НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ ИГРОВОГО СЕРВЕРА

Антон Гоцюк
Ведущий программист IT Territory/My.Games

Наша студия

- 200+ сотрудников
- 15+ успешных проектов
- 100+ миллионов игроков

IT TERRITORY

- Аллоды Онлайн
- Hawk: Freedom Squadron
- Space Justice
- World Above
- Rush Royale

IT TERRITORY:

ПЛАН

ПЛАН

- Какие проекты мы тестируем

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

ИСХОДНЫЕ ДАННЫЕ

Метрики наших проектов

- TCP + синхронный прикладной протокол
- 10+k CCU
- Тысячи запросов в секунду (RPS)
- latency 99% < 50 ms

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?

MY.GAMES

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

```
{ "actions":[
   { "type“: "message", "message": { "request": "Cheat", "action": "unlockstory", "id": 44 } },
   { "type“: "sleep", "duration": 100 },
   { "type": "loop", "count": 30,
    "action“: [
     { "type": "mission", "complete": true },
     { "type": "message", "message": { "request": "Buy", “resource": 3746016796, "id": 124 } },
     { "type": "sleep", "duration": 1000 }
    ]
   }
] }
```

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

```
{ "actions":[
    { "type“: "message", "message": { "request": "Cheat", "action": "unlockstory", "id": 44 } },
    { "type“: "sleep", "duration": 100 },
    { "type": "loop", "count": 30,
      "action“: [
        { "type": "mission", "complete": true },
        { "type": "message", "message": { "request": "Buy", “resource": 3746016796, "id": 124 } },
        { "type": "sleep", "duration": 1000 }
      ]
    }
] }
```

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

```
{ "actions":[
   { "type“: "message", "message": { "request": "Cheat", "action": "unlockstory", "id": 44 } },
   { "type“: "sleep", "duration": 100 },
   { "type": "loop", "count": 30,
    "action“: [
     { "type": "mission", "complete": true },
     { "type": "message", "message": { "request": "Buy", “resource": 3746016796, "id": 124 } },
     { "type": "sleep", "duration": 1000 }
    ]
   }
] }
```

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

```
{ "actions":[
   { "type“: "message", "message": { "request": "Cheat", "action": "unlockstory", "id": 44 } },
   { "type“: "sleep", "duration": 100 },
   { "type": "loop", "count": 30,
     "action“: [
       { "type": "mission", "complete": true },
       { "type": "message", "message": { "request": "Buy", “resource": 3746016796, "id": 124 } },
       { "type": "sleep", "duration": 1000 }
     ]
   }
] }
```

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

```
{ "actions":[
   { "type“: "message", "message": { "request": "Cheat", "action": "unlockstory", "id": 44 } },
   { "type“: "sleep", "duration": 100 },
   { "type": "loop", "count": 30,
     "action“: [
       { "type": "mission", "complete": true },
       { "type": "message", "message": { "request": "Buy", “resource": 3746016796, "id": 124 } },
       { "type": "sleep", "duration": 1000 }
     ]
   }
] }
```

Count?

MY.GAMES

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку

MY.GAMES

ПЕРВЫЕ ПОПЫТКИ

- Как было на Аллодах?
- Golang на скорую руку
- Нужно идти от требований

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими

MY.GAMES

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Использовать существующий код протокола

MY.GAMES

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Использовать существующий код протокола
- На боевом стеке

MY.GAMES

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Использовать существующий код протокола
- На боевом стеке
- Сценарии могут писать QA

MY.GAMES

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Использовать существующий код протокола
- На боевом стеке
- Сценарии могут писать QA
- Автоматизированный анализ результатов

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Gatling

Преимущества:

- Асинхронность
- Простота написания кейсов для разработчика
- Интеграция с Jenkins
- Отличная визуализация результатов

Недостатки:

- Нет TCP из коробки
- Сложности с распределением нагрузки в бесплатной версии

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Apache JMeter

Преимущества:

- Поддерживает большое количество протоколов
- Большое комьюнити, неплохая поддержка
- Наличие графического интерфейса

Недостатки:

- Достаточно высокий порог вхождения
- Более сложная разработка нетривиальных сценариев (сравнивая с Gatling)
- Повышенное потребление ресурсов
- XML

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Tsung

Экзотика на Erlang

Преимущества:

- Поддерживает большое количество протоколов
- Асинхронность (один поток – несколько пользователей)
- Отказоустойчивость
- Анализ метрик ОС из коробки

Недостатки:

- Только *nix
- Отсутствие графической оболочки
- Далекий от нас язык
- Скромное комьюнити

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Grinder

Jython: не Java,
но и не Python

Преимущества:

- Поддерживает большое количество протоколов
- Большая гибкость написания скриптов
- Продвинутая поддержка HTTP

Недостатки:

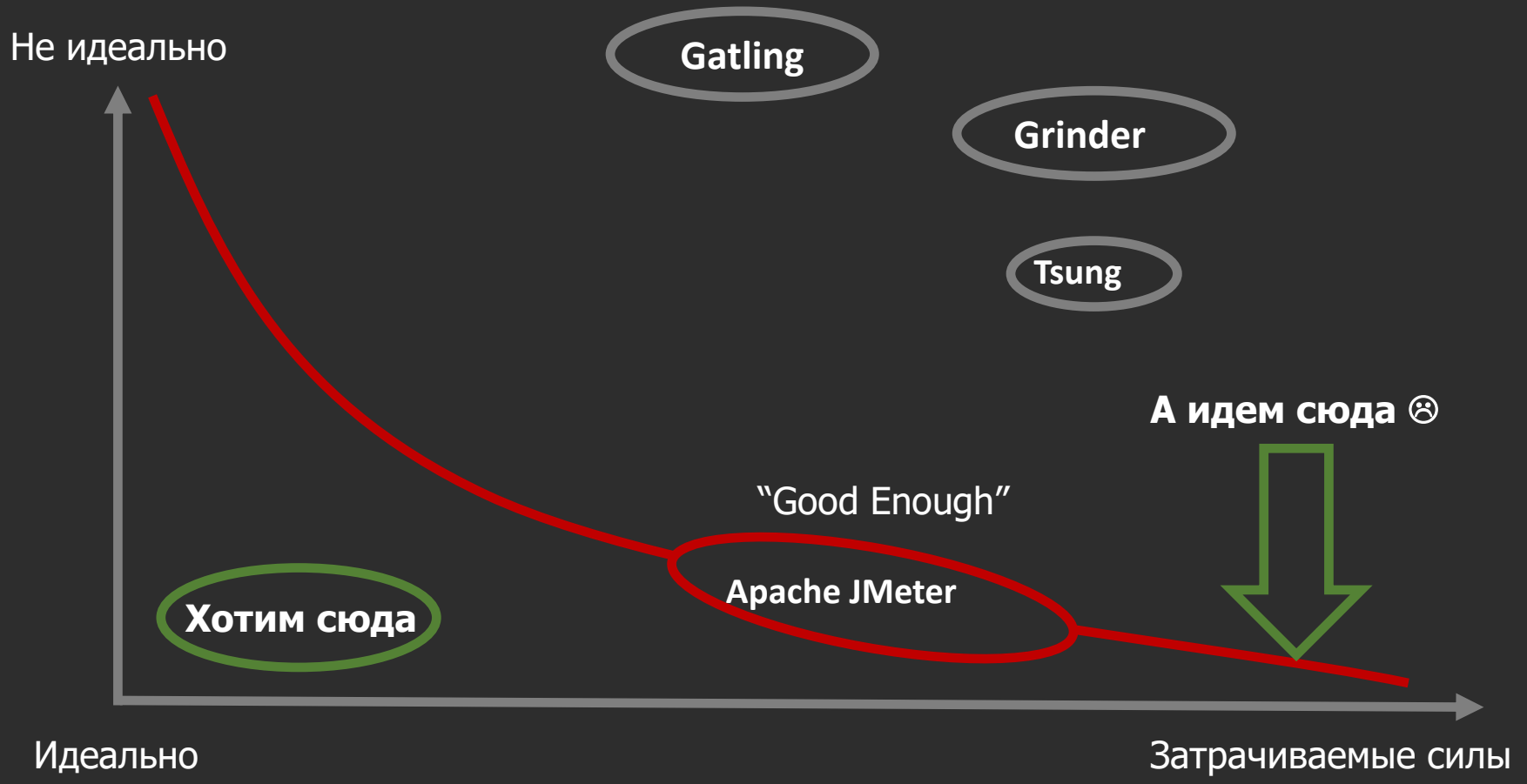
- Высокий порог вхождения
- Нет удобного инструмента для написания скриптов
- Отчеты уступают по качеству конкурентам
- Маленькое комьюнити

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Остальные

- Locust, FunkLoad : только HTTP
- Loader.io: скудный бесплатный план
- Loadrunner, WebLoad: всего 50 «пользователей» в бесплатной версии
- LoadView, StressStimulus, LoadNinja: только платная версия

ВЫБОР


Не идеально
Gatling
Grinder
Tsung
А идем сюда ☹
"Good Enough"
Apache JMeter
Хотим сюда
Идеально
Затрачиваемые силы

MY.GAMES

ТЕХНОЛОГИИ

Vert.x

Open-source framework для построения реактивных распределенных event-driven приложений работающих на JVM.

Мультиязычный, гибкий и эффективный инструмент.

И мы уже используем его в production.

MY.GAMES

VERT.X

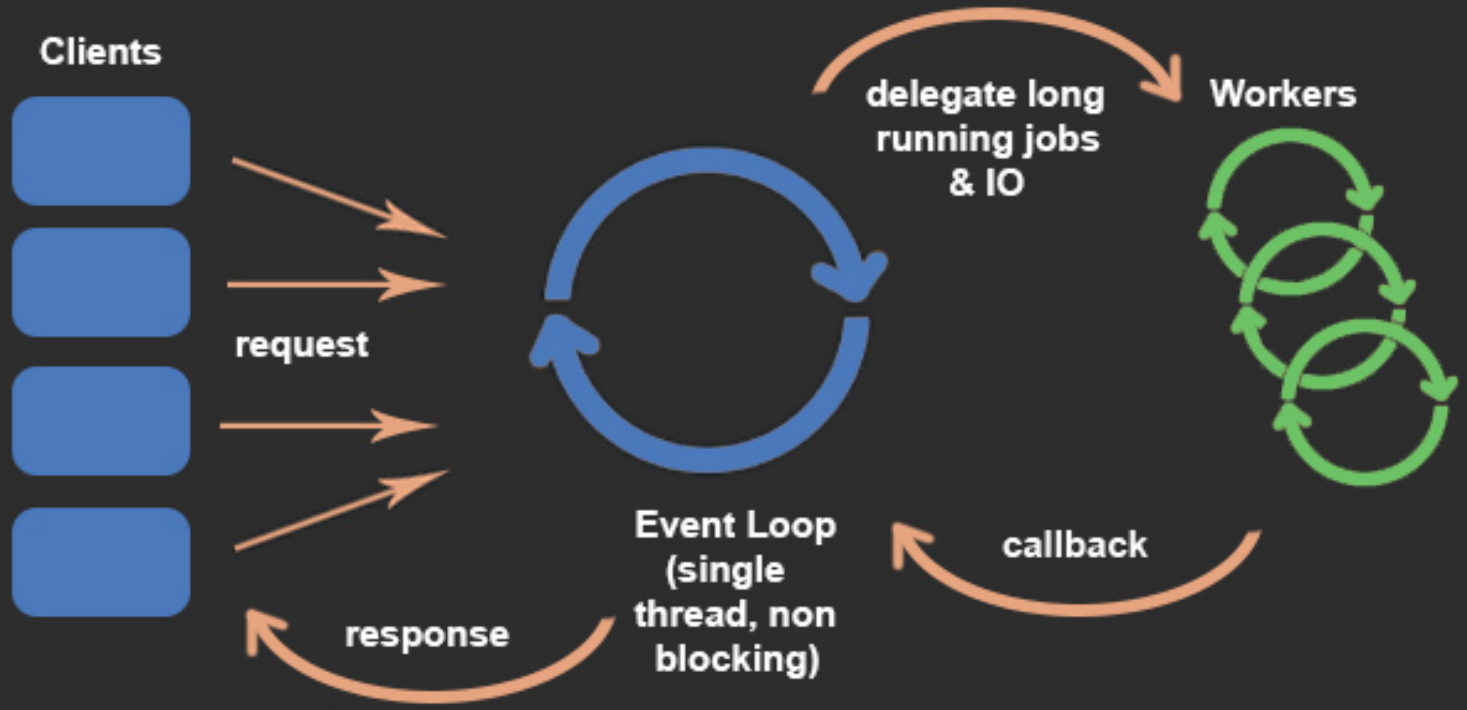
- Приложение — это набор verticle’ов
- Verticle — actor-like агент системы, микросервис
- Общение через шину: Event Bus
- Хранение распределенных данных: Hazelcast, Apache Ignite и другие

wikipedia.org/wiki/Actor_model

MY.GAMES

VERT.X

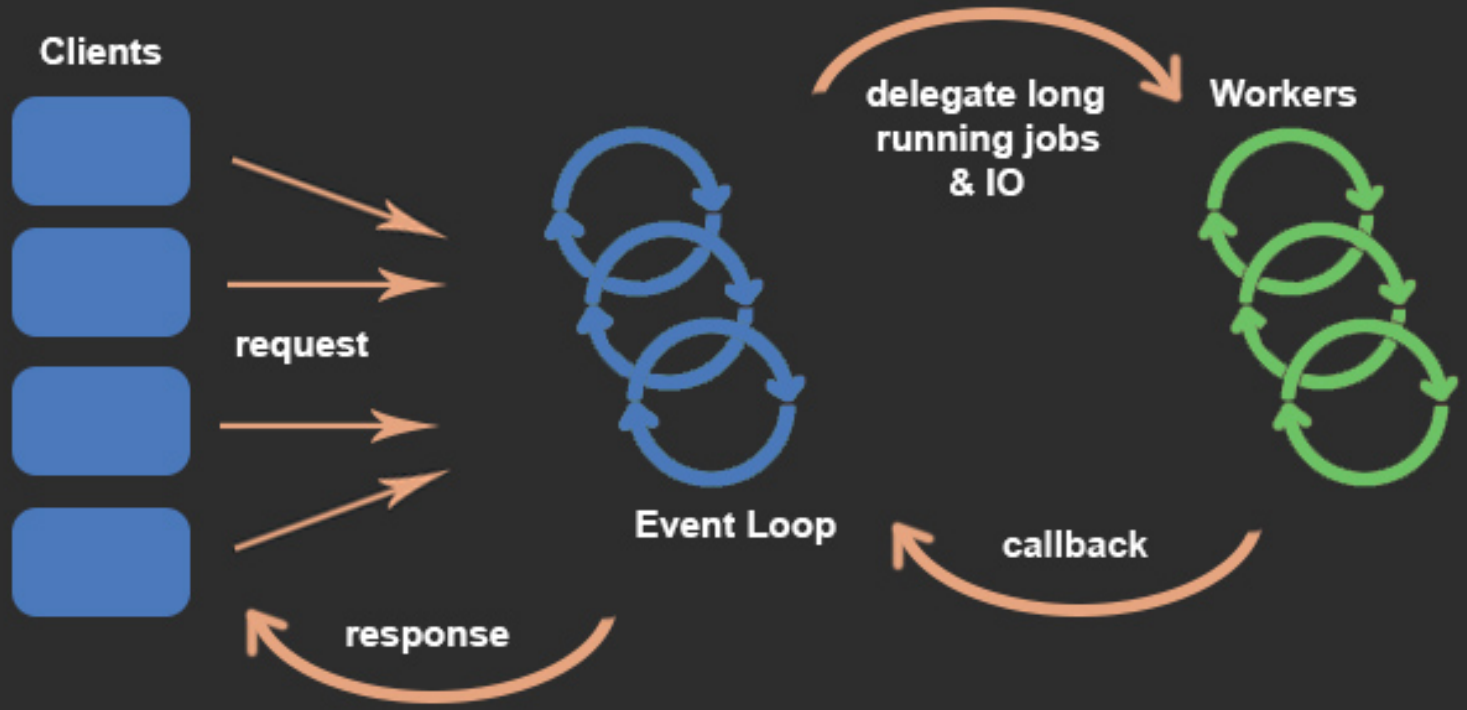
Шаблон Реактор



wikipedia.org/wiki/Reactor_pattern

VERT.X

Мульти-Реактор



wikipedia.org/wiki/Reactor_pattern

VERT.X

Принципы построения приложений

MY.GAMES

VERT.X

Принципы построения приложений

- Hollywood Principle: don’t call us, we’ll call you
- Don’t block me!

MY.GAMES

VERT.X

Hollywood Principle — Event-driven

- таймер
- запрос от клиента
- чтение с диска
- …

MY.GAMES

VERT.X

Hollywood Principle — HTTP request handler

```
{
  doSomethingBefore();

  server.requestHandler(request -> {
    request.response().end("hello world!");
  });

  doSomethingAfter();
}
```

MY.GAMES

VERT.X

Hollywood Principle — HTTP request handler

```
{
  doSomethingBefore();

  server.requestHandler(request -> {
    request.response().end("hello world!");
  });

  doSomethingAfter();
}
```

MY.GAMES

VERT.X

Hollywood Principle — HTTP request handler

```
{
  doSomethingBefore();

  server.requestHandler(request -> {
    request.response().end("hello world!");
  });

  doSomethingAfter();
}
```

MY.GAMES

VERT.X

Принципы построения приложений

- Hollywood Principle: don’t call us, we’ll call you
- Don’t block me!

MY.GAMES

VERT.X

Don’t block me!

- Асинхронный API
- Эффективность и масштабирование

VERT.X

Распределенные данные

MY.GAMES

VERT.X

Распределенные данные

- key-value

MY.GAMES

VERT.X

Распределенные данные

- key-value

```
vertx.sharedData().getAsyncMap("mymap", res -> {
 if (res.succeeded()) {
  var map = res.result();
  map.put("foo", "bar", resPut -> {
   if (resPut.succeeded()) {
    System.out.println("The value is stored cluster wide");
   }
  });
}});
```

MY.GAMES

VERT.X

Распределенные данные

- key-value

```
vertx.sharedData().getAsyncMap("mymap", res -> {
 if (res.succeeded()) {
  var map = res.result();
  map.put("foo", "bar", resPut -> {
   if (resPut.succeeded()) {
    System.out.println("The value is stored cluster wide");
   }
  });
}});
```

MY.GAMES

VERT.X

Распределенные данные

- key-value

```
vertx.sharedData().getAsyncMap("mymap", res -> {
 if (res.succeeded()) {
  var map = res.result();
  map.put("foo", "bar", resPut -> {
   if (resPut.succeeded()) {
    System.out.println("The value is stored cluster wide");
   }
  });
}});
```

MY.GAMES

VERT.X

Распределенные данные

- key-value

MY.GAMES

VERT.X

Распределенные данные

- key-value
- мониторы

MY.GAMES

VERT.X

Распределенные данные

- key-value
- мониторы
- счетчики

ТЕХНОЛОГИИ

Vert.x

Отличный выбор для эффективной работы большого количества экземпляров нагрузочных сценариев.

MY.GAMES

ТЕХНОЛОГИИ

Проблема: Callback Hell

```
sharedData.getCounter("mycounter", result1 -> {
  if (result1.succeeded()) {
    result1.result().addAndGet(stats.longValue(), result2 -> {
      if (result2.succeeded()) {
        System.out.println("Incremented to " + result2.result());
      }
    }
  }
});
```

MY.GAMES

ТЕХНОЛОГИИ

Проблема: Callback Hell

```
sharedData.getCounter("mycounter", result1 -> {
  if (result1.succeeded()) {
    result1.result().addAndGet(stats.longValue(), result2 -> {
      if (result2.succeeded()) {
        System.out.println("Incremented to " + result2.result());
      }
    }
  }
});
```

MY.GAMES

ТЕХНОЛОГИИ

Проблема: Callback Hell

Quasar?

docs.paralleluniverse.co/quasar/

MY.GAMES

ТЕХНОЛОГИИ

Проблема: Callback Hell

~~Quasar~~

Project Loom

openjdk.java.net/projects/loom/

MY.GAMES

ТЕХНОЛОГИИ

Проблема: Callback Hell

~~Quasar~~

~~Project Loom~~

Kotlin Coroutines + Suspend Functions

MY.GAMES

ТЕХНОЛОГИИ

Проблема: Callback Hell

```
//код ~~курильщика~~ Vert.x + Java

sharedData.getCounter("mycounter", result1 -> {
  if (result1.succeeded()) {
    result1.result().addAndGet(stats.longValue(), result2 -> {
      if (result2.succeeded()) {
        System.out.println("Incremented to " + result2.result());
      }
    }
  }
});
```

MY.GAMES

ТЕХНОЛОГИИ

Решение: Suspend Functions

```
//код ~~здорового человека~~ Vert.x + Kotlin

val result1 = vertx.sharedData().getCounterAwait("mycounter")
val result2 = result1.addAndGetAwait(stats.toLong())
println("Incremented to $result2")
```

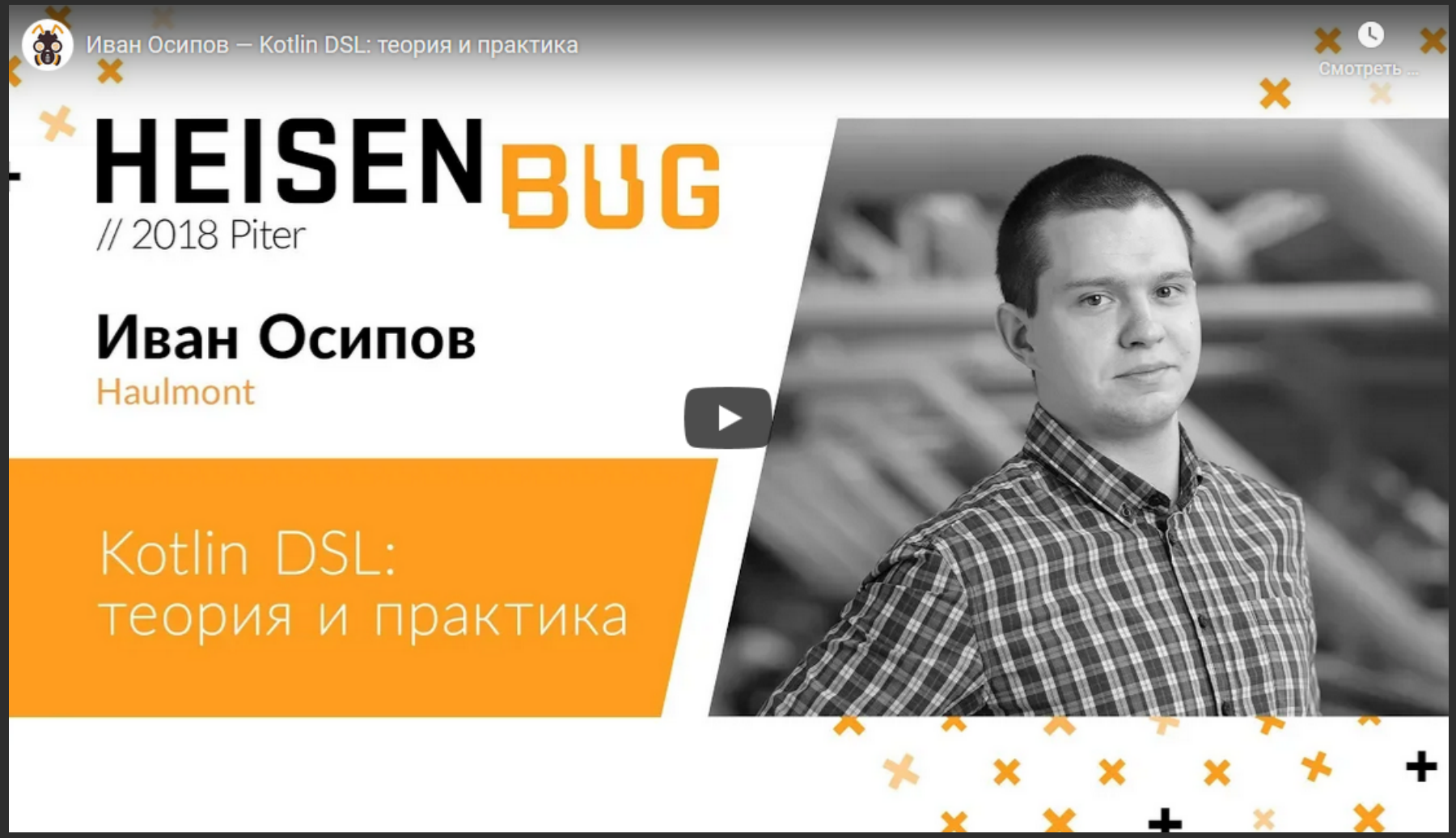
MY.GAMES

ТЕХНОЛОГИИ

DSL

- Лямбды с приемниками
- Функции-расширения
- Псевдонимы типов
- Использование лямбд вне скобок метода

ТЕХНОЛОГИИ



2018.heisenbug-piter.ru

ТЕХНОЛОГИИ

Kotlin

ТЕХНОЛОГИИ

Kotlin

- Coroutines + Vert.x

MY.GAMES

ТЕХНОЛОГИИ

Kotlin

- Coroutines + Vert.x
- Suspend Functions

MY.GAMES

ТЕХНОЛОГИИ

Kotlin

- Coroutines + Vert.x
- Suspend Functions
- DSL

MY.GAMES

ТЕХНОЛОГИИ

Kotlin

- Coroutines + Vert.x
- Suspend Functions
- DSL
- Hype!

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

TRUE STORY

TRUE STORY

Prerequisites

TRUE STORY

Prerequisites

- Конфигурация

TRUE STORY

Prerequisites

- Конфигурация
- Авторизация

MY.GAMES

TRUE STORY

Prerequisites

- Конфигурация
- Авторизация
- Протокол

MY.GAMES

TRUE STORY

Prerequisites

- Конфигурация
- Авторизация
- Протокол
- Модель – данные игрока

MY.GAMES

TRUE STORY

Профиль нагрузки

- Метрики запросов
- Длительность сессии
- Боевая БД

MY.GAMES

TRUE STORY

Talk is cheap. Show me the code.

Linus Torvalds

MY.GAMES

TRUE STORY

```
val session = user.login()
with(session) {
    repeat(10) { request { BuyRequest(CRYSTALS, 10) } }

    if (!model.quests.contains(SOME_QUEST)) {
        request { TakeQuestsRequest(SOME_QUEST) }
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, 1000) }
    request { CompleteQuestRequest(SOME_QUEST) }

    while (model.currencies[GOLD].value < 100) {
        request { StartMissionRequest(1, 1, SPARROW) }
        val flight = waitForEvent<StartMissionEvent>().flight_id
        request { CompleteMissionRequest(flight) }
    }
}
```

MY.GAMES

TRUE STORY

```
val session = user.login()
with(session) {
    repeat(10) { request { BuyRequest(CRYSTALS, 10) } }

    if (!model.quests.contains(SOME_QUEST)) {
        request { TakeQuestsRequest(SOME_QUEST) }
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, 1000) }
    request { CompleteQuestRequest(SOME_QUEST) }

    while (model.currencies[GOLD].value < 100) {
        request { StartMissionRequest(1, 1, SPARROW) }
        val flight = waitForEvent<StartMissionEvent>().flight_id
        request { CompleteMissionRequest(flight) }
    }
}
```

MY.GAMES

TRUE STORY

```
val session = user.login()
with(session) {
    repeat(10) { request { BuyRequest(CRYSTALS, 10) } }

    if (!model.quests.contains(SOME_QUEST)) {
        request { TakeQuestsRequest(SOME_QUEST) }
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, 1000) }
    request { CompleteQuestRequest(SOME_QUEST) }

    while (model.currencies[GOLD].value < 100) {
        request { StartMissionRequest(1, 1, SPARROW) }
        val flight = waitForEvent<StartMissionEvent>().flight_id
        request { CompleteMissionRequest(flight) }
    }
}
```

MY.GAMES

TRUE STORY

```
val session = user.login()
with(session) {
    repeat(10) { request { BuyRequest(CRYSTALS, 10) } }

    if (!model.quests.contains(SOME_QUEST)) {
        request { TakeQuestsRequest(SOME_QUEST) }
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, 1000) }
    request { CompleteQuestRequest(SOME_QUEST) }

    while (model.currencies[GOLD].value < 100) {
        request { StartMissionRequest(1, 1, SPARROW) }
        val flight = waitForEvent<StartMissionEvent>().flight_id
        request { CompleteMissionRequest(flight) }
    }
}
```

TRUE STORY

```
val session = user.login()
with(session) {
    repeat(10) { request { BuyRequest(CRYSTALS, 10) } }

    if (!model.quests.contains(SOME_QUEST)) {
        request { TakeQuestsRequest(SOME_QUEST) }
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, 1000) }
    request { CompleteQuestRequest(SOME_QUEST) }

    while (model.currencies[GOLD].value < 100) {
        request { StartMissionRequest(1, 1, SPARROW) }
        val flight = waitForEvent<StartMissionEvent>().flight_id
        request { CompleteMissionRequest(flight) }
    }
}
```

TRUE STORY

```
scenario {
    story {
        type = "TrueStory“
        config { iterations = 100500 }
    }
    authentication {
        strategy {
            type = "existing“
            range {
                offset = 11000001
                limit = 3
            }
        }
    }
    rps = 1.0f
}
```

TRUE STORY

```
scenario {
    story {
        type = "TrueStory“
        config { iterations = 100500 }
    }
    authentication {
        strategy {
            type = "existing“
            range {
                offset = 11000001
                limit = 3
            }
        }
    }
    rps = 1.0f
}
```

MY.GAMES

TRUE STORY

```
scenario {
    story {
        type = "TrueStory“
        config { iterations = 100500 }
    }
    authentication {
        strategy {
            type = "existing“
            range {
                offset = 11000001
                limit = 3
            }
        }
    }
    rps = 1.0f
}
```

MY.GAMES

TRUE STORY

```
scenario {
    story {
        type = "TrueStory“
        config { iterations = 100500 }
    }
    authentication {
        strategy {
            type = "existing“
            range {
                offset = 11000001
                limit = 3
            }
        }
    }
    rps = 1.0f
}
```

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})
```

MY.GAMES

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})
```

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})
```

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})
```

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})

enum class Action(
    val weight: Float,
    val execute: suspend Session<Model>.() -> Unit
) {
    Buy(3, { request { BuyRequest(CRYSTALS, 10) } }),
    PvE_Mission(7, { /* ... */ }),
    TakeQuest(1, { /* ... */ }),
    // and so on...
}
```

MY.GAMES

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})

enum class Action(
    val weight: Float,
    val execute: suspend Session<Model>.() -> Unit
) {
    Buy(3, { request { BuyRequest(CRYSTALS, 10) } }),
    PvE_Mission(7, { /* ... */ }),
    TakeQuest(1, { /* ... */ }),
    // and so on...
}
```

MY.GAMES

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})

enum class Action(
    val weight: Float,
    val execute: suspend Session<Model>.() -> Unit
) {
    Buy(3, { request { BuyRequest(CRYSTALS, 10) } }),
    PvE_Mission(7, { /* ... */ }),
    TakeQuest(1, { /* ... */ }),
    // and so on...

    companion object { operator fun invoke() = random.pick(values()) { a -> a.weight } }
}
```

MY.GAMES

TRUE STORY

```
class TrueStory(config: Configuration) : Story<Model>({ user ->
    val session = user.login()
    repeat(config.value("iterations")) {
        Action().execute(session)
    }
})

enum class Action(
    val weight: Float,
    val execute: suspend Session<Model>.() -> Unit
) {
    Buy(3, { request { BuyRequest(CRYSTALS, 10) } }),
    PvE_Mission(7, { /* ... */ }),
    TakeQuest(1, { /* ... */ }),
    // and so on...

    companion object { operator fun invoke() = random.pick(values()) { a -> a.weight } }
}
```

ЗАПУСК СЦЕНАРИЕВ

ЗАПУСК СЦЕНАРИЕВ

- Локально

ЗАПУСК СЦЕНАРИЕВ

- Локально
- Jenkins

MY.GAMES

ЗАПУСК СЦЕНАРИЕВ

- Локально
- Jenkins
- Docker

ЗАПУСК СЦЕНАРИЕВ

- Локально
- Jenkins
- Docker
- Ansible

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

MY.GAMES

ПЛАН

- Какие проекты мы тестируем
- Какие требования предъявляем
- Какие решения используем и почему именно их
- Как пишем и запускаем нагрузочные сценарии
- Как анализируем результаты

АНАЛИЗ РЕЗУЛЬТАТОВ

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX

MY.GAMES

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX
- Grafana

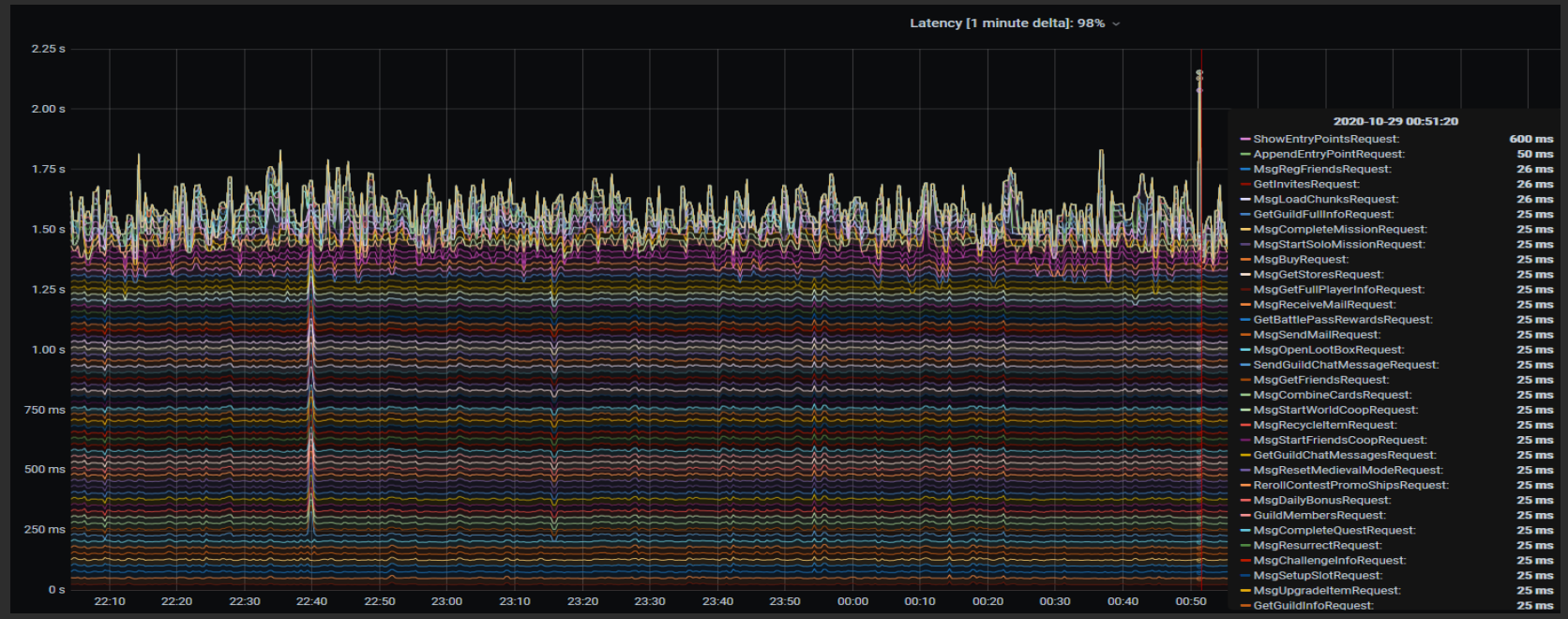
АНАЛИЗ РЕЗУЛЬТАТОВ

Grafana
ccu

АНАЛИЗ РЕЗУЛЬТАТОВ

Grafana

Latency



MY.GAMES

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX
- Grafana

MY.GAMES

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX
- Grafana
- и… свой инструмент для сравнения

MY.GAMES

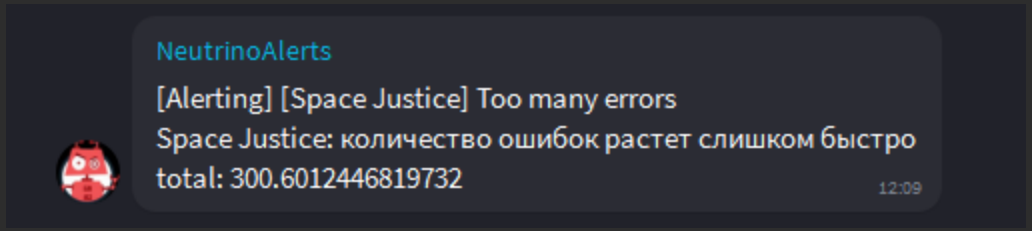
АНАЛИЗ РЕЗУЛЬТАТОВ

Правила сравнения

- Сравниваем N и (N-1) запуски
- Допустимый процент отклонения (x3)
- Допустимое значение отклонения (x3)
- Допустимый лимит (количество ошибок)

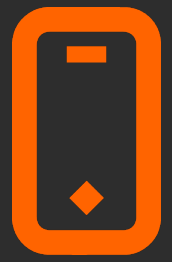
АНАЛИЗ РЕЗУЛЬТАТОВ

Алерты



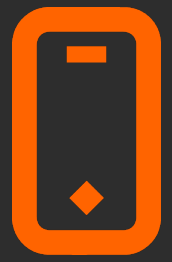
ШАГ ЗА ШАГОМ

ШАГ ЗА ШАГОМ



Запуск: Jenkins, Docker, Ansible

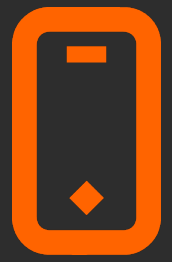
ШАГ ЗА ШАГОМ



Запуск: Jenkins, Docker, Ansible

Метрики: Prometheus

ШАГ ЗА ШАГОМ



Запуск: Jenkins, Docker, Ansible

Метрики: Prometheus

Анализ: Grafana, сравнение, алерты

MY.GAMES

ЧТО ПОЛУЧИЛОСЬ

- Быстрая разработка благодаря использованию привычного стека
- Недорогая поддержка за счет максимальной автоматизации
- Профиль нагрузки максимально близок к боевому
- Сценарии пишут QA
- Можно тестировать не только на нагрузку

ЧТО ПОЛУЧИЛОСЬ

ЧТО ХОТИМ УЛУЧШИТЬ

- Инструмент сравнения
- DSL — нет предела совершенству

MY.GAMES

ИТОГО

- Глаза боятся, руки делают
- QA + Vert.x + Kotlin Coroutines

=

- Простые, мощные и эффективные сценарии

АНТОН ПОЦЮС

Бас-гитарист и Java-программист в одном лице

potsyus@corp.mail.ru

СПАСИБО!