**Custom Cache Library**

High [performance](https://crunchify.com/in-java-what-is-a-difference-between-identityhashmap-and-hashmap-performance-comparision/) scalable web applications often use a distributed in-memory data cache. A cache is an area of local [memory](https://crunchify.com/java-runtime-get-free-used-and-total-memory-in-java/) that holds a copy of frequently accessed data.

The characteristic of the program:

* Items will expire based on a time to live period.
* Cache will keep most recently used items (apache [common collections](https://crunchify.com/how-to-override-equals-and-hashcode-method-in-java/) has a LRUMap, which, removes the least used entries).
* For the expiration of items we can timestamp the last access and in a separate thread remove the items when the time to live limit is reached. This is nice for reducing [memory](https://crunchify.com/how-to-generate-out-of-memory-oom-in-java-programatically/) pressure for applications that have long idle time in between accessing the cached objects.
* Thread safe due to synchronized keyword.

**Socket.io is preferred to access the cache**

Why socket.io?

* Reconnects Automatically if Connection drops.
* Enables real-time, bidirectional and event-based communication between client and server.
* It allows synchronized communication to take place simply within your app.
* Supports a large number of browsers, even old ones.
* Very fast, most online games use it.