

Лабораторна робота №2

Золотов Іван ФБ-31мп

Посилання на GitHub

<https://github.com/SideOctopus/Lab2>

1. Встановлення Hazelcast:

```
sudo docker run -d --name hazelcast1 -p 5704:5701 hazelcast/hazelcast:latest
```

```
sudo docker run -d --name hazelcast2 -p 5705:5701 hazelcast/hazelcast:latest
```

```
sudo docker run -d --name hazelcast3 -p 5706:5701 hazelcast/hazelcast:latest
```

2. Розгортання Management Center:

```
sudo docker run -d -p 8081:8080 hazelcast/management-center
```

3. Підключаємо кластер до Management Center:

4. Distributed Map

```
1 import hazelcast
2
3 def main():
4     # Create a Hazelcast client and connect to the cluster
5     client = hazelcast.HazelcastClient(cluster_members=["183.158.69.103:4704", "183.158.69.103:4705", "183.158.69.103:4706"])
6
7     # Get a distributed map named "capitals"
8     distributed_map = client.get_map("capitals").blocking()
9
10    # Insert 1000 key-value pairs into the map
11    for i in range(1000):
12        distributed_map.put(i, f"City {i}")
13
14    # Print the size of the map
15    print(f"Map size: {distributed_map.size()}")
16
17    # Shutdown the Hazelcast client
18    client.shutdown()
19
20 if __name__ == "__main__":
21     main()
```

Перевіряємо розподіл даних між вузлами в Management Center:

Member ^	Entries	Gets	Puts	Removals	Sets	Entry Memory	Events
172.17.0.2:5700	351	0	351	0	0	45.25 kB	0
172.17.0.3:5700	333	0	333	0	0	42.85 kB	0
172.17.0.4:5700	316	0	316	0	0	40.70 kB	0
TOTAL	1,000	0	1,000	0	0	128.80 kB	0

Вимикаємо одну ноду

```
sudo docker stop hazelcast1
```

Member ^	Entries	Gets	Puts	Removals	Sets
172.17.0.3:5700	510	0	333	0	0
172.17.0.4:5700	490	0	316	0	0
TOTAL	1,000	0	649	0	0

Вимикаємо ще одну ноду

```
sudo docker stop hazelcast2
```

4. Topic

```
1 import hazelcast
2 import time
3
4 def task4_1():
5     # Create a Hazelcast client and connect to the cluster
6     client = hazelcast.HazelcastClient(cluster_members=["183.158.69.103:4704", "183.158.69.103:4705", "183.158.69.103:4706"])
7
8     try:
9         # Get a distributed topic named "my-distributed-topic"
10        topic = client.get_topic("my-distributed-topic")
11
12        # Publish numbers 1 to 100 to the topic
13        for i in range(1, 101):
14            topic.publish(i)
15            print(f"Sent: {i}")
16            time.sleep(0.1) # Sleep to simulate delay between publishes
17
18    finally:
19        # Shutdown the Hazelcast client
20        client.shutdown()
21
22 if __name__ == "__main__":
23     task4_1()
```

Отримуємо:

```
Sent: 1
Sent: 2
Sent: 3
Sent: 4
Sent: 5
Sent: 6
Sent: 7
Sent: 8
Sent: 9
Sent: 10
Sent: 11
Sent: 12
Sent: 13
Sent: 14
Sent: 15
Sent: 16
Sent: 17
Sent: 18
```

Клієнт повідомлень без паузи:

```
1 import hazelcast
2
3 def task4_2():
4     # Create a Hazelcast client and connect to the cluster
5     client = hazelcast.HazelcastClient(cluster_members=["183.158.69.103:4704", "183.158.69.103:4705", "183.158.69.103:4706"])
6
7     try:
8         # Get the distributed topic named "my-distributed-topic"
9         topic = client.get_topic("my-distributed-topic")
10
11        # Define a listener function to handle incoming messages
12        listener = topic.add_listener(lambda message: print(f"Received: {message.message}"))
13
14        # Wait for user input to stop the listener (typically press Enter)
15        input("Press Enter to stop...\n")
16
17    finally:
18        # shutdown the Hazelcast client
19        client.shutdown()
20
21 if __name__ == "__main__":
22     task4_2()
```

```
Press Enter to stop...
Received: 1
Received: 2
Received: 3
Received: 4
Received: 5
Received: 6
Received: 7
Received: 8
Received: 9
Received: 10
Received: 11
Received: 12
Received: 13
Received: 14
Received: 15
Received: 16
Received: 17
Received: 18
Received: 19
Received: 20
Received: 21
Received: 22
```

Клієнт повідомлень з паузою:

```
1 import hazelcast
2 import time
3
4 def task4_2():
5     # Create a Hazelcast client and connect to the cluster
6     client = hazelcast.HazelcastClient(cluster_members=["183.158.69.103:4704", "183.158.69.103:4705", "183.158.69.103:4706"])
7
8     try:
9         # Get the distributed topic named "my-distributed-topic"
10        topic = client.get_topic("my-distributed-topic")
11
12        # Define a listener function to handle incoming messages
13        listener = topic.add_listener(lambda message: print(f"Received: {message.message}"))
14
15        # Pause execution for 10 seconds to listen for messages
16        time.sleep(10)
17
18        # Wait for user input to stop the listener (typically press Enter)
19        input("Press Enter to stop...\n")
20
21    finally:
22        # Shutdown the Hazelcast client
23        client.shutdown()
24
25 if __name__ == "__main__":
26     task4_2()
```

5. Bounded Queue

```
1 import hazelcast
2 import time
3
4 def produce():
5     # Create a Hazelcast client and connect to the cluster
6     client = hazelcast.HazelcastClient(cluster_members=["183.158.69.103:4704", "183.158.69.103:4705", "183.158.69.103:4706"])
7
8     try:
9         # Get the distributed queue named "bounded-queue"
10        queue = client.get_queue("bounded-queue").blocking()
11
12        # Produce items (numbers 1 to 100) and enqueue them
13        for i in range(1, 101):
14            queue.put(i)
15            print(f"Produced: {i}")
16            time.sleep(0.1) # Introducing a small delay for demonstration
17
18    finally:
19        # Shutdown the Hazelcast client to release resources
20        client.shutdown()
21
22 if __name__ == "__main__":
23     produce()
```

```
1 import hazelcast
2 import time
3
4 def consume():
5     # Create a Hazelcast client and connect to the cluster
6     client = hazelcast.HazelcastClient(cluster_members=["183.158.69.103:4704", "183.158.69.103:4705", "183.158.69.103:4706"])
7
8     try:
9         # Get the distributed queue named "bounded-queue"
10        queue = client.get_queue("bounded-queue").blocking()
11
12        # Continuously consume items from the queue
13        while True:
14            item = queue.take()
15            print(f"Consumed: {item}")
16
17    finally:
18        # Shutdown the Hazelcast client to release resources
19        client.shutdown()
20
21 if __name__ == "__main__":
22     consume()
```

На виході отримуємо 2 термінали з парними і непарними номерами.