

# Проектування високонавантажених систем

## Лабораторна робота №1

### Реалізація каунтера з використанням Hazelcast

Виконав: Мартиненко Денис ФБ-42мп

#### 1. Встановити і налаштувати Hazelcast.

```
PS C:\Users\denis\OneDrive\Рабочий стол\Новая папка>

PS C:\Users\denis\OneDrive\Рабочий стол\Новая папка>
PS C:\Users\denis\OneDrive\Рабочий стол\Новая папка> docker-compose up -d
time="2024-10-25T23:51:26+03:00" level=warning msg="C:\\Users\\denis\\OneDrive\\Рабочий стол\\Новая папка\\docker-
to avoid potential confusion"
- hazelcast-1 [██████] Pulling
- hc-management Pulling
- hazelcast-3 Pulling
PS C:\Users\denis\OneDrive\Рабочий стол\Новая папка> docker network create -d bridge hazelcast-network
708f738584948a2b3ade0b3a29f219d11aec6b61c7a56842caf37ab700ed7a35
PS C:\Users\denis\OneDrive\Рабочий стол\Новая папка> docker-compose up -d
time="2024-10-25T23:52:11+03:00" level=warning msg="C:\\Users\\denis\\OneDrive\\Рабочий стол\\Новая папка\\docker-
to avoid potential confusion"
[+] Running 17/4
✓ hazelcast-3 Pulled
✓ hazelcast-1 Pulled
✓ hc-management Pulled
✓ hazelcast-2 Pulled
[+] Running 4/4
✓ Container node1 Started
✓ Container node3 Started
✓ Container node2 Started
✓ Container lab1-mc Started
PS C:\Users\denis\OneDrive\Рабочий стол\Новая папка>
```

Код файлу docker-compose.yml:

```
version: "3"
name: lab1
services:
  hazelcast-1:
    network_mode: 'hazelcast-network'
    container_name: 'node1'
    environment:
      - 'HZ_NETWORK_PUBLICADDRESS=192.168.0.104:5701'
      - HZ_CLUSTERNAME=lab1
      - HAZELCAST_CONFIG=conf.xml
    volumes:
      - ./conf.xml:/opt/hazelcast/conf.xml
    ports:
      - '5701:5701'
    image: 'hazelcast/hazelcast:5.4.0'
  hazelcast-2:
    network_mode: 'hazelcast-network'
    container_name: 'node2'
    environment:
      - 'HZ_NETWORK_PUBLICADDRESS=192.168.0.104:5702'
```

```

- HZ_CLUSTERNAME=lab1
- HAZELCAST_CONFIG=conf.xml
volumes:
- ./conf.xml:/opt/hazelcast/conf.xml
ports:
- '5702:5701'
image: 'hazelcast/hazelcast:5.4.0'
hazelcast-3:
network_mode: 'hazelcast-network'
container_name: 'node3'
environment:
- 'HZ_NETWORK_PUBLICADDRESS=192.168.0.104:5703'
- HZ_CLUSTERNAME=lab1
- HAZELCAST_CONFIG=conf.xml
volumes:
- ./conf.xml:/opt/hazelcast/conf.xml
ports:
- '5703:5701'
image: 'hazelcast/hazelcast:5.4.0'
hc-management:
container_name: lab1-mc
image: hazelcast/management-center:5.4.0
network_mode: hazelcast-network
depends_on:
- hazelcast-1
- hazelcast-2
- hazelcast-3
ports:
- 8080:8080

```

2. Сконфігурувати і запустити 3 ноди (інстанси) об'єднані в кластер.

Переходимо на <http://localhost:8080/> та реєструємося в hazelcast management center.

Connect Cluster

Connect Directly    Upload Client Config

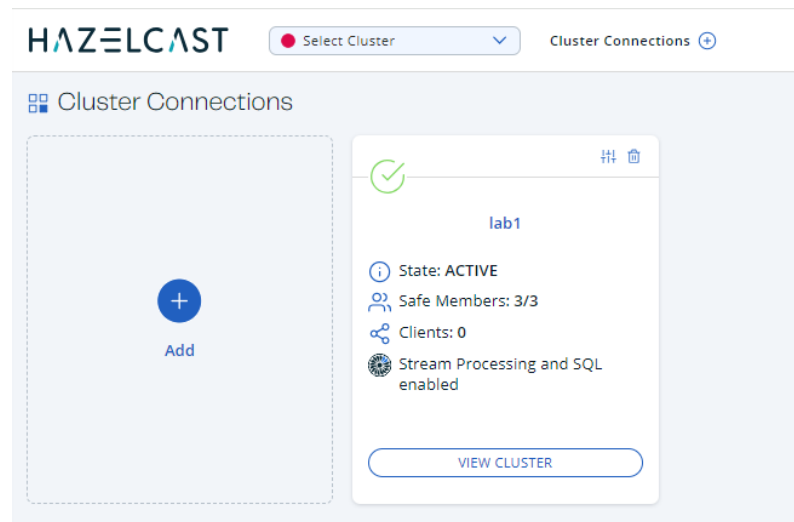
Connect Directly

Cluster Name ⓘ  
lab1

Member Addresses ⓘ  
192.168.0.104

Enabled ☒ ON

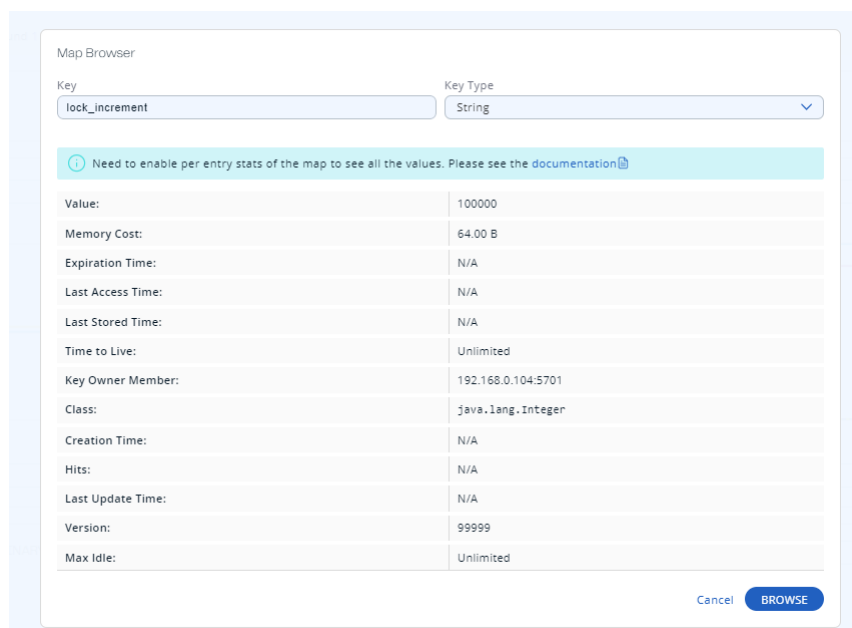
Cancel    CONNECT



3. Далі, на основі прикладу з Distributed Map, напишіть код який буде емулювати інкремент значення для одного й того самого ключа у циклі до 10К. Це необхідно робити у 10 потоках.

```
def increment_with_lock(key):
    """Функція для інкрементації з локальним блокуванням."""
    global global_counter
    lock = Lock()
    for _ in range(10000):
        with lock:
            global_counter += 1
            dist_map.put(key, global_counter)
```

```
--- Виконання завдання: lock_increment ---
Час виконання: 59.6010 секунд
Фінальне значення: 100000
```



- Це очікуваний результат, оскільки використовується локальне блокування для синхронізації потоків.

4. На основі прикладу реалізуйте каунтер без блокувань. Поміряйте час виконання, та подивитися чи коректне кінцеве значення каунтера ви отримаєте.

```
def simple_increment(key):  
    """Функція для інкрементації без блокування."""  
    dist_map.put(key, 0)  
    for _ in range(10000):  
        current_value = dist_map.get(key)  
        dist_map.put(key, current_value + 1)
```

```
--- Виконання завдання: simple_increment ---  
Час виконання: 82.2170 секунд  
Фінальне значення: 10464
```

Map Browser

Key:  Key Type:

*Need to enable per entry stats of the map to see all the values. Please see the documentation*

Value:	10464
Memory Cost:	64.00 B
Expiration Time:	N/A
Last Access Time:	N/A
Last Stored Time:	N/A
Time to Live:	Unlimited
Key Owner Member:	192.168.0.104-5701
Class:	java.lang.Integer
Creation Time:	N/A
Hits:	N/A
Last Update Time:	N/A
Version:	100009
Max Idle:	Unlimited

[Cancel](#) [BROWSE](#)

- Фінальне значення лічильника тут не відповідає очікуваному значенню 100000. Це підтверджує проблему, пов'язану з гонками потоків, що виникають при відсутності блокувань.

5. На основі прикладу реалізуйте каунтер з використанням песимістичного блокування. Поміряйте час виконання, та подивитися чи коректне кінцеве значення каунтера ви отримаєте.

```
def pessimistic_increment(key):  
    """Функція для інкрементації з песимістичним блокуванням."""  
    if not dist_map.contains_key(key):  
        dist_map.put(key, 0)  
    for _ in range(10000):  
        dist_map.lock(key)  
        try:  
            current_value = dist_map.get(key)  
            dist_map.put(key, current_value + 1)  
        finally:  
            dist_map.unlock(key)
```

```
--- Виконання завдання: pessimistic_increment ---  
Час виконання: 857.0108 секунд  
Фінальне значення: 100000
```


Map Browser

Key

pessimistic\_increment

Key Type

String

 Need to enable per entry stats of the map to see all the values. Please see the [documentation](#)

Value:	100000
Memory Cost:	64.00 B
Expiration Time:	N/A
Last Access Time:	N/A
Last Stored Time:	N/A
Time to Live:	Unlimited
Key Owner Member:	192.168.0.104:5701
Class:	java.lang.Integer
Creation Time:	N/A
Hits:	N/A
Last Update Time:	N/A
Version:	100009
Max Idle:	Unlimited

Cancel

BROWSE

- Це очікуваний результат, але час виконання є значно більшим, що характерно для песимістичних блокувань, які значно уповільнюють обробку.

6. На основі прикладу реалізуйте каунтер з використанням оптимістичного блокування. Поміряйте час виконання, та подивитися чи коректне кінцеве значення каунтера ви отримаєте.

```
def optimistic_increment(key):  
    """Функція для інкрементації з оптимістичним блокуванням."""  
    if not dist_map.contains_key(key):  
        dist_map.put(key, 0)  
    for _ in range(10000):  
        while True:  
            current_value = dist_map.get(key)  
            new_value = current_value + 1  
            if dist_map.replace_if_same(key, current_value, new_value):  
                break
```

```
--- Виконання завдання: optimistic_increment ---  
Час виконання: 1503.9769 секунд  
Фінальне значення: 100000
```

Map Browser

Key:  Key Type:

Need to enable per entry stats of the map to see all the values. Please see the [documentation](#)

Value:	100000
Memory Cost:	64.00 B
Expiration Time:	N/A
Last Access Time:	N/A
Last Stored Time:	N/A
Time to Live:	Unlimited
Key Owner Member:	192.168.0.104:5701
Class:	java.lang.Integer
Creation Time:	N/A
Hits:	N/A
Last Update Time:	N/A
Version:	100007
Max Idle:	Unlimited

[Cancel](#) [BROWSE](#)

- Час виконання ще більший, оскільки оптимістичне блокування зазвичай вимагає кількох повторних спроб у разі колізій, що збільшує загальний час виконання.

7. Реалізуйте каунтер з використанням `IAAtomicLong` та увімкнувши підтимку CP Sysbssystem на основі трьох нод.

- Для виконання цього завдання файл `conf.xml` містить наступну конфігурацію:

```
<cp-subsystem>
  <cp-member-count>3</cp-member-count>
  <group-size>3</group-size>
  <session-time-to-live-seconds>300</session-time-to-live-seconds>
  <session-heartbeat-interval-seconds>5</session-heartbeat-interval-seconds>
  <missing-cp-member-auto-removal-seconds>14400</missing-cp-member-auto-removal-seconds>
  <fail-on-indeterminate-operation-state>false</fail-on-indeterminate-operation-state>
  <persistence-enabled>false</persistence-enabled>
  <base-dir>/opt/hazelcast/cp-data</base-dir>
  <data-load-timeout-seconds>120</data-load-timeout-seconds>
  <cp-member-priority>0</cp-member-priority>
  <raft-algorithm>
    <leader-election-timeout-in-millis>2000</leader-election-timeout-in-millis>
    <leader-heartbeat-period-in-millis>5000</leader-heartbeat-period-in-millis>
    <max-missed-leader-heartbeat-count>5</max-missed-leader-heartbeat-count>
    <append-request-max-entry-count>100</append-request-max-entry-count>
    <commit-index-advance-count-to-snapshot>10000</commit-index-advance-count-to-snapshot>
    <uncommitted-entry-count-to-reject-new-appends>100</uncommitted-entry-count-to-reject-new-appends>
    <append-request-backoff-timeout-in-millis>100</append-request-backoff-timeout-in-millis>
  </raft-algorithm>
  <semaphores/>
  <locks/>
</cp-subsystem>
```

CP Subsystem

All CP members are accessible

CP Members

Configured 3 / 3 Accessible

Promote Member to CP

PROMOTE

Remove CP Member

REMOVE

Restart CP Subsystem

RESTART

CP Subsystem Stats

Member	Nodes	Destroyed Groups	Active Members	Missing Members
192.168.0.104:5703	2	0	3	0
192.168.0.104:5702	2	0	3	0
192.168.0.104:5701	2	0	3	0

1 - 3 of 3 Rows 10

CP Group Metrics

Group Name	Group Id	Member Count
METADATA	0	3
default	9658	3

1 - 2 of 2 Rows 10

```
def atomic_increment_with_cp(key):
    """Функція для інкрементації з використанням атомарного лічильника в CP Subsystem."""
    # Отримання IAtomicLong, яке підтримується CP Subsystem
    atomic_long = client.cp_subsystem.get_atomic_long(key).blocking()
    for _ in range(10000):
        atomic_long.add_and_get(1)

def execute_task_with_cp(task_function, task_name):
    """Функція для виконання завдання в багатопоточному режимі."""
    print(f"--- Виконання завдання: {task_name} ---")
    start_time = time.time()
    threads = [Thread(target=task_function, args=(task_name,)) for _ in range(10)]

    for thread in threads:
        thread.start()
    for thread in threads:
        thread.join()

    elapsed_time = time.time() - start_time
    final_value = client.cp_subsystem.get_atomic_long(task_name).blocking().get()
    print(f"Час виконання: {elapsed_time:.4f} секунд")
    print(f"Фінальне значення: {final_value}")

def reset_atomic_counter_with_cp(key):
    """Скидання атомарного лічильника для CP Subsystem."""
    atomic_long = client.cp_subsystem.get_atomic_long(key).blocking()
    atomic_long.set(0)
```

Name	Group	Value
atomic_increment	default	300 000
cp_atomic_increment	default	100 000

## Logs:

```
PS C:\Users\denis> docker logs node1 > node1.log
>> docker logs node2 > node2.log
>> docker logs node3 > node3.log
>>
```

```
#####
# JAVA=/usr/bin/java
# JAVA_OPTS=-add-modules java.se --add-exports java.base/jdk.internal.ref=ALL-UNNAMED --add-opens java.base/java.lang=ALL-
# CLASSPATH=/opt/hazelcast/*:/opt/hazelcast/lib:/opt/hazelcast/lib/*:/opt/hazelcast/bin/user-lib:/opt/hazelcast/bin/user-ll
#####
2024-10-25 21:05:00,119 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.c.AbstractConfigLocator[[m]: Loading configuration '/opt/ha
2024-10-25 21:05:00,124 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.c.AbstractConfigLocator[[m]: Using configuration file at /o
2024-10-25 21:05:02,871 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.c.ExternalConfigurationOverride[[m]: Detected external co
2024-10-25 21:05:03,049 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.AddressPicker[[m]: [LOCAL] [lab1] [5.4.0] Using public addre
2024-10-25 21:05:03,175 [[32m INFO[[m] [[36mmain[[m] [[34m.h.s.s.log[[m]: [192.168.0.104]:5701 [lab1] [5.4.0]
#####
      o      o      o---o      o---o      o---o      o      o---o      o---o
      |      |      |      |      |      |      |      |      |
      o---o      o      o      o---o      o      o      o---o      o
      |      |      |      |      |      |      |      |      |
      o      o      *      o      o---o      o---o      o---o      o
      |      |      |      |      |      |      |      |      |
      o      o      o      o      o      o      o      o      o
#####
2024-10-25 21:05:03,176 [[32m INFO[[m] [[36mmain[[m] [[34m.h.system[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Copyright (c)
2024-10-25 21:05:03,185 [[32m INFO[[m] [[36mmain[[m] [[34m.h.system[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Hazelcast Plat
2024-10-25 21:05:03,187 [[32m INFO[[m] [[36mmain[[m] [[34m.h.system[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Cluster name:
2024-10-25 21:05:03,189 [[32m INFO[[m] [[36mmain[[m] [[34m.h.system[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Integrity Che
2024-10-25 21:05:03,190 [[32m INFO[[m] [[36mmain[[m] [[34m.h.system[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Jet is enable
2024-10-25 21:05:22,992 [[32m INFO[[m] [[36mmain[[m] [[34m.h.s.security[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Enable DEF
2024-10-25 21:05:23,721 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.i.Node[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Using Multi
2024-10-25 21:05:23,726 [[32m INFO[[m] [[36mmain[[m] [[34m.h.c.CPSubsystem[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] CP Sub
2024-10-25 21:05:32,585 [[32m INFO[[m] [[36mmain[[m] [[34m.h.j.i.JetServiceBackend[[m]: [192.168.0.104]:5701 [lab1] [5.4.0]
2024-10-25 21:05:35,444 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.d.Diagnostics[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] Diagn
2024-10-25 21:05:35,906 [[32m INFO[[m] [[36mmain[[m] [[34m.h.c.LifecycleService[[m]: [192.168.0.104]:5701 [lab1] [5.4.0] [:
2024-10-25 21:05:39,132 [[32m INFO[[m] [[36mmain[[m] [[34m.h.i.c.MulticastJoiner[[m]: [192.168.0.104]:5701 [lab1] [5.4.0]
2024-10-25 21:05:39,769 [[32m INFO[[m] [[36mhz.angry_banzai.IO.thread-in-0[[m] [[34m.h.i.s.t.TcpServerConnection[[m]: [192
2024-10-25 21:05:41,298 [[32m INFO[[m] [[36mhz.angry_banzai.IO.thread-in-1[[m] [[34m.h.i.s.t.TcpServerConnection[[m]: [192
2024-10-25 21:05:41,342 [[32m INFO[[m] [[36mhz.angry_banzai.generic-operation.thread-1[[m] [[34m.h.i.c.ClusterService[[m]:
#####
Members [size:3, ver:3] [
  Member [192.168.0.104]:5703 - b72c2609-0113-4700-b7fe-066168de02a0
  Member [192.168.0.104]:5702 - 0ce95cad-15e9-4da5-9476-2207708c4373
  Member [192.168.0.104]:5701 - 76b22cfe-8adc-4993-aabd-218abc073c2a this
#####
```



