Углубленное программирование на Java
Лекция 5 / 1
«QA и всё такое»

Александр Акбашев

О себе

Образование:

2005 - 2011 МГТУ им. Баумана - магистр

2011 - ? МГТУ им.Баумана - аспирант

2011 - 2013 Open University Skolkovo

Ключевой поворот:

2011 - Mail.Ru Games (Allods Team)

Senior QA Engineer, команда сервера Skyforge

Содержание лекции

1. Контроль качества разрабатываемого программного обеспечения
2. Юнит-тесты
3. Функциональное тестирование
4. Нагрузочное тестирование
5. Профилирование: сервера, памяти, контента

QA

QA

QA

- Критерии качества разрабатываемого ПО

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО
- Формальные проверки

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО
- Формальные проверки

QA

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО
- Формальные проверки

QA Engineer

QA

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО
- Формальные проверки

QA Engineer

- Инспекции

QA

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО
- Формальные проверки

QA Engineer

- Инспекции
- Тесты

QA

QA

- Критерии качества разрабатываемого ПО
- Оценка качества разрабатываемого ПО
- Формальные проверки

QA Engineer

- Инспекции
- Тесты
- Хуки

Тестирование

Черный ящик

ничего не знаем

QA/тестировщики

Белый ящик

знаем всё

разработчики

Виды тестов

Unit тесты

- 100% контроль окружения
- Пишут все программисты
- Перед коммитом изменения проверяются самим программистом
- После коммита изменения проверяются системой Continuous Integration
- Метрика - покрытие кода тестами

Виды тестов

Функциональные тесты

- Пишутся для ключевых элементов
- Всё по-настоящему
- Заданы лишь начальные значения
- “Быстрое” покрытие
- Плохое покрытие

Виды тестов

Нагрузочное тестирование

- Проверяют, выдерживает ли сервер заявленное число пользователей
- Выявление наиболее популярных и наиболее редких багов
- Метрика - максимальное число пользователей в один момент времени (PCCU)

UnitTest

Лучше написать и выполнить неполные тесты,
чем не выполнить полные.

Мартин Фаулер

UnitTest

Проверка отдельных модулей

UnitTest

Безопасный рефакторинг

UnitTest

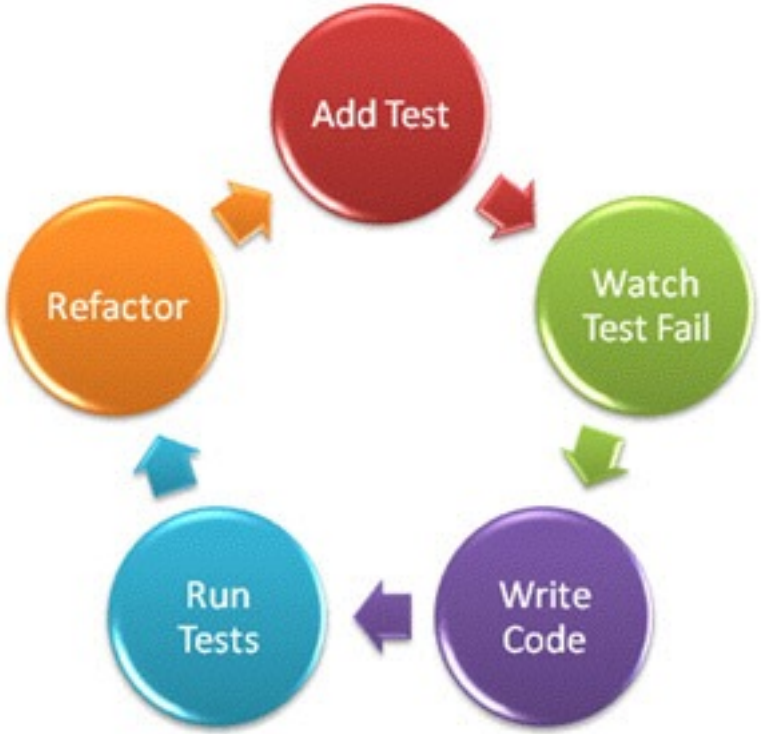
Совместимость модулей

UnitTest

Защита от повторения ошибок

UnitTest

Test Driven Development



UnitTest

Hello, world!

IDEA

Включить Plugins => jUnit

Создать тест Go to => Test

```
public class simpleTest {
    @Before
    public void setUp() throws Exception {
    }

    @After
    public void tearDown() {
     }

    @Test
    public void testSomething() {
     }
}
```

UnitTest

Простейший Unit тест

```
public class Math {
    private int a;
    private int b;

    public Math (int a, int b) {
        this.a = a;
        this.b = b;
    }

    public int getSum() {
        return this.a + this.b;
    }

    public int getDiv() {
        return this.a % this.b;
    }
}
```

```
public class simpleTest {
    private Math math;
    private int a = 3;
    private int b = 1;

    @Before
    public void setUp() throws Exception {
        math = new Math(a, b);
    }
    @Test
    public void testGood() {
        assertEqual(a+b, math.getSum());
    }
    @Test
    public void testBad() {
        assertEqual(a/b, math.getDiv());
    }
```

UnitTest

Организация теста

```
@Test
 public void testGood() {
    // Установка начальных значений
    final Frontend frontend = new Frontend();
    frontend.registerNewUser("test", "test");

    // Действо
    int users = frontend.getUserCount();

    // Проверка
    Assert.equals(users, 1);
}
```

UnitTest

Требования к тестам

UnitTest

Требования к тестам

- Независимость от внешних факторов

UnitTest

Требования к тестам

- Независимость от внешних факторов
- Независимость от порядка выполнения

UnitTest

Требования к тестам

- Независимость от внешних факторов
- Независимость от порядка выполнения
- Зависимость от кода

UnitTest

Требования к тестам

- Независимость от внешних факторов
- Независимость от порядка выполнения
- Зависимость от кода
- Минимум “перекрытий”

UnitTest

Требования к тестам

- Независимость от внешних факторов
- Независимость от порядка выполнения
- Зависимость от кода
- Минимум “перекрытий”
- Один тест - одна проверка

UnitTest

Требования к тестам

- Независимость от внешних факторов
- Независимость от порядка выполнения
- Зависимость от кода
- Минимум “перекрытий”
- Один тест - одна проверка
- Повторяемость результатов

UnitTest

Test Doubles - Дублёры

UnitTest

Test Doubles - Дублёры

- Повышение скорости тестов

UnitTest

Test Doubles - Дублёры

- Повышение скорости тестов
- Независимость от окружения

UnitTest

Test Doubles - Дублёры

- Повышение скорости тестов
- Независимость от окружения
- Полное покрытие

UnitTest

Test Doubles - Дублёры

- Повышение скорости тестов
- Независимость от окружения
- Полное покрытие
- “Свой человек”

UnitTest

Test Doubles - Дублёры

Test Doubles - Дублёры

Dummy - просто заполнить параметр

UnitTest

Test Doubles - Дублёры

Dummy - просто заполнить параметр

Fake - упрощенная и/или неподходящая для “боевого” сервера с работающей реализацией

UnitTest

Test Doubles - Дублёры

Dummy - просто заполнить параметр

Fake - упрощенная и/или неподходящая для “боевого” сервера с работающей реализацией

Stubs - заглушка для вызовов в тесте, проверяет состояние

UnitTest

Test Doubles - Дублёры

Dummy - просто заполнить параметр

Fake - упрощенная и/или неподходящая для “боевого” сервера с работающей реализацией

Stubs - заглушка для вызовов в тесте, проверяет состояние

Spies - stub, следящий за информацией о вызовах

UnitTest

Test Doubles - Дублёры

Dummy - просто заполнить параметр

Fake - упрощенная и/или неподходящая для “боевого” сервера с работающей реализацией

Stubs - заглушка для вызовов в тесте, проверяет состояние

Spies - stub, следящий за информацией о вызовах

Mocks - объекты, заменяющие “настоящие” с заранее заданными условиями

UnitTest

Dummy

```
@Test
 public void testGood() {
    // Установка начальных значений
    final Frontend frontend = new Frontend();
    frontend.registerNewUser("test", "test");

    // Действо
    int users = frontend.getUserCount();

    // Проверка
    Assert.equals(users, 1);
}
```

UnitTest

Fake

UnitTest

Fake

- InMemory Database: H2 etc.

UnitTest

Fake

- InMemory Database: H2 etc.
- Debug WebServer

UnitTest

Fake

- InMemory Database: H2 etc.
- Debug WebServer
- Своё

UnitTest

Stubs

```
public class TestMessage extends Msg {
    private boolean wasExecute;

    public void exec()   {
        wasExecute = true;
    }
}

public boolean wasExecute() {   return wasExecute;  }

@Test
public void messageSystemIsReallyExecMessage() {
        final Msg testMessage = new TestMessage();
        new MessageSystem.send(testMessage);
        Assert.assertTrue(testMessage.wasExecute());
}
```

UnitTest

Spies

```
@Test
public void messageSystemIsReallyExecMessage() {
        final Msg testMessage = mock(Msg.class);
        new MessageSystem.send(testMessage);
        verify(testMessage, atLeastOnce()).exec();
}
```

UnitTest

Mocks

```
public class FrontendTest {
    final private static HttpServletRequest request = mock(HttpServletRequest.class) ;
    ...
    @Test
    public void testDoGet() throws Exception {
        final StringWriter stringWrite = new StringWriter();
        final PrintWriter writer = new PrintWriter(stringWrite);
        final String url = "/login" ;

        when(response.getWriter()).thenReturn(writer);
        when(request.getSession()).thenReturn(httpSession);
        when(request.getPathInfo()).thenReturn(url);

        frontend.doGet(request, response);

        verify(request, atLeastOnce()).getParameter("username");                    // Spy?
        Assert.assertTrue(stringWrite.toString().contains("<!DOCTYPE html>"));  // Stub?
    }
}
```

UnitTest

Устанавливаем Mockito

```
<dependency>
    <groupId>org.mockito</groupId>
    <artifactId>mockito-all</artifactId>
    <version>1.8.4</version>
</dependency>
```

UnitTest

jUnit. Rules.

- Временные файлы
- Ожидание таймаута
- Ожидание исключения

UnitTest

jUnit. Ожидание исключения.

```
@Test(expected = ParseException.class)
  public void ensureThatInvalidFormatThrowException() throws ParseException {
      final DateFormat formatter = new SimpleDateFormat("HH:mm:ss");
      formatter.parse("2013-12-12 12:12:02");
  }

@Rule
   public final ExpectedException thrown = ExpectedException.none();

@Test
   public void ensureExceptionWithRule() throws ParseException {
       thrown.expect(NullPointerException.class);
       thrown.expectMessage("Oups");
       throw new NullPointerException("Oups");
   }
```

FunctionalTest

Необходимое, но недостаточное условие работоспособности.

Практика

FunctionalTest

Функциональное тестирование

Реализует ли требуемый функционал

- Поднимается ли сервер?
- Можно ли зайти в игру?
- Можно ли играть?

Ручное тестирование должно быть более глубоким

FunctionalTest

Selenium

Без единой строки кода (plugin Firefox)

- Быстро
- Слабо контролируемо
- Несерьезно

Много кода на Java

- Не так быстро
- Полный контроль
- Путь джедая

FunctionalTest

Установка Selenium

```
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>2.4.0</version>
</dependency>
```

FunctionalTest

Схема работы теста

FunctionalTest

Реализация тестирования аутентификации

```
public void testLogin(@NotNull String url,@NotNull String username,@NotNull String password) {
        WebDriver driver = new HtmlUnitDriver(true);
        driver.get(url);
        // Find the text input element by its name
        WebElement element = driver.findElement(By.name("userName"));
        element.sendKeys(username);
        WebElement element = driver.findElement(By.name("password"));
        element.sendKeys(password);

        // Now submit the form. WebDriver will find the form for us from the element
        element.submit();

        // Wait for the page to load, timeout after 10 seconds
        (new WebDriverWait(driver, 10)).until(new ExpectedCondition<Boolean>() {
                @Override
                @NotNull
                public Boolean apply(@NotNull WebDriver d) {
                        final int id = d.findElement(By.name("id"));
                        return id != 0;
                        }
                });
         driver.quit();

}
```

LoadTest

[robic]: привет, нужно просетапить сервер чтоб держал нагрузку

[robic]: нужен mysql и apache

[ad_minic]: какая нагрузка?

[robic]: большая

[ad_minic]: круто

[ad_minic]: тогда ставь большой апач и большой мускуль на большой сервер

bash.org

LoadTest

Моделирование

тетрация - Microsoft Excel

Home Insert Page Layout Formulas Data Review View

B1 =$A1^A1

	A	B	C	D	E	F	G	H	I	J
1	0,1	0,794328	0,160573	0,690919	0,203742	0,625544	0,23684	0,579642	0,263244	0,545451
2	0,2	0,72478	0,311459	0,605759	0,377218	0,544924	0,416021	0,511934	0,438706	0,49358
3	0,3	0,696845	0,43215	0,594344	0,488911	0,555084	0,512576	0,539492	0,522289	0,533219
4	0,4	0,693145	0,529871	0,61538	0,569005	0,593704	0,580418	0,587528	0,583713	0,585757
5	0,5	0,707107	0,612547	0,654041	0,635498	0,643719	0,640061	0,641686	0,640964	0,641285
6	0,6	0,736022	0,686617	0,704166	0,697882	0,700125	0,699323	0,69961	0,699508	0,699544
7	0,7	0,779056	0,757395	0,76327	0,761672	0,762106	0,761988	0,76202	0,762012	0,762014
8	0,8	0,836512	0,829724	0,830982	0,830748	0,830792	0,830784	0,830785	0,830785	0,830785
9	0,9	0,909533	0,90862	0,908707	0,908699	0,908699	0,908699	0,908699	0,908699	0,908699
10	1	1	1	1	1	1	1	1	1	1
11	1,1	1,110534	1,11165	1,111768	1,111781	1,111782	1,111782	1,111782	1,111782	1,111782
12	1,2	1,244565	1,254718	1,257043	1,257576	1,257698	1,257726	1,257733	1,257734	1,257734
13	1,3	1,406457	1,446293	1,461489	1,467327	1,469576	1,470444	1,470779	1,470908	1,470958
14	1,4	1,601693	1,714163	1,780276	1,820322	1,845016	1,860409	1,87007	1,876159	1,880007

LoadTest

Ожидаемое использование

LoadTest

Нескольких пользователей

Одновременно

LoadTest

Как происходит?

- Формируется список активностей геймммеханики
- Формируются параметры игрового дизайна
- Реализуется бот
- Проводится испытание
- Анализируются результаты

LoadTest

Рекомендуемые инструменты

jMeter

LoadBalancer

Yandex-tanki

LoadTest

Анализируются результаты

1. Потребление CPU, %
2. Диск (iowait)
3. Сеть, Mb/s
4. Время отклика, ms
5. CCU, users
6. Load, parrots
7. Использование памяти, МБ

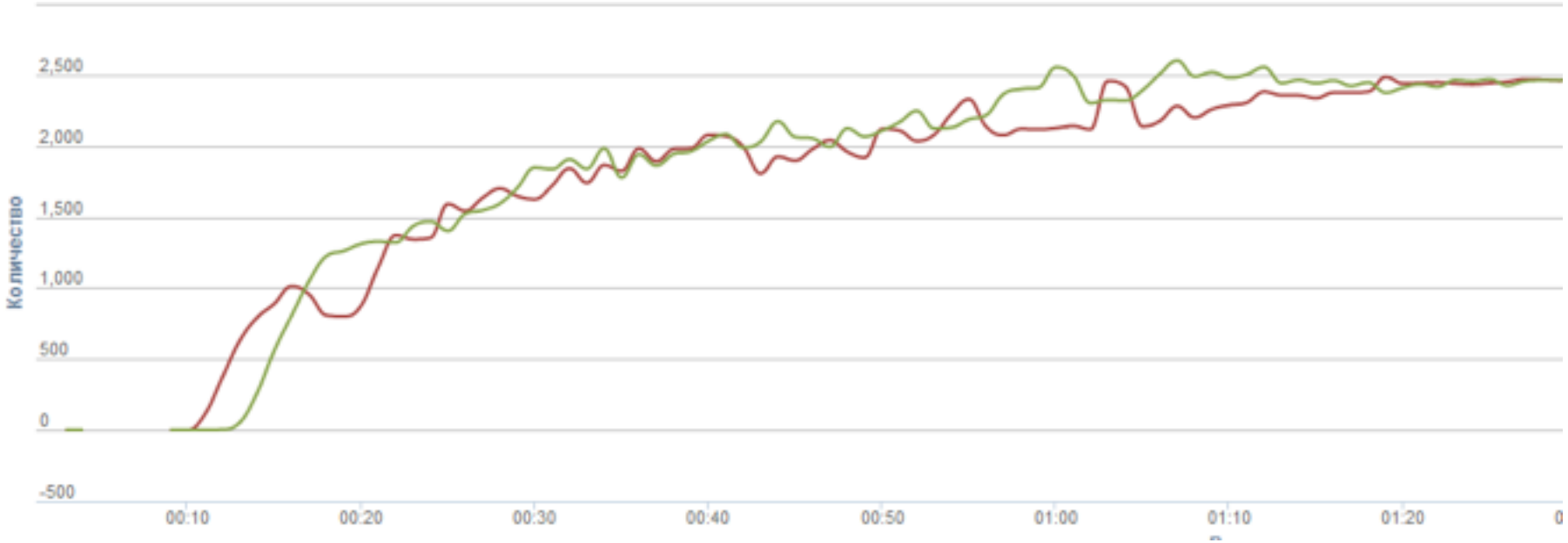
Highload от Быкова

LoadTest

CCU, users

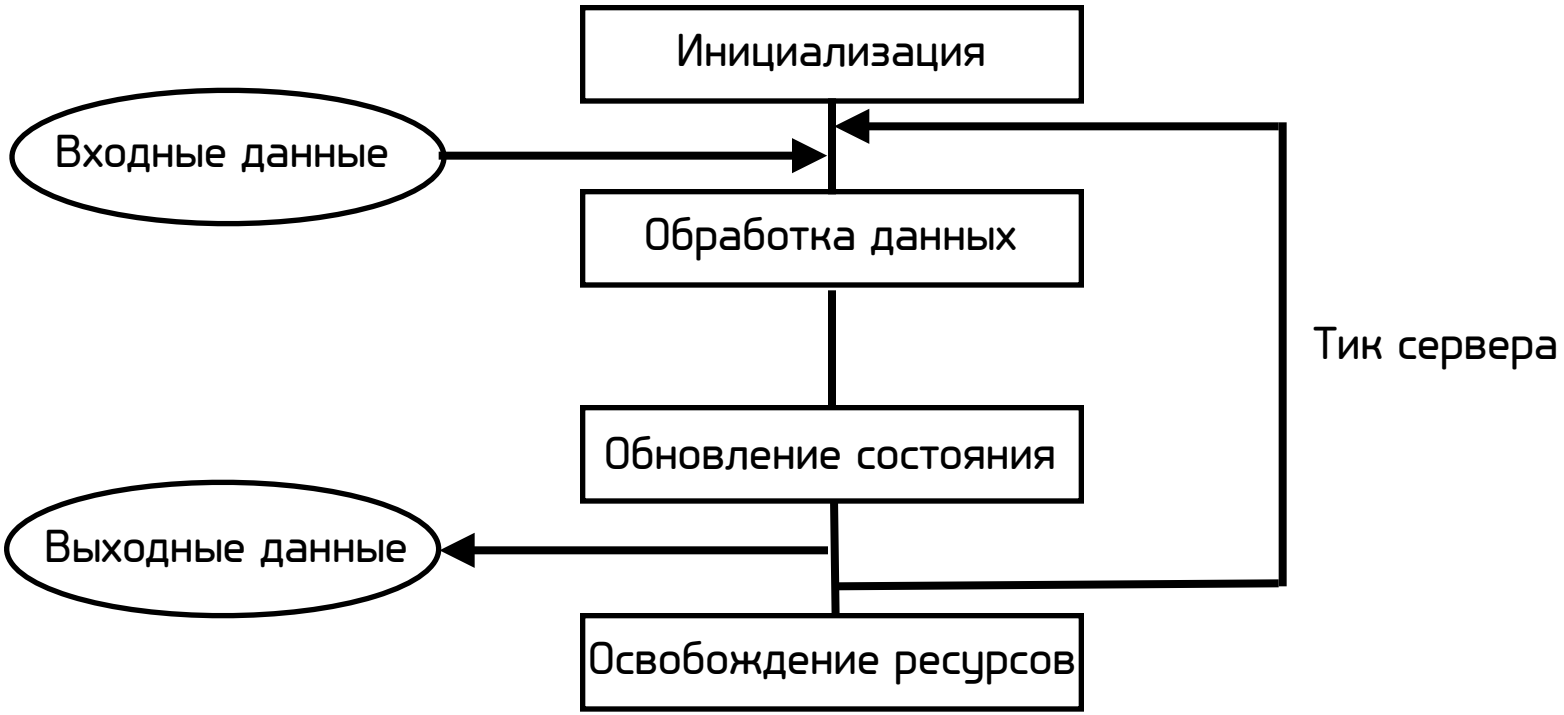
```
private Map<Integer, Avatar> avatars = new HashMap<>();

public final int getTotalAvatars() {
    return avatars.size();
}
```



LoadTest

Server FPS



LoadTest

Load [parrots]

```
public void run () {
    while(true) {
        doJob(this);
        Thread.sleep(100);    // фиксированное время ожидания потока
    }
}

private static final int TICK_TIME = 20;
public void run () {
    while(true) {
        long startTime = System.currentTimeMillis();
        doJob(this);
        int deltaTime = System.currentTimeMillis() - startTime;
        float load = deltaTime / TICK_TIME;
        if (load < 1)              // динамическое время ожидания потока
            Thread.sleep( TICK_TIME - deltaTime );
    }
}
```

LoadTest

Многопоточность

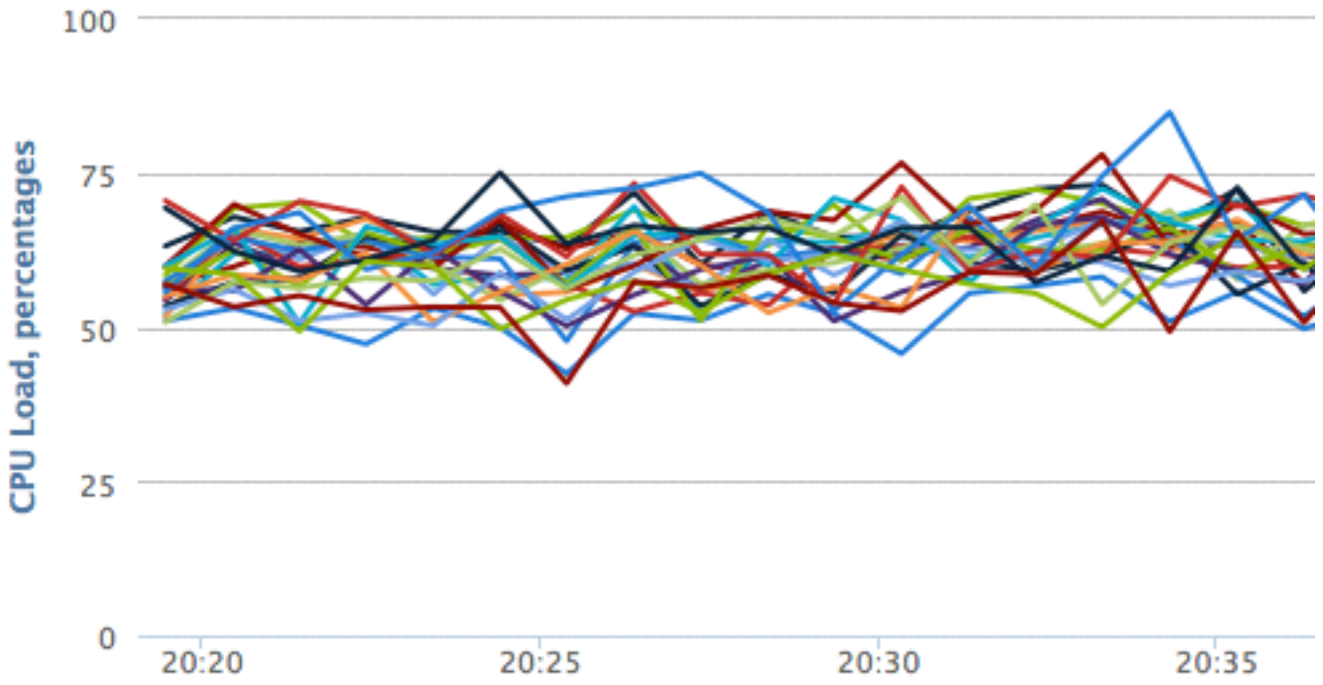
Не спать вредно - CPU starvation

Много спать тоже вредно

Чем больше нагрузка на CPU, тем лучше

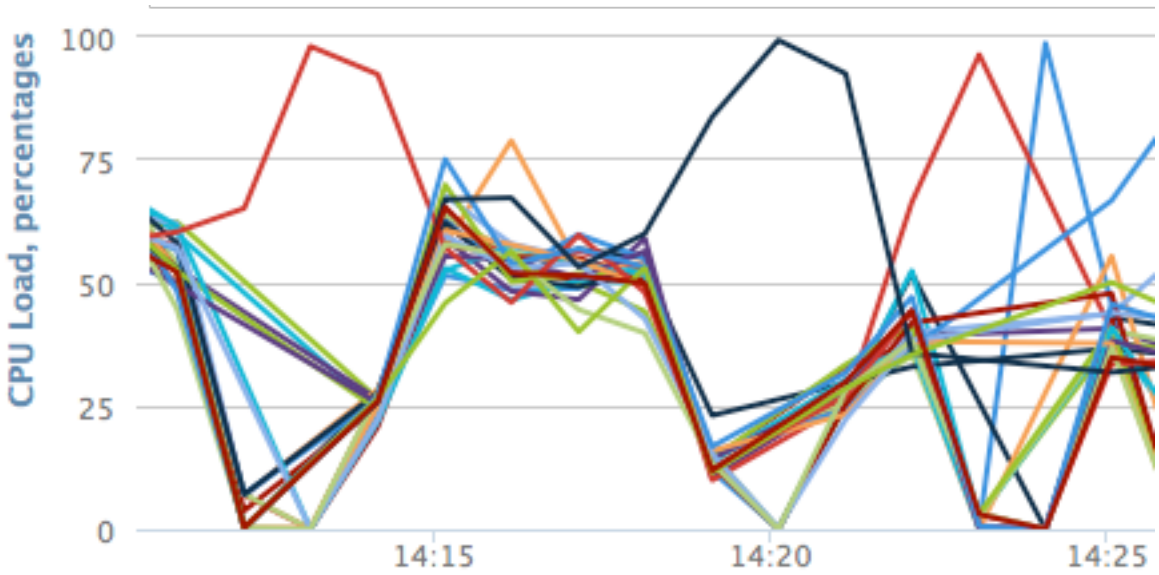
LoadTest

Нагрузка на процессор



LoadTest

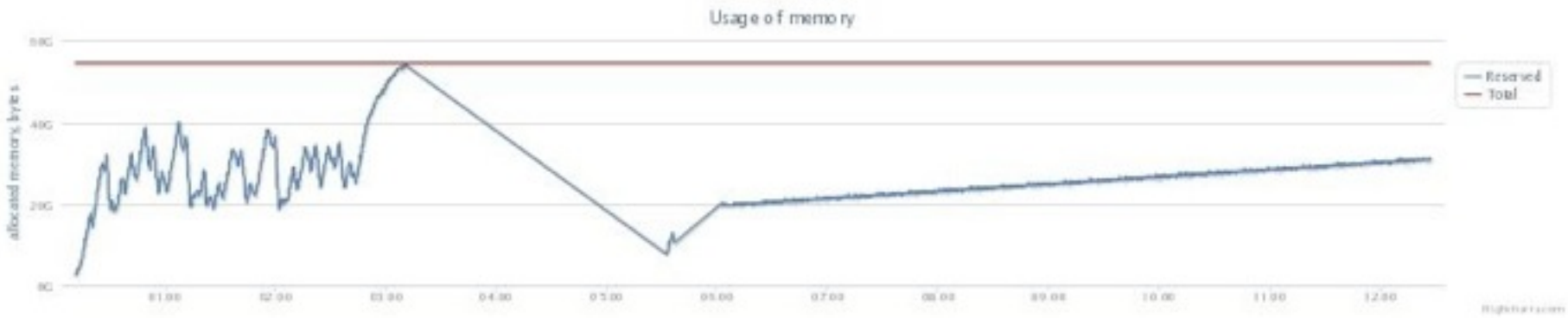
Нагрузка на процессор



LoadTest

Память

```
long free = Runtime.getRuntime().freeMemory();
long max = Runtime.getRuntime().maxMemory();
```



Углубленное программирование на Java
Лекция 5 / 2
«Вся правда о GC»

Александр Акбашев

Garbage Collector

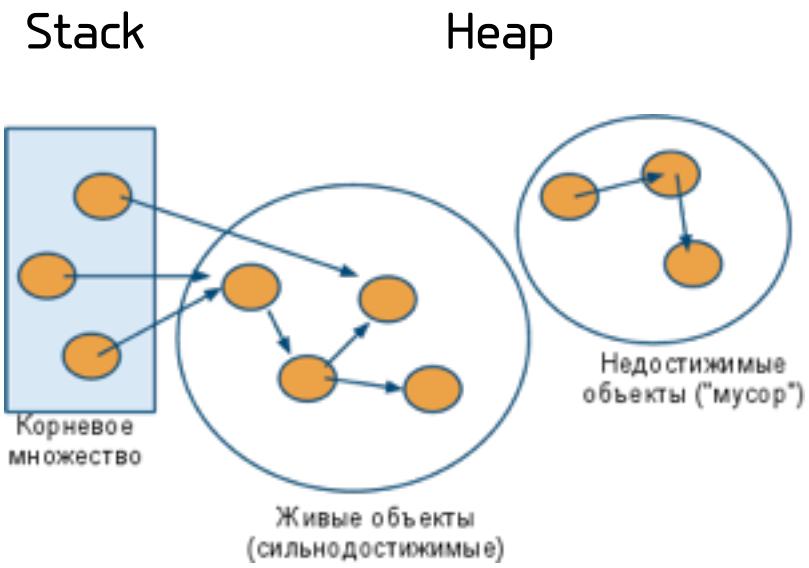
BetrayeR: В java сборкой мусора управляет виртуальная машина, а не пользователь.
Вызов метода gc() - это лишь запрос на сборку мусора.
Спецификация не регламентирует время сборки мусора и не гарантирует, начнется ли она вообще.
BetrayeR: вот зачем тогда этот метод gc() нужен?
BetrayeR: типо:
- java, пожалуйста, начни сборку мусора
- нет
- ну и фу такой быть!

bash.org

Garbage Collector

Garbage collector (GC)

- Поиск недостижимых объектов
- Освобождение памяти
- Вызов finalize



Типы ссылок

- Strong references
- Weak references
- Soft references
- Phantom references

Garbage Collector

Finalize

Когда нужно переопределять?

Garbage Collector

Finalize

Когда нужно переопределять?

НИКОГДА

Garbage Collector

Autocloseable

```
class Resource implements AutoCloseable {
@Override
public void close() throws Exception {}
}


try (Resource resource = new Resource()) {
  throw new Exception();
}
```

Garbage Collector

Strong References

Объект не удаляется

new

Garbage Collector

Soft References

Объект удаляется...если нужна память

- Учитывает количество ссылок
- Используется для memory-sensitive cache

```
new SoftReference <T> (T obj);
```

Garbage Collector

Weak References

Объект удаляется

Атомарно очищает все слабые ссылки на объект

Если надо, помещает объекты на финализацию

Перед использовать - проверять на NULL

WeakHashMap

```
new WeakReference <T> (T obj);
```

Garbage Collector

Phantom References

Объект удаляется с преферансом и куртизанками

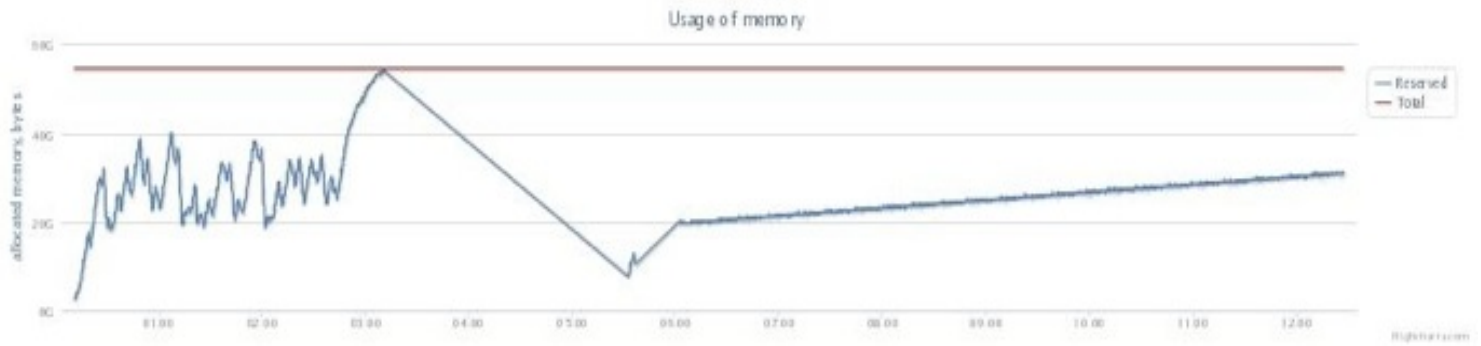
Выполняется метод finalize()

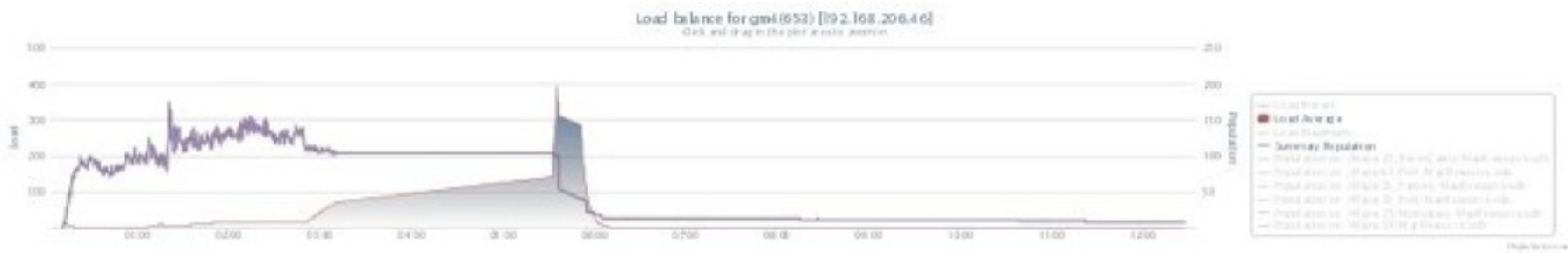
Ссылка помещается в очередь ReferenceQueue

```
new PhantomReference <T> (T obj, ReferenceQueue<? super T> queue);
new ReferenceQueue <T> ( );
```

Garbage Collector

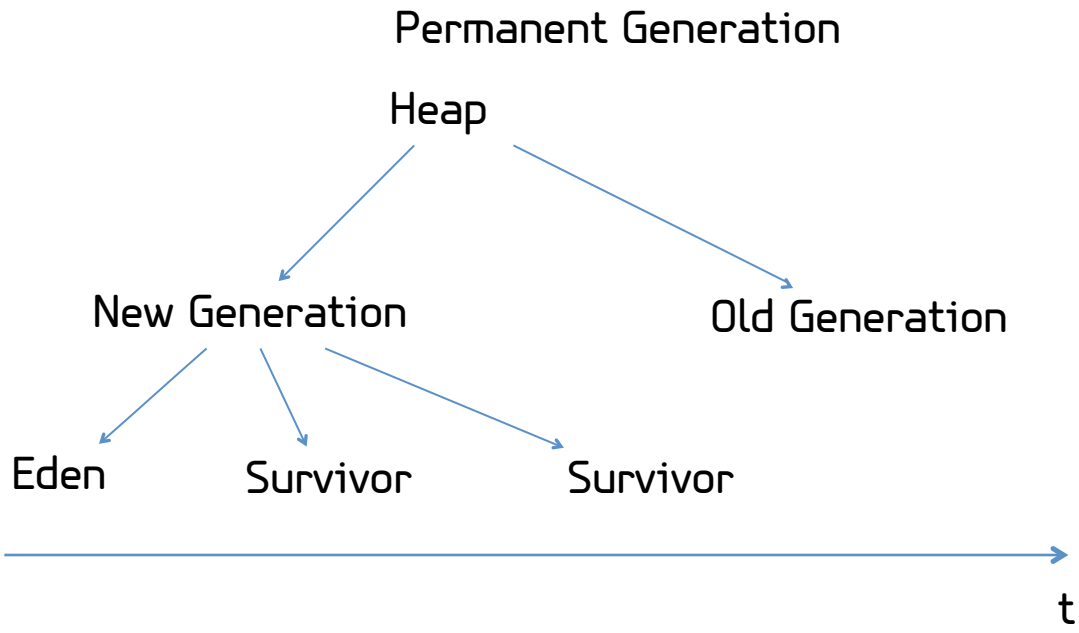
Немного о потреблении памяти





Garbage Collector

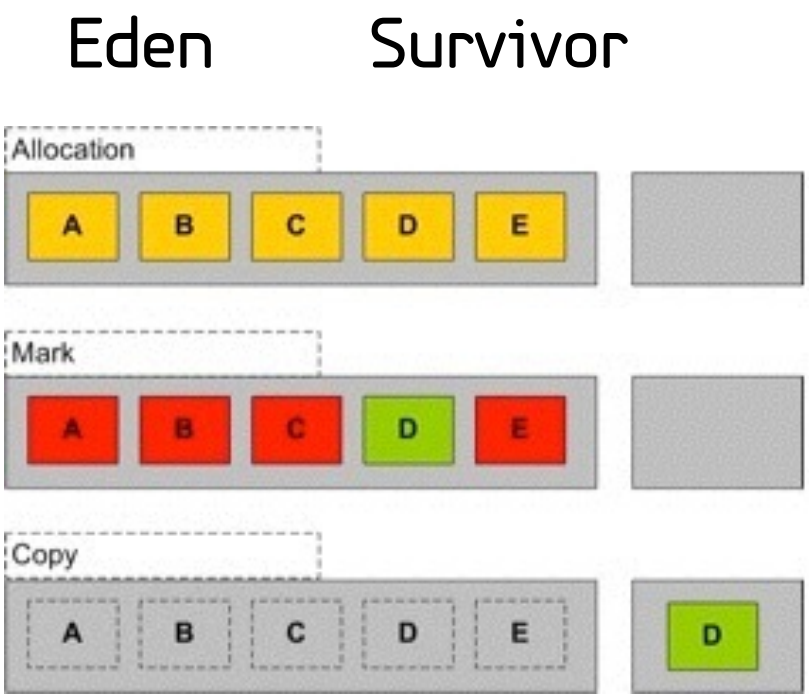
HotSpot JVM



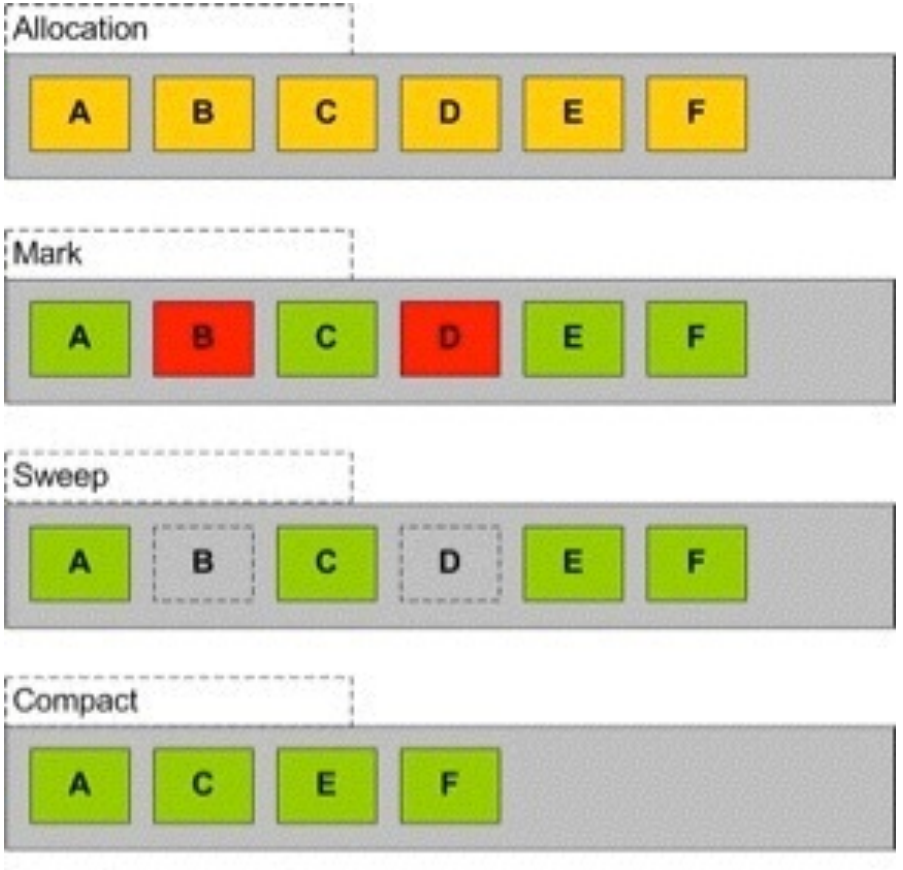
-XX:PermSize **-XX:MaxPermSize**

-Xms **-Xmx**

Garbage Collector

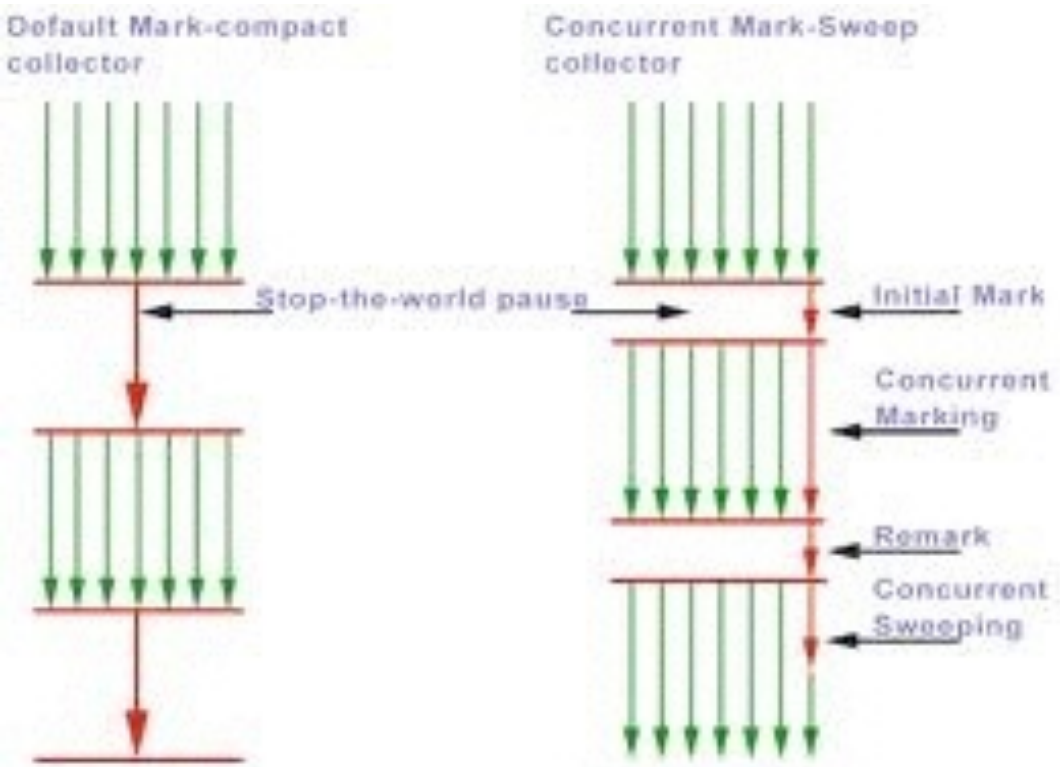


Copy Collection



Mark-compact algorithm

Garbage Collector



Garbage Collector

Что делать? Как настраивать?

jvisualvm входит в JDK

Итого

Полезное

http://www.oracle.com
http://martinfowler.com
“Effective Unit Testing”, Lasse Koskela

http://www.highcharts.org

Спасибо за внимание

Акбашев Александр,

a.akbashev@corp.mail.ru


```
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>2.4.0</version>
</dependency>
 <dependency>
      <groupId>org.mockito</groupId>
      <artifactId>mockito-all</artifactId>
      <version>1.8.4</version>
  </dependency>
```