Daniel R. Cender

Grand Canyon University

CST-361: Design Patterns in Java

Charbel Elkhoury

April 24, 2021

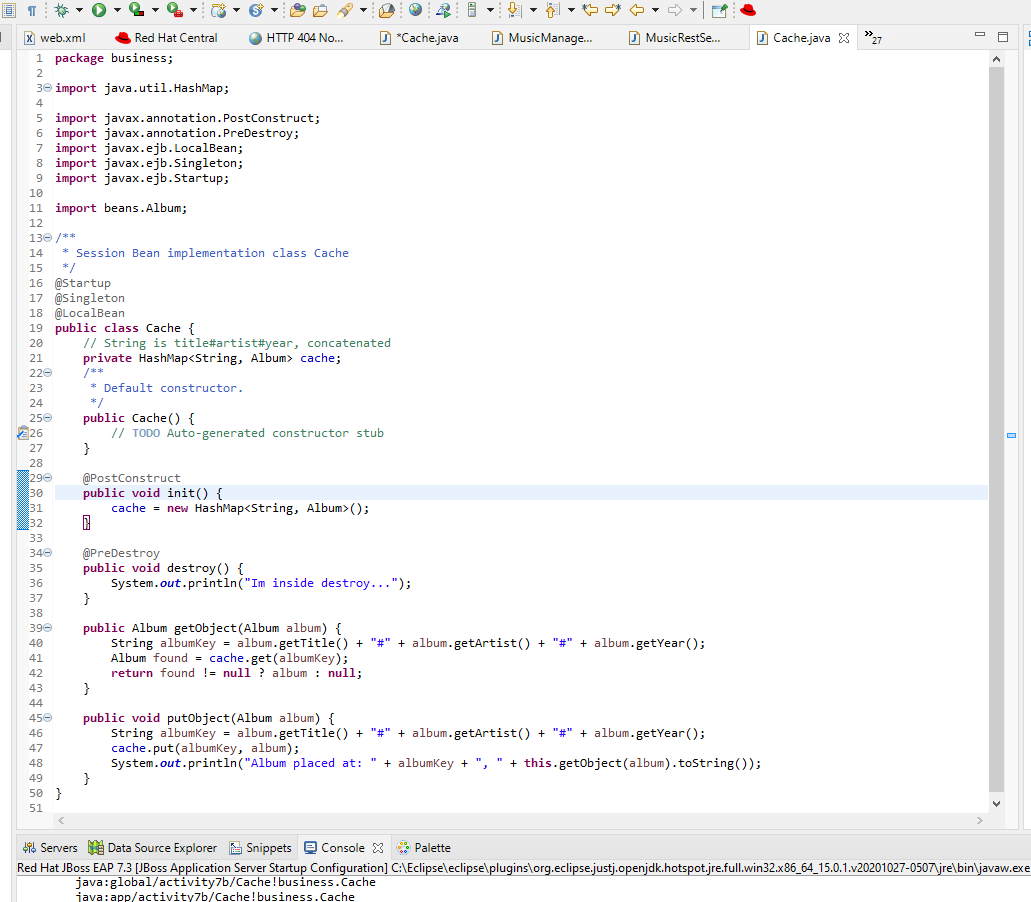
**Write-Up**

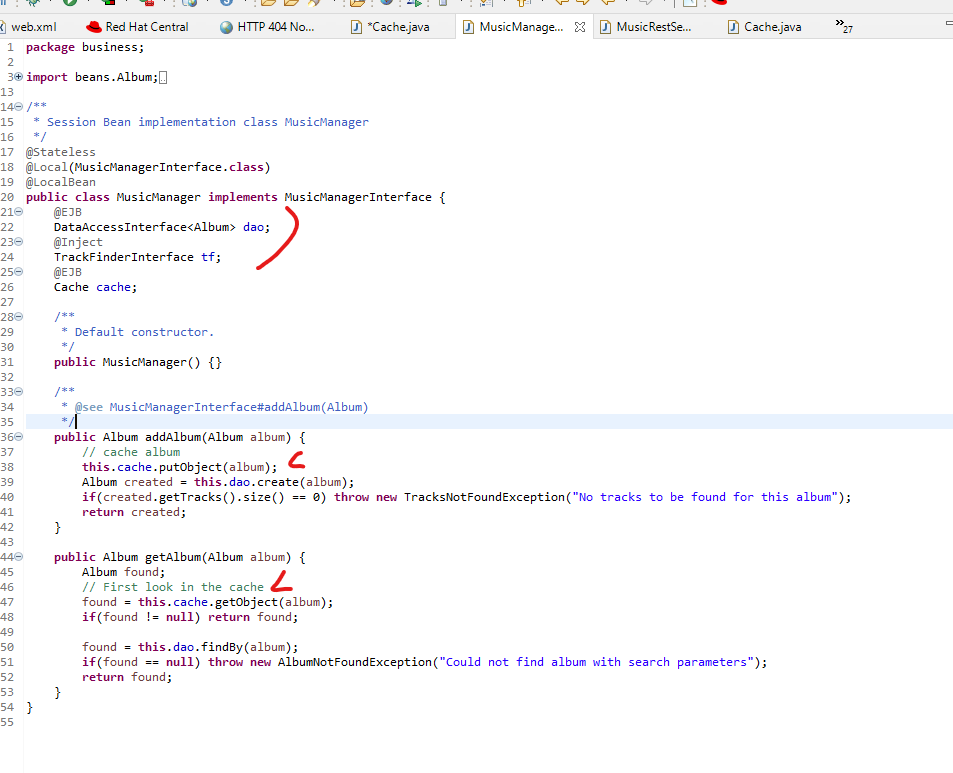
**Chapter 4 Questions**

* **Where and how did this activity demonstrate the use of the Singleton Design Pattern?**The EJB created for the local Cache class was specifically annotated so that, at most, only one instance of the class would be created for each application session.
* **Why was the Singleton Design Pattern required in this activity?** We would only want to have one instance of a session-wide cache, because a user would expect that data to persist across page redirects.
* **Where and how did this activity demonstrate technical encapsulation?**In the Cache implementations, we do not allow for a consuming class to directly access the data structure. We provide a level of encapsulating abstraction (the getObject and putObject methods) for an EJB consumer to use in order to properly mutate the cached data.
* **How could the design of the EJB Singleton Cache be improved?**The Cache would ideally be linked to an externally-connected caching technology, such as a Redis cache server. This would allow for data to be cached based on geographical request location and/or the user requesting the information, irrespective of a browser session.
* **What precautions do you have to take when designing a Cache?**In the same way that a database engine must protect against stale reads or inappropriate writes, a cache should do the same. A simple improvement would be adding @Lock(LockType.WRITE) or @Lock(LockType.READ) to the accessor/getter cache methods.
* **Research the Internet and identify 2 open source Cache Frameworks that could be used as a replacement for the Singleton EBJ Cache**. A couple highly-rated caching frameworks for JavaEE applications are OSCache and JBoss Cache, which is maintained by Red Hat.
* **Provide one additional use case where a Singleton could be used in an Enterprise Java application.** There are likely utilities that only need one instance in an enterprise application. Only one error logger is needed for every application instance, for example. There’s no reason for more to exist, and error logging should not occur in that high of a volume in production anyways.

**Deliverables**

Below are a set of screenshots showing the code used to implement the Cache class. The JBoss server started refusing to run this activity application, despite working the week before. The first screenshot shows the Cache class, and the second shows the EJB in use in the MusicManager class.





**References**