

ZODB

Un database ad oggetti
scritto in Python (e C)

Chi siamo

Riccardo Lemmi

Antonio Tirabasso

Tips

Se devi usare un ORM allora usa lo ZODB

Cos'è lo ZODB

- È un database ad oggetti
- I dati vengono gestiti con lo stesso paradigma dell'applicazione:
 - Non si è legati allo storage:
 - Nessun mapping
 - Nessun 'glue code'
 - Codice più semplice, più robusto e più facile da testare

Cosa fornisce lo ZODB

- Persistenza oggetti
 - Tramite il modulo pickle
- Supporto transazioni
 - ACID: atomicity, consistency, isolation, durability
- Storage alternativi
- History/Undo
 - Se lo storage lo supporta
- Scalabilità: ZEO

Come funziona

- L'oggetto viene serializzato tramite il modulo pickle e vengono aggiunte informazioni sulla transazione
- ZODB si preoccupa di leggere e scrivere gli oggetti sullo storage
- ZODB si occupa della gestione della cache

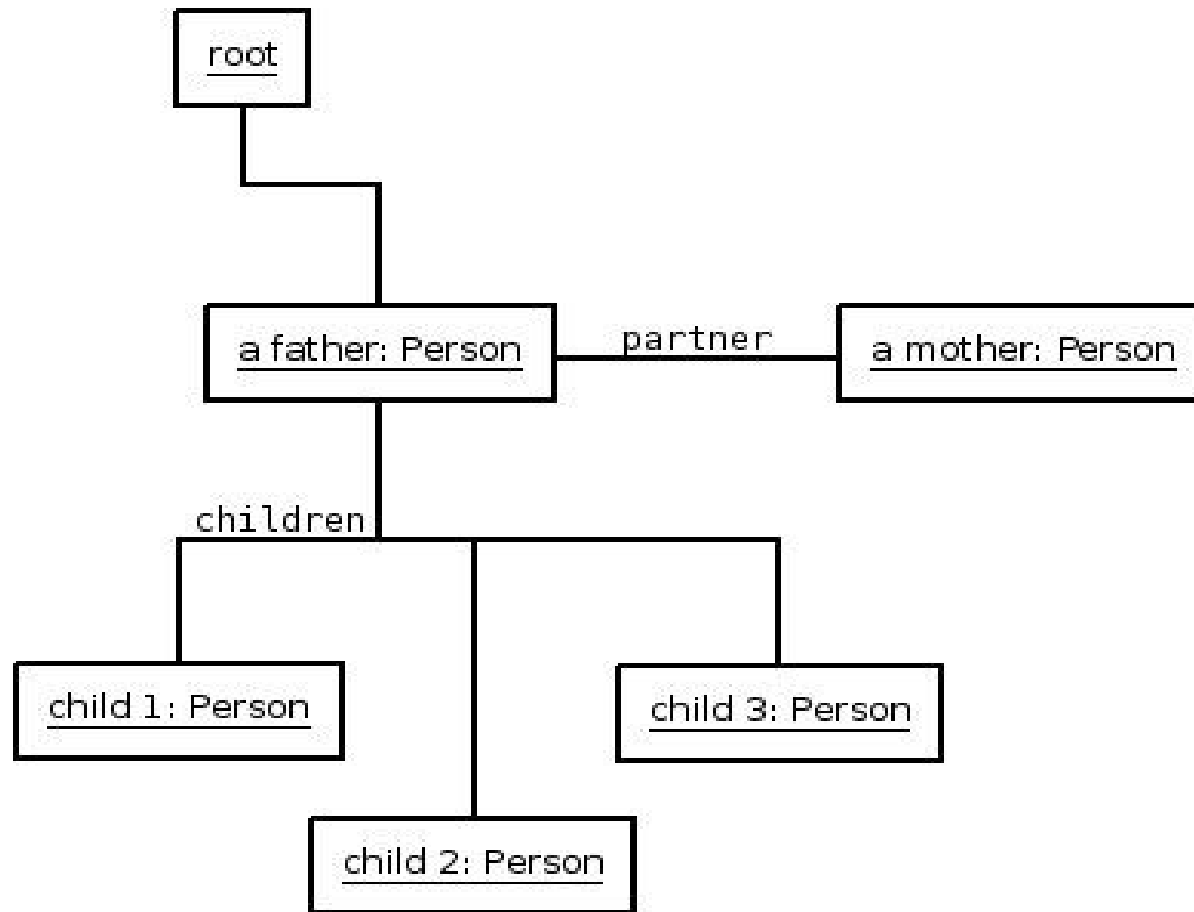
Regole per scrivere classi "persistenti"

- Classi:
 - Persistent: gestito automaticamente ma se contiene riferimenti a:
 - Integer, string, float, boolean: non si deve fare nulla
 - List and dict: da gestire con l'attributo `_p_changed`

Regole per scrivere classi "persistenti"

- Le classi devono ereditare da Persistent
 - in caso contrario si generano serializzazioni multiple
- Per serializzare un oggetto è necessario connetterlo con un altro oggetto persistente
 - al limite la radice del DB
- Fai sempre un commit delle transazioni
- Se modifichi sotto-oggetti non-Persistent devi settare `_p_changed` a True.
- Usa PersistentList, PersistentMapping e BTree

L'esempio: l'albero genealogico



L'esempio: l'albero genealogico

- Inizializzazione zodb
- Operazioni
 - Creazione
 - Cancellazione
 - Aggiunta
 - Undo
- La ricerca
 - come si fa

Links

- <http://www.zodb.org/overview.html>
- <http://faassen.n--tree.net/blog/view/weblog/2008/06/20/0>