УНИВЕРСИТЕТ ИТМО

Факультет программной
инженерии и компьютерной техники

Лабораторная работа № 4
по дисциплине
«Методы и средства программной инженерии»

Вариант 666333

Преподаватель: Покид Александр Владимирович
Выполнили: Геллер Л. А., Блинова М. А.
Группа: P3230

Санкт-Петербург
2021 г.

Задание

1. Для своей программы из лабораторной работы #4 по дисциплине "Программирование интернет-приложений" реализовать:

- MBean, считающий общее число установленных пользователем точек, а также число точек, попадающих в область. В случае, если координаты установленной пользователем точки вышли за пределы отображаемой области координатной плоскости, разработанный MBean должен отправлять оповещение об этом событии.
- MBean, определяющий процентное отношение "попаданий" к общему числу кликов пользователя по координатной плоскости.

2. С помощью утилиты JConsole провести мониторинг программы:

- Снять показания MBean-классов, разработанных в ходе выполнения задания 1.
- Определить имена всех потоков, выполняющихся при запуске программы.

3. С помощью утилиты VisualVM провести мониторинг и профилирование программы:

- Снять график изменения показаний MBean-классов, разработанных в ходе выполнения задания 1, с течением времени.
- Определить имя класса, объекты которого занимают наибольший объём памяти JVM; определить пользовательский класс, в экземплярах которого находятся эти объекты.

4. Получить HeapDump, и с помощью утилиты VisualVM локализовать и устранить "утечку памяти" в программе ниже

1. MBean

- PointsCount

```
@Component
public class PointCount extends NotificationBroadcasterSupport implements PointCountMBean {

    private long sequenceNumber = 1;
    private int shots = 0;
    private int hits = 0;

    @Override
    public int getTotal() { return shots; }
    @Override
    public int getHits() { return hits; }
    public void setTotal(int n) { shots = n; }
    public void setHits(int n) { hits = n; }

    public void takeAShot(Data point) {
        shots++;
        if (point.getResult()) hits++;
        if (Math.abs(point.getX()) > point.getR()*1.15 || Math.abs(point.getY()) > point.getR()*1.15) {
            System.out.println("Out of displayed area!");
            Notification n = new Notification( type: "OutOfDisplayedArea", this.getClass().getName(),
                    sequenceNumber,
                    message: "Point coordinates (\" + point.getX() + \";\" + point.getY() + \") " +
                            "are out of displayed area of coordinate plane");
            sendNotification(n);
            sequenceNumber++;
        }
    }
```

```
@Override
public MBeanNotificationInfo[] getNotificationInfo() {
    String[] types = new String[]{AttributeChangeNotification.ATTRIBUTE_CHANGE};
    String name = AttributeChangeNotification.class.getName();
    String description = "Point coordinates are out of displayed area of coordinate plane";
    MBeanNotificationInfo info = new MBeanNotificationInfo(types, name, description);
    return new MBeanNotificationInfo[]{info};
}
```

- HitPercent

```
@Component
public class HitPercent extends NotificationBroadcasterSupport implements HitPercentMBean {

    private double percent = 0;

    @Override
    public double getPercent() { return percent; }

    public void setPercent(int total, int hits) { percent = (double)hits/total; }

    public void updatePercent(Data point, int total) {
        if (point.getResult()) percent = (percent * total + 1) / (total + 1);
        else percent *= (double) total / (total + 1);
    }
```

2. JConsole

Показания MBean-классов:

PointsCount:

Attribute values

Name	Value
Hits	**8**
Total	**12**

HitPercent:

Attribute values

Name	Value
Percent	**0.6666666666666666**

Notifications:

Notification buffer

TimeStamp	Type	UserData	SeqNum	Message	Event	Source
11:36:57:189	OutOfDisplayedA...		2	Point coordinates...	javax.manageme...	web.lab4.mbeans.Poi...
11:36:49:181	OutOfDisplayedA...		1	Point coordinates...	javax.manageme...	web.lab4.mbeans.Poi...

Point coordinates (1.5;0.0) are out of displayed area of

Количество классов, загруженных в JVM в процессе выполнения программы:

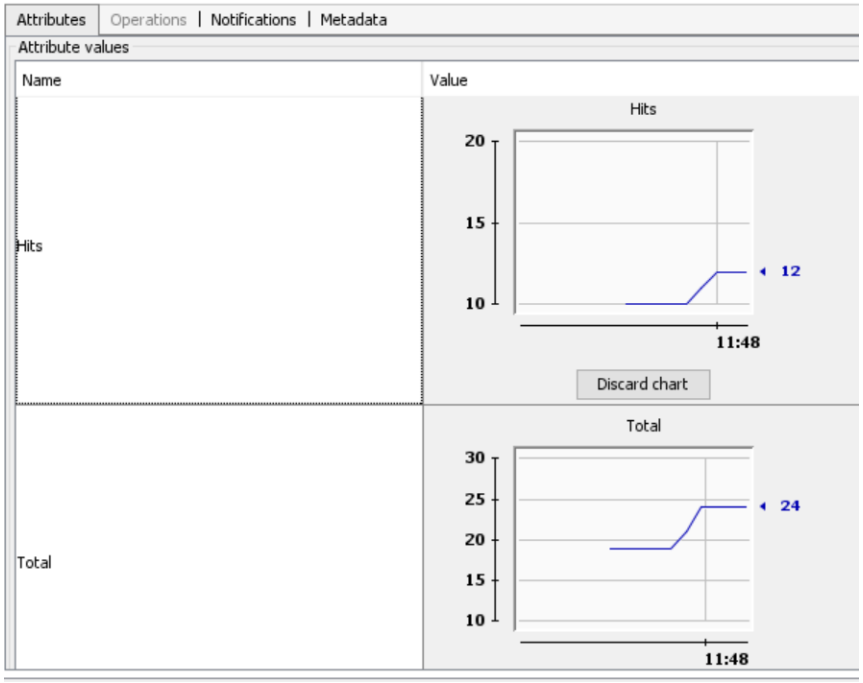
Current classes loaded: 64 064

Total classes loaded: 64 106

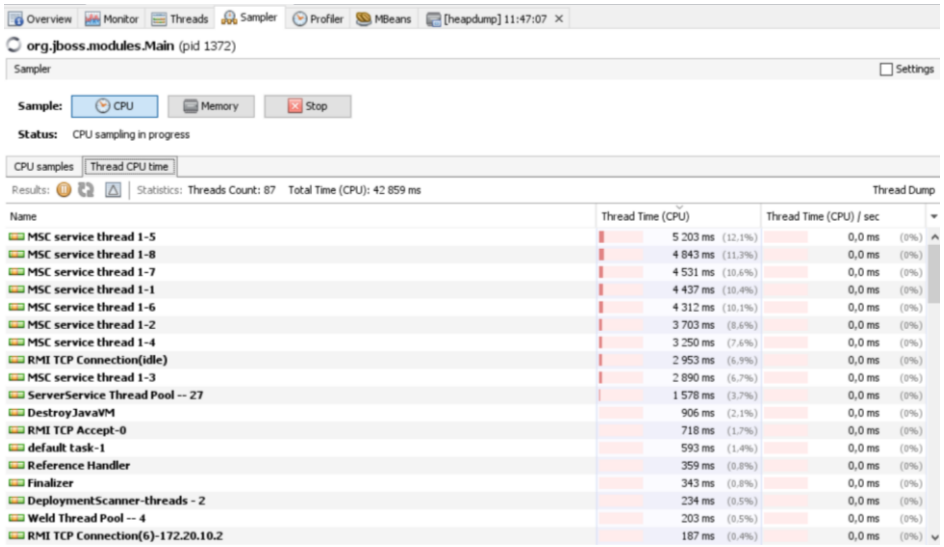
Total classes unloaded: 42

3. VisualVM

График изменения показаний MBean-классов:



Поток, потребляющий наибольший процент времени CPU:

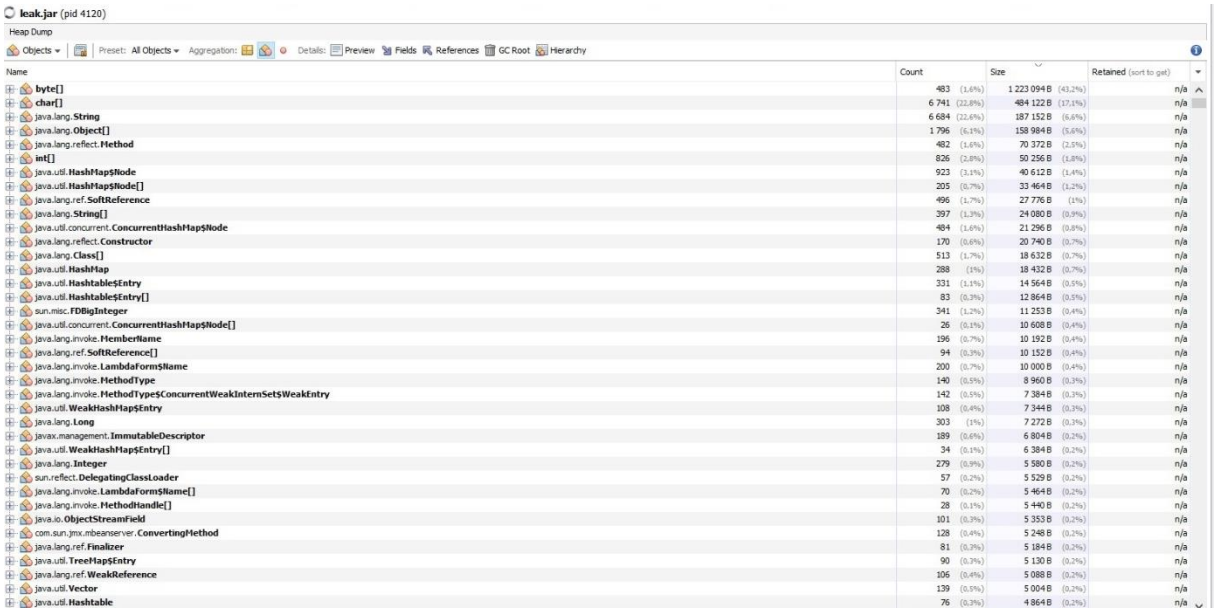


Больше всего процессорного времени занимает поток MSC service thread 1-5

Утечка памяти

Проблема: предоставленная программа падает с OutOfMemoryException

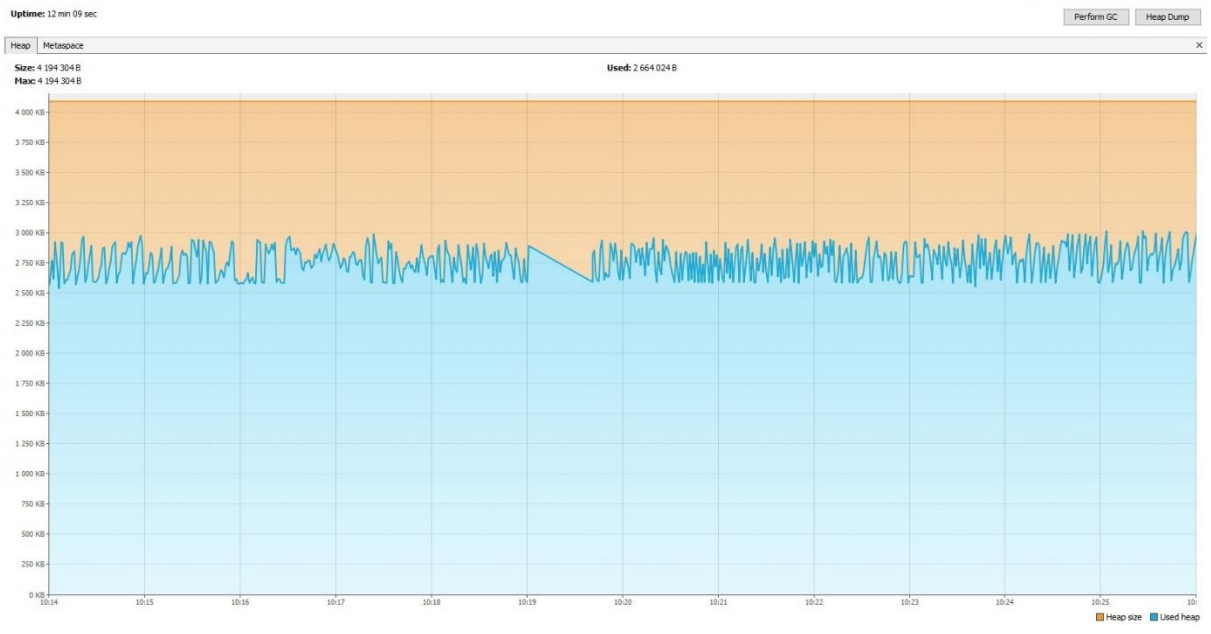
HeapDump:



Решение:

- очищать буферы
- убрать ненужные String.valueOf()

Результат:



Вывод

При выполнении данной лабораторной работы мы написали свой первый MBean и познакомились с JConsole и VisualVM