## Caching in Hibernate

Hibernate caching improves the performance of the application by pooling the object in the cache.

There are mainly two types of caching: first level cache and second level cache.

## First Level Cache

Session object holds the first level cache data. It is enabled by default. The first level cache data will not be available to entire application. An application can use many session object.

## Second Level Cache

SessionFactory object holds the second level cache data. The data stored in the second level cache will be available to entire application. But we need to enable it explicitely.

Second Level Cache implementations are provided by different vendors such as:

EH (Easy Hibernate) Cache
Swarm Cache
OS Cache
JBoss Cache

SessionFactory holds the second level cache data. It is global for all the session objects and not enabled by default.
Each implementation provides different cache usage functionality. There are four ways to use second level cache.

**read-only**: caching will work for read only operation.
**nonstrict-read-write**: caching will work for read and write but one at a time.
**read-write**: caching will work for read and write, can be used simultaneously.
**transactional**: caching will work for transaction.

The cache-usage property can be applied to class or collection level in hbm.xml file. The example to define cache usage is given below:

```
<cache usage="read-only" />
```

Let's see the second level cache implementation and cache usage.

| Implementation | read-only | nonstrict-read-write | read-write | transactional |
|---|---|---|---|---|
| **EH Cache** | Yes | Yes | Yes | No |

| | | | | |
|---|---|---|---|---|
| **Swarm Cache** | Yes | Yes | Yes | No |
| **OS Cache** | Yes | Yes | No | No |
| **JBoss Cache** | No | No | No | Yes |

1 extra steps for second level cache example using EH cache

1) Add 2 configuration setting in hibernate.cfg.xml file

```
<property name="cache.provider_class">org.hibernate.cache.EhCacheProvider</property>

<property name="hibernate.cache.use_second_level_cache">true</property>
```

2) Add cache usage setting in hbm file

```
<cache usage="read-only" />
```

3) Create ehcache.xml file

```
<?xml version="1.0"?>
<ehcache>

<defaultCache
maxElementsInMemory="100"
eternal="false"
timeToIdleSeconds="120"
timeToLiveSeconds="200" />

</ehcache>
```

You need to create ehcache.xml file to define the cache property.

**defaultCache** will be used for all the persistent classes. We can also define persistent class explicitely by using the cache element.

**eternal** If we specify eternal="true", we don't need to define timeToIdleSeconds and timeToLiveSeconds attributes because it will be handled by hibernate internally. Specifying eternal="false" gives control to the programmer, but we need to define timeToIdleSeconds and timeToLiveSeconds attributes.

**timeToIdleSeconds** It defines that how many seconds object can be idle in the second level cache.

**timeToLiveSeconds** It defines that how many seconds object can be stored in the second level cache whether it is idle or not.