

Выбор сборщика мусора в Java

JSP & Servlets
18 уровень, 6 лекция

ОТКРЫТА

7.1* Как правильно выбрать сборщик мусора

Если у твоего приложения нет строгих требований ко времени задержки, тебе стоит просто запустить приложение и предоставить выбор правильного сборщика самой JVM.

В большинстве случаев настройки по умолчанию отлично работают. При необходимости можно настроить размер кучи для повышения производительности. Если производительность по-прежнему не соответствует ожиданиям, попробуйте изменить сборщик в соответствии с требованиями вашего приложения.

- **Последовательный.** Если в приложении небольшой набор данных (примерно до 100 МБ), и/или оно будет работать на одном процессоре без каких-либо требований к времени задержки.
- **Параллельный.** Если приоритет — пиковая производительность приложения, и требования к времени задержки отсутствуют (или допустимы паузы в одну секунду и более).
- **CMS/G1.** Если время отклика важнее, чем общая пропускная способность, и паузы при сборке мусора должны быть короче одной секунды.
- **ZGC.** Если у времени отклика высокий приоритет и/или задействована очень большая куча.

7.2* Рекомендации по сбору мусора

Избегайте ручных триггеров

Помимо основных механизмов сборки мусора, один из важнейших моментов относительно этого процесса в Java — недетерминированность. То есть невозможно предсказать, когда именно во время выполнения она произойдет.

С помощью методов `System.gc()` или `Runtime.gc()` можно включить в код подсказку для запуска сборщика мусора, но это не гарантирует, что он действительно запустится.

Пользуйтесь инструментами для анализа

Если у тебя недостаточно памяти для запуска приложения, вы столкнетесь с замедлениями, длительным временем сбора мусора, событиями “остановки мира” и в конечном итоге ошибками из-за нехватки памяти. Возможно, это указывает, что куча слишком мала, но также может и означать, что в приложении произошла утечка памяти.

Ты можешь прибегнуть к помощи инструмента мониторинга, например, `jstat` или `Java Flight Recorder` и увидеть, растет ли использование кучи бесконечно, что может указывать на ошибку в коде.

Отдавайте предпочтение настройкам по умолчанию

Если у тебя небольшое автономное Java-приложение, тебе, скорее всего, не понадобится настраивать сборку мусора. Настройки по умолчанию отлично тебе послужат.

Пользуйтесь флагами JVM для настройки

Лучший подход к настройке сборки мусора в Java — установка JVM-флагов. С помощью флагов можно задать сборщик мусора (например, `Serial`, `G1` и так далее), начальный и максимальный размер кучи, размер разделов кучи (например, Молодого поколения, Старшего поколения) и многое другое.

Выбирайте сборщик правильно

Хороший ориентир в плане начальных настроек — характер настраиваемого приложения. К примеру, параллельный сборщик мусора эффективен, но часто вызывает события “остановки мира”, что делает его более подходящим для внутренней

обработки, где допустимы длительные паузы.

В то же время, сборщик мусора CMS предназначен для минимизации задержек, а значит идеально подходит для веб-приложений, где важна скорость реагирования.

[← Предыдущая лекция](#)

[Следующая лекция →](#)

 +18 

Комментарии

[популярные](#) [новые](#) [старые](#)

JavaCoder

Введите текст комментария



У ЭТОЙ СТРАНИЦЫ ЕЩЕ НЕТ НИ ОДНОГО КОММЕНТАРИЯ

ОБУЧЕНИЕ

- [Курсы программирования](#)
- [Курс Java](#)
- [Помощь по задачам](#)
- [Подписки](#)
- [Задачи-игры](#)

СООБЩЕСТВО

- [Пользователи](#)
- [Статьи](#)
- [Форум](#)
- [Чат](#)
- [Истории успеха](#)

КОМПАНИЯ

- [О нас](#)
- [Контакты](#)
- [Отзывы](#)
- [FAQ](#)
- [Поддержка](#)



JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

Русский 

СКАЧИВАЙТЕ НАШИ ПРИЛОЖЕНИЯ

