Университет ИТМО

Факультет ПИиКТ

Дисциплина: МиСПИ

Лабораторная работа №4.

Вариант 35

              Выполнил:

Абульфатов Руслан Мехтиевич,

группа Р32312

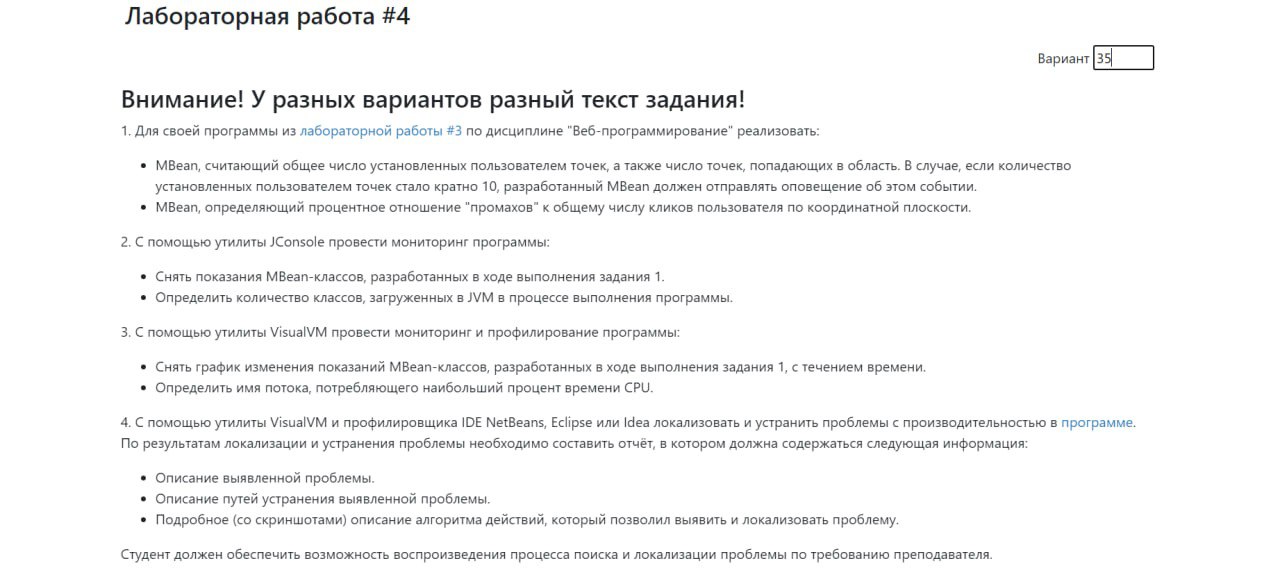
Преподаватель:

Исаев Илья Владимирович

г. Санкт-Петербург, 2023 год

Задание:

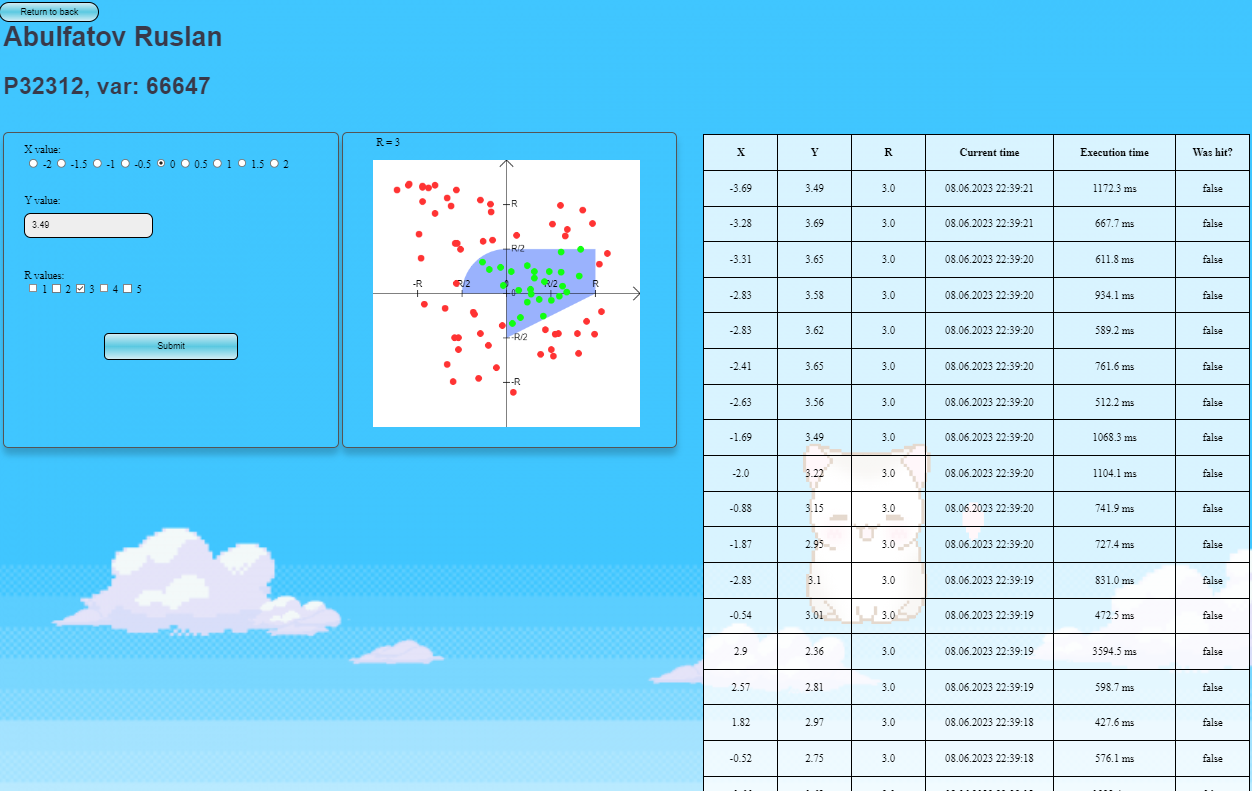
Лабораторная работа №4



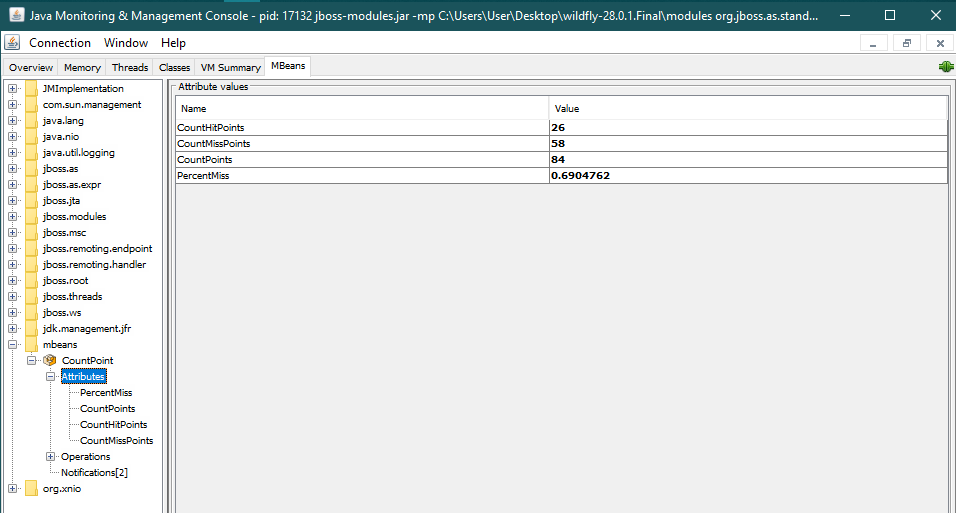
Исходный код MBeans: ----------------------

**JConsole**

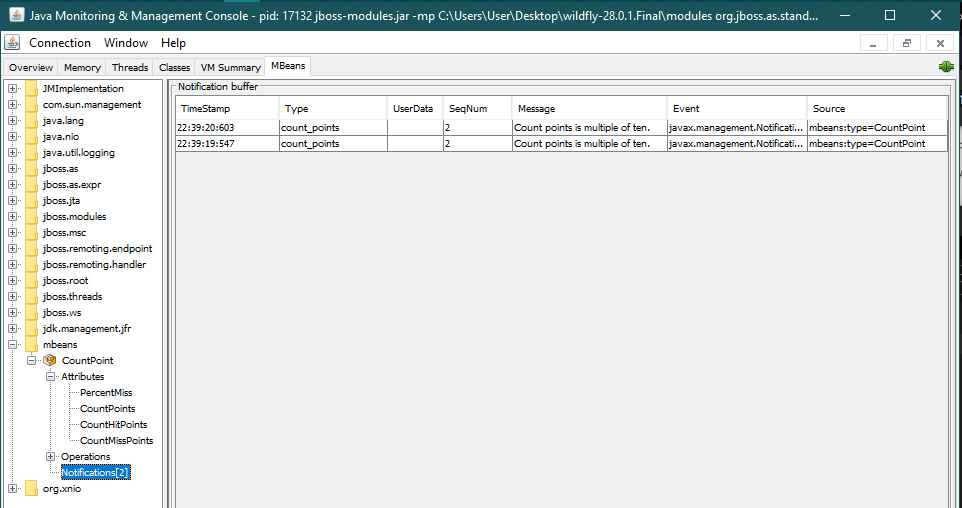
Запущенное приложение на WildFly



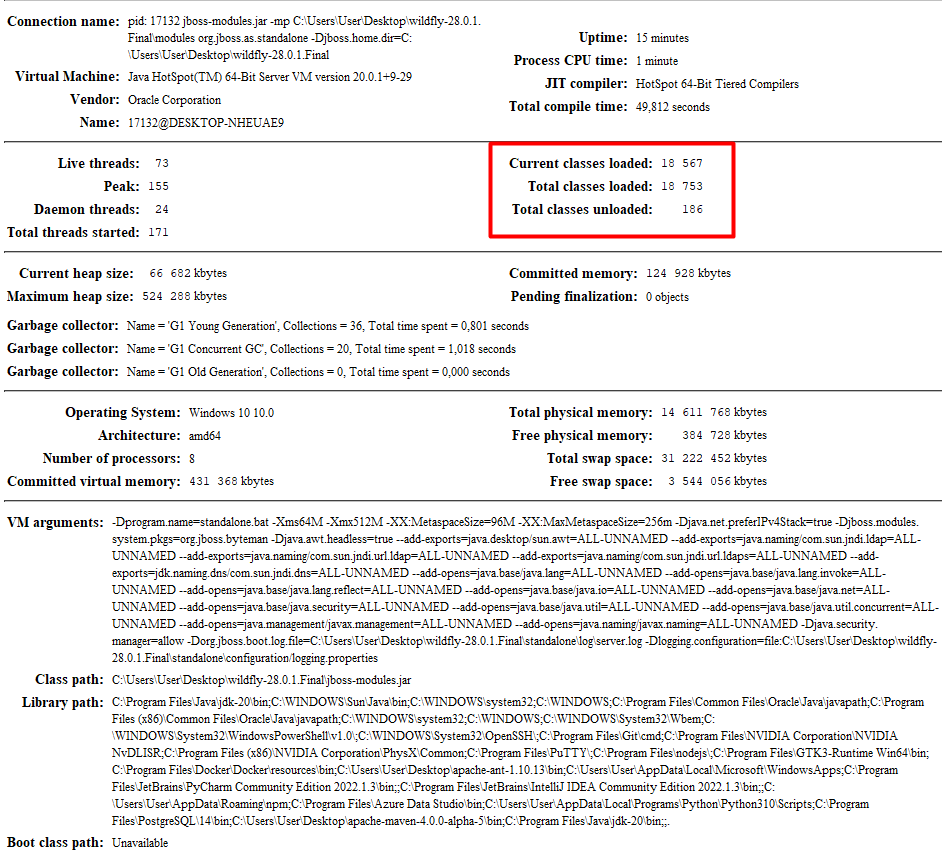
Атрибуты MBean - CountPoints



Уведомления MBean - CountPoints



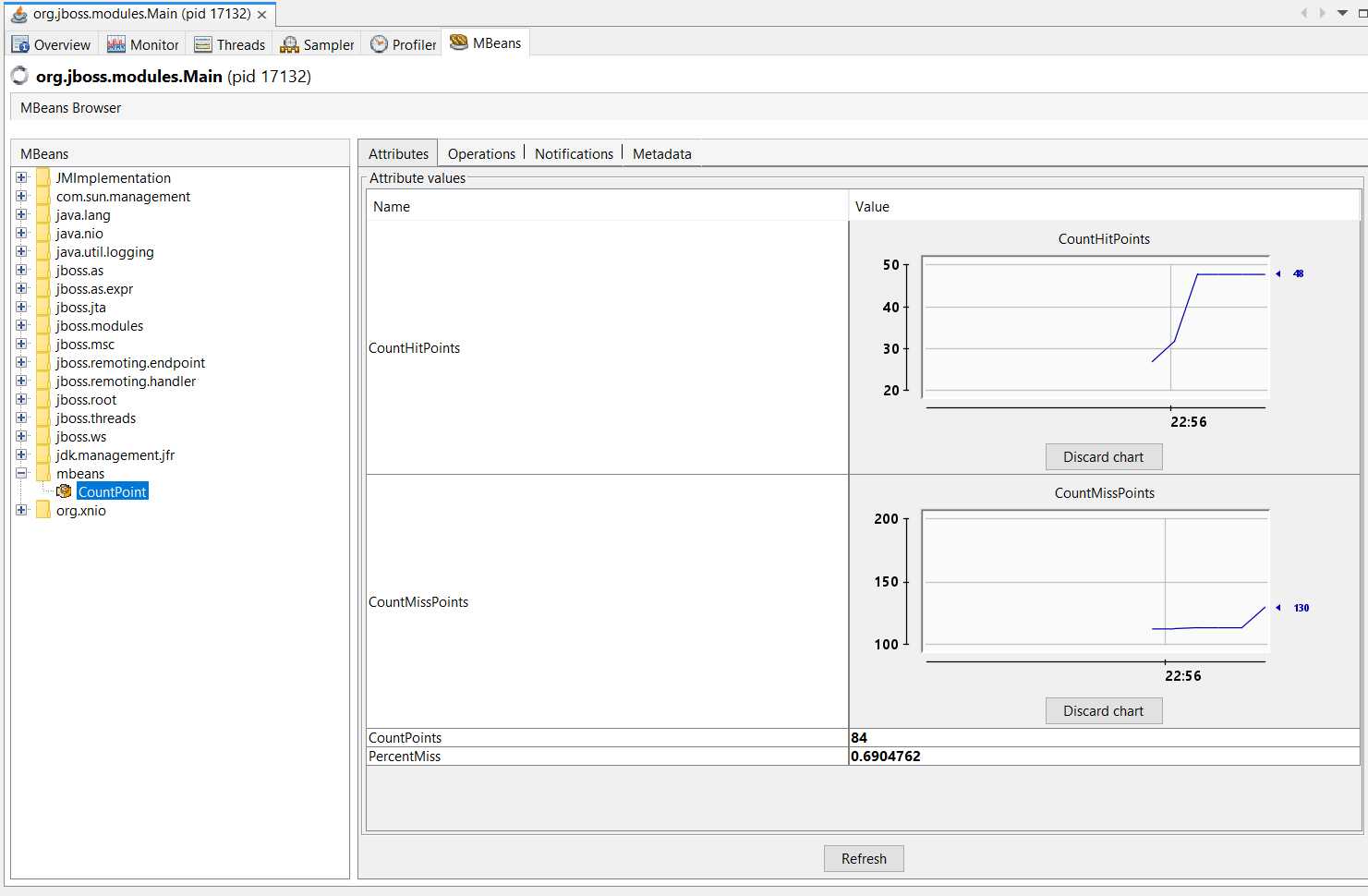
Вся информация о запущенном процессе WildFly

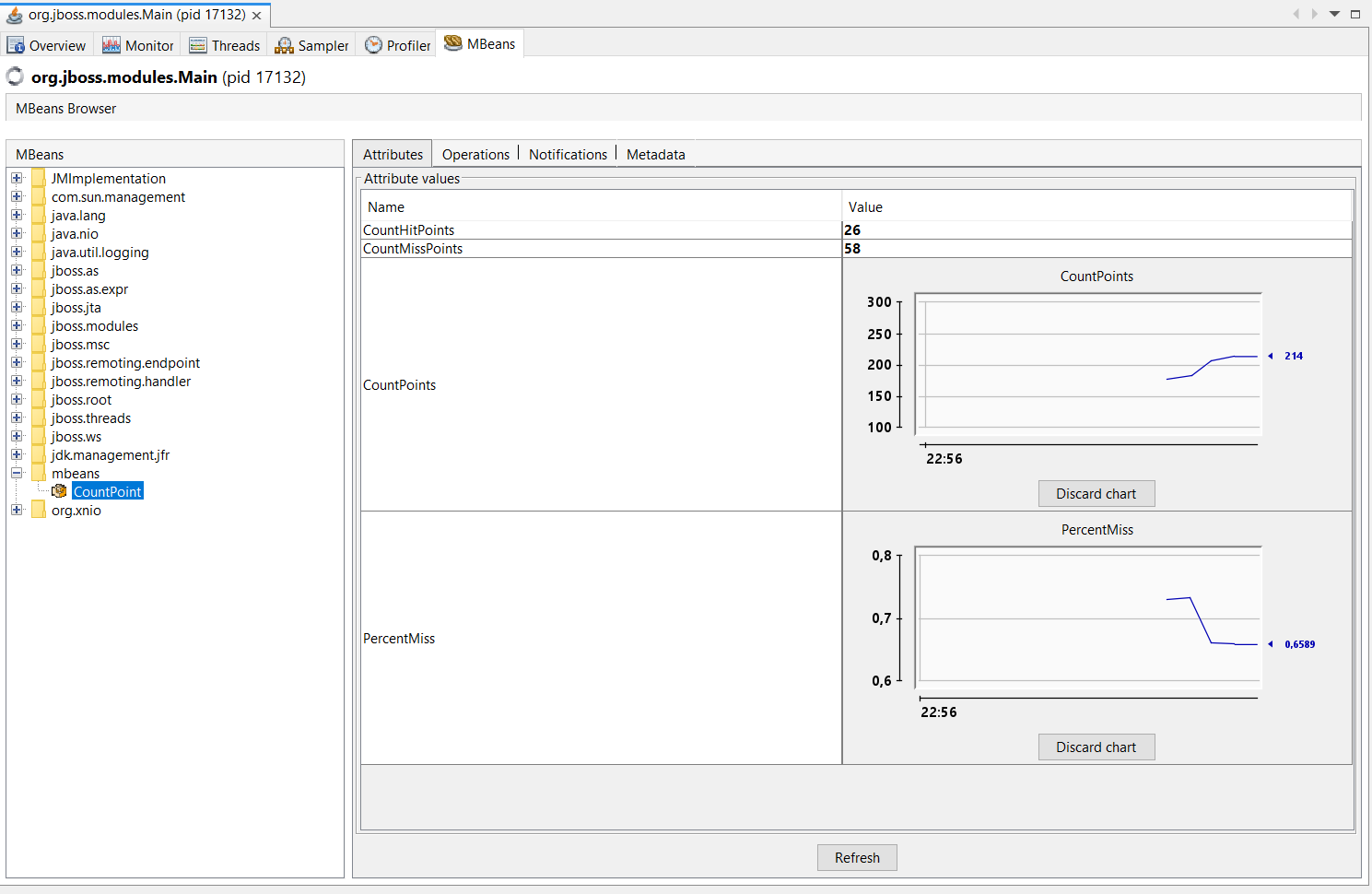


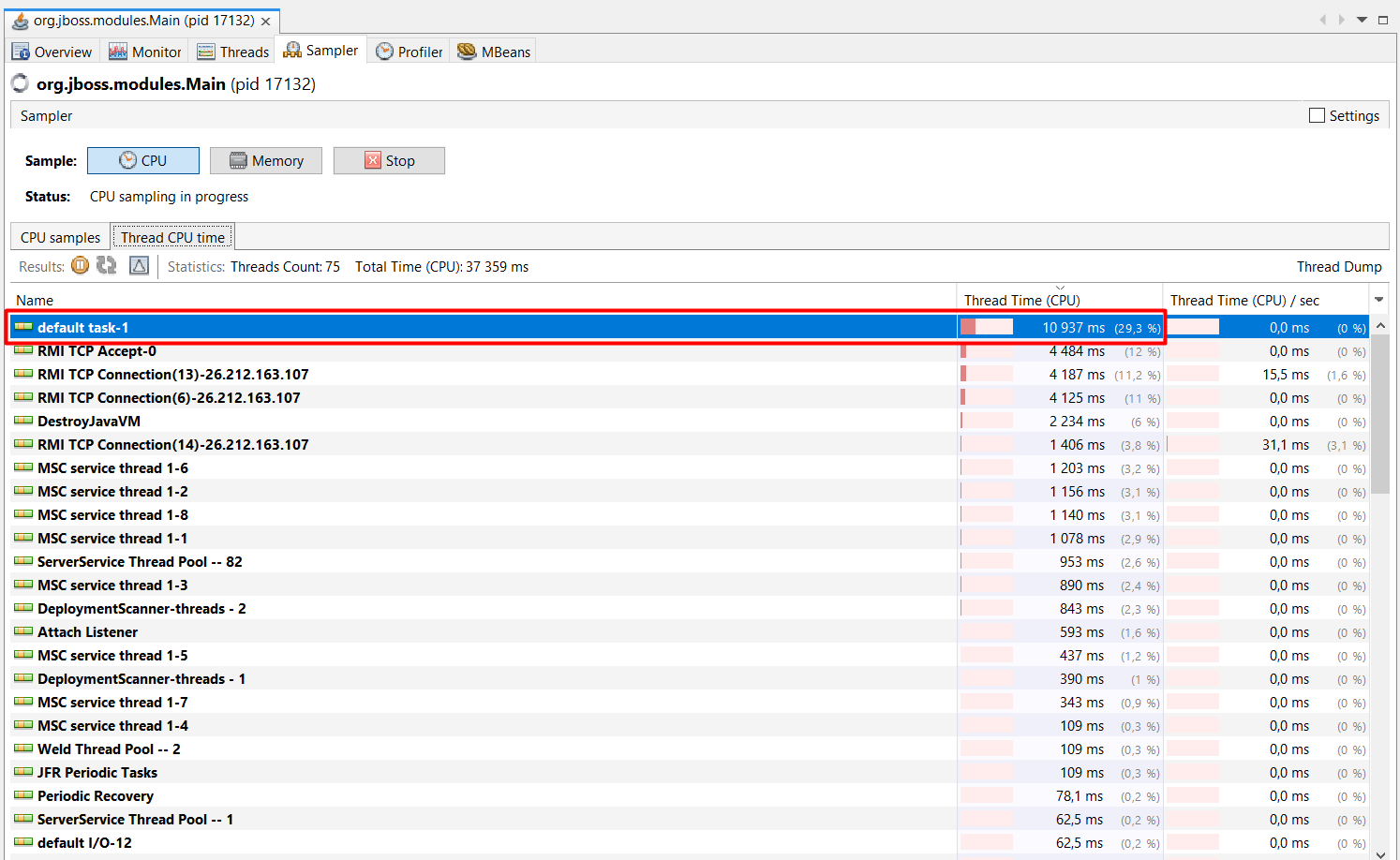
Количество загруженный классов – 18567 и 186 не загружены. Всего было загружено – 18753 класса.

**VisualVM:**

Отслеживание переменных из созданных MBean с течением времени.





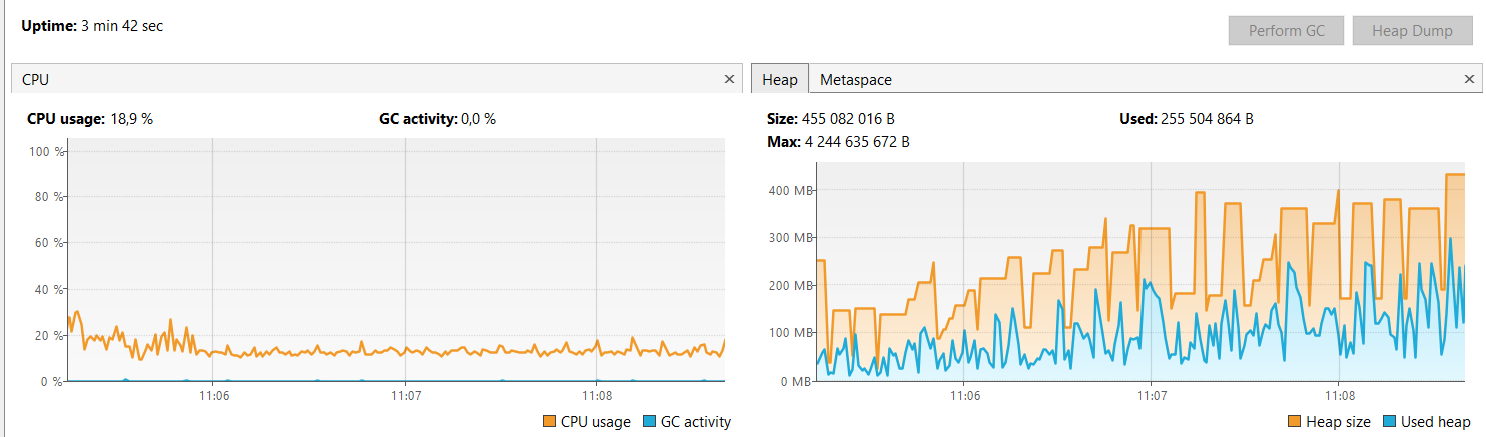
Имя потока, потребляющего наибольший процент времени CPU. 

**VisualVM + Idea:**

1. Описание выявленной проблемы

За 4 минуты работы программы, были сняты следующие показатели:

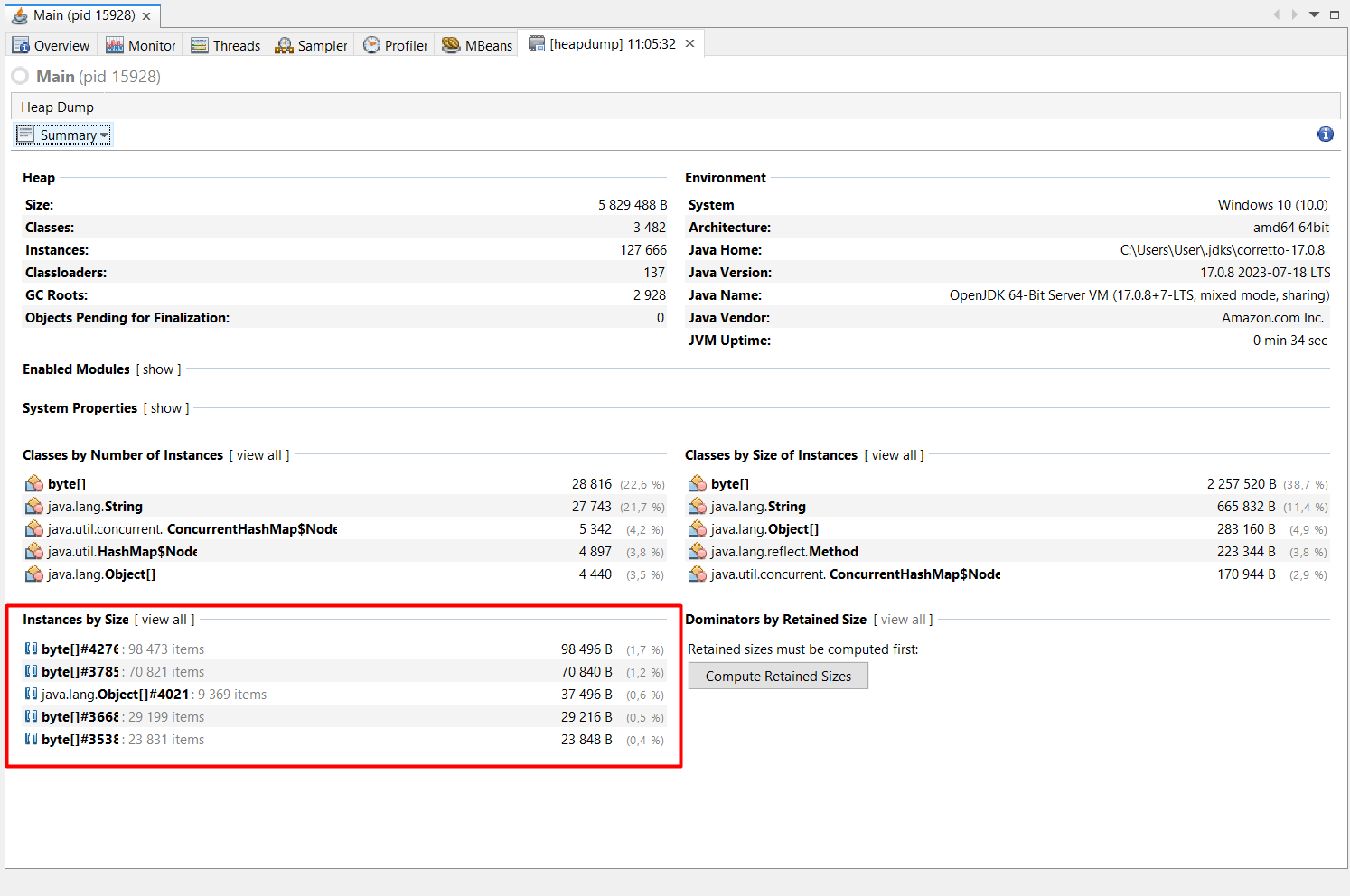
* нагрузка на процессор стабильна
* нагрузка по памяти увеличивается



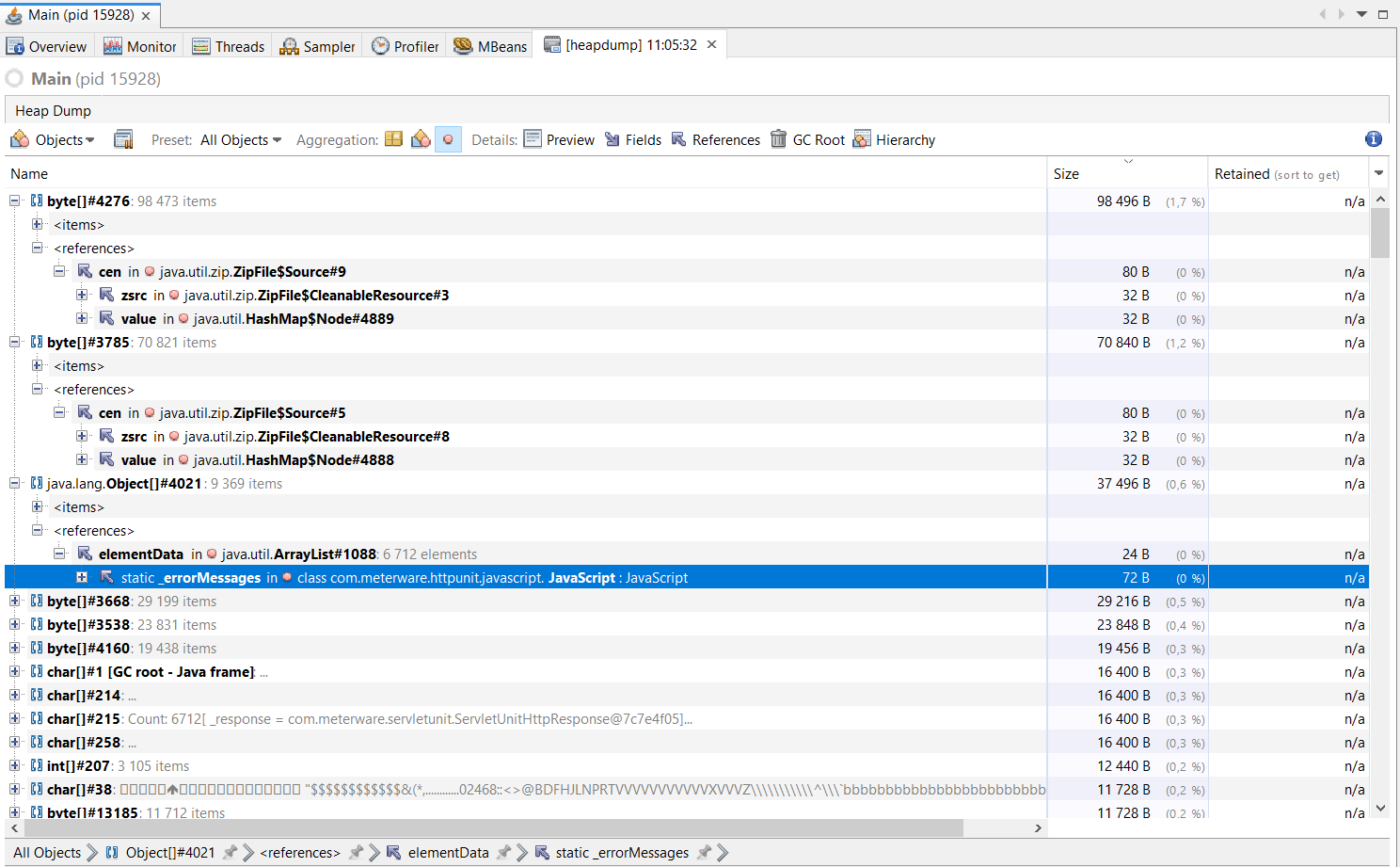
Из этого можно предположить, что идет некорректное взаимодействие с памятью при работе программы.

1. Описание путей устранения проблемы

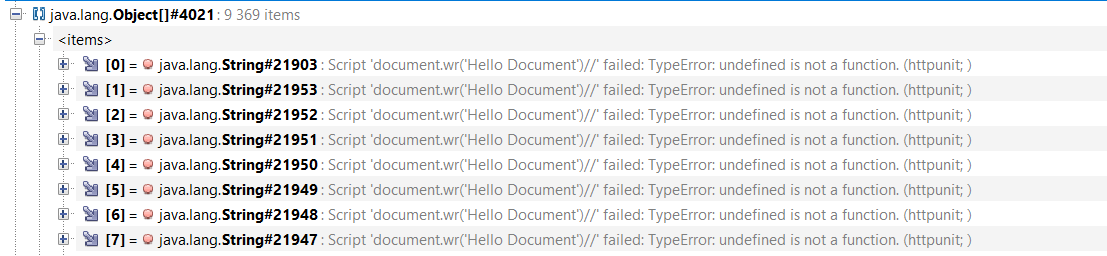
После остановки программы, посмотрим собранную статистику, чтобы понять каких объектов было больше в памяти.



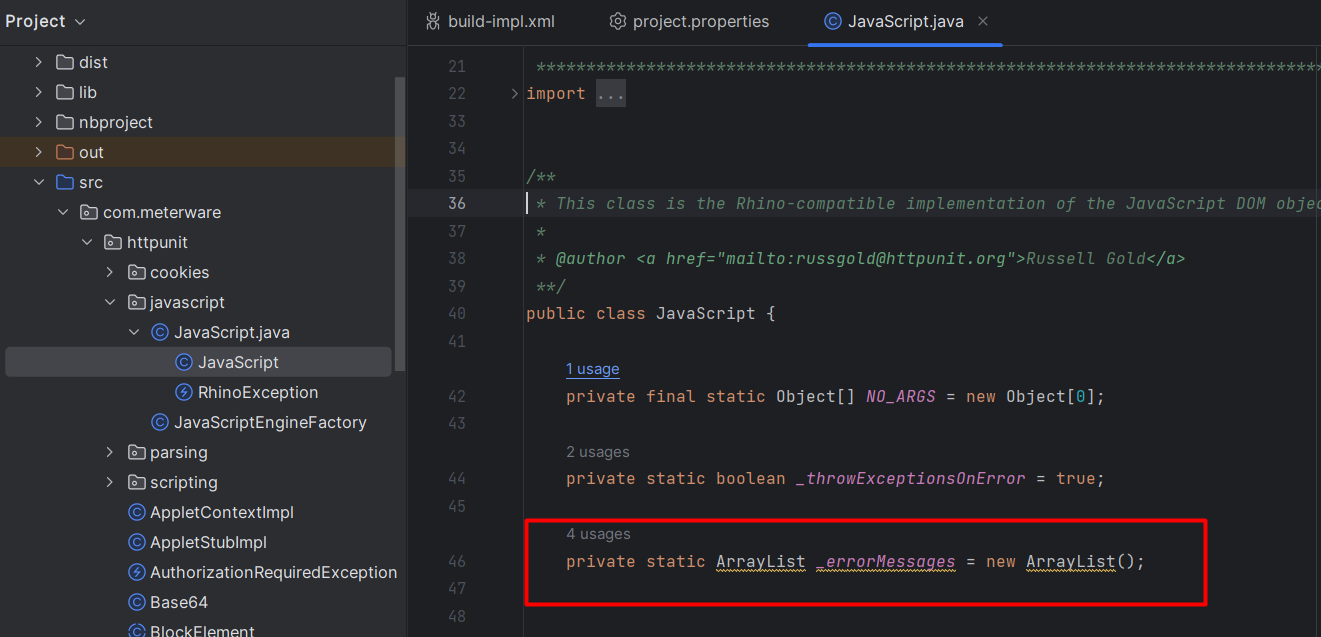
Попытаемся найти что нагружает память. Самый большой размер у переменной \_errorMessages в классе com.meterware.httpunit.javascript.JavaScript.



Если посмотреть данные из вкладки items, то можно сделать вывод, что данная переменная используется для накопления логов ошибок во время работы программы.



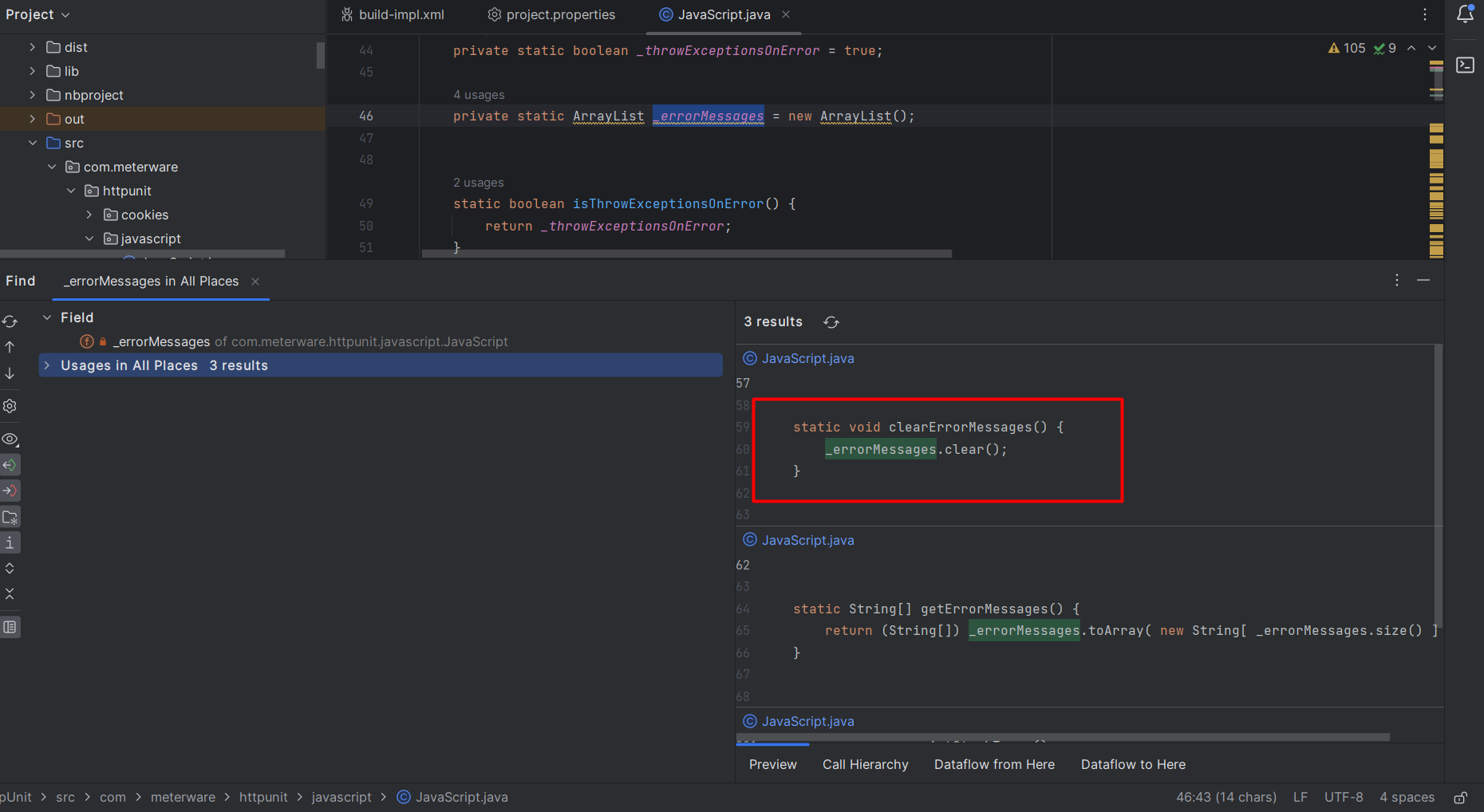
Найдем данную переменную в коде.

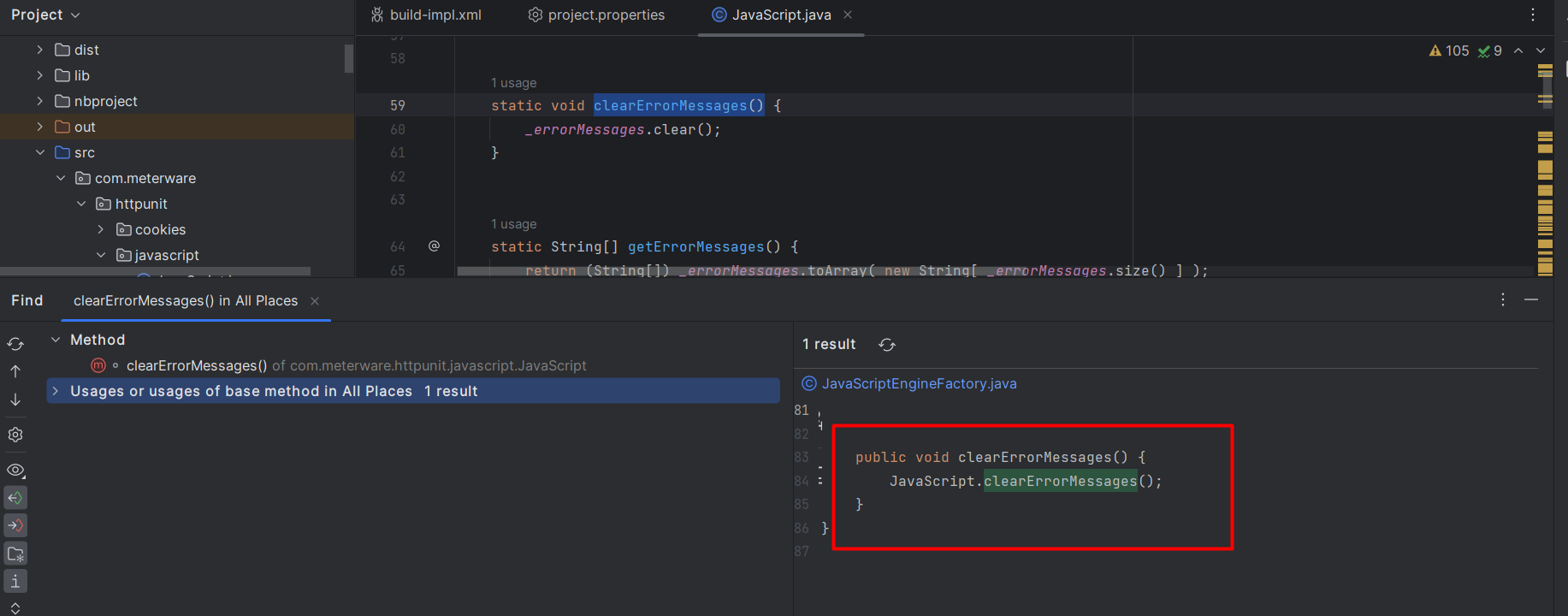


Она является статичной, следовательно существует все время, и не может быть очищена автоматически, только вручную.

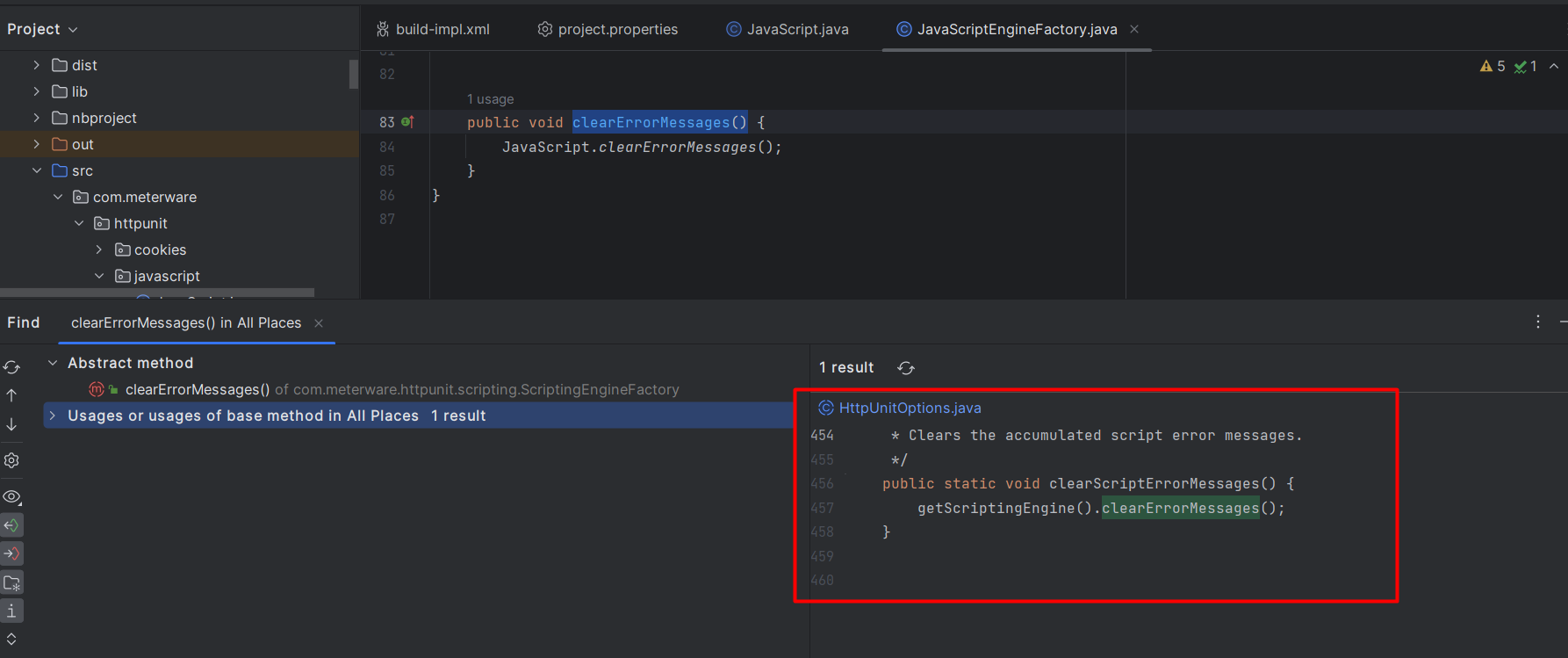
1. Описание алгоритма решения проблемы

Посмотрим где в данном классе используется переменная \_errorMessages.



Она используется в 3х местах, и даже есть метод для удаления всех элементов списка. Посмотрим где вызывается он. 

Теперь он используется только в одном месте. Исследуем дальше.



Он также используется только в одном месте.

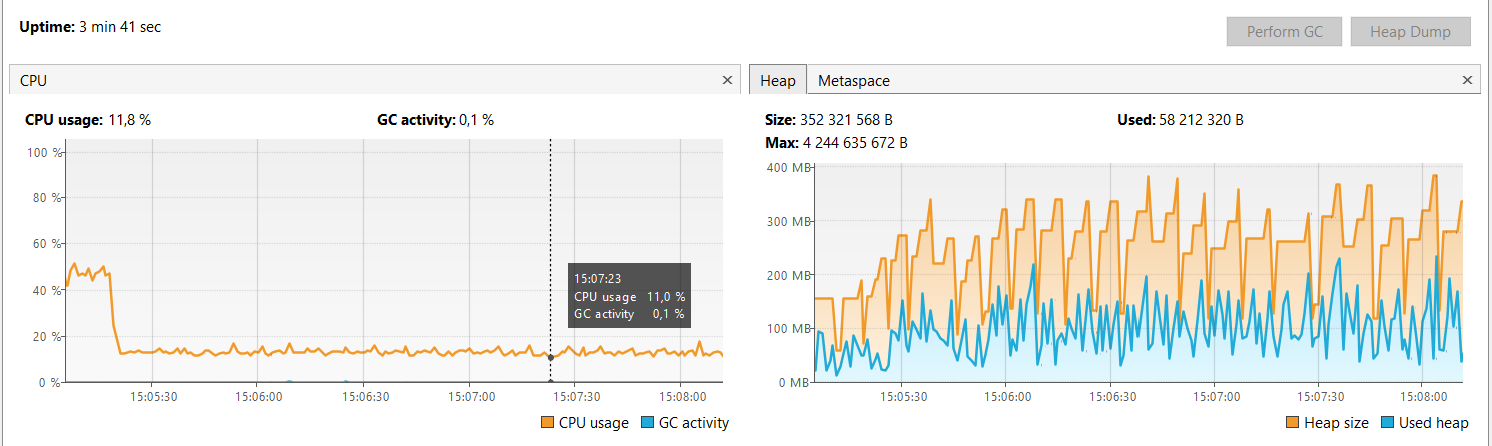


Благодаря IDE мы сразу можем увидеть, что данный метод нигде не вызывается.

Добавим принудительную очистку в главный метод.



Проверим новый данные в VisualVM.



Проведем сравнение старых и новых показателей.

CPU.

|  |  |
| --- | --- |
| Было | Стало |
|  |  |

Memory.

|  |  |
| --- | --- |
| Было | Стало |
|  |  |

Показатели пришли к более горизонтальному виду.

**Вывод:**

Были произведен мониторинг и профилирование работы программы с помощью утилит JConsole и VisualVM. Утилиты позволяют отслеживать показания MBean’ов, графики изменения показаний с течением времени, вызывать операции, а также отслеживать уведомления. VisualVM также позволяет получить более подробную информацию о нагрузке на CPU/RAM. Например, узнать какие потоки программы нагружают CPU и как долго они это делают.