Программная инженерия

Дмитриев Андрей Владиславович
andrei-dmitriev@yandex.ru
http://in4mix2006.narod.ru/
2008

Обнаружение узких мест и анализ утечек памяти

Обнаружение узких мест и анализ утечек памяти

Дмитриев Андрей Владиславович
andrei-dmitriev@yandex.ru
http://in4mix2006.narod.ru/
2008

Качество программы

Качество можно оценивать по следующим критериям:

- Результативность – решает ли программа поставленную задачу?
- Быстродействие – насколько быстро программа решает задачу?
- Надежность – насколько можно доверять результатам работы программы?
- Ресурсоемкость – как много ресурсов нужно программе для решения задачи?
- …

Программа

- Определения.
- Возможности профилировщика.
- Запуск.
- Мониторинг приложения.
- Анализ производительности.
- Анализ использования памяти.

Определения (1/2)

- *Куча (heap)* – область памяти, где размещаются объекты, создаваемые виртуальной машиной.
- *Сборка мусора* – удаление объектов, которые больше не используются.
- *Утечка памяти* – объект, который более не используется, но, тем не менее, не может быть удален сборщиком мусора.

Определения (2/2)

- *Собственное время исполнения* – время, необходимое методу для выполнения своих инструкций. Время на выполнение вызываемых из него методов не учитывается.
- *Hot spot* – метод, имеющий сравнительно большое значение собственного времени исполнения.
- *Профилировщик* – утилита, показывающая параметры работы приложения.
- *Профилирование* – обнаружение проблем производительности измерительным методом.

Цель профилирования

- По статистике, относительно небольшое число методов занимают больше всего процессорного времени.
  - Поиск таких методов и устранение задержек позволяет улучшить скорость работы приложения.
- Неудаленные объекты, находясь все время в памяти, замедляют скорость работы приложения.
  - Обнаружение и устранение утечек памяти является важной составляющей повышения качества приложения.

Состав профилировщика

- Агент
  - Загружается при старте виртуальной машины Java (Java Virtual Machine, JVM).
  - Взаимодействует с JVM с использованием протокола JVM TI (Tool Interface).
- Клиент
  - Предоставляет пользовательский интерфейс.
  - Взаимодействие с агентом:
    - переключение режимов профилирования,
    - запись результатов измерений в файл.
  - Визуализирует результаты профилирования.

Вопрос

Зачем заниматься исследованием потребления **памяти**, если существует автоматическая сборка мусора?

Ответ

Эффективность потребления **памяти** Java программой может существенно влиять на производительность и надежность этой программы.

Почему важно избавляться от утечек?

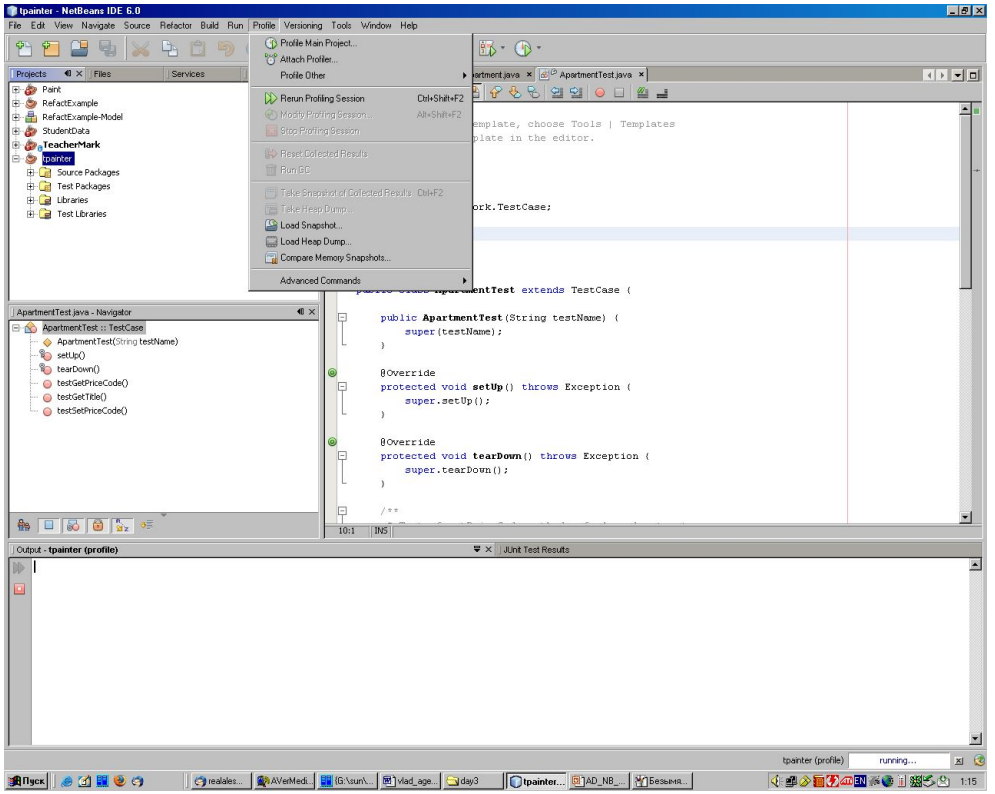
- В случае невозможности выделить память под новый объект возникает OutOfMemoryError.
  - Аварийное завершение программы или ее дальнейшая нестабильная работа.
  - Возможна потеря результатов работы и/или повреждение файлов данных.
- Использование больших объемов памяти может привести к увеличению частоты обращения ОС к файлу подкачки.
  - Общее снижение производительности системы.
- При создании большого количества временных объектов VM тратит больше времени на сборку мусора.

Запуск

- С версии NetBeans 6.0 профилировщик входит в состав среды разработки.
- Для начала профилирования проекта нужно выбрать соответствующий пункт меню.
- Провести калибрацию (определение специфичных для данной среды работы условий) окружения.

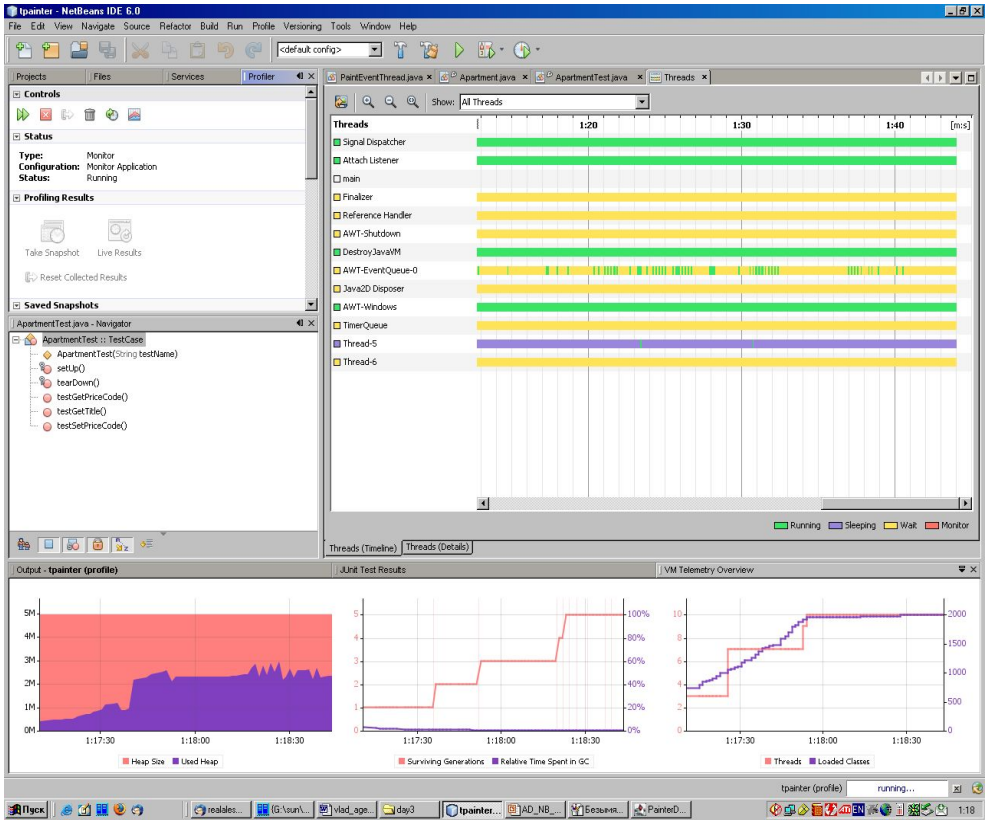
Возможности профилировщика

- Взаимодействие с SE, EE и ME приложениями.
- Оценка скорости выполнения отдельных частей программы.
- Отслеживание создаваемых Java объектов.
- Поиск утечек памяти.
- Создание отчетов.



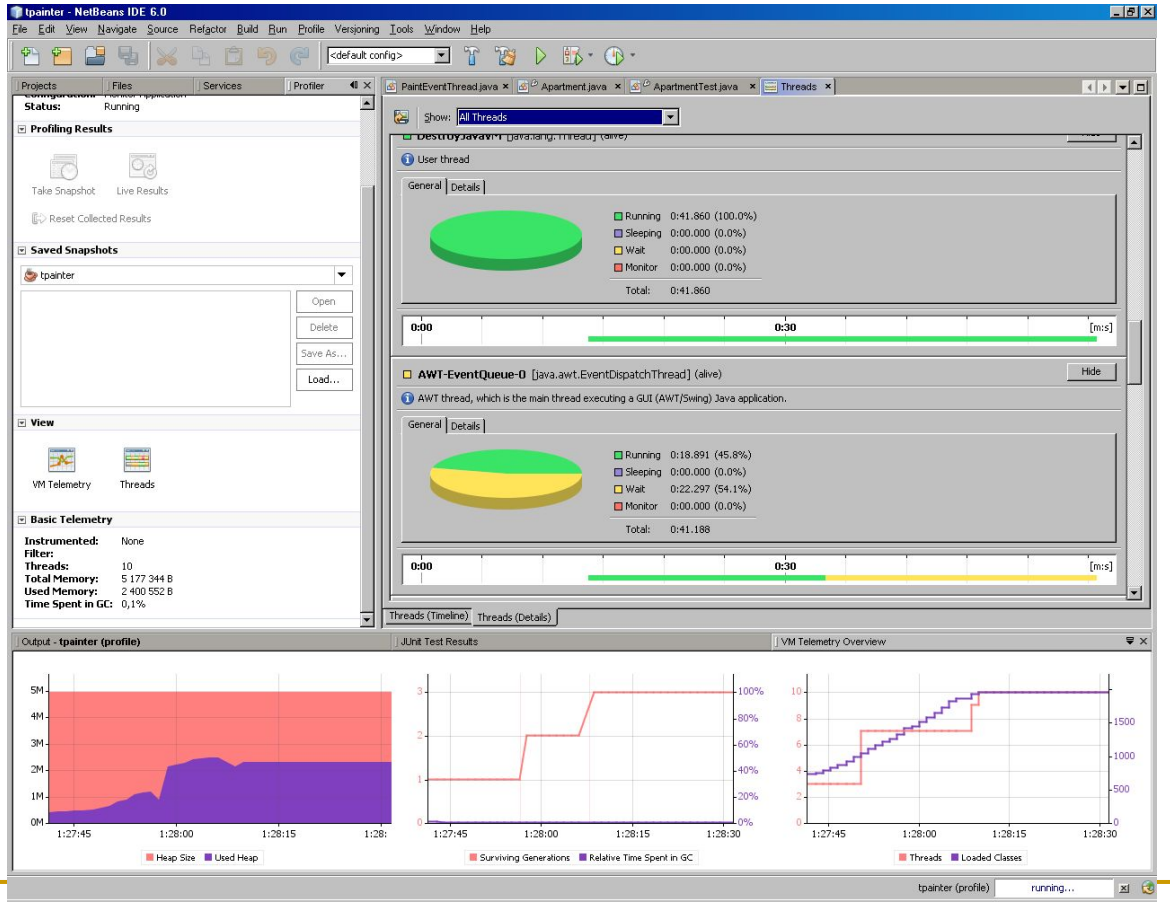
Мониторинг приложения

- Полезен для получения общей статистики о работе приложения.
- Отображает в реальном времени сведения :
  - Об использованной памяти,
  - О работе сборщика мусора,
  - О потоках Java.



Мониторинг приложения: статистика

Приводит наглядную статистику параметров работы.

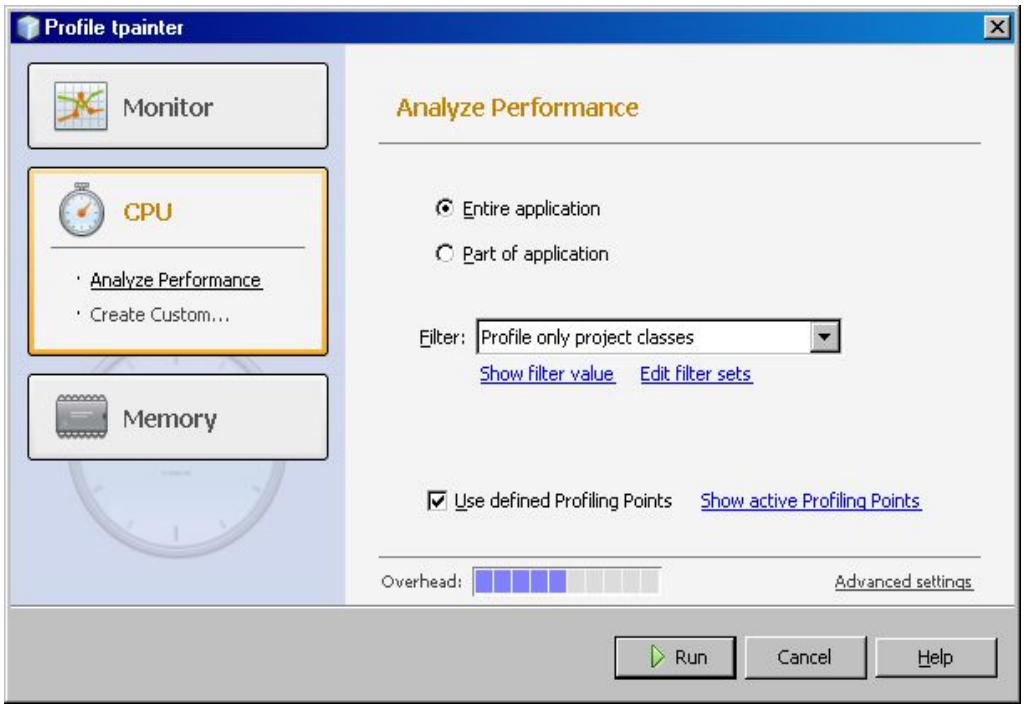


Мониторинг приложения: другие возможности

- Сохранение результатов измерений в файл для последующего просмотра.
- Вызов сборщика мусора.

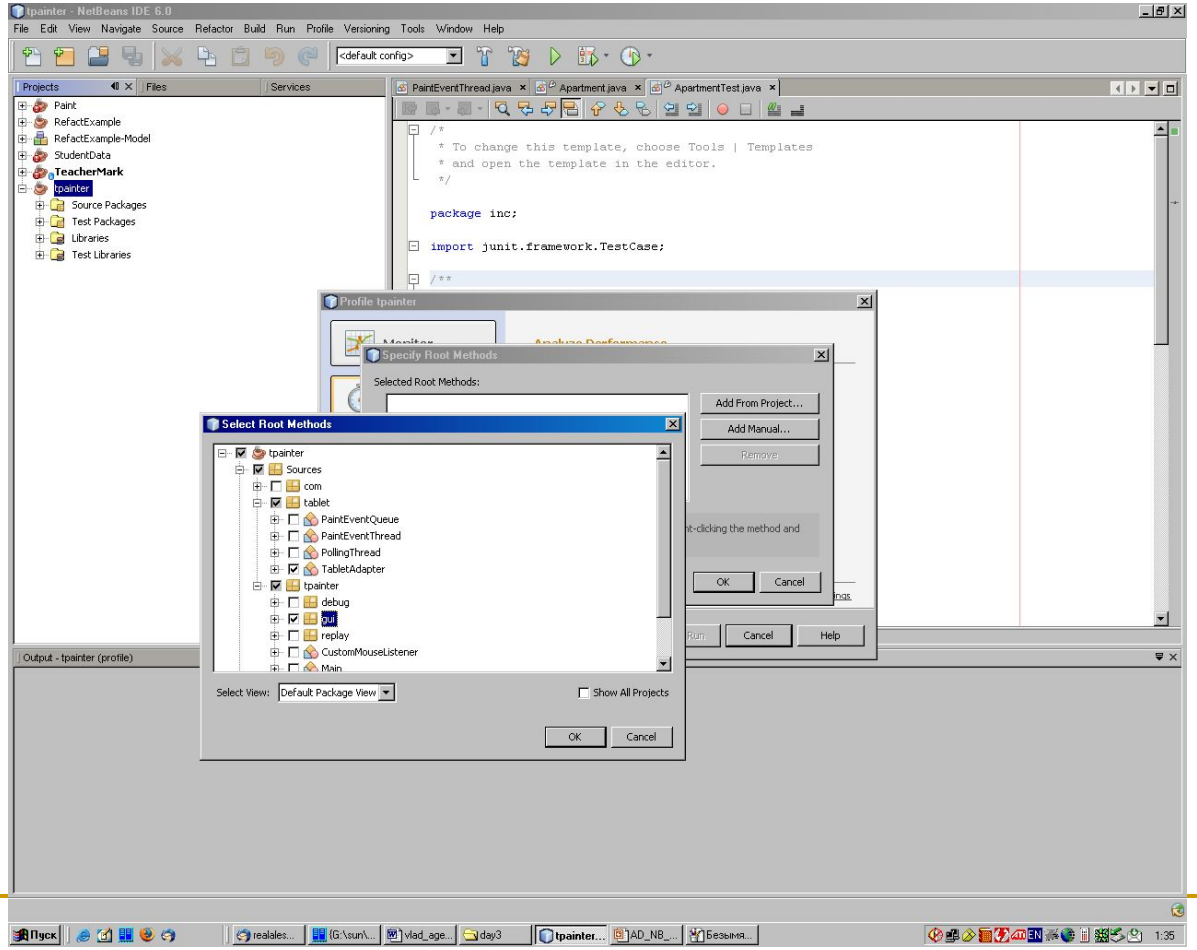
Анализ производительности

- Необходим для получения детализированной информации о методах приложения:
  - Время исполнения,
  - Количество вызовов.
- Можно анализировать:
  - Все приложение,
  - Приложение и классы JDK,
  - Только некоторые методы.



Производительность блоков кода

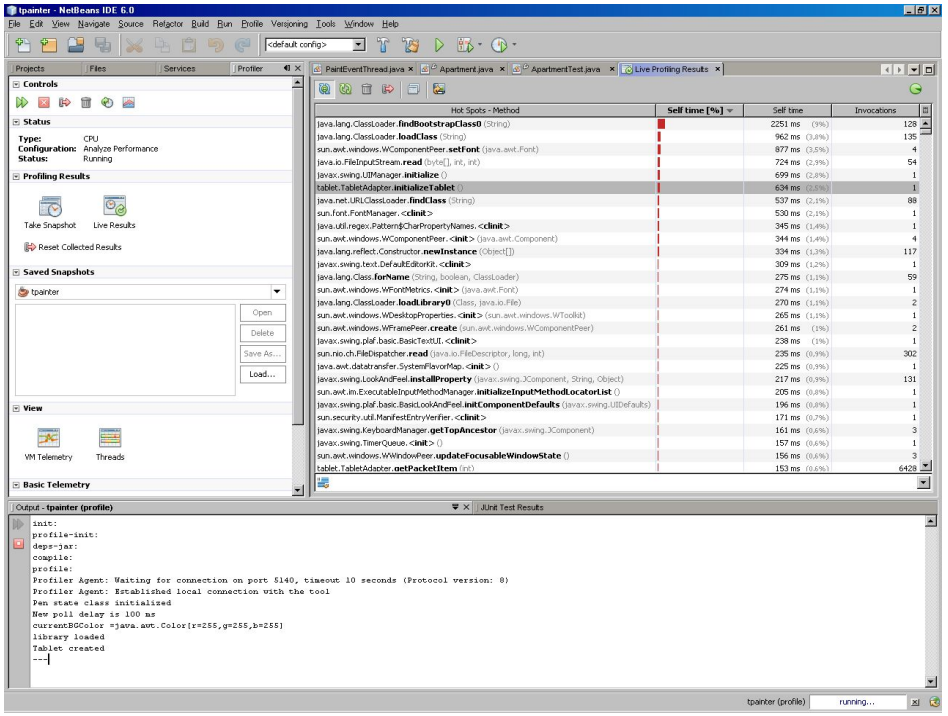
Для проведения анализа блока кода нужно выбрать интересующие классы и/или методы.



Производительность: результаты в реальном времени

Каждый метод снабжен:

- временем исполнения,
- количеством вызовов,
- процентной шкалой от всего времени работы приложения.

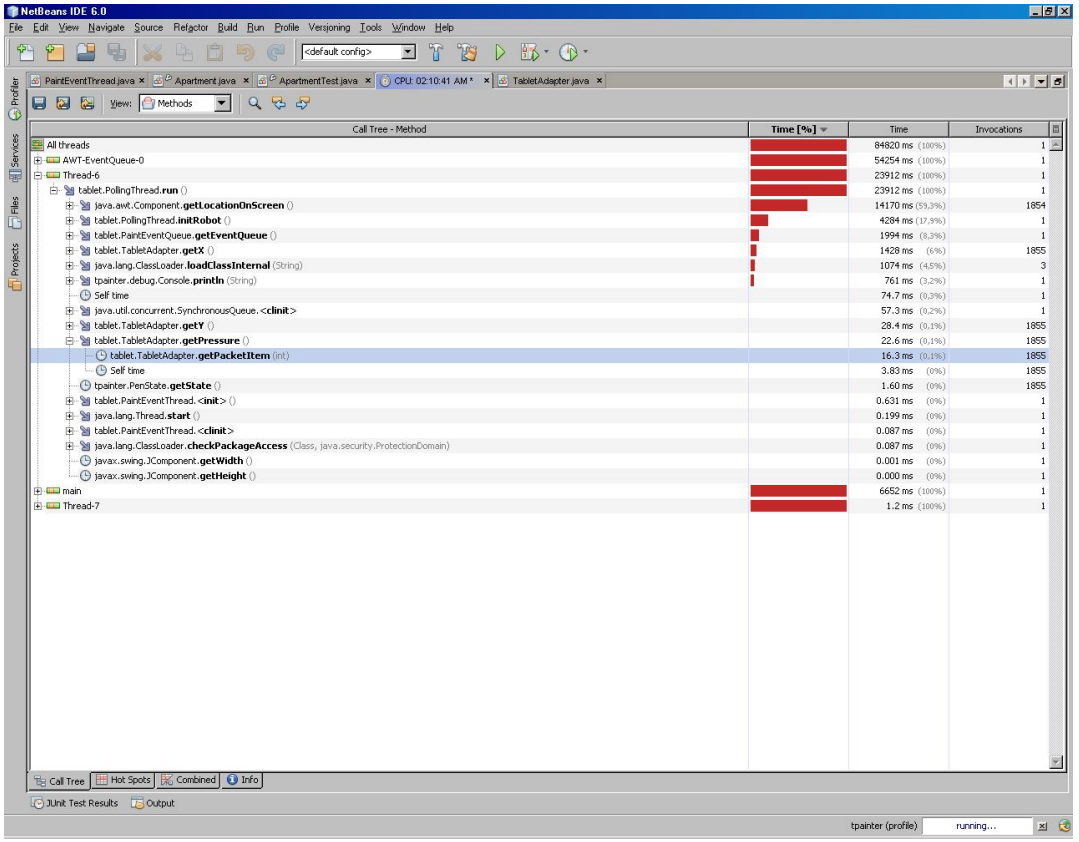


Производительность: сохраненные результаты

- Для более детального анализа можно сделать слепок состояния приложения.
- По сохраненным данным анализа можно построить:
  - Дерево вызовов.
  - Отсортированный список всех методов.

Дерево вызовов

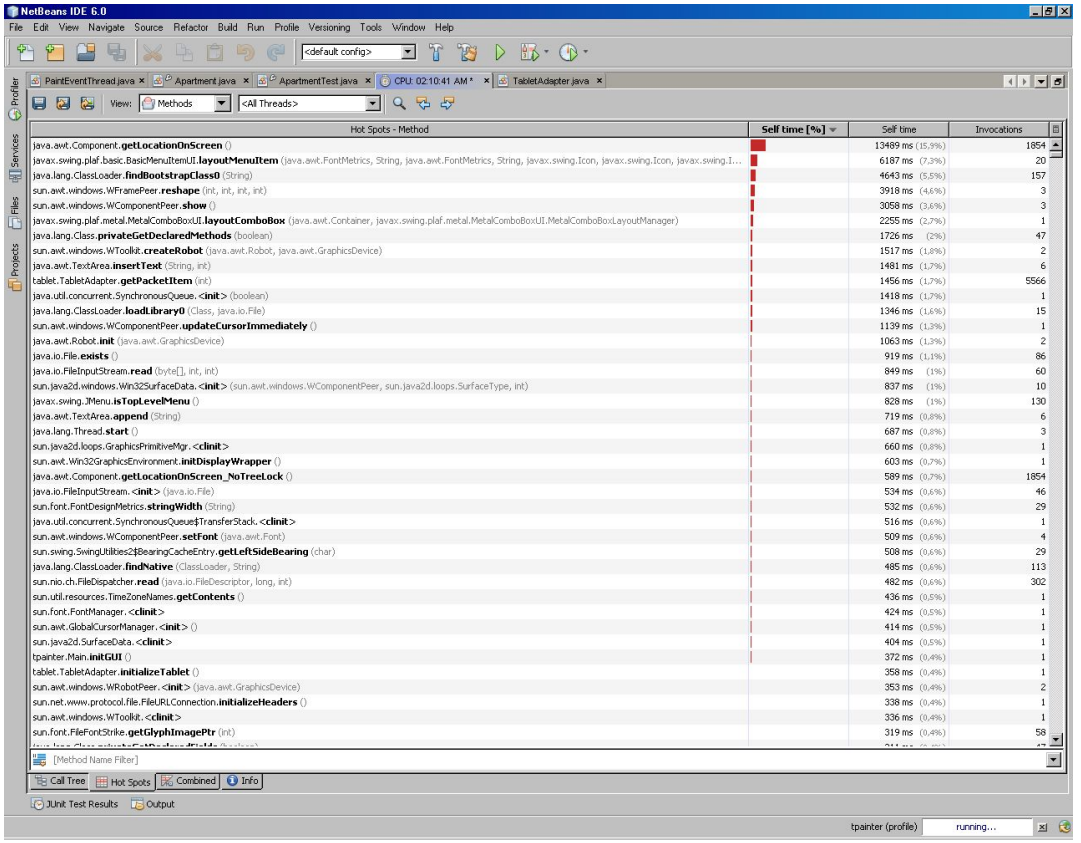
- Все вызванные методы помещаются в дерево
- Каждому узлу ставится в соответствие время работы и количество вызовов.



Отсортированный список всех методов

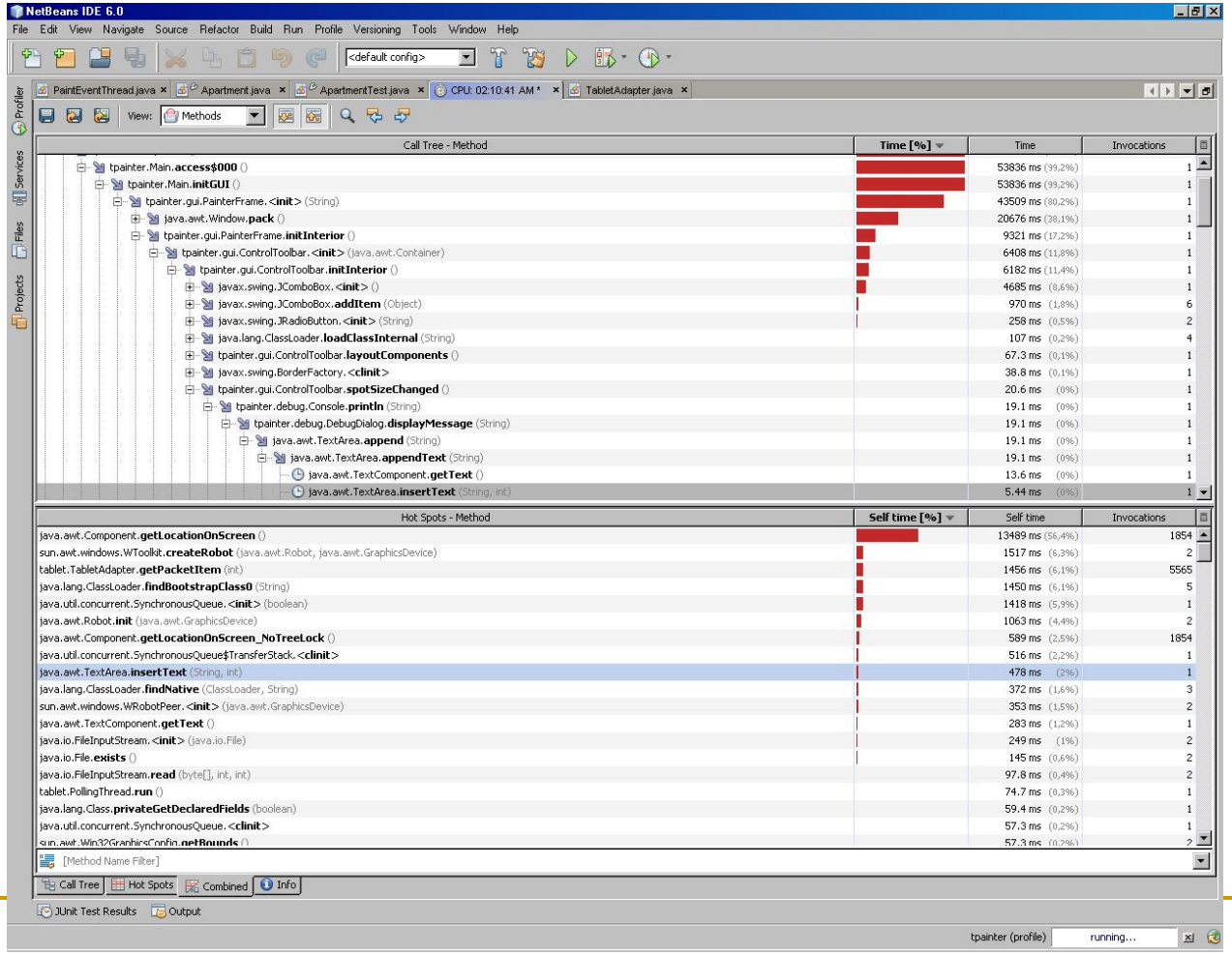
Каждый метод снабжен:

- временем исполнения,
- количеством вызовов,
- процентной шкалой от всего времени работы приложения.



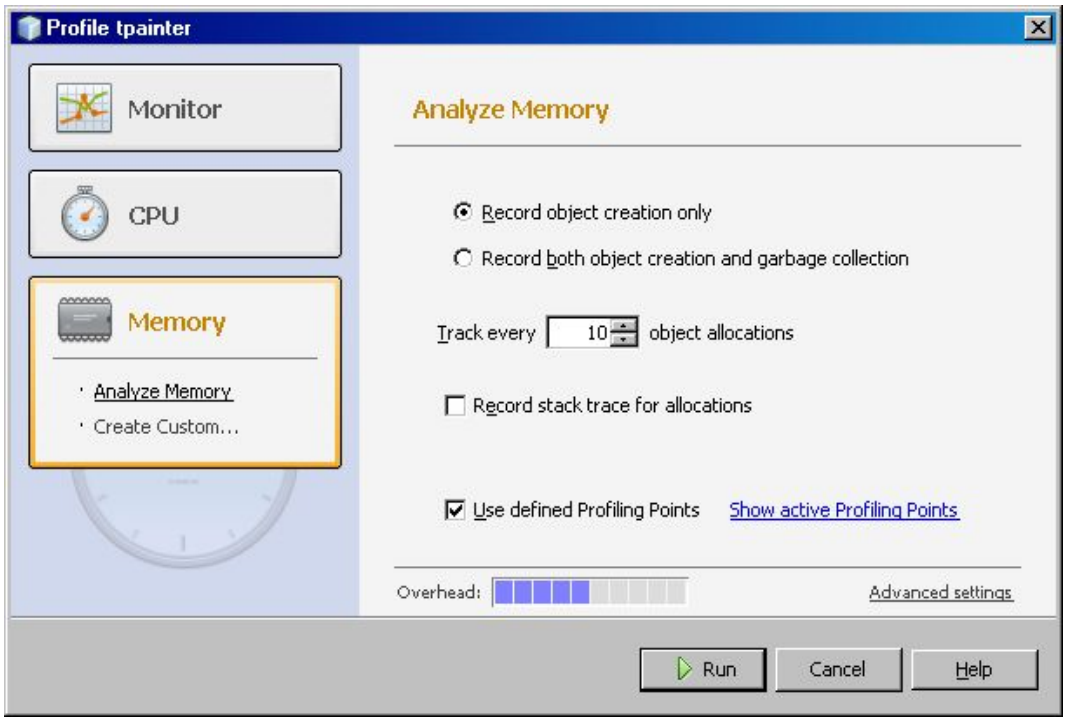
Комбинированное представление

Отображается соответствие стека вызова имени метода.



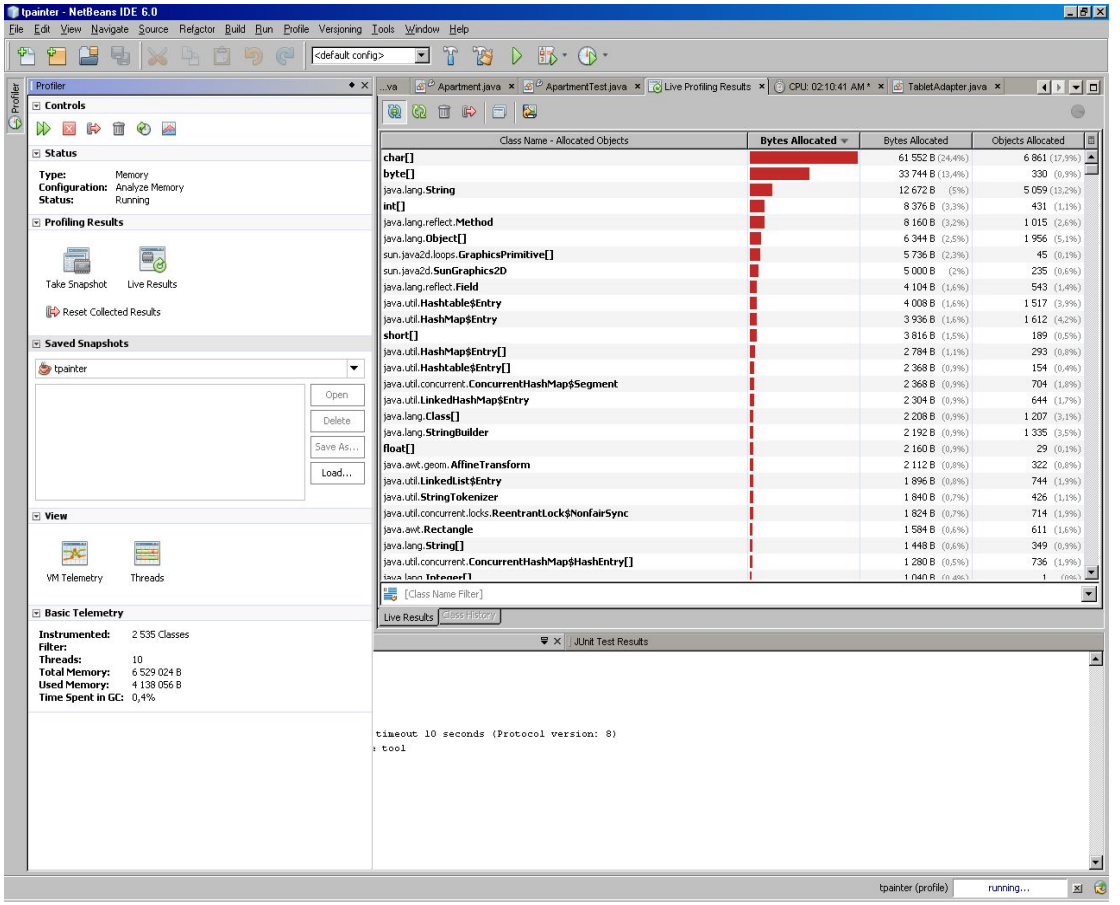
Анализ использования памяти

- Используется
  - для оценки объема используемой памяти
  - для нахождения неудаляемых объектов.



Анализ памяти: результаты в реальном времени

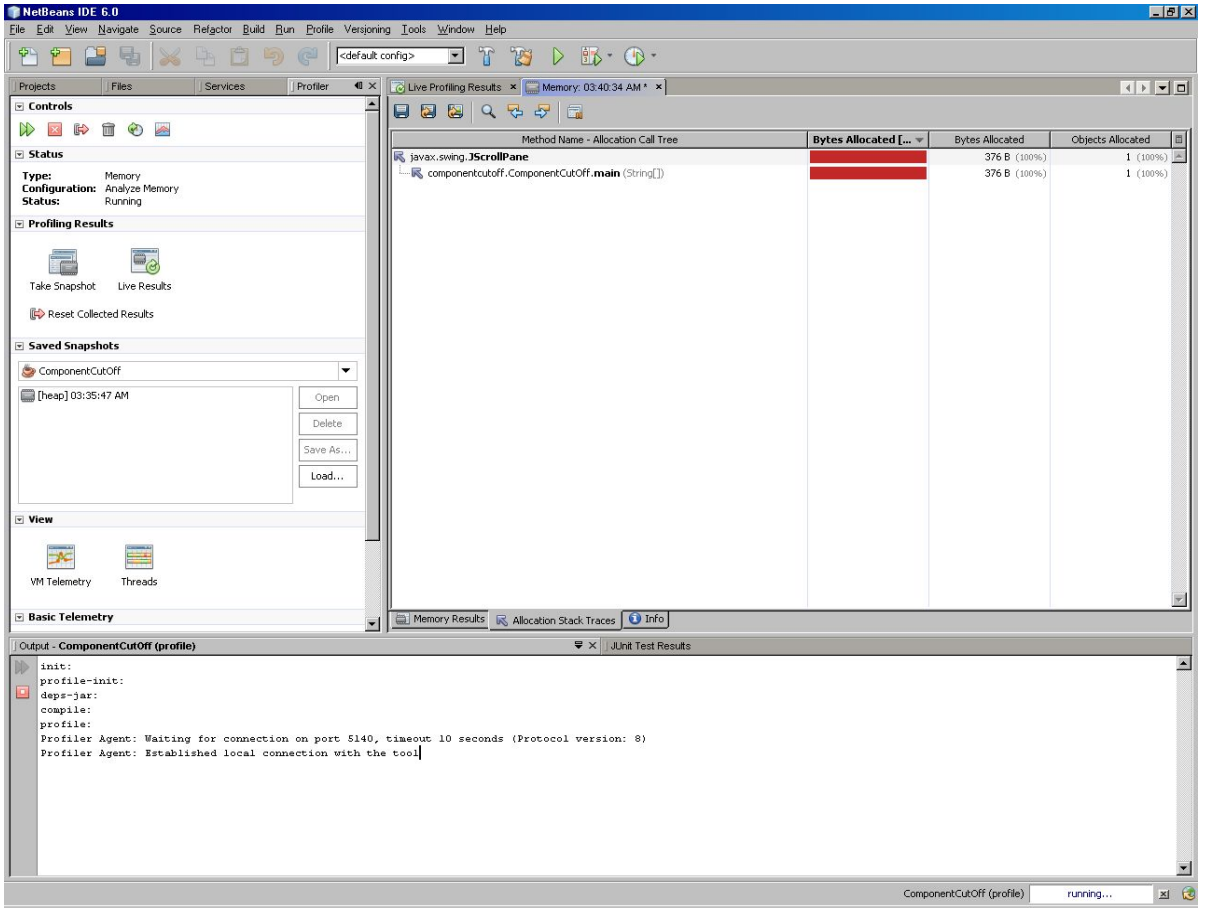
- Отслеживание состава кучи (heap).
- Таблица содержит:
  - Имя класса.
  - Количество доступных (живых) объектов.
  - Расход памяти (в байтах).
  - Усредненное поколение* объекта.
  - Число поколений объекта.



*Поколение - число, отражающее количество пережитых данным объектом сборок мусора)

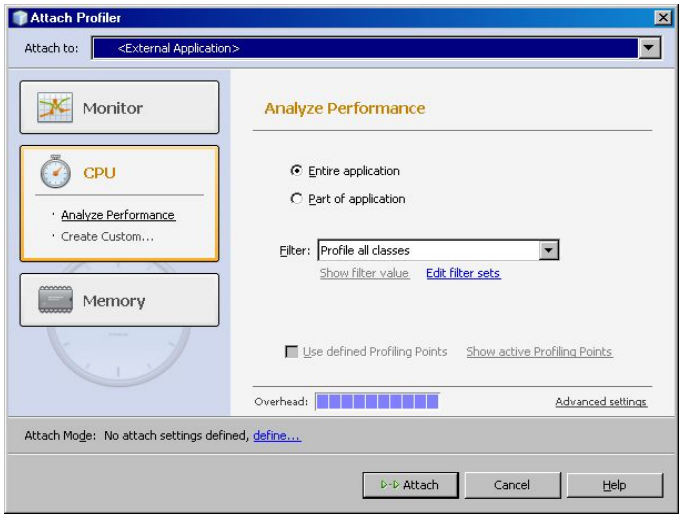
Анализ памяти: сохраненные результаты

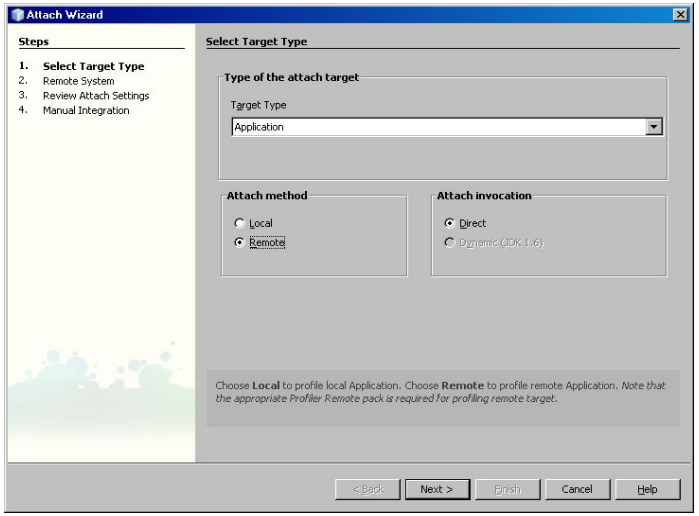
Позволяет получить стек создания каждого объекта.



Профилирование внешней программы

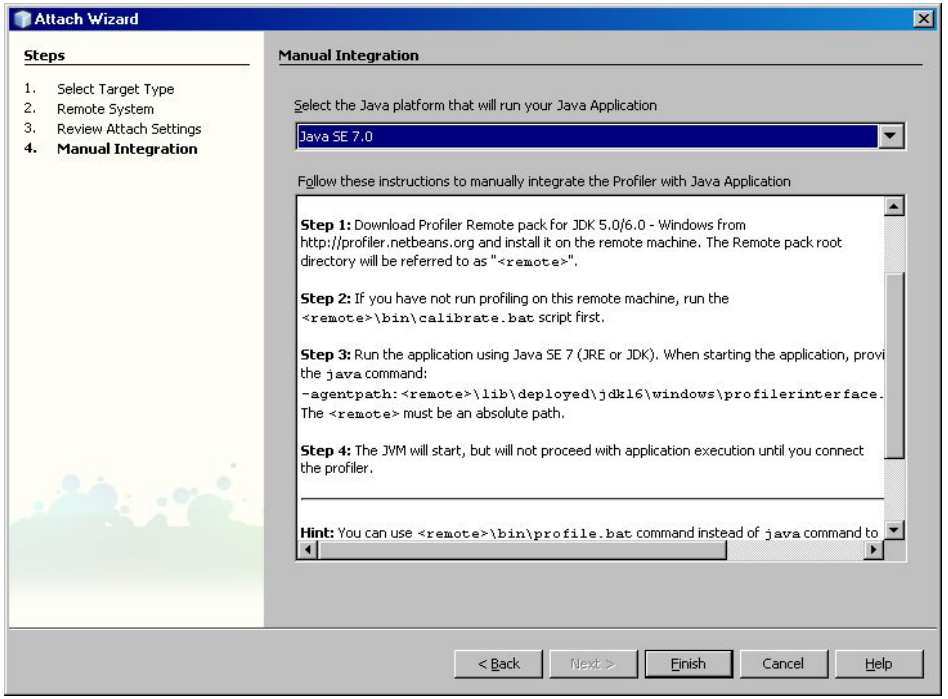
Среда разработки позволяет проводить анализ приложения, запущенного вне среды разработки.





Профилирование внешней программы: запуск целевого приложения

- Для удаленного анализа потребуется запустить целевое приложение с указанными параметрами.
- Среда предоставляет рекомендованные параметры для использования.



Ссылки

- Сайт NetBeans:
  - http://netbeans.org/
- Описание профилировщика:
  - http://www.netbeans.org/kb/60/java/profiler-intro.html
- Поиск утечек памяти:
  - http://www.netbeans.org/kb/articles/nb-profiler-uncoveringleaks_pt1.html
- Онлайн-курсы:
  - http://javapassion.com/
- NetBeans IDE Field Guide by Patrick Keegan, Ludovic Champenois, etc.
- Язык программирования Java и среда NetBeans

Q&A

Спасибо!

Дмитриев Андрей Владиславович
andrei-dmitriev@yandex.ru