

Scelte Implementative

Ingegneria del Software

Stefano Ravetta, Simone Renzo, Matteo Scarpone, Mattia Tollari

29 Febbraio, 2016

1 Introduzione

1.1 Informazioni su questo documento

In questo documento verranno trattate e discusse le scelte implementative che sono state prese per la realizzazione del progetto "Team Diary Management" e le motivazioni delle stesse .

1.2 Contenuti

Si porrà l'attenzione in modo particolare su

- Software per la creazione dei diagrammi
- Software per la stesura della documentazione
- Linguaggio di programmazione
- Formato e struttura della base di dati
- Librerie per la creazione dell'interfaccia utente
- Struttura dei sistemi di sicurezza adottati

2 Linguaggio

2.1 Introduzione

É stato scelto il linguaggio di programmazione Python¹ per alcune caratteristiche piuttosto interessanti che lo possono far risaltare nell'ecosistema dei linguaggi di programmazione odierno.

¹<https://www.python.org/>

2.2 Alto livello

Python consente di programmare ad un livello molto alto: con poche righe riesce a gestire in modo semplice e diretto operazioni complesse senza che il codice risulti troppo complesso da leggere e comprendere.

Questo ha permesso al team di sviluppo di concentrarsi di più sulle parti più astratte e progettuali (come i design pattern e la rimozione di vari code smell) che su questioni più vicine alla comprensione del funzionamento del linguaggio.

2.3 Filosofia

Una parte di un famoso "easter egg"² recita:

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Queste righe sembrano denotare con toni sarcastici la filosofia del linguaggio. Uno degli obiettivi del linguaggio è appunto far sì che il codice scritto risulti il miglior compromesso tra stringatezza e comprensibilità. Si intende quindi che codice estremamente compatto non è sinonimo di efficienza a lungo termine in quanto provocherà quasi sicuramente un dilatamento delle tempistiche sia in fasi di sviluppo successive sia in fase di testing ed ottimizzazione. Un esempio che può far notare il modo in cui questo tipo di filosofia è inserito nella struttura del linguaggio possa effettivamente influire sul codice è quello dell'indentazione: per forzare il programmatore ad indentare sempre e il codice (e a farlo in modo corretto) questo linguaggio non usa caratteri di begin ed end nella definizione di classi, espressioni condizionali, espressioni iterative, funzioni e procedure (come fa ad esempio Java con le parentesi graffe) ma usa solo l'indentazione. In tal modo da una parte i programmatori che di norma non indentano il loro codice risulteranno "costretti" a farlo, e dall'altra i programmatori che già di norma indentano il proprio codice non dovranno inserire altri simboli per esprimere ciò che concettualmente può essere già espresso dall'indentazione.

Secondo il nostro team di sviluppo questa caratteristica risultava molto interessante per la gestione di un progetto in cui è richiesta particolare attenzione alla qualità del codice prodotto. Inoltre a livello pratico tutto ciò è risultato addirittura comodo dato che ogni membro del gruppo presentava un proprio stile di stesura del codice e si è mostrata la necessità di uno stile univoco con cui lavorare.

²Questo easter egg, "The Zen of Python" è stato inserito nell'interprete Python e che può apparire appena si prova ad importare la libreria "this".

2.4 Portabilità

Python è un linguaggio precompilato e il suo interprete è disponibile per numerose³ piattaforme. La presenza di questa caratteristica non è stata sicuramente ignorata dato che i quasi tutti i membri del team di sviluppo usano piattaforme ed ambienti diversi. Anche la distribuzione dell'applicazione e di pacchetti contenenti le varie dipendenze richieste risulta estremamente semplificata grazie alla portabilità di Python.

2.5 Dizionari

Una delle funzionalità che sono state più apprezzate per la stesura del codice relativa al progetto è senza dubbio la gestione dei dizionari che Python offre e la loro semplice e diretta interazione con classi e anche dati Json.

3 Librerie per la creazione dell'interfaccia utente

3.1 Introduzione

Sono state scelte le librerie Qt⁴ che fanno parte di uno storico framework open source.

3.2 Potenzialità

Le librerie Qt sono in grado di gestire interfacce grafiche, basi di dati, attività multimediali, flussi di rete, gestione di file e notifiche di sistema. Una caratteristica piuttosto interessante di queste librerie è che possono essere usate attraverso molti⁵ linguaggi di programmazione e non sono legate a nessuna piattaforma specifica. Permettono anche possibile lo sviluppo di applicazioni mobile⁶

3.3 Ambienti di sviluppo

Per lo sviluppo dell'interfaccia grafica è stato comodo usare lo strumento QtDesigner⁷ (che segue il paradigma WYSIWYG) che permette di disegnare GUI, modificare le proprietà dei vari oggetti e generare codice XML dell'interfaccia realizzata.

³L'elenco delle piattaforme supportate è reperibile a <https://www.python.org/download/other/>

⁴<http://www.qt.io/>

⁵È possibile visionare tutti i linguaggi supportati a questa pagina:
<http://wiki.qt.io/Category:LanguageBindings>

⁶il progetto SailfishOs può dare un'idea delle potenzialità delle librerie Qt:
<http://sailfishos.org/develop/sdk-overview/>

⁷<http://doc.qt.io/qt-4.8/designer-manual.html>

3.4 interazione con Python

Per far interagire il file XML generato da QtDesigner con Python è stato usato lo strumento Pyuic4⁸ che svolge il lavoro meccanico di conversione da file che descrive un'interfaccia ad uno script Python che mette a disposizione classi che rappresentano gli oggetti usati ed i metodi per accedervi.

4 Struttura dei sistemi di sicurezza adottati

4.1 Codifica delle password

4.2 Token, autenticazione e gestione delle sessioni

4.3 Password dimenticate

⁸<http://pyqt.sourceforge.net/Docs/PyQt4/designer.html>