

Get Started with Hazelcast IMDG

In this hands-on exercise, you will:

- Download and install Hazelcast IMDG on your local device
- Download and install Hazelcast Management Center
- Work with a simple map structure using Hazelcast IMDG in embedded mode
- Use the same map structure with Hazelcast IMDG in client/server mode
- Use Management Center to monitor cluster operations

Before You Start

Hazelcast IMDG is a Java application and requires the Java Runtime Environment (JRE), version 8.0 or later. Make sure you have JRE installed. [Download JRE](#)

Part 1: Acquire Hazelcast IMDG

Choose one of the following options:

Maven

You can find Hazelcast in standard Maven repositories. If your project uses Maven, you do not need to add additional repositories to your `pom.xml` or add `hazelcast-<version>.jar` file into your classpath - Maven does that for you.

1. Open a new Maven project in your preferred IDE. Add the following lines to your `pom.xml`

```
<dependencies>
  <dependency>
    <groupId>com.hazelcast</groupId>
    <artifactId>hazelcast</artifactId>
    <version>Hazelcast IMDG version-number e.g. 4.0</version>
  </dependency>
</dependencies>
```

ZIP or TAR

1. Browse to hazelcast.org.
2. Download the package `hazelcast-<version>.zip` or `hazelcast-<version>.tar.gz`.
3. Extract the downloaded file.
4. Open a new project in your IDE. Add the file `hazelcast-<version>.jar` to your classpath.

Docker (Client/Server mode only)

1. Launch the Hazelcast Docker container.

```
$ docker run hazelcast/hazelcast:$HAZELCAST_VERSION
```

Note: If you use Docker, skip the Embedded Mode portion of this lab exercise.

Part 2: Embedded Mode

Note: Embedded mode only uses Java

From your IDE, run the following code:

```
import com.hazelcast.core.*;
import com.hazelcast.config.*;
import com.hazelcast.map.IMap.*;
import com.hazelcast.collection.IQueue.*;

public class GettingStarted {
    public static void main(String[] args) {
        Config cfg = new Config();

        //Start an instance of Hazelcast IMDG
        HazelcastInstance hz = Hazelcast.newHazelcastInstance(cfg);

        //Create a new distributed map
        IMap<Integer, String> mapCust = hz.getMap("customers");

        //Populate the map
        mapCust.put(1, "Joe");
        mapCust.put(2, "Ali");
        mapCust.put(3, "Avi");
        mapCust.put(4, "Ben");
        mapCust.put(5, "Emi");
        mapCust.put(6, "Sam");

        //Extract and print data from the map

        System.out.println("Customer with key 1: " + mapCust.get(1));
        System.out.println("Map Size: " + mapCust.size());

        //Create a new distributed queue
        IQueue<String> qCust = hz.getQueue("customers");
        qCust.offer("John");
        qCust.offer("Mary");
        qCust.offer("Jane");
        qCust.offer("Fred");
        qCust.offer("Luka");
        qCust.offer("Mike");

        //Extract and print data from the queue
        System.out.println("First customer: " + qCust.poll());
        System.out.println("Second Customer: " + qCust.peek());
        System.out.println("Queue size: " + qCust.size());
    }
}
```

From your IDE run window, observe the startup sequence of the Hazelcast IMDG instance, as well as the output printed by your code.

```
INFO: [192.168.10.115]:5701 [dev] [4.0.11]

Members {size:1, ver:1} [
    Member [192.168.10.115]:5701 - 2d9598b7-f0be-43cf-b842-34488cc74dc2 this
]

Apr 25, 2020 1:45:27 PM com.hazelcast.core.LifecycleService
INFO: [192.168.10.115]:5701 [dev] [4.0.1] [192.168.10.115]:5701 is STARTED
Apr 25, 2020 1:45:27 PM com.hazelcast.internal.partition.impl.PartitionStateManager
INFO: [192.168.10.115]:5701 [dev] [4.0.1] Initializing cluster partition table
arrangement...
Customer with key 1: Joe
Map Size: 6
First customer: Tom
Second Customer: Mary
Queue size: 5
```

Part 3: Client/Server Mode – Starting the Cluster

Starting the cluster will depend on the method you used to acquire Hazelcast.

Maven

Open a new code window. Run the following code twice to start two instances of Hazelcast.

```
import com.hazelcast.config.Config;
import com.hazelcast.core.Hazelcast;

public class StartMember {

    public static void main(String[] args) {
        Config config = new Config();

        Hazelcast.newHazelcastInstance(config);
    }
}
```

ZIP/TAR

Go to the `bin` directory and use the `start.sh` script (or `start.bat` if you are on Windows) to start two instances of Hazelcast.

Docker

Use the `docker run hazelcast/hazelcast` command twice to start two instances of Hazelcast.

For all methods, you should see output similar to the following, indicating that the two cluster members are running and have found each other:

```
Members {size:2, ver:2} [
  Member [192.168.10.115]:5701 - e2a3f163-4d6e-4ef9-8030-7dfdfbd734f4
  Member [192.168.10.115]:5702 - 8f904e45-5417-46ca-a7e7-70d71969d76d this
]
```

Part 4: Management Center

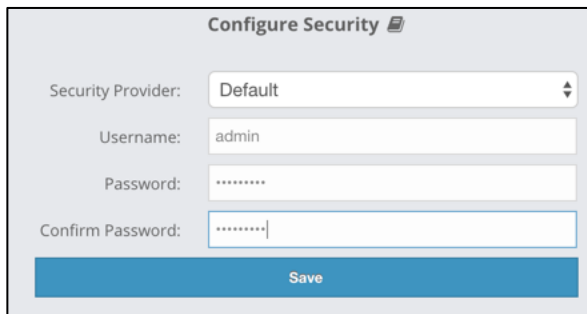
1) Start Management Center

Maven/ZIP/TAR: Go to the directory `management-center` under the main Hazelcast directory. Run `start.sh` (or `start.bat` if you are on Windows).

Docker: `docker run -p 8080:8080 hazelcast/management-center`

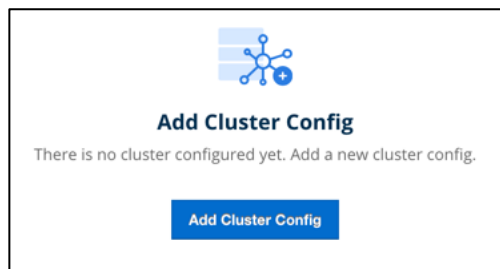
2) Open a browser window to <http://localhost:8080>.

3) Create a username and password under the **Default** security provider.



4) Log in to Management Center.

5) Click on the **Add Cluster Config** button.



6) For Maven/ZIP/TAR, leave the settings at the default and click **Save**.

For Docker, find out the IP addresses used for the cluster and use that address for Member Addresses. (Docker usually defaults to 172.17.0.2 for the first instance and .0.3 for the second).

```
Members {size:2, ver:2} [  
  Member [172.17.0.2]:5701 - c6a5e083-d1de-4968-a182-1593edf5011a  
  Member [172.17.0.3]:5701 - 125d6afc-ce98-4b76-9e6b-0af60c9cf929 this  
]
```

← **Add Cluster Config** Upload Config File

Cluster Name ⓘ
dev

Cluster Config State ⓘ
Enabled

Member Addresses ⓘ
localhost x

Maven/ZIP/TAR: Localhost
Docker: Docker IP address of cluster

Discard Changes Save

7) Open the cluster view by clicking **Select**.

• dev ✎ 🗑

⚙️ Config State: **Enabled**

📋 Cluster State: **ACTIVE**

👤 Members: 2

Select

Part 5: Client/Server – Using the Client, Cluster Elasticity

1) Modify your code from Part 2 as shown below:

```
//Import client libraries
import com.hazelcast.client.HazelcastClient;
import com.hazelcast.client.config.ClientConfig;
import com.hazelcast.core.*;
import com.hazelcast.config.*;
import com.hazelcast.map.IMap.*;
import com.hazelcast.collection.IQueue.*;

public class GettingStarted {
    public static void main(String[] args) {

        //change the config to ClientConfig
        ClientConfig cfg = new ClientConfig();

        //Start an instance of the Hazelcast Java client
        HazelcastInstance hz = HazelcastClient.newHazelcastClient(cfg);

        //Create a new distributed map
        IMap<Integer, String> mapCust = hz.getMap("customers");

        //Populate the map
        mapCust.put(1, "Joe");
        mapCust.put(2, "Ali");
        mapCust.put(3, "Avi");
        mapCust.put(4, "Ben");
        mapCust.put(5, "Emi");
        mapCust.put(6, "Sam");

        //Extract and print data from the map

        System.out.println("Customer with key 1: " + mapCust.get(1));
        System.out.println("Map Size: " + mapCust.size());

        //Create a new distributed queue
        IQueue<String> qCust = hz.getQueue("customers");
        qCust.offer("John");
        qCust.offer("Mary");
        qCust.offer("Jane");
        qCust.offer("Fred");
        qCust.offer("Luka");
        qCust.offer("Mike");

        //Extract and print data from the queue
        System.out.println("First customer: " + qCust.poll());
        System.out.println("Second Customer: " + qCust.peek());
        System.out.println("Queue size: " + qCust.size());
    }
}
```


- 2) Run your code and observe your results. You should see the same system output as you did in Part 2, but the startup sequence will show the client connecting to the running Hazelcast cluster.

```
Apr 24, 2020 8:57:54 PM com.hazelcast.client.impl.connection.ClientConnectionManager
INFO: hz.client_1 [dev] [4.0.1] Authenticated with server
[192.168.10.115]:5702:385dd679-4cae-4a5e-9c62-a856834dl60e, server version: 4.0.1,
local address: /192.168.10.115:57596
```

- 3) In Management Center, under **Storage > Maps**, observe how the three map entries are distributed across the two cluster members. (Below is an example screen shot; your data may differ).

Map Statistic Data Table (In-Memory Format: BINARY)

Member	Entries	Gets	Puts	Removals	Entry Mem...	Backups	Backup Me...
192.168.10.115:5701	2	0	3	0	262.00 B	4	524.00 B
192.168.10.115:5702	4	1	3	0	524.00 B	2	262.00 B
TOTAL	6	1	6	0	786.00 B	6	786.00 B

Spend a few moments exploring the type of information you can retrieve using Management Center.

- 4) In Management Center, under **Messaging > Queues**, observe how the queue is distributed across the two cluster members. Again, spend some time exploring Management Center.

Queue Statistics

Member	Items	Backups
192.168.10.115:5701	0	5
192.168.10.115:5702	5	0

Wondering why the map entries are distributed across cluster members, but the queue entries aren't? Watch the lesson on partitioning in the Intro to Hazelcast course.

- 5) Start a third instance of Hazelcast IMDG.

- 6) Use Management Center to observe the changes in map data distribution, and the lack of any changes to queue distribution

Member ↕	↕ Entries	↕ Gets	↕ Puts	↕ Removals	↕ Entry Mem...	↕ Backups	↕ Backup Me...
192.168.10.115:5701	2	0	3	0	262.00 B	4	524.00 B
192.168.10.115:5702	1	1	3	0	131.00 B	1	131.00 B
192.168.10.115:5703	3	0	0	0	393.00 B	1	131.00 B
TOTAL	6	1	6	0	786.00 B	6	786.00 B

- 7) Use Management Center to stop one of the first two cluster members. (**Cluster > Members**, select a member, click **Shutdown Member**)
- 8) Use Management Center to observe the changes in map and queue distribution.

Member ↕	↕ Entries	↕ Gets	↕ Puts	↕ Removals	↕ Entry Mem...	↕ Backups	↕ Backup Me...
192.168.10.115:5702	2	1	3	0	262.00 B	4	524.00 B
192.168.10.115:5703	4	0	0	0	524.00 B	2	262.00 B
TOTAL	6	1	3	0	786.00 B	6	786.00 B

Member ↕	↕ Items	↕ Backups
192.168.10.115:5702	5	0
192.168.10.115:5703	0	5

- 9) Save your client code for use with the Getting Started with Hazelcast Cloud exercise!

Part 6: Other Clients (Optional)

Visit <https://hazelcast.org/imdg/clients-languages/> to review client compatibility/availability and to access the client software for the following languages:

- .NET
- C#
- C++
- node.js
- Python
- Go

Download your preferred client. Start your cluster as described in Part 3. Use your client to communicate with the Hazelcast cluster.