Day 4

Thursday, August 18, 2022 1:29 PM

Object States in Hibernate

the objects we create with hibernate can have 1 of 3 states

These states are in context to the session object

1. Transient
   - An object is created in code, but not associated with a session
   - Has yet to be stored in thn db

2. Persistant
   - the object is persisted in the database
   - done with .save() .persist() .saveOrUpdate()
   - hibernate will will watch for changes and try to synchronize the state with the db
   - these will be able to queried from the db

3. Detached
   - We have access to a persisted object but the session it is attached to closes
   - Any changes to this object will not be synched
   - We can reattach with .update() .merge()
   - We can create detached objects with .close() .evict() .clear

All the methods above come from the session

Caching In Hibernate

Hibernate perform caching to increase performance

Two levels of caching

first-Level / L1:
- first place hibernate looks for data
- automatically implemented
- Associated with the session object
- Useful when you have repeated queries on the same session
- The hibernate just has the data on hand cuts out the database query

Second-Level Cache
- The second place hibernate looks
- This one is optional, the developer must provide a caching library for hibernate
- Associated with the session factory

- Associated with the session factory

To configure use the hibernate.cfg.xml

property with name cache.use_second_level_cache = true
property with name hibernate.cache.region.factory_class classpath for lib

To tell hibernate to cache an object/class
use @Cacheble above the class

Service Locator Design Pattern

A design pattern used to encapsulate the processes involved in obtaining a service in a layer of abstraction

It has a central registry which is known as the Service locator
- this is responsible for returning instances of our services

Components of this design:

Client: the part of the app looking for a service

Service Locator: returns instances of a service

Initial Context: creates, registers, and caches services

Service factory: provides lifecycle management for the services, which helps create, register and look up services

Service: the services being found