MY.GAMES

SQA DAYS#28

НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ ИГРОВЫХ СЕРВЕРОВ

Дмитрий Демин
QA-специалист, IT Territory, VRN

MY.GAMES SQA DAYS#28

Наша студия

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

MY.GAMES SQA DAYS#28

IT TERRITORY VRN

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

ПЛАН

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

MY.GAMES SQA DAYS#28

Мобильные клиенты наших приложений, которые общаются с игровым сервером.

Unity, C#

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

Мобильные клиенты наших приложений, которые общаются с игровым сервером.

Unity, C#

Серверная часть наших игр. Выступает в роли «валидатора» действий игроков и в качестве связующего звена между игроками.

Java

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

MY.GAMES SQA DAYS#28

Мобильные клиенты наших приложений, которые общаются с игровым сервером.

Unity, C#

Серверная часть наших игр. Выступает в роли «валидатора» действий игроков и в качестве связующего звена между игроками.

Java

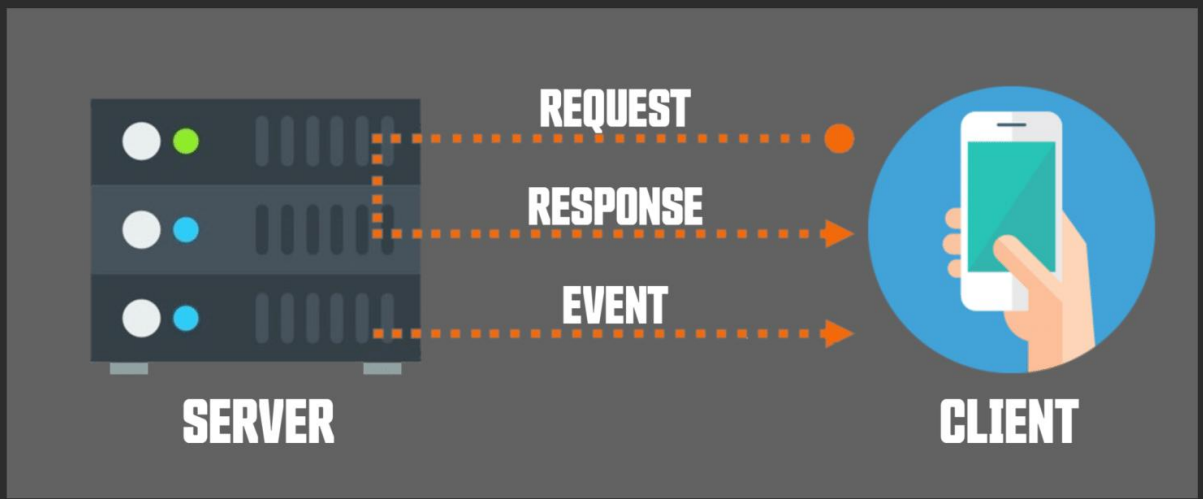
Регулярная поддержка проектов с выходом новых версий клиента и сервера раз в 1 - 1.5 месяца.

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

- TCP + синхронный прикладной протокол

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

- TCP + синхронный прикладной протокол



КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

- TCP + синхронный прикладной протокол
- 10+k CCU

MY.GAMES SQA DAYS#28

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

- TCP + синхронный прикладной протокол
- 10+k CCU
- Тысячи запросов в секунду (RPS)

MY.GAMES SQA DAYS#28

КАКИЕ ПРОЕКТЫ МЫ ТЕСТИРУЕМ

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

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

MY.GAMES

SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

- Сложность в обработке ответов и эвентов

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

- Сложность в обработке ответов и эвентов
- Необходимость постоянной поддержки изменений протокола

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

MY.GAMES SQA DAYS#28

ПЕРВЫЕ ПОПЫТКИ

Сценарий на Golang

- Сложность в обработке ответов и эвентов
- Необходимость постоянной поддержки изменений протокола
- Сложность в описании условий и циклов

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

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими

MY.GAMES SQA DAYS#28

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Простота написания сценариев

MY.GAMES SQA DAYS#28

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Простота написания сценариев
- Использовние существующего кода протокола

MY.GAMES SQA DAYS#28

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Простота написания сценариев
- Использовние существующего кода протокола
- С использованием боевго стека

MY.GAMES SQA DAYS#28

ПОСТАНОВКА ЗАДАЧИ

- Нагрузочные сценарии должны быть гибкими
- Простота написания сценариев
- Использовние существующего кода протокола
- С использованием боевго стека
- Возможность запуска 10k+ ботов

MY.GAMES SQA DAYS#28

СУЩЕСТВУЮЩИЕ РЕШЕНИЯ

Остальные

- JMeter: высокий порог вхождения, повышенное потребление ресурсов
- Gatling: нет TCP из коробки
- Loadrunner, WebLoad: всего 50 «пользователей» в бесплатной версии
- Grinder: высокий порог вхождения, маленькое комьюнити
- LoadView, StressStimulus, LoadNinja: только платная версия

MY.GAMES

SQA DAYS#28

ТЕХНОЛОГИИ

- Язык программирования Java

MY.GAMES SQA DAYS#28

ТЕХНОЛОГИИ

- Язык программирования Java

1) Позволит гибко описывать сценарии
2) Прямое использование кода протокола

ТЕХНОЛОГИИ

- Язык программирования Java
- Фреймворк Vert.x

MY.GAMES SQA DAYS#28

ТЕХНОЛОГИИ

- Язык программирования Java
- Фреймворк Vert.x

1) Позволяет создавать распределенные приложения, работающие на JVM.

2) Можно запускать большое количество экземпляров нагрузочных сценариев.

3) Потребляет мало ресурсов.

MY.GAMES SQA DAYS#28

CALLBACK HELL

```
request(TakeQuestsRequest(SOME_QUEST), res1 -> {
 if (res1.succeeded()) {
  Integer counter = res1.response.counter;
  request(CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter), res2 -> {
   if (res2.succeeded()) {
    request(CompleteQuestRequest(SOME_QUEST), res3 -> {
     if (res3.succeeded()) {
      System.out.println("Quest [" + SOME_QUEST + "] completed!");
     }
    });
   }
  });
 }
});
```

MY.GAMES SQA DAYS#28

CALLBACK HELL

```
request(TakeQuestsRequest(SOME_QUEST), res1 -> {
 if (res1.succeeded()) {
  Integer counter = res1.response.counter;
  request(CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter), res2 -> {
   if (res2.succeeded()) {
    request(CompleteQuestRequest(SOME_QUEST), res3 -> {
     if (res3.succeeded()) {
      System.out.println("Quest [" + SOME_QUEST + "] completed!");
     }
    });
   }
  });
 }
});
```

MY.GAMES SQA DAYS#28

CALLBACK HELL

```
request(TakeQuestsRequest(SOME_QUEST), res1 -> {
 if (res1.succeeded()) {
  Integer counter = res1.response.counter;
  request(CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter), res2 -> {
   if (res2.succeeded()) {
    request(CompleteQuestRequest(SOME_QUEST), res3 -> {
     if (res3.succeeded()) {
      System.out.println("Quest [" + SOME_QUEST + "] completed!");
     }
    });
   }
  });
 }
});
```

MY.GAMES SQA DAYS#28

CALLBACK HELL

```
request(TakeQuestsRequest(SOME_QUEST), res1 -> {
 if (res1.succeeded()) {
  Integer counter = res1.response.counter;
  request(CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter), res2 -> {
   if (res2.succeeded()) {
    request(CompleteQuestRequest(SOME_QUEST), res3 -> {
     if (res3.succeeded()) {
      System.out.println("Quest [" + SOME_QUEST + "] completed!");
     }
    });
   }
  });
 }
});
```

MY.GAMES SQA DAYS#28

CALLBACK HELL

```
request(TakeQuestsRequest(SOME_QUEST), res1 -> {
 if (res1.succeeded()) {
  Integer counter = res1.response.counter;
  request(CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter), res2 -> {
   if (res2.succeeded()) {
    request(CompleteQuestRequest(SOME_QUEST), res3 -> {
     if (res3.succeeded()) {
      System.out.println("Quest [" + SOME_QUEST + "] completed!");
     }
    });
   }
  });
 }
});
```

KOTLIN

- Совместимость с Java

MY.GAMES

SQA DAYS#28

KOTLIN

- Совместимость с Java
- Coroutines

MY.GAMES

SQA DAYS#28

KOTLIN

- Совместимость с Java
- Coroutines
- Suspend functions

MY.GAMES

SQA DAYS#28

KOTLIN

- Совместимость с Java
- Coroutines
- Suspend functions
- Поддержка использования Kotlin в Vert.x

MY.GAMES SQA DAYS#28

ИЗБАВЛЯЕМСЯ ОТ CALLBACK HELL

Старый код на Java:

```
request(TakeQuestsRequest(SOME_QUEST), res1 -> {
 if (res1.succeeded()) {
  Integer counter = res1.response.counter;
  request(CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter), res2 -> {
   if (res2.succeeded()) {
    request(CompleteQuestRequest(SOME_QUEST), res3 -> {
     if (res3.succeeded()) {
      System.out.println("Quest [" + SOME_QUEST + "] completed!");
     }
    });
   }
  });
 }
});
```

MY.GAMES SQA DAYS#28

ИЗБАВЛЯЕМСЯ ОТ CALLBACK HELL

Новый код с Kotlin:

```
val counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
request { CompleteQuestRequest(SOME_QUEST) }
```

MY.GAMES SQA DAYS#28

ИЗБАВЛЯЕМСЯ ОТ CALLBACK HELL

Новый код с Kotlin:

```
val counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
request { CompleteQuestRequest(SOME_QUEST) }
```

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

MY.GAMES SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР СЦЕНАРИЯ

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

    var counter = 10
    if (!model.quests.contains(SOME_QUEST)) {
        counter = request { TakeQuestsRequest(SOME_QUEST) }.response.counter
    }
    request { CheatRequest(Cheat.SetQuestCounter, SOME_QUEST, counter) }
    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 SQA DAYS#28

ПРИМЕР КОНФИГУРАЦИИ

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

MY.GAMES SQA DAYS#28

ПРИМЕР КОНФИГУРАЦИИ

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

MY.GAMES SQA DAYS#28

ПРИМЕР КОНФИГУРАЦИИ

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

MY.GAMES SQA DAYS#28

ПРИМЕР КОНФИГУРАЦИИ

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

MY.GAMES SQA DAYS#28

ПРИМЕР КОНФИГУРАЦИИ

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

TRUE STORY

- Сценарий, полностью дублирующий действия игроков

MY.GAMES

SQA DAYS#28

TRUE STORY

- Сценарий, полностью дублирующий действия игроков
- Использование копии боевой БД

TRUE STORY

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

MY.GAMES SQA DAYS#28

TRUE STORY

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

MY.GAMES SQA DAYS#28

TRUE STORY

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

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

MY.GAMES SQA DAYS#28

TRUE STORY

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

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

MY.GAMES SQA DAYS#28

TRUE STORY

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

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

MY.GAMES SQA DAYS#28

TRUE STORY

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

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

MY.GAMES SQA DAYS#28

TRUE STORY

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

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

ЗАПУСК СЦЕНАРИЕВ

- Локально

MY.GAMES SQA DAYS#28

ЗАПУСК СЦЕНАРИЕВ

- Локально
- Jenkins

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

MY.GAMES SQA DAYS#28

ПЛАН

- Какие проекты мы тестируем
- Какой инструмент мы выбрали для нагрузочного тестирования
- Как пишем и запускаем сценарии (ботов)
- Как анализируем результаты
- Что мы получили в итоге

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus

MY.GAMES SQA DAYS#28

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX

MY.GAMES SQA DAYS#28

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX
- Grafana

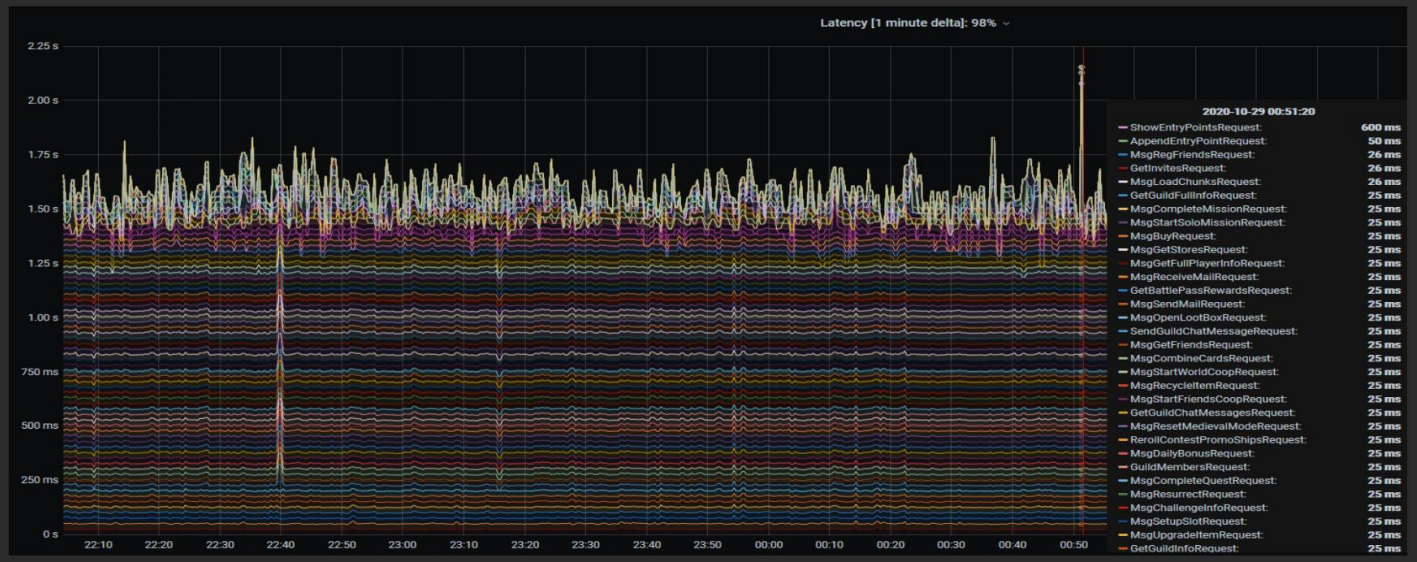
АНАЛИЗ РЕЗУЛЬТАТОВ

Grafana
ccu

MY.GAMES SQA DAYS#28

АНАЛИЗ РЕЗУЛЬТАТОВ

Grafana
Latency



MY.GAMES SQA DAYS#28

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX
- Grafana

MY.GAMES SQA DAYS#28

АНАЛИЗ РЕЗУЛЬТАТОВ

- Prometheus
- JMX
- Grafana
- Свой инструмент автоматического сравнения

АНАЛИЗ РЕЗУЛЬТАТОВ

Правила сравнения

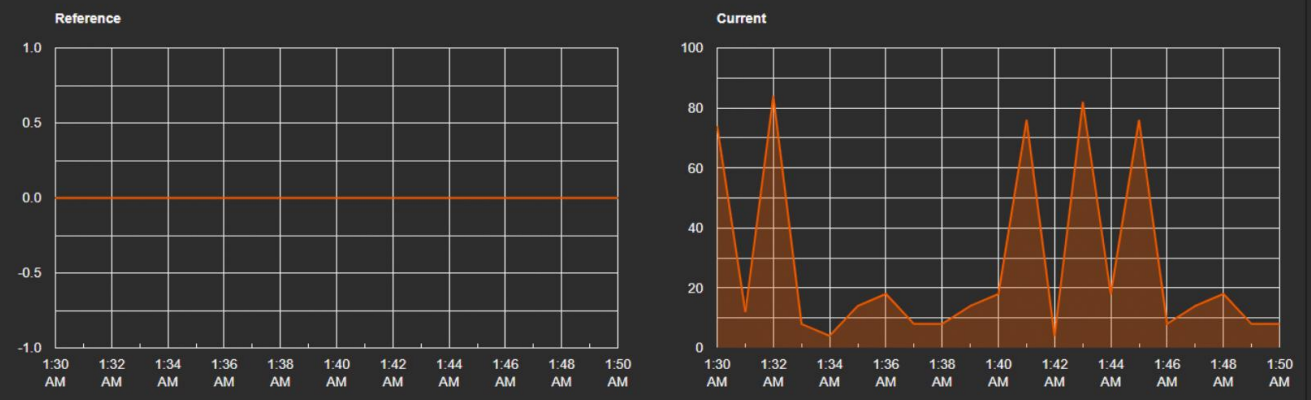
- Сравниваем первый (эталонный) и последний запуски
- Или сравниваем N и (N-1) запуски
- Допустимый процент отклонения
- Допустимое значение отклонения
- Допустимый лимит (количество ошибок)

MY.GAMES SQA DAYS#28

АНАЛИЗ РЕЗУЛЬТАТОВ

Формирование отчета

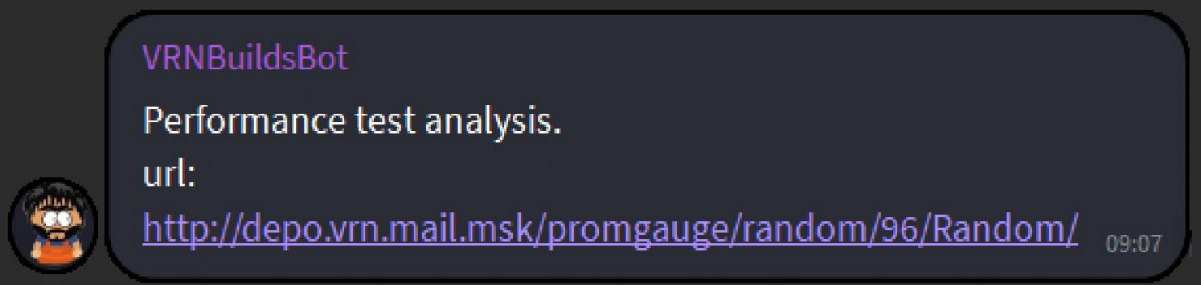
Errors



- 2020-12-16 22:31:00.000 AlarmConfig$ErrorThreshold. Ref value: 0.0, current value: 12.0
- 2020-12-16 22:32:00.000 AlarmConfig$ErrorThreshold. Ref value: 0.0, current value: 84.0
- 2020-12-16 22:33:00.000 AlarmConfig$ErrorThreshold. Ref value: 0.0, current value: 8.0

АНАЛИЗ РЕЗУЛЬТАТОВ

Alert с выявленными проблемами



ПОСЛЕДОВАТЕЛЬНОСТЬ ТЕСТИРОВАНИЯ

ПОСЛЕДОВАТЕЛЬНОСТЬ ТЕСТИРОВАНИЯ

MY.GAMES SQA DAYS#28

Запуск с помощью Jenkins

ПОСЛЕДОВАТЕЛЬНОСТЬ ТЕСТИРОВАНИЯ

MY.GAMES SQA DAYS#28

Запуск с помощью Jenkins

Сбор метрик с помощью Prometheus

ПОСЛЕДОВАТЕЛЬНОСТЬ ТЕСТИРОВАНИЯ

MY.GAMES SQA DAYS#28

Запуск с помощью Jenkins

Сбор метрик с помощью Prometheus

Получение результата с помощью Grafana, механизма сравнения и алертов

MY.GAMES SQA DAYS#28

ЧТО ПОЛУЧИЛОСЬ

- Быстрая разработка сценариев с привычным стеком

MY.GAMES SQA DAYS#28

ЧТО ПОЛУЧИЛОСЬ

- Быстрая разработка сценариев с привычным стеком
- Недорогая поддержка за счет максимальной автоматизации

MY.GAMES SQA DAYS#28

ЧТО ПОЛУЧИЛОСЬ

- Быстрая разработка сценариев с привычным стеком
- Недорогая поддержка за счет максимальной автоматизации
- Профиль нагрузки максимально близок к боевому

MY.GAMES SQA DAYS#28

ЧТО ПОЛУЧИЛОСЬ

- Быстрая разработка сценариев с привычным стеком
- Недорогая поддержка за счет максимальной автоматизации
- Профиль нагрузки максимально близок к боевому
- Можно тестировать не только на нагрузку

ПЛАНЫ НА БУДУЩЕЕ

- Улучшить механизм сравнения результатов

MY.GAMES SQA DAYS#28

ПЛАНЫ НА БУДУЩЕЕ

- Улучшить механизм сравнения результатов
- Проводить нагрузочное тестирование на боевом стенде

MY.GAMES SQA DAYS#28

ИТОГО

- Нагрузочное тестирование своими руками
- QA + Vert.x + Kotlin Coroutines
- Простые, мощные и эффективные сценарии

СПАСИБО!

ДМИТРИЙ ДЕМИН

QA-специалист, IT Territory, VRN

dmitriy.demin@corp.mail.ru