrea De Lucia*

**Studenti:**

Raffaele Vitiello

Alessio Cuccurullo

Francesco Giuliano

**OBJECT DESIGN DOCUMENT**

[1. INTRODUZIONE 4](#_Toc534300428)

[**1.1** **Object design trade-offs** 4](#_Toc534300429)

[**1.2** **Linee Guida per la Documentazione delle interfacce** 5](#_Toc534300430)

[**1.3** **Definizioni, acronimi e abbreviazioni** 6](#_Toc534300431)

[**1.4** **Riferimenti** 6](#_Toc534300432)

[2 DESIGN PATTERN 7](#_Toc534300433)

[**2.1** **Design pattern globali** 7](#_Toc534300434)

[3 PACKAGE COMPONENTS 8](#_Toc534300435)

[4 CLASS INTERFACES 9](#_Toc534300436)

[5 GLOSSARIO 10](#_Toc534300437)

1. **INTRODUZIONE**
   1. **Object design trade-offs**

Dopo aver realizzato i documenti RAD ed SDD, dove sono stati omessi gli aspetti che riguardano l’implementazione del sistema, si va a stilare il documento di Object Design che ha come obbiettivo la creazione di un modello che interagisca con tutte le componenti che sono state analizzate nei documenti citati. In questo documento vengono definite le interfacce delle classi, le operazioni, i tipi, gli argomenti e le signature dei sottosistemi definiti nel SDD. Si specificano quinidi i trade-off e le linee guida.

Funzionalità vs Tempo

Non saranno implementate le funzionalità secondarie per presentare il prima possibile il progetto, come ad esempio registrazione, modifica informazioni personali.

Per questo motivo porremo maggiore attenzione alle funzionalità chiave di Sine Charta.

Leggibilità del codice vs Velocità di sviluppo

Il codice dovrà essere comprensibile per semplificare manutenzione e modifiche future. Utilizzeremo quanto più possibile dei commenti per indicare lo scopo delle classi e metodi, rallentando lo sviluppo del sistema.

Scalabilità vs Manutenibilità

Sine Requie è un gioco in continua evoluzione. Il suo regolamento, infatti, è soggetto a piccoli cambiamenti nel corso degli anni. Sine Charta deve essere progettato in maniera tale da poter tenere il passo con lo sviluppo del gioco reale.

Usabilità vs Sicurezza

Sine Charta non è e non vuole essere un software per memorizzare dati sensibili. Pertanto ci occuperemo soprattutto di migliorare l'esperienza di gioco dell'utente finale attraverso un sistema usabile, risspetto all'implementazione di sistemi di sicurezza dei dati ad hoc.

* 1. **Linee Guida per la Documentazione delle interfacce**

Per lo scrittura del codice si seguiranno le seguenti linee guida:

**Naming Convention**

Utilizzeremo le seguenti convenzioni per i nomi:

* Descrittivi
* Pronunciabili
* Di uso comune
* Di lunghezza media
* Non abbreviati
* Utilizzando solo caratteri consentiti (a-z, A-Z, 0-9)

**Varibili**

* I nomi delle variabili devono sempre iniziare con una lettera minuscola e le parole successive con una maiuscola. Le dichiarazioni delle variabili devono essere fatte all’inizio del blocco di codice, le variabili dello stesso tipo sono dichiarate sulla stessa riga.
* In alcuni casi viene utilizzato il carattere underscore “\_”, per le variabili costanti oppure quando vengono utilizzate delle proprietà statiche.

**Metodi**

* I nomi dei metodi devono iniziare con una lettera minuscola, e le parole successive con lettera maiuscola. Solitamente il nome di un metodo consiste in un verbo che identifica l’azione da svolgere, seguite dal nome di un oggetto.
* I nomi dei metodi per l’accesso alle variabili devono essere del tipo “getNomeVaribile()”, mentre i metodi per la modifica delle variabili devono essere del tipo “setNomeVariabile()”.
* Se viene utilizzata una variabile all’interno di un metodo, questa deve essere sempre dichiarata prima del suo utilizzo e deve essere utilizzata per un solo scopo, per facilitare la leggibilità del codice.
* Ad ogni metodo viene aggiunta una descrizione per specificare la loro funzione, come i valori riguardanti gli argomenti, i valori di ritorno e le eccezioni. Questa descrizione deve essere posizionata prima della dichiarazione del metodo.

**Classi e pagine**

* I nomi delle classi devono iniziare con una lettera maiuscola e anche le parole successive, mentre i nomi delle pagine possono iniziare sia con minuscole che con maiuscole.
  1. **Definizioni, acronimi e abbreviazioni**

**Acronimi:**

**Abbreviazioni:**

* 1. **Riferimenti**
* Documento RAD Sine Charta.
* Documento SDD Sine Charta.
* B. Bruegge, A. H. Dutoit, Object Oriented Software Engineering - Using UML, Pattern and Java, Prentice Hall, 3rd edition, 2009 .
* Design Patterns: Elements of Reusable Object-Oriented Software, Gang of four, 1994.

1. **DESIGN PATTERN**
   1. **Design pattern globali**
2. **PACKAGE COMPONENTS**
3. **CLASS INTERFACES**

1. **GLOSSARIO**