

## РАЗРАБОТКА ИНСТРУМЕНТА ДЛЯ ДОСТУПА К ИСТОРИЧЕСКИМ ДАННЫМ JAVA-КОЛЛЕКЦИЙ В СРЕДЕ РАЗРАБОТКИ INTELLIJ IDEA

Сидорова Е.А. (Университет ИТМО, компания JetBrains)

Java-коллекции - наиболее часто используемая часть стандартной библиотеки языка Java. Во время отладки программы на языке Java для такого типа как коллекция интерес представляет не только то, какие элементы находятся в коллекции в данный момент, но и то, где именно эти элементы были добавлены (удалены). Однако получить эту информацию с помощью существующих средств достаточно трудно. Был разработан инструмент, который позволяет пользователю получать во время отладки данные об изменении содержимого Java-коллекций. Полученные результаты свидетельствуют об эффективности реализованного подхода.

**Введение.** Очень много времени разработчика уходит на поиск ошибок в программном обеспечении и их исправление. Поэтому развитие инструментов для отладки является очень важной задачей. Кроме базового инструментария для отладчика программ на языке Java среды разработки IntelliJ IDEA реализованы и более продвинутые возможности: например, особые виды точек останова (Method breakpoints, Exception breakpoints, Field Watchpoints) и фильтры для них (инстанс-фильтр, класс-фильтр, фильтрация по условию и проч.), а также возможность отлаживать асинхронный код. Тем не менее, могут возникнуть проблемы, если мы захотим отладить программу, в которой используются Java-коллекции (типы данных, реализующие интерфейсы `java.util.Collection` или `java.util.Map`).

Рассмотрим следующий пример. Допустим, у нас есть некоторый класс с полем-коллекцией. Пусть пользователь запустил программу под отладчиком, остановился на некоторой точке останова и видит, что в коллекции оказался некоторый набор элементов. Если предположить, например, что в коллекции оказались не те элементы, что должны (т. е. где-то в программе есть ошибка), то у пользователя в такой ситуации может возникнуть вопрос: откуда взялись эти элементы в коллекции? Или почему отсутствуют элементы, которые предположительно должны в данный момент в коллекции находиться? То есть для такого типа данных как коллекция интерес представляет история содержимого коллекции. Чтобы понять, где в программе ошибка, было бы полезно знать то, как менялась коллекция в ходе работы программы: когда и где (т.е. из какого стека вызовов) происходило добавление и удаление элементов и какие именно это были элементы.

В данный момент, чтобы получить эту информацию, пользователь может использовать логирование или существующие точки останова. Самым ближайшим аналогом в данной ситуации является Field Watchpoint. Но данное решение обладает следующими недостатками:

- сложность поиска нужной информации об изменении в коллекции (если с коллекцией проводилось много операций, то данных будет много и в них легко запутаться)
- скорость работы

Целью данной работы была разработка инструмента, который позволял бы легко получать информацию об изменениях в коллекции и при этом работал бы на порядок быстрее своего ближайшего аналога.

**Основная часть.** В IntelliJ IDEA был добавлен новый тип точек останова, который работает с полями классов, имеющими тип коллекция. Инструмент может работать в двух режимах: во-первых, он может работать как обычная точка останова — он останавливает программу в тот момент, когда содержимое коллекции меняется, во-вторых, он может записывать происходящие изменения без остановки и потом показывать их пользователю.

Как известно, Java является компилируемым и интерпретируемым языком одновременно. Java компилируется в байт-код, который потом интерпретируется JVM. Для того, чтобы отслеживать изменения содержимого коллекции, было необходимо, во-первых, детектировать

изменения значения поля, а, во-вторых, детектировать изменения в самой коллекции и записывать их при необходимости. Суть решения состояла в том, что для захвата этой информации изменялся байт-код классов-коллекций и их родителей, а также изменялся байт-код класса, внутри которого находилось отслеживаемое поле, и его наследников (если поле имело модификатор доступа `protected`): нужные инструкции вставлялись в байт-код классов, затем класс перезагружался в JVM.

Одна из трудностей реализации данного решения состояла в том, что если мы изменяем байт-код какого-то метода класса, и в данный момент этот метод лежит на стеке, то сделанные изменения будут доступны только при повторном вызове этого метода. В общем случае эта проблема решалась тем, что ставился временный Field Watchpoint на отслеживаемое поле до тех пор, пока мы не вышли из всех конструкторов (или статических блоков инициализации для статических полей), которые находились на стеках потоков в то время, когда пользователь ставил точку останова на коллекцию.

**Выводы.** Был разработан инструмент, который позволяет пользователю получать доступ к данным об истории изменения содержимого Java-коллекций во время отладки. Инструмент реализован как новый тип точек останова в IntelliJ IDEA Community. Были проведены измерения скорости работы для этих точек останова и их ближайшего аналога. Результаты показывают, что разработанный инструмент работает значительно быстрее своего аналога.

Сидорова Е.А. (автор)