JAVA helper

BOOKS

SITES

VIDEO

Командная строка

* Посмотреть java\_home : echo %JAVA\_HOME%
* Компиляция javac className.java
* Можно скомпилировать с использованием шаблонного символа подстановки: javac class\*.java
* Команды javac:   
  -verbose – увидим загрузку всех классов (работает также с java)
* Пути к классам указываем через –cp
* Варианты запуска: 1) java –cp ./app.jar;./app2.jar ru.lessons.Main (т.е. передаем список библиотек) 2) META-INF/MANIFEST.MF (добавить Main-Class и Class-Path: ./utils.jar) 3) все собираем в один архив 4) доп. библиотеки положить в "java.ext.dirs" 5) запустить с опцией –Djava.ext.dirs=.
* -verbose:gc – будет работу gc выдавать в system.out

Память

* Существует stack, heap и permgen
* Стеков столько сколько создается потоков выполнения(на стеке хранятся фреймы методов, локальные переменные)
* Из permgen классы не удаляются, но на самом деле могут выгрузиться (при этом статические поля занулятся)
* Классы выгружаются если выгружается их classLoader
* Хорошая вещь WeakReference (слабее чем софт) и SoftReference

Сборщик мусора

* Runtime.getRuntime() – получить ссылку на runtime, у него можно получить totalMemory и freeMemory
* Есть пару вариантов: 1) подсчет ссылок (в java не используется, т.к. могут быть циклические ссылки) 2) поиск достижимых
* В java объекты в памяти могут перемещаться (чтобы не было фрагментации памяти), поэтому нет адресной арифметики и ссылку нельзя привести к числу
* Называется инкрементальный если работает по чуть-чуть (FULLGC значит что перебирает всю память)
* Finalize не стоит использовать (проблемы: 1) подсистема памяти отдельно учитывает классы с finalize 2) в finalize можно оживить объект и если снова потерять то gc убьет объект уже без finalize)