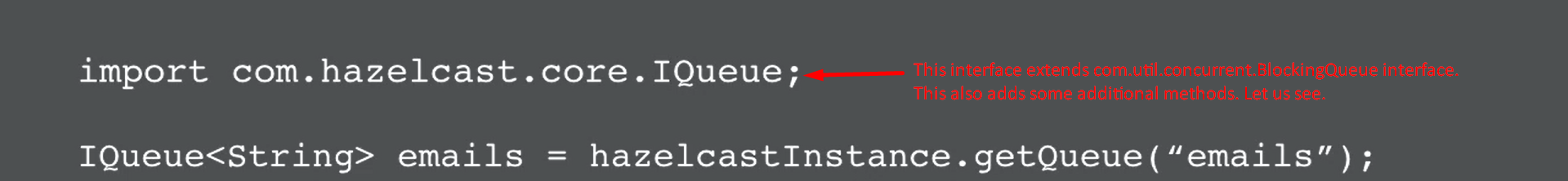
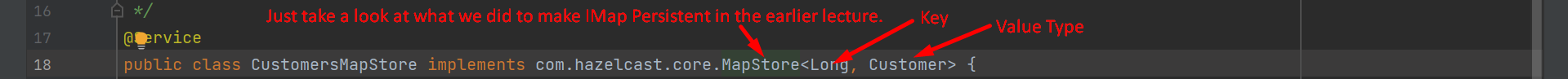
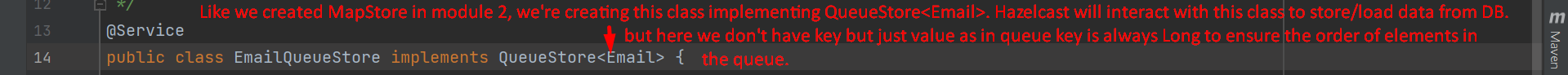
1. 
2.   
   We can get reference to a queue by calling **HazelcastInstance.getQueue**(<queue\_name>);
3. **Methods**: As this interface add some additional methods. Let’s have a look.
   1. **poll()**:  
       Text

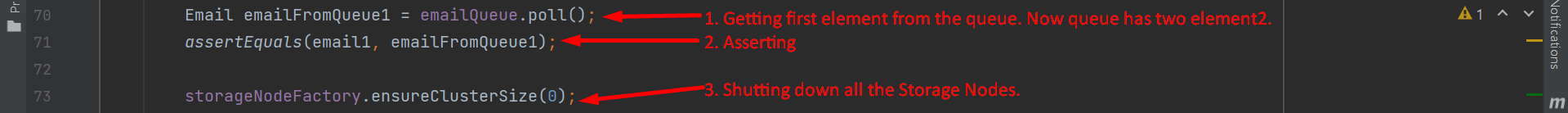
      Description automatically generated
      1. Non-Blocking method.
      2. Will return the 1st available element from the queue else null.
      3. There are other versions of this method which are blocking methods.
   2. poll**(Long Duration**, **TimeUnit.Seconds)**:  
      Text

      Description automatically generated
      1. This waits for the element to be available for the specified time.
      2. If element is available within the time period, then it will return immediately.  
         If time duration expires, then it will return null and the thread will continue.
   3. **take()**  
        
      Text

      Description automatically generated
      1. This is fully blocking method.
4. Queue Data Structure can be configured to be persistent like IMap Data Structure.
5. Let’s see one example of it.
6. A picture containing graphical user interface

   Description automatically generated
7. We need to create a class that will interact with the Hazelcast like we create **CustomerMapStore.java** which **extended MapStore.**java interface.  
     
     
     
   The remaining code is same as we wrote for MapStore.   
   Refer to “8.Persistence Storage” lecture in module 2 🡺 Section 2 Basic Map Usage
8. Let’s inform the Hazelcast about **QueueStore**.  
   Text

   Description automatically generated
9. Let’s test our QueueStore.  
   Text

   Description automatically generated  
     
   **Line#73**: When all the storage nodes are shut down, before it, all the data will be persisted into DB.  
   Text

   Description automatically generated   
   