

федеральное государственное автономное образовательное
учреждение высшего образования «Национальный исследовательский
университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №4

Вариант 815

Студент:
Кулагин Вячеслав, Р3209
Преподаватель:
Карташев Владимир
Сергеевич

Санкт-Петербург
2025

Оглавление

Задание	4
1. Для своей программы из лабораторной работы #3 по дисциплине "Веб-программирование" реализовать:	4
МBean, считающий общее число установленных пользователем точек, а также число точек, попадающих в область. В случае, если пользователь совершил 4 "промаха" подряд, разработанный MBean должен отправлять оповещение об этом событии.	4
MBean, определяющий процентное отношение "промахов" к общему числу кликов пользователя по координатной плоскости.	4
2. С помощью утилиты JConsole провести мониторинг программы:	4
Снять показания MBean-классов, разработанных в ходе выполнения задания 1.	4
Определить имена всех потоков, выполняющихся при запуске программы.....	4
3. С помощью утилиты VisualVM провести мониторинг и профилирование программы:.....	4
Снять график изменения показаний MBean-классов, разработанных в ходе выполнения задания 1, с течением времени.	4
Определить имя класса, объекты которого занимают наибольший объём памяти JVM; определить пользовательский класс, в экземплярах которого находятся эти объекты.....	4
4. С помощью утилиты VisualVM и профилировщика IDE NetBeans, Eclipse или Idea локализовать и устранить проблемы с производительностью в программе. По результатам локализации и устранения проблемы необходимо составить отчёт, в котором должна содержаться следующая информация:	4
Описание выявленной проблемы.	4
Описание путей устранения выявленной проблемы.	4
Подробное (со скриншотами) описание алгоритма действий, который позволил выявить и локализовать проблему.	4
Студент должен обеспечить возможность воспроизведения процесса поиска и локализации проблемы по требованию преподавателя.	4
Часть 1	4

jboss-modules.jar (pid 84570)

Overview Saved data Details

PID: 84570
 Host: 0.0.0.0
 Main class: jboss-modules.jar
 Arguments: -mp /home/studs/408946/wildfly/wildfly-33.0.1.Final/modules org.jboss.as.standalone -Djboss.home.dir=/home/studs/s409517/wildfly/wildfly-33.0.1.Final -Djboss.server.base

JVM: OpenJDK 64-Bit Server VM (17.0.12+7-1, mixed mode, sharing)
 Java version 17.0.12 2024-07-16, vendor OpenJDK BSD Porting Team
 Java Home: /usr/local/openjdk17
 JVM Flags: <none>

Heap dump on OOME: disabled

Saved data	JVM arguments	System properties
Thread Dumps: 0 Heap Dumps: 0 Profiler Snapshots: 0 JFR Snapshots: 0	-D[Standalone] -Djdk.serialFilter=maxbytes=10485760,maxdepth=128,maxarray=100000,maxrefs=300000 -Xms64m -Xmx512m -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256m -Djava.net.preferIPv4Stack=true -Djboss.modules.system.pkgs=org.jboss.byteman -Djava.awt.headless=true --add-exports=java.desktop/sun.awt=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.ldap=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.url.ldap=ALL-UNNAMED --add-exports=java.naming/com.sun.jndi.url.ladps=ALL-UNNAMED --add-exports=jdk.naming.dns/com.sun.jndi.dns=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/java.lang.invoke=ALL-UNNAMED --add-opens=java.base/java.lang.reflect=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/java.net=ALL-UNNAMED --add-opens=java.base/java.security=ALL-UNNAMED --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.util.concurrent=ALL-UNNAMED --add-opens=java.management/javax.management=ALL-UNNAMED --add-opens=java.naming/javax.naming=ALL-UNNAMED -Djava.security.manager=allow -Dorg.jboss.boot.log.file=/home/studs/408946/wildfly/wildfly-33.0.1.Final/standalone/log/server.log -Dlogging.configuration=file:/home/studs/408946/wildfly/wildfly-33.0.1.Final/standalone/configuration/logging.properties -XX:MaxHeapSize=1G -XX:MaxMetaspaceSize=128m	

12

Исследование на утечки	13
Вывод	15

Задание

1. Для своей программы из лабораторной работы #3 по дисциплине "Веб-программирование" реализовать:

MBean, считающий общее число установленных пользователем точек, а также число точек, попадающих в область. В случае, если пользователь совершил 4 "промаха" подряд, разработанный MBean должен отправлять оповещение об этом событии.

MBean, определяющий процентное отношение "промахов" 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. С помощью утилиты VisualVM и профилировщика IDE NetBeans, Eclipse или Idea локализовать и устранить проблемы с производительностью в программе. По результатам локализации и устранения проблемы необходимо составить отчёт, в котором должна содержаться следующая информация:

Описание выявленной проблемы.

Описание путей устранения выявленной проблемы.

Подробное (со скриншотами) описание алгоритма действий, который позволил выявить и локализовать проблему.

Студент должен обеспечить возможность воспроизведения процесса поиска и локализации проблемы по требованию преподавателя.

Часть 1

Web.lab.jmx.MBeanRegistrar

```
package web.lab.jmx;

import jakarta.annotation.PostConstruct;
import jakarta.enterprise.context.ApplicationScoped;
import jakarta.inject.Inject;
import javax.management.*;
import java.lang.management.ManagementFactory;
import java.util.logging.Level;
import java.util.logging.Logger;

@ApplicationScoped
public class MBeanRegistrar {
    private static final Logger logger = Logger.getLogger(MBeanRegistrar.class.getName());

    @Inject private PointsStats pointsStats;
    @Inject private MissRateStats missRateStats;

    @PostConstruct
    public void register() {
        try {
            MBeanServer mbs = ManagementFactory.getPlatformMBeanServer();
            mbs.registerMBean(pointsStats, new ObjectName("web31:type=PointStats"));
            mbs.registerMBean(missRateStats, new ObjectName("web31:type=MissRatio"));
        }
    }
}
```

```

        } catch (Exception e) {
            logger.log(Level.SEVERE, "Failed to register MBeans", e);
        }
    }

    private void registerMBean(MBeanServer server, Object mbean, String object-
Name) {
        try {
            ObjectName name = new ObjectName(objectName);
            if (!server.isRegistered(name)) {
                server.registerMBean(mbean, name);
                logger.info(() -> "Successfully registered MBean: " + object-
Name);
            } else {
                logger.warning(() -> "MBean already registered: " + objectName);
            }
        } catch (InstanceAlreadyExistsException e) {
            logger.warning(() -> "MBean already exists: " + objectName);
        } catch (MalformedObjectNameException e) {
            logger.severe(() -> "Invalid object name format: " + objectName);
        } catch (Exception e) {
            logger.log(Level.SEVERE, "Error registering MBean: " + objectName,
e);
        }
    }
}
}

```

Web.lab.jmx.MissRateStats

```

package web.lab.jmx;

import jakarta.enterprise.context.ApplicationScoped;
import java.util.concurrent.atomic.AtomicInteger;

@ApplicationScoped
public class MissRateStats implements MissRateStatsMBean {
    private final AtomicInteger totalClicks = new AtomicInteger();
    private final AtomicInteger missCount = new AtomicInteger();

    public void recordClick(boolean hit) {
        totalClicks.incrementAndGet();
        if (!hit) {
            missCount.incrementAndGet();
        }
    }

    @Override
    public int getTotalClicks() {
        return totalClicks.get();
    }

    @Override
    public int getMissCount() {
        return missCount.get();
    }

    @Override
    public double getMissPercentage() {
        int attempts = totalClicks.get();
        return attempts == 0 ? 0.0 : (100.0 * missCount.get()) / attempts;
    }
}

```

Web.lab.jmx.MissRateStatsMBean

```
package web.lab.jmx;

public interface MissRateStatsMBean {
    int getTotalClicks();
    int getMissCount();
    double getMissPercentage();
}
```

Web.lab.jmx.PointStats

```
package web.lab.jmx;

import jakarta.enterprise.context.ApplicationScoped;
import javax.management.*;
import java.util.concurrent.atomic.AtomicInteger;
import java.util.concurrent.atomic.AtomicLong;

@ApplicationScoped
public class PointsStats extends NotificationBroadcasterSupport implements PointsStatsMBean {
    private final AtomicInteger totalClicks = new AtomicInteger();
    private final AtomicInteger hitsCount = new AtomicInteger();
    private final AtomicInteger consecutiveMisses = new AtomicInteger();
    private final AtomicLong sequenceNumber = new AtomicLong(1);

    public void recordClick(boolean hit) {
        totalClicks.incrementAndGet();

        if (hit) {
            hitsCount.incrementAndGet();
            consecutiveMisses.set(0);
        } else {
            int currentMisses = consecutiveMisses.incrementAndGet();
            if (currentMisses >= 4) {
                try {
                    Notification notification = new Notification(
                        "web.lab.jmx.PointsStats.misses",
                        this,
                        sequenceNumber.getAndIncrement(),
                        System.currentTimeMillis(),
                        "4 промаха подряд"
                    );
                    sendNotification(notification);
                } finally {
                    consecutiveMisses.set(0);
                }
            }
        }
    }

    @Override
    public MBeanNotificationInfo[] getNotificationInfo() {
        return new MBeanNotificationInfo[] {
            new MBeanNotificationInfo(
                new String[]{"web.lab.jmx.PointsStats.misses"},
                Notification.class.getName(),
                "4 consecutive misses notification"
            )
        };
    }
}
```

```

    }

    @Override
    public int getTotalClicks() {
        return totalClicks.get();
    }

    @Override
    public int getHitsCount() {
        return hitsCount.get();
    }
}

```

Web.lab.jmx.PointsStatsMBean

```

package web.lab.jmx;

public interface PointsStatsMBean {
    int getTotalClicks();
    int getHitsCount();
}

```

Сохранение значений в ResultBean:

```

package web.lab.web31;

import com.google.gson.Gson;
import jakarta.annotation.PostConstruct;
import jakarta.enterprise.context.ApplicationScoped;
import jakarta.inject.Inject;
import jakarta.inject.Named;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import jakarta.persistence.TypedQuery;
import jakarta.transaction.Transactional;
import lombok.Getter;
import lombok.Setter;
import web.lab.entity.ResultEntity;
import web.lab.entity.Point;
import web.lab.jmx.MBeanRegistrar;
import web.lab.jmx.MissRateStats;
import web.lab.jmx.PointsStats;

import java.io.Serializable;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.List;

@Named
@ApplicationScoped
@Getter
@Setter
public class ResultsBean implements Serializable {

    @Inject
    private MBeanRegistrar mbeanRegistrar;

    @PersistenceContext(unitName = "my-persistence-unit")
    private EntityManager entityManager;

    private List<Point> results = new ArrayList<>();
}

```

```

private boolean isInitialized = false;

@PostConstruct
private void init() {
    if (!isInitialized) {
        loadResults();
        isInitialized = true;
    }
}

private void loadResults() {
    TypedQuery<ResultEntity> query = entityManager.createQuery(
        "SELECT r FROM ResultEntity r", ResultEntity.class);

    List<ResultEntity> resultEntities = query.getResultList();
    results = new ArrayList<>();
    for (ResultEntity entity : resultEntities) {
        results.add(new Point(entity.getX(), entity.getY(), entity.getR(),
entity.isHit()));
    }
}

@Inject
private PointsStats stats;

@Inject
private MissRateStats missRateStats;

@Transactional
public void addResult(Point point) {
    ResultEntity resultEntity = ResultEntity.builder()
        .x(point.getX())
        .y(point.getY())
        .r(point.getR())
        .hit(point.isHit())
        .timestamp(LocalDateTime.now())
        .build();

    boolean hit = point.isHit();
    stats.recordClick(hit);
    missRateStats.recordClick(hit);

    entityManager.persist(resultEntity);
    results.add(new Point(resultEntity.getX(), resultEntity.getY(), resultEntity.getR(), resultEntity.isHit()));
}

public String getJsonResults() {
    return new Gson().toJson(results);
}
}

```

Overview Memory Threads Classes VM Summary MBeans

MBeanInfo

Name	Value
Info:	
ObjectName	web_lab3:type=MissRatio
ClassName	org.server.beans.MissRatio
Description	Information on the management interface of the MBean
Constructor-0:	
Name	org.server.beans.MissRatio
Description	Public constructor of the MBean

Descriptor

Name	Value
Info:	
immutableInfo	true
interfaceClassName	org.server.beans.MissRatioMBean
mxbean	false

The screenshot shows the JBoss JMX Management tool interface. The left pane displays a tree view of MBeans under the 'web_lab3' domain. The 'MissRatio' MBean is selected. The right pane contains two tables: 'MBeanInfo' and 'Descriptor'. The 'MBeanInfo' table lists the management interface details, including the object name, class name, and a description of the interface. The 'Descriptor' table provides information about the descriptor, such as immutable status and the interface class name.

Overview Memory Threads Classes VM Summary MBeans

MBeanInfo

Name	Value
Info:	
ObjectName	web_lab3:type=PointStats
ClassName	org.server.beans.PointStats
Description	Information on the management interface of the MBean
Constructor-0:	
Name	org.server.beans.PointStats
Description	Public constructor of the MBean

Descriptor

Name	Value
Info:	
immutableInfo	true
interfaceClassName	org.server.beans.PointStatsMBean
mxbean	false

This screenshot is identical to the one above, showing the 'PointStats' MBean details in the JBoss JMX Management tool. The tree view on the left shows the 'PointStats' MBean selected. The right pane displays the 'MBeanInfo' and 'Descriptor' tables with the same data as the first screenshot, providing details about the management interface and descriptor.

Attribute values

Name	Value
HitPoints	22
MissPercentage	45.0
TotalPoints	40

Attribute values

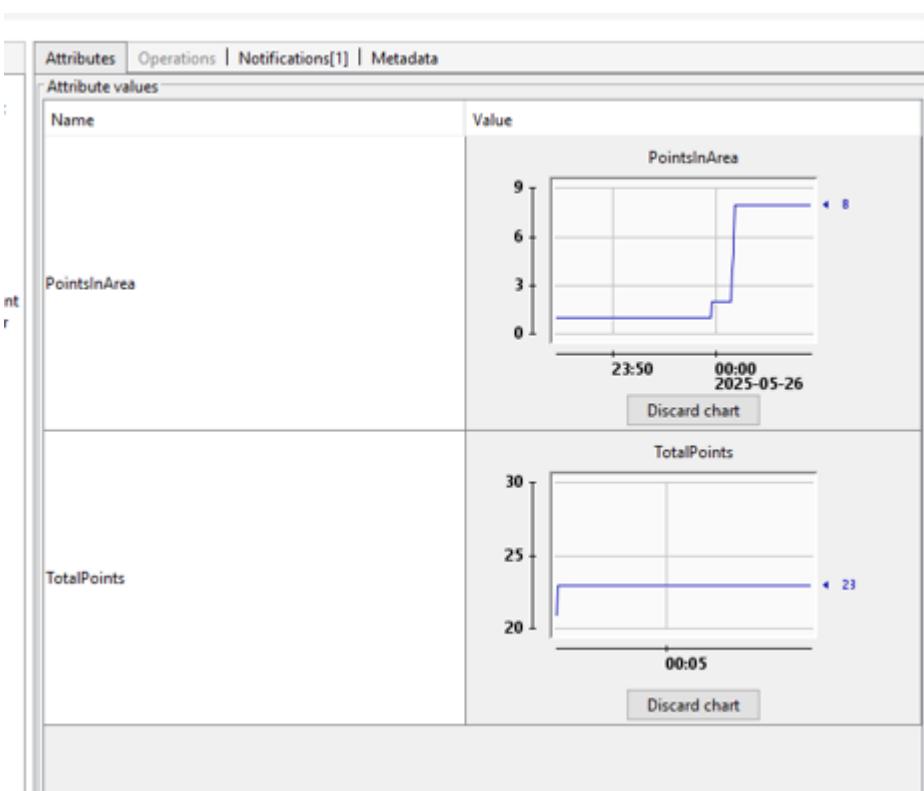
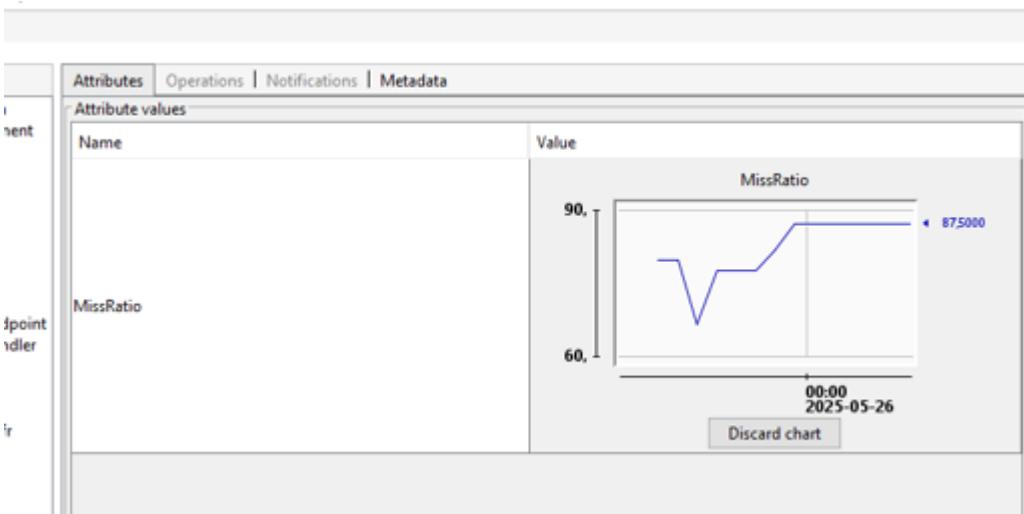
Name	Value
HitPoints	22
TotalPoints	40

Connection name:	pid: 48591 jboss-modules.jar -mp /home/felipy/otp-lab4/wildfly-preview-26.1.3.Final/modules org.jboss.as.standalone -Djboss.home.dir=/home/felipy/otp-lab4/wildfly-preview-26.1.3.Final -Djboss.server.base.dir=/home/felipy/otp-lab4/wildfly-preview-26.1.3.Final/standalone -b 0.0.0.0 -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.authenticate=false -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.port=28002 -Dcom.sun.management.jmxremote.rmi.port=28002 -Djava.rmi.server.hostname=0.0.0.0 -Dboss.bind.address.management=0.0.0.0 -Dboss.bind.address=0.0.0.0	Uptime: 10 minutes
Virtual Machine:	OpenJDK 64-Bit Server VM version 21.0.7+6-Ubuntu12.04	Process CPU time: 42.540 seconds
Vendor:	Ubuntu	JIT compiler: HotSpot 64-Bit Tiered Compilers
Name:	48591@DESKTOP-4ICOU9J	Total compile time: 26.412 seconds
Live threads:	85	Current classes loaded: 23,106
Peak:	155	Total classes loaded: 23,117
Daemon threads:	29	Total classes unloaded: 11
Total threads started:	166	
Current heap size:	186,197 kbytes	Committed memory: 229,376 kbytes
Maximum heap size:	524,288 kbytes	Pending finalization: 0 objects
Garbage collector:	Name = 'G1 Young Generation', Collections = 52, Total time spent = 0.184 seconds	
Garbage collector:	Name = 'G1 Concurrent GC', Collections = 16, Total time spent = 0.035 seconds	
Garbage collector:	Name = 'G1 Old Generation', Collections = 0, Total time spent = 0.000 seconds	
Operating System:	Linux 6.6.87.1-microsoft-standard-WSL2	Total physical memory: 7,704,794 kbytes
Architecture:	amd64	Free physical memory: 5,188,368 kbytes
Number of processors:	16	Total swap space: 2,097,152 kbytes
Committed virtual memory:	1,715,500 kbytes	Free swap space: 2,097,152 kbytes
VM arguments:	-Dstandalone -Xss64m -Xms512m -XX:MaxMetaspaceSize=96M -Djava.net.preferIPv4Stack=true -Dboss.modules.system.plugins=org.jboss.lyra:type=jaxrs,avt,bessless=true -add-export=javax.management.com.sun.jndi.ldap:ALL-UNNAMED -add-export=javax.naming.com.sun.jndi.url.http:ALL-UNNAMED -add-open=java.base/javax.lang.invoke=ALL-UNNAMED -add-open=java.base/javax.lang.reflect=ALL-UNNAMED -add-open=java.base/javax.security=ALL-UNNAMED -add-open=java.base/java.awt:ALL-UNNAMED -add-open=java.base/java.util.concurrent=ALL-UNNAMED -add-open=java.base/java.util.concurrent=java.management:type=jaxm,management=ALL-UNNAMED -add-open=java.naming(javax.naming=ALL-UNNAMED -Djava.security.manager=allow -Dorg.jboss.boot.log.file=/home/felipy/otp-lab4/wildfly-preview-26.1.3.Final/standalone/configuration/logging.properties	
Class path:	/home/felipy/otp-lab4/wildfly-preview-26.1.3.Final/jboss-modules.jar	
Library path:	/usr/java/packages/lib/usr/lib/x86_64-linux-gnu/jni/lib/x86_64-linux-gnu/usr/lib/jni/lib/usr/lib	
Boot class path:	Unavailable	

Notification buffer

TimeStamp	Type	UserData	SeqNum	Message	Event	Source
21:02:05:078	Point.OutOfBounds		11	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:02:03:546	Point.OutOfBounds		10	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:02:02:188	Point.OutOfBounds		9	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:02:00:642	Point.OutOfBounds		8	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:59:602	Point.OutOfBounds		7	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:58:428	Point.OutOfBounds		6	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:57:319	Point.OutOfBounds		5	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:56:337	Point.OutOfBounds		4	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:55:419	Point.OutOfBounds		3	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:53:960	Point.OutOfBounds		2	Point is out of bounds	javax.management...	web_lab3:type=PointStats
21:01:51:320	Point.OutOfBounds		1	Point is out of bounds	javax.management...	web_lab3:type=PointStats

Графики



jboss-modules.jar (pid 84570)

Saved data Details

Overview

PID: 84570
Host: 0.0.0.0
Main class: jboss-modules.jar
Arguments: -mp /home/studs/408946/wildfly/wildfly-33.0.1.Final/modules org.jboss.as.standalone -Dboss.home.dir=/home/studs/s409517/wildfly/wildfly-33.0.1.Final -Dboss.server.base

JVM: OpenJDK 64-Bit Server VM (17.0.12+7-1, mixed mode, sharing)
Java: version 17.0.12 2024-07-16, vendor OpenJDK BSD Porting Team
Java Home: /usr/local/openjdk17
JVM Flags: <none>

Heap dump on OOME: disabled

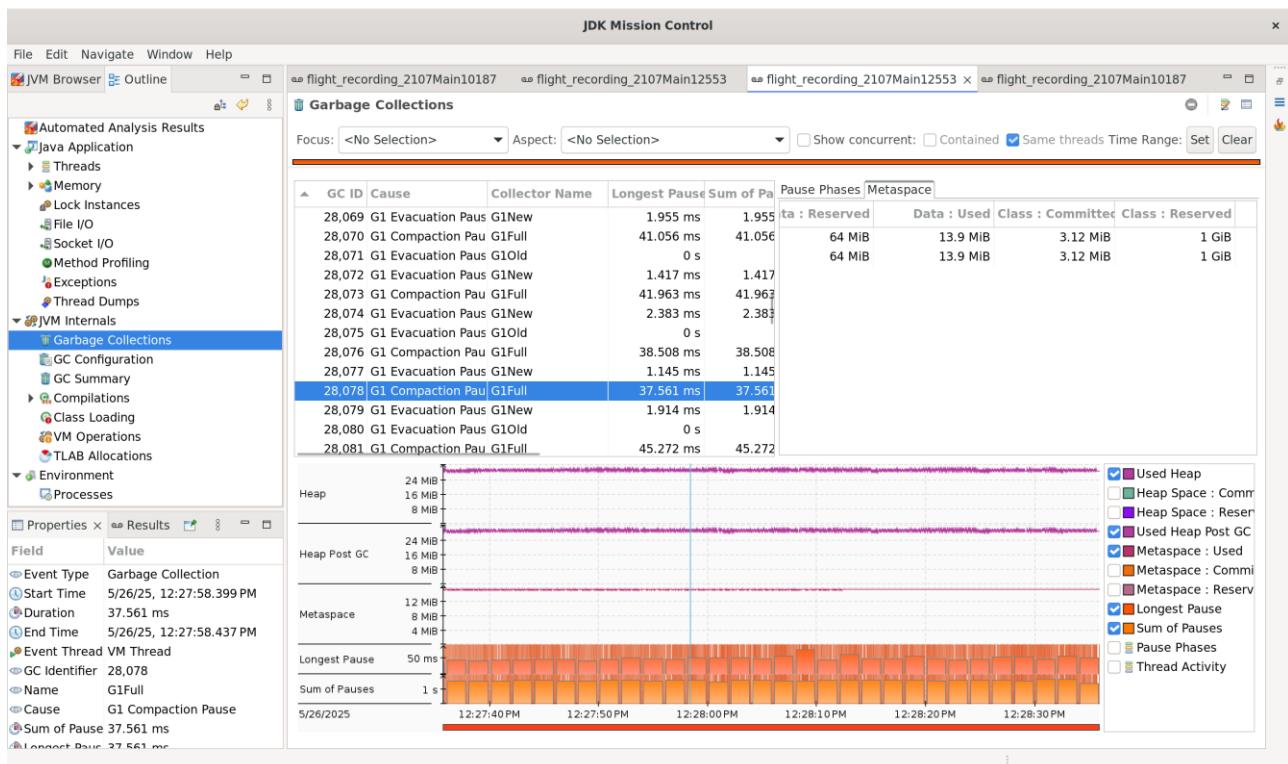
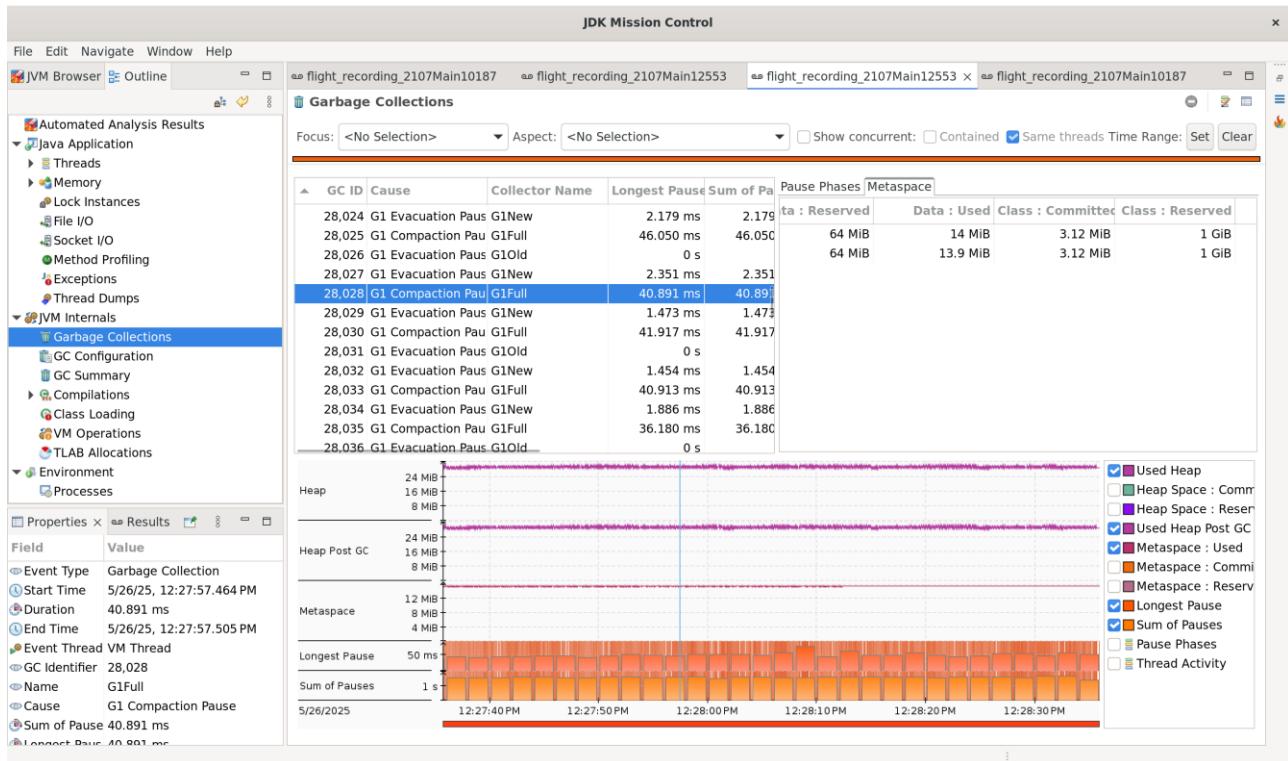
Saved data JVM arguments System properties

Thread Dumps: 0
Heap Dumps: 0
Profiler Snapshots: 0
JFR Snapshots: 0

```
-D[Standalone]
-Djdk.serialFilter=maxbytes=10485760;maxdepth=128;maxarray=100000;maxrefs=300000
-Xms64m
-Xmx512m
-XX:MetaspaceSize=96M
-XX:MaxMetaspaceSize=256m
-Djava.net.preferIPv4Stack=true
-Djboss.modules.system.pkgs=org.jboss.byteman
-Djava.awt.headless=true
--add-exports=java.desktop/sun.awt=ALL-UNNAMED
--add-exports=java.naming/com.sun.jndi.ldap=ALL-UNNAMED
--add-exports=java.naming/com.sun.jndi.url=ALL-UNNAMED
--add-exports=java.naming/com.sun.jndi.url.daps=ALL-UNNAMED
--add-exports=java.naming.dns/com.sun.jndi.dns=ALL-UNNAMED
--add-opens=java.base/java.lang=ALL-UNNAMED
--add-opens=java.base/java.lang.invoke=ALL-UNNAMED
--add-opens=java.base/java.lang.reflect=ALL-UNNAMED
--add-opens=java.base/java.io=ALL-UNNAMED
--add-opens=java.base/java.net=ALL-UNNAMED
--add-opens=java.base/java.security=ALL-UNNAMED
--add-opens=java.base/java.util=ALL-UNNAMED
--add-opens=java.base/java.util.concurrent=ALL-UNNAMED
--add-opens=java.management/javax.management=ALL-UNNAMED
--add-opens=java.naming/javax.naming=ALL-UNNAMED
-Djava.security.manager=allow
-Dorg.jboss.boot.log.file=/home/studs/408946/wildfly/wildfly-33.0.1.Final/standalone/log/server.log
-Dlogging.configuration=file:/home/studs/408946/wildfly/wildfly-33.0.1.Final/standalone/configuration/logging.properties
-XX:MaxHeapSize=1G
-XX:MaxMetaspaceSize=128m
```



Исследование на утечки



GC работает очень часто, при этом практически не очищая место. То есть он срабатывает, но никак не очищает место в памяти, а иногда даже наоборот оно увеличивается, либо лишь незначительно уменьшается. Это свидетельствует об утечке памяти.

JDK Mission Control

File Edit Navigate Window Help

JVM Browser Outline

Automated Analysis Results

- Java Application
 - Threads
 - Memory
 - Lock Instances
 - File I/O
 - Socket I/O
 - Method Profiling
 - Exceptions
 - Thread Dumps
- JVM Internals
 - Garbage Collections
 - GC Configuration
 - GC Summary
 - Compilations
 - Class Loading
 - VM Operations
 - TLAB Allocations
- Environment Processes

Properties x Results

Field Value

Event Type Garbage Collection
Start Time 5/26/25, 12:27:58.399 PM
Duration 37.561 ms
End Time 5/26/25, 12:27:58.437 PM
Event Thread VM Thread
GC Identifier 28,078
Name G1Full
Cause G1 Compaction Pause
Sum of Pause 37.561 ms
Longest Pause 37.561 ms

Garbage Collections

Focus: <No Selection> Aspect: <No Selection>

Collector Name	Longest Pause	Sum of Pauses	Final Refers	Pause Phases	Metaspace
vacuation Paus G1Old	0 s	0 s			
vacuation Paus G1New	2.506 ms	2.506 ms			
mpaction Pau G1Full	40.208 ms	40.208 ms			
vacuation Paus G1New	1.955 ms	1.955 ms			
mpaction Pau G1Full	41.056 ms	41.056 ms			
vacuation Paus G1Old	0 s	0 s			
vacuation Paus G1New	1.417 ms	1.417 ms			
mpaction Pau G1Full	41.963 ms	41.963 ms			
vacuation Paus G1New	2.383 ms	2.383 ms			
vacuation Paus G1Old	0 s	0 s			
mpaction Pau G1Full	38.508 ms	38.508 ms			
vacuation Paus G1New	1.145 ms	1.145 ms			
mpaction Pau G1Full	37.561 ms	37.561 ms			

GC Threshold When GC ID Start Time

33.2 MB Before GC 28,078 5/26/25, 12:27:58.3
33.2 MB After GC 28,078 5/26/25, 12:27:58.4

Heap

Used Heap
Heap Space : Comm
Used Heap Post GC
Used Heap : Used
Metaspace : Comm
Metaspace : Reserv
Longest Pause
Sum of Pauses
Pause Phases
Thread Activity

Далее находим что стало ее причиной

JDK Mission Control

File Edit Navigate Window Help

JVM Browser Outline

Automated Analysis Results

- Java Application
 - Threads
 - Memory
 - Live Objects
- JVM Internals
 - Garbage Collections
 - GC Configuration
 - GC Summary
 - Compilations
 - Class Loading
 - VM Operations
 - TLAB Allocations
- Environment

Properties x Results

Field Value

Event Type Old Object Sample
Start Time 5/26/25, 12:23:44.254 PM
Duration 47.107 ms
End Time 5/26/25, 12:23:44.301 PM
Event Thread main
Allocation Tin 5/26/2025, 12:21:47.000 PM -
Object Size 120 B - 278 KIB
Object Age 141.758 ms - 1 min 57 s
Last Known l 20.3 MIB - 26.2 MIB
Object Too many values

Live Objects

Allocation 5/26/2025 12:21:45PM 12:22:00PM 12:22:15PM 12:22:30PM 12:22:45PM 12:23:00PM 12:23:15PM 12:23:30PM

Live Object Sample

Col Description

- ArrayList
 - Object[]
 - Class
 - ArrayList
 - Object[]
 - String
 - String
 - String
 - String
 - String
 - String
 - String

Stack Trace Flame Graph

Stack Trace	Samples	Percentage
byte[] java.util.Arrays.copyOfRangeByt(byte[], int, int)	160	99.4 %
byte[] java.util.Arrays.copyOfRangeByt(int, int)	160	99.4 %
void java.lang.String.<init>(AbstractStringBuilder, Void)	160	99.4 %
void java.lang.String.<init>(StringBuilder)	160	99.4 %
String java.lang.StringBuilder.toString()	160	99.4 %
void com.meterware.httpunit.javascript.JavaScript\$JavaScriptEngine.handleScriptExc	160	99.4 %
String com.meterware.httpunit.javascript.JavaScript\$JavaScriptEngine.executeScript(160	99.4 %
String com.meterware.httpunit.javascript.JavaScript\$Window.executeScript(String, St	160	99.4 %
String com.meterware.httpunit.scripting.ScriptableDelegate.runScript(String, String)	160	99.4 %
String com.meterware.httpunit.parsing.ScriptFilter.getTranslatedScript(String, String)	160	99.4 %
void com.meterware.httpunit.parsing.ScriptFilter.endElement(QName, Augmentations	160	99.4 %
void org.cyberneko.html.filters.DefaultFilter.endElement(QName, Augmentations)	160	99.4 %

Recording My Recording: (51%)

Можно также исследовать heap:

Class Name	Shallow Heap	Retained Heap	Percent
jdk.internal.loader.ClassLoaders\$AppClassLoader @ 0xfeed0e8	96 B	11.02 MB	43.42%
classes java.util.ArrayList @ 0xfeeef248	24 B	10.95 MB	43.17%
elementData java.lang.Object[474] @ 0xfe2c8f38	1.87 KB	10.95 MB	43.17%
class com.meterware.httpunit.javascript.JavaScript @ 0xfe200e10	16 B	10.91 MB	42.98%
_errorMessages java.util.ArrayList @ 0xfe20aed8	24 B	10.91 MB	42.98%
java.lang.Object[1] @ 0xfe7e5a90	24 B	24 B	0.00%
NO_ARGS java.lang.Object[] @ 0xfe201fb0	16 B	16 B	0.00%
Total: 3 of 3 entries;			
class.org.cyberneko.html.HTMLElements @ 0xfe297ff0	256 B	12.63 KB	0.05%
class.org.mozilla.javascript.Parser @ 0xfe2968c0	32 B	5.39 KB	0.02%
class.org.apache.xerces.util.URI @ 0xfe2bf9a0	80 B	3.29 KB	0.01%
class.org.cyberneko.html.HTMLScanner @ 0xfe2c1590	160 B	2.60 KB	0.01%
class.org.mozilla.javascript.ScriptRuntime @ 0xfe2cbab0	184 B	1.84 KB	0.01%
class.org.mozilla.classfile.ClassFileWriter @ 0xfe207a70	10 B	1.72 KB	0.01%

Actual Data			
Attributes	Statics	Value	Overview
Type	Name	Value	
ref	parent	jdk.internal.loader.ClassLoaders\$PlatformClassLoader @ 0xfeeec9c8	
ref	ucp	jdk.internal.loader.URLClassPath @ 0xferb7658	
ref	nameToModule	java.util.concurrent.ConcurrentHashMap @ 0xeeef3c0	
ref	moduleToReader	java.util.concurrent.ConcurrentHashMap @ 0xfeeef770	
ref	resourceCache	java.lang.ref.SoftReference @ 0xferb7630	
ref	pdcache	java.util.concurrent.ConcurrentHashMap @ 0xfeeef380	
ref	parent	jdk.internal.loader.ClassLoaders\$PlatformClassLoader @ 0xfeeec9c8	
ref	name	app	
ref	unnamedModule	java.lang.Module @ 0xfeed150	

Утечка в классе com.meterware.httpunit.JavaScript

Вывод

В ходе выполнения лабораторной работы я освоил практику создания MBeans в веб-приложениях, изучил утилиты JConsole, VisualVM и JFR для мониторинга и профилирования программ, а также приобрел навыки выявления и устранения утечек памяти на основе полученных данных.