

# Быстрое прототипирование бэкенда игры с геолокацией на OpenResty, Redis и Docker

Александр Гладыш  
CTO, LogicEditor



Профессиональная конференция  
разработчиков высоконагруженных  
систем

# Содержание

1. Кейс
2. Задача
3. Философия
4. План разработки
5. Docker
6. Вопросы?

# Обо мне

- В разработке ПО с 2002-го,
- большую часть этого времени — в геймдеве (разработка, проектирование, управление),
- вне геймдева — нагруженные интернет-решения, enterprise ПО и др.
- Организатор [meetup.com/Lua-in-Moscow](https://meetup.com/Lua-in-Moscow)

# Мобильные игры с геолокацией



# Цели прототипирования в общем

- Проверить ряд подходов к построению игрового процесса,
- выработать новые идеи,
- найти, в чём фан, а в чём — нет.

# Цели технологической части прототипирования

- С минимальными затратами получить код,
- дающий возможность быстро итерироваться по вариантам геймплея.
- Прояснить на практике технические ограничения жанра.

# Результаты

За два календарных месяца (менее 100 человеко-часов) разработан прототип серверной части игры с геолокацией и рудиментарный клиент для неё.

Ведётся быстрое итерирование по вариантам построения игрового процесса и дальнейшая разработка технологической части проекта.

# О чём этот доклад?

- Доклад — технологической части проекта,
- не о геймдизайне
- и не о монетизации.



# Зачем этот доклад?

Делать игры с геолокацией сейчас проще чем когда-либо.

Я покажу как начать.

# Задача

- Максимально быстро написать сервер для быстрого прототипирования.
- Параллельно с сервером реализовать минимальный клиент.
- Проверять гипотезы уже во время написания, если возможно.
- Выявить основные технические ограничения на проект.

# Стадии разработки

- Препродакшен
- Продакшен
- Поддержка

# Стадии разработки: Препродакшен

- Препродакшен
  - Поиск геймплея, эскизы, выяснение ограничений.
  - Более глубокая проработка удачных вариантов геймплея.
  - Подготовка к продакшену выбранного варианта.

# Продакшен: приоритеты разработки

- Восприятие проекта
- Устойчивость ко взлому
- Масштабируемость
- Производительность
- Стабильность
- Гибкость
- Скорость итерирования
- Простота

# Препродакшен: время на вес золота

## • Скорость итерирования

- Гибкость
- Простота
- Восприятие проекта
- Стабильность
- Производительность
- Масштабируемость
- Устойчивость ко взлому

# Фокус на скорость и лёгкость разработки

- Механизмы, а не решения.
- Лучше быстро чем правильно.
- Много коротких итераций.

# Лучше быстро чем правильно

- Чем меньше кода тем лучше.
- Плохой код лучше сложного.
- Хаки в механизмах допустимы, хаки в решениях — нет.
- Рефакторить нужно только то, что болит.



# Основные этапы разработки

- Выбор геймплея пилотного прототипа.
- Выбор технологического стека.
- Реализация минимального пилотного прототипа.
- Итерирование с гейм-дизайнерами.

# Выбор геймплея пилотного прототипа: Задачи

- Максимально простой
- но играбельный
- повод реализовать все базовые игровые сущности и механизмы прототипа.

# Геймплей первого прототипа

- Игрок, перемещаясь по карте, ищет расставленных на ней мобов;
- найдя моба может попробовать его поймать с заданной вероятностью успеха;
- успешная поимка моба увеличивает счётчик в характеристиках игрока;
- пойманный моб исчезает с карты;
- через некоторое время моб респавнится в том же месте;
- администраторы могут добавлять новых мобов на карту.

# Критерии выбора технологического стека

- Что-то знакомое, на чём можно написать быстро,
- или что-то интересное и зажигающее.
- Главное не забыть вовремя выбросить весь код.

# Технологический стек

- Сервер:
  - Redis,
  - OpenResty,
  - Docker.
- Клиент:
  - Одностраничное веб-приложение в браузере,
  - HTML5.

# Почему не что-то готовое?

Not Invented Here Syndrome?

И да и нет.

# Redis

- Надёжное, хорошо зарекомендовавшее себя решение.
- Работа с координатами из коробки:
  - GEOADD key longitude latitude member
  - GEORADIUS key longitude latitude radius m
- Достаточно удобный набор примитивов для хранения игровых объектов.
- Хранимые процедуры на Lua.

# OpenResty

- Дистрибутив nginx с поддержкой Lua, Redis и многим другим из коробки.
- Очень быстро работает, достаточно дружелюбен, хорошо поддерживается.
- Пригоден как для быстрого прототипирования, так и для продакшена.



# Docker

- Воспроизводимая кроссплатформенная среда для разработки.
- Хорошо снимает боль по настройке окружения разработчика.
- Окружение разработчика можно быстро превратить в прототип серверного окружения.
- Требуется обновления до достаточно свежей версии.

# Браузер, HTML5

- На начальном этапе сервер важнее.
- Бои в augmented reality и прочие рюшечки делать не нужно, можно представлять в голове.
- На HTML5 можно быстро написать дешёвый и сердитый клиент.
- Есть ограниченный (но достаточный) доступ к данным геолокации.

# Docker: как установить на Ubuntu

- В Ubuntu, традиционно, старая версия.
- За `wget | sh` больно бьём по рукам.
- `docker` и `docker-machine` устанавливаем из apt-репозитория докера.
- `docker-compose` устанавливаем через `pip install`.
- Про установку на других платформах читаем официальную документацию.

# docker: архитектура одного прототипа (схема с редисом, nginx)

- TODO

# docker: базовые образы redis, openresty/openresty

- TODO

# docker: dockerfiles

- TODO

# docker: nginx config, трюки

- TODO

docker: nginx config, трюки: lua code cache, routers

- TODO



# docker-compose: больше одного прототипа

- TODO

# docker-compose: ymls; довольно про сисадминство

- TODO

код: проектирование механизмов для первого прототипа,  
задачи

- TODO

# игровой объект: обзор

- TODO

# игровой объект: характеристики

- TODO

# игровой объект: прототипы

- TODO

# игровой объект: действия

- TODO

# игровой объект: права на действия

- TODO



# игровой объект: координаты

- TODO

игровой объект: как это ложится на базу, плевать на производительность, плевать на атомарность, плевать на читеров

- TODO

# игровой объект: почему так?

- TODO

# апи: статус

- TODO

# апи: действия

- TODO

# апи: отложенные действия

- TODO

апи: нет админки (пока), есть внутриигровые админские действия

- TODO

# апи: сброс и патч базы; проще, ещё проще!

- TODO



клиент: первый клиент это curl, демонстрация curl

- TODO

# клиент: демонстрация html5

- TODO

клиент: html5 геолокация, https-only (кроме localhost)

- TODO

# клиент: как работает версия на js

- TODO

# клиент, трюки: имена объектов

- TODO

# клиент, трюки: гуглекарта

- TODO

# клиент, трюки: асинхронная перерисовка

- TODO

три вектора развития: геймплей, технологии, фичи;  
приоритеты; не закопаться!

- TODO



# как работать с фидбеком от геймдизайнеров: механизмы, не хаки

- TODO

как работать с фидбеком от геймдизайнеров: приоритеты:  
баги, новые механизмы, далее по убыванию боли

- TODO

# как работать с фидбеком от геймдизайнеров: быстрые итерации, садиться рядом и кодить

- TODO

как работать с фидбеком от геймдизайнеров: админка  
только тогда, когда без неё станет совсем больно

- TODO

# что получилось: описание, трудозатраты, оценка успешности

- TODO

# что дальше: дорога к релизу

- TODO

# проблемы: геолокация шумит

- TODO

# проблемы: нет геолокации в зданиях

- TODO



# проблемы: нет проблем, подстраивайте под них геймплей

- TODO

чего не хватает из механизмов: по-крупному — события,  
много мелочей

Например, счётчика пройденных метров.

- TODO

ошибки: геолокация появилась поздновато, лишний map

- TODO

# ошибки: карта в клиенте появилась поздновато

- TODO

# ошибки: больше выходить на улицу

- TODO

# ошибки: ещё?

- TODO

# Вопросы?

@agladysh

ag@logiceditor.com

meetup.com/Lua-in-Moscow