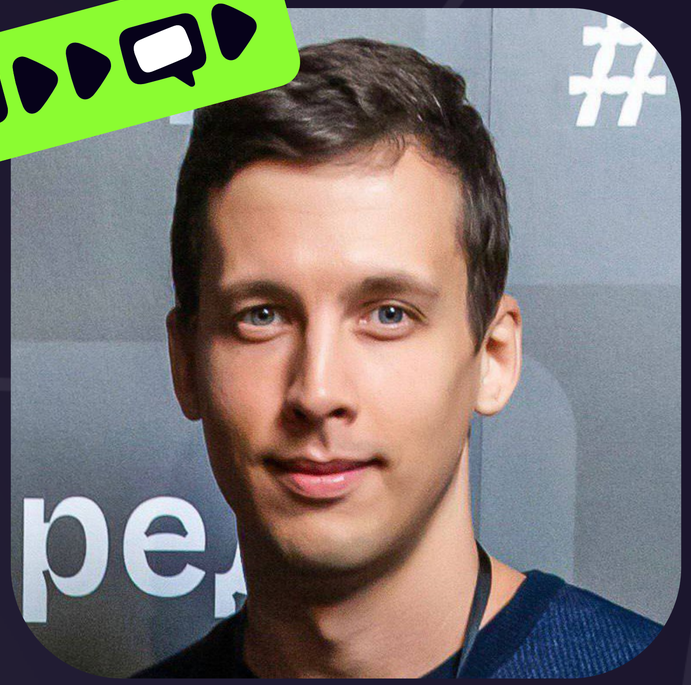
Как сделать из Cocos Creator крепкий орешек:
Строим TS-архитектуру вокруг игры

Дмитрий Третьяков



Дмитрий Третьяков

Solution Architect
IDDQD

@DmitriyTretyakov

Cocos Creator – что за зверь

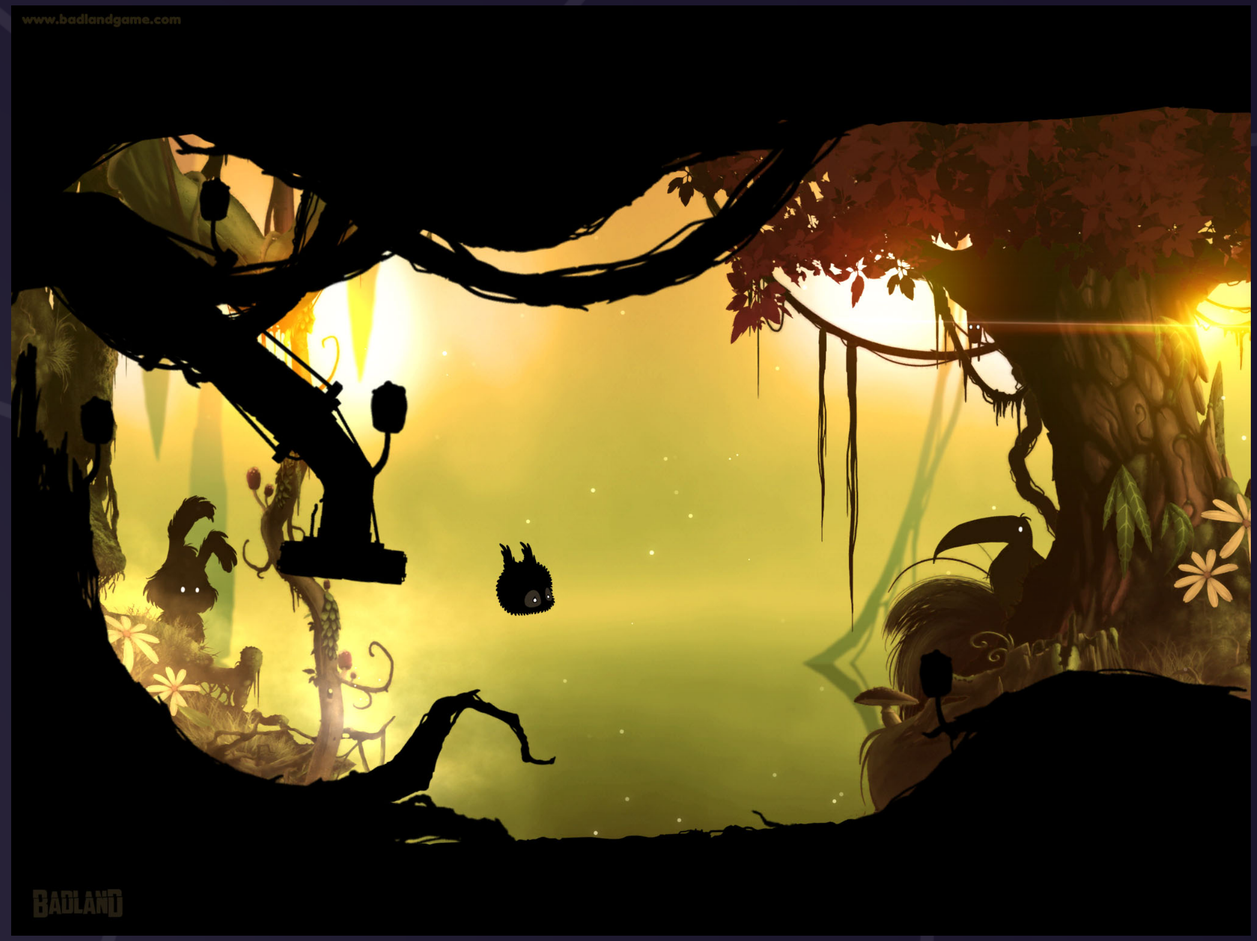
Cocos Creator - это **бесплатный** игровой движок, китайский аналог и конкурент Unity 3D, использующий в качестве языка разработки TypeScript. Это универсальный инструмент разработки, который позволяет разработчикам создавать игры для всех основных платформ, таких как **iOS, Android, Facebook Instant Games, WeChat Mini Games, HTML5 и настольных ПК.**

Популярные игры - где используют

Cocos creator и cocos2d-x используются крупными студиями, такими как Ubisoft, Glu Mobile, Rovio, Crescent Moon Games и другими, самые популярные проекты:

Badland, Shadow Blade, Plague Inc., AFK Arena, Clash of Kings и др.

Популярные игры - Badland



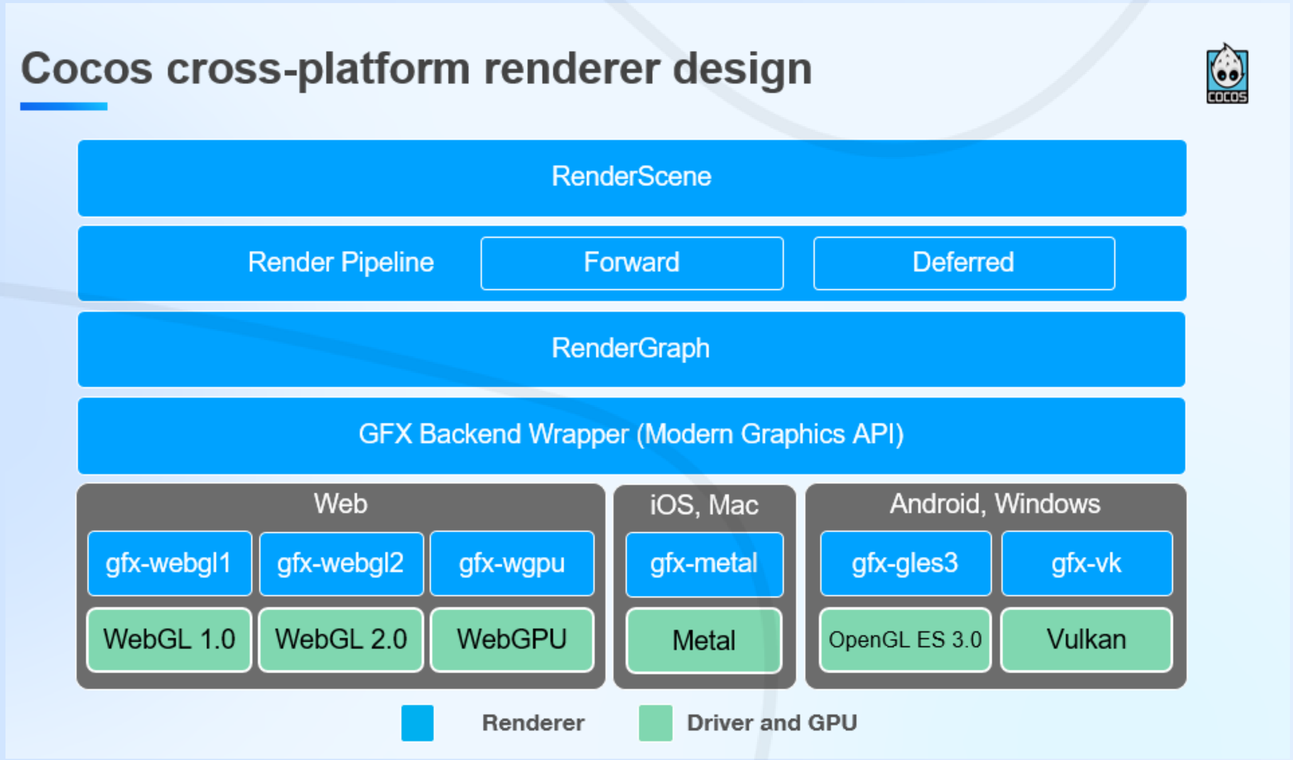
Популярные игры - Shadow Blade

Популярные игры - AFK Arena

Популярные игры - Plague Inc.

Популярные игры - Clash of kings

Кросспатформенность Cocos Creator

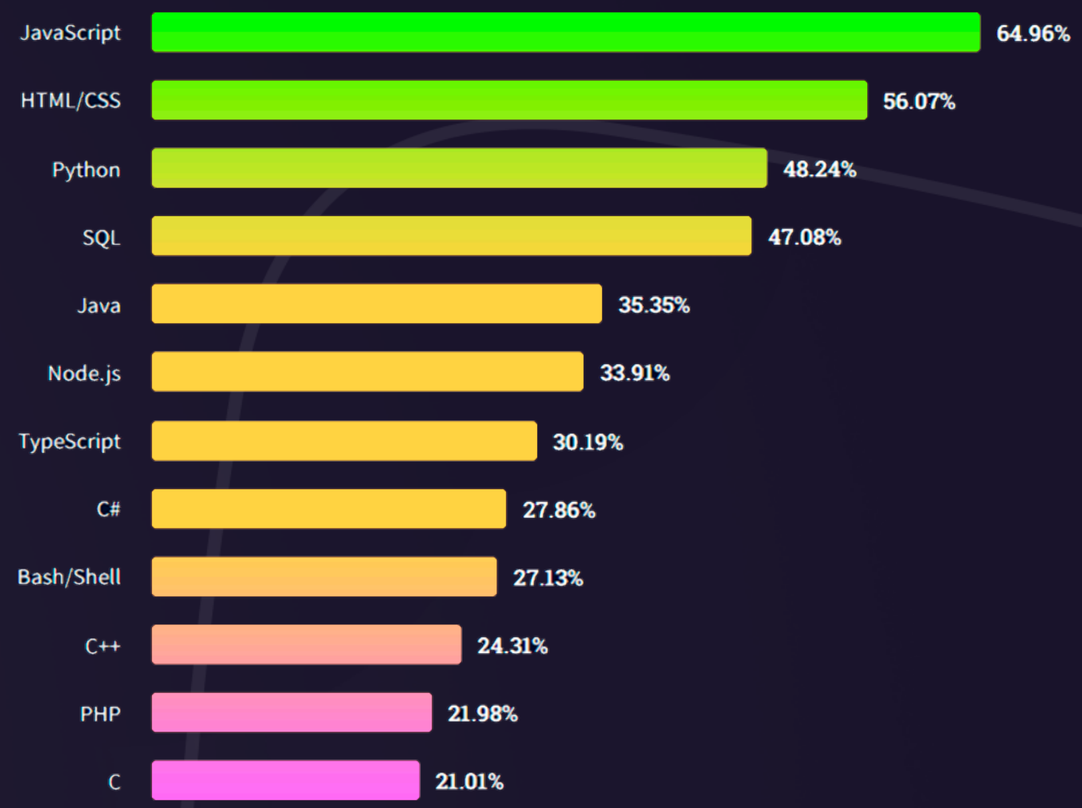


Эффективность — одна из причин, по которой многие разработчики предпочитают использовать Cocos для разработки многопользовательских игр. «Эффективность» здесь заключается не только в возможности быстро создавать контент с нуля, но и в процессе упаковки, сборки и выпуска проекта.

Cocos Creator поддерживает все основные платформы, позволяя разработчикам публиковать контент на веб-/игровых/нативных платформах одним щелчком мыши, снижая стоимость создания многоплатформенных версий.

Причины выбора Cocos Creator

- Язык разработки TypeScript - максимально популярная технология и проще найти специалистов;
- Кроссплатформенность;
- Поддержка технологии WebGPU;
- Схожесть интерфейса с Unity, а значит легкое освоение специалистами, работавшими с Unity;
- Бесплатный движок.

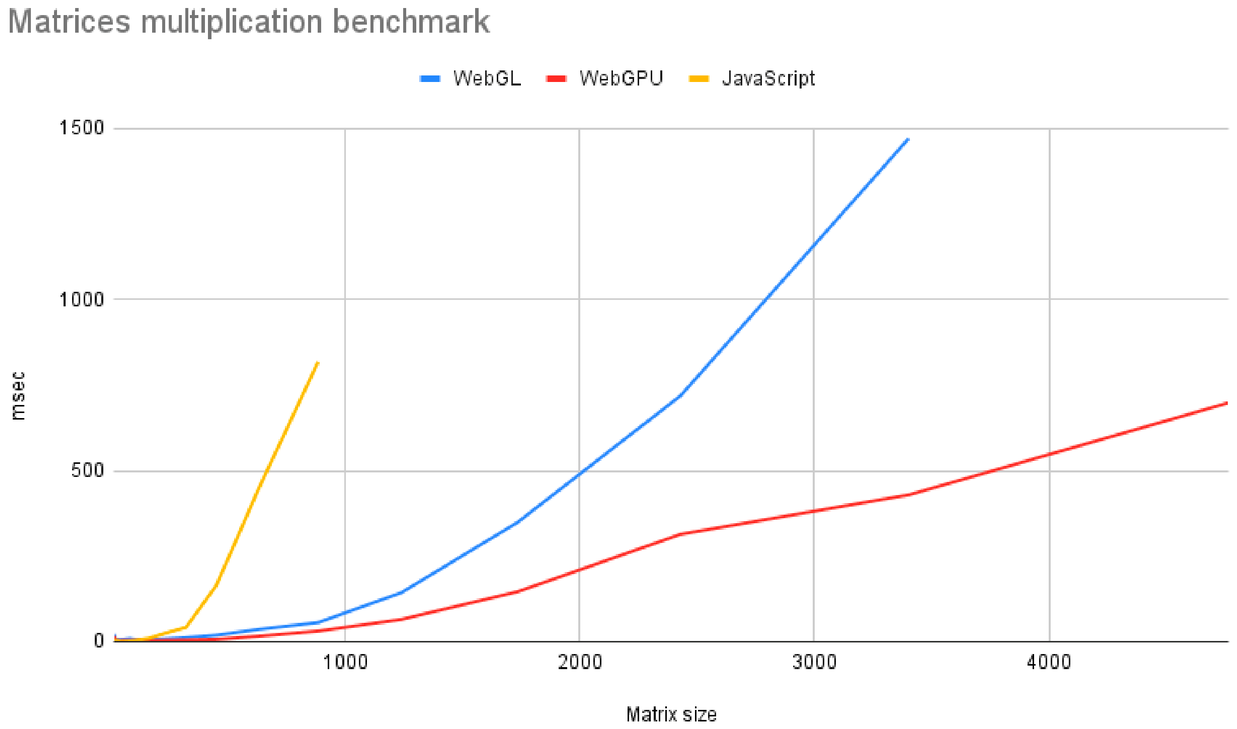


Источник:
https://insights.stackoverflow.com/survey/2021#most-popular-technologies-language

Причины выбора Cocos Creator

WebGL vs WebGPU

WebGPU — сменщик WebGL, новый API для работы с графическим ускорителем в браузере и появился в Chrome в начале 2023 года. По сравнению с WebGL он обещает лучшее быстродействие и лучшую совместимость с последними моделями видеокарт, но самой большим нововведением WebGPU является специальный функционал для вычислений на GPU.

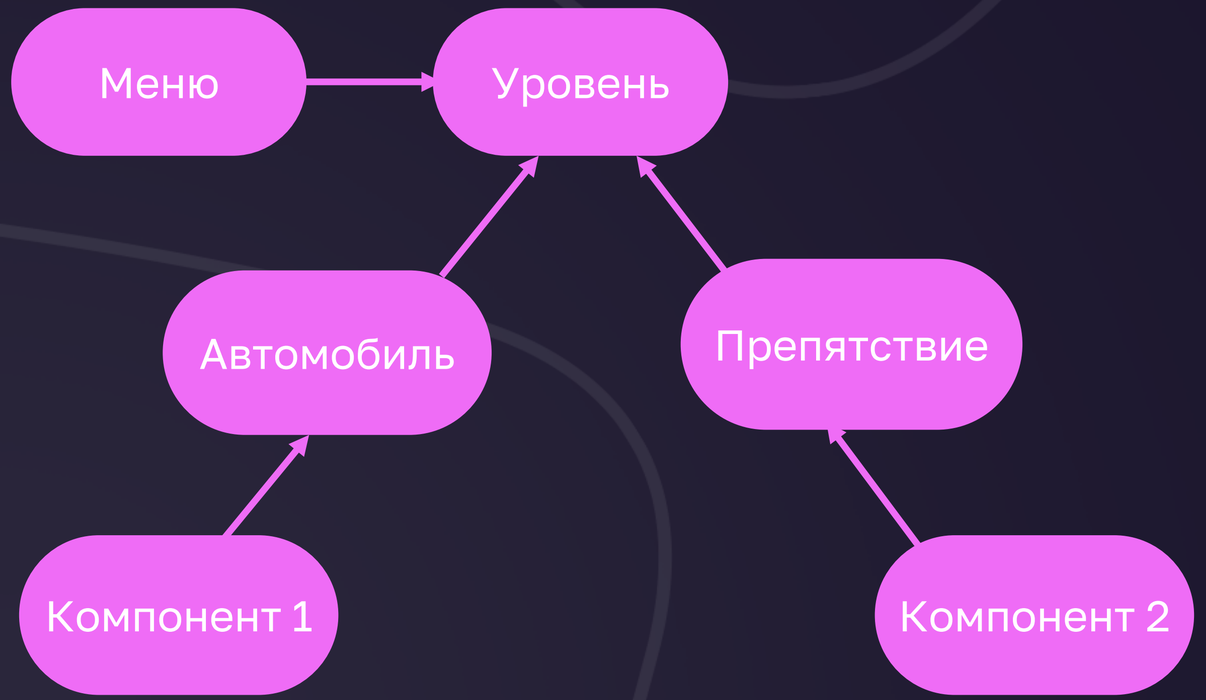


Источник:
https://pixelscommander.com/ru/javascript/webgpu-computations-performance-in-comparison-to-webgl/

Cocos Creator уже поддерживает WebGPU

Все ли так идеально?

Компонентно-ориентированная архитектура



Хранит в себе
скорость автомобиля

Для работы требуется
скорость автомобиля

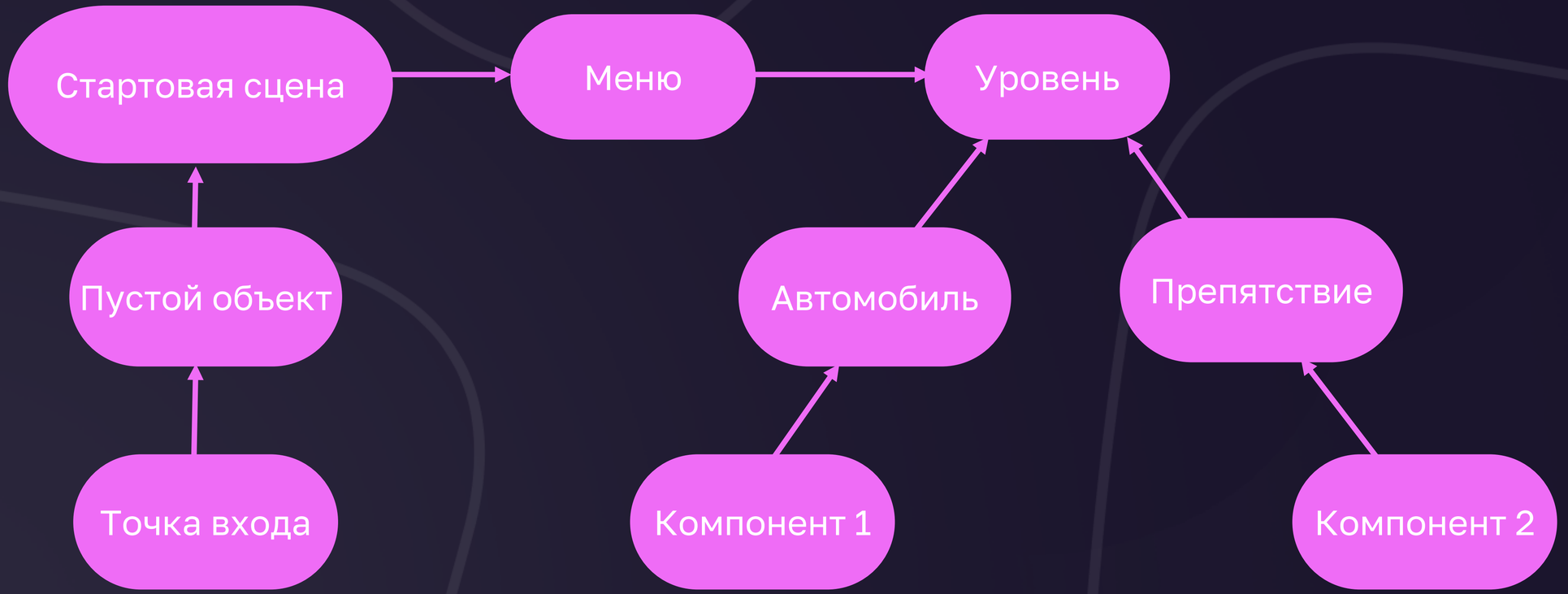
- Весь код игры хранится в компонентах;
- Компонент это обработчик событий, который прикрепляется к объекту в сцене;
- Объекты на сцене загружаются асинхронно;
- Точки входа нет.

Вывод:

мы не знаем, будет ли известна скорость автомобиля при загрузке препятствия

Все ли так идеально?

Компонентно-ориентированная архитектура



Точка входа

Требования:

- Создание глобальных переменных и сервисов;
- Запросы к серверу для получения настроек.

По большому счету это полноценная асинхронная функция, исполнения которой игра должна дождаться до старта сцены.

Механизм внедрения зависимостей

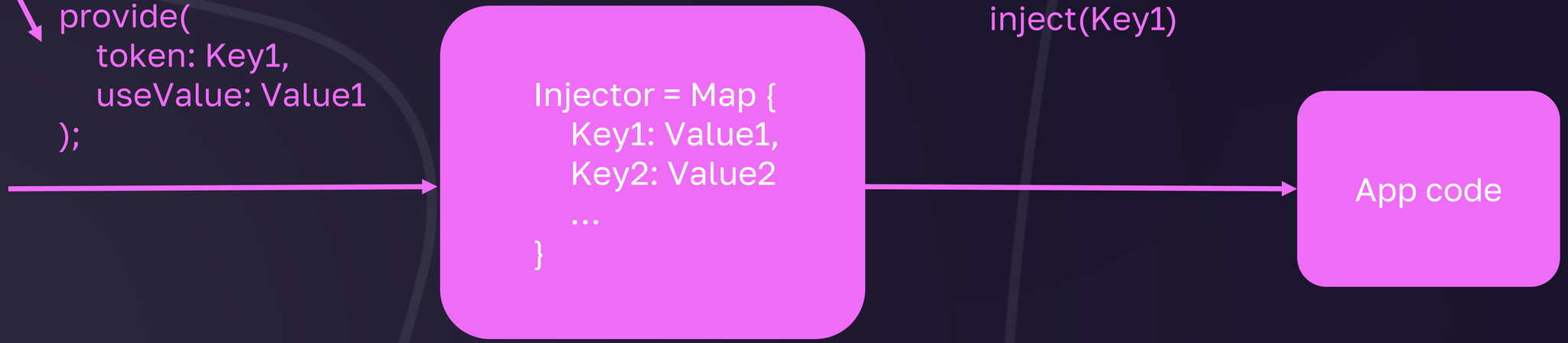
Токен может быть не только строкой, но и классом

Добавление:

```
provide(
    token: Key1,
    useValue: Value1
);
```

Использование:

```
inject(Key1)
```



Механизм внедрения зависимостей

Преимущества

- Возможность писать тесты не изменяя исходный код;
- Удобство поддержи;
- Повторное использование.

Пример:

Допустим, мы разрабатываем приложение, но у нас еще не готово для него API:

```
provide(
    token: ApiService,
    useClass: MockApiService
);
```

а внутри приложения можно написать inject(ApiService)

Механизм внедрения зависимостей

Свое решение

- Не подошли популярные решения: inversify/Inversifyjs, microsoft/tsyringe и другие
- Причина: Ограничение на работу декораторов TS в COCOS

Варианты использования:

- useClass: ClassName - для внедрения singleton зависимости
- useValue: "Some value" - для внедрения значения
- useFactory: () => {...some code} - для внедрения в зависимости от условия

Область видимости:

Для токена можно указать где он доступен: " game", "scene", "some scene name"

Переменные окружения

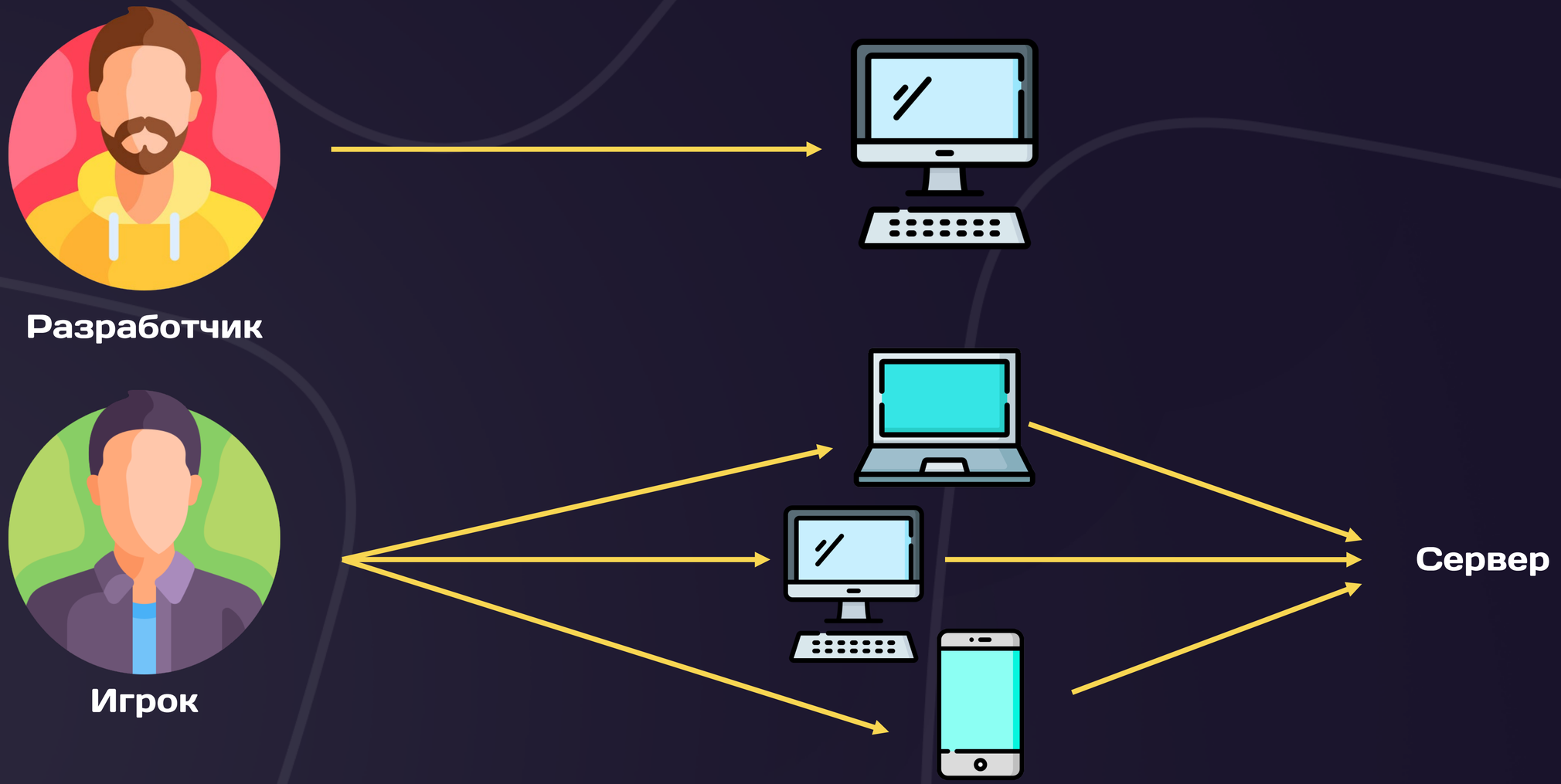
Переменные окружения позволяют единый код, а все зависимости вынести в отдельный файл, например env.json

```
{
  "platform": "yandex-games",
  "environment": "local",
  "debug": true,
    ..
}
```

DI + ENV = Love

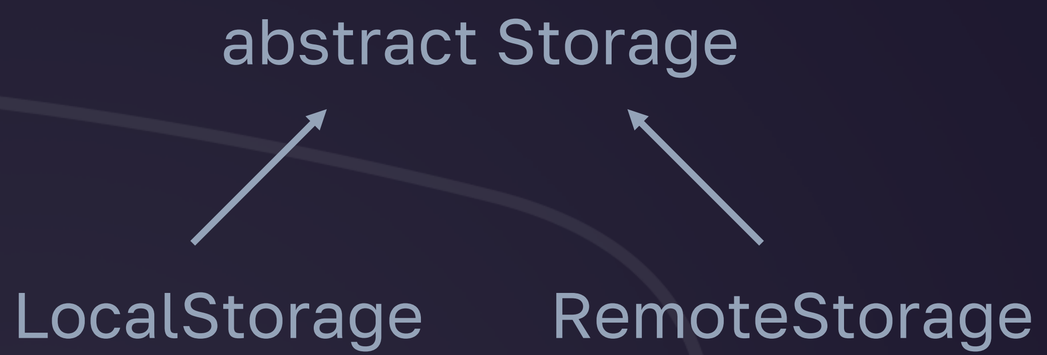
Удаленное хранилище

Суть задачи



Удаленное хранилище

Решение




```
provide(
    token: Storage,
    useFactory: () => {
        return env() === ‘production’
        ? new RemoteStorage(SOME_URL)
        : new LocalStorage()
    }
```

Удаленная конфигурация

У игры есть баланс, он хранится в игре в виде JSON файла и у геймдизайнера должна быть возможность править этот баланс.

Возможные варианты

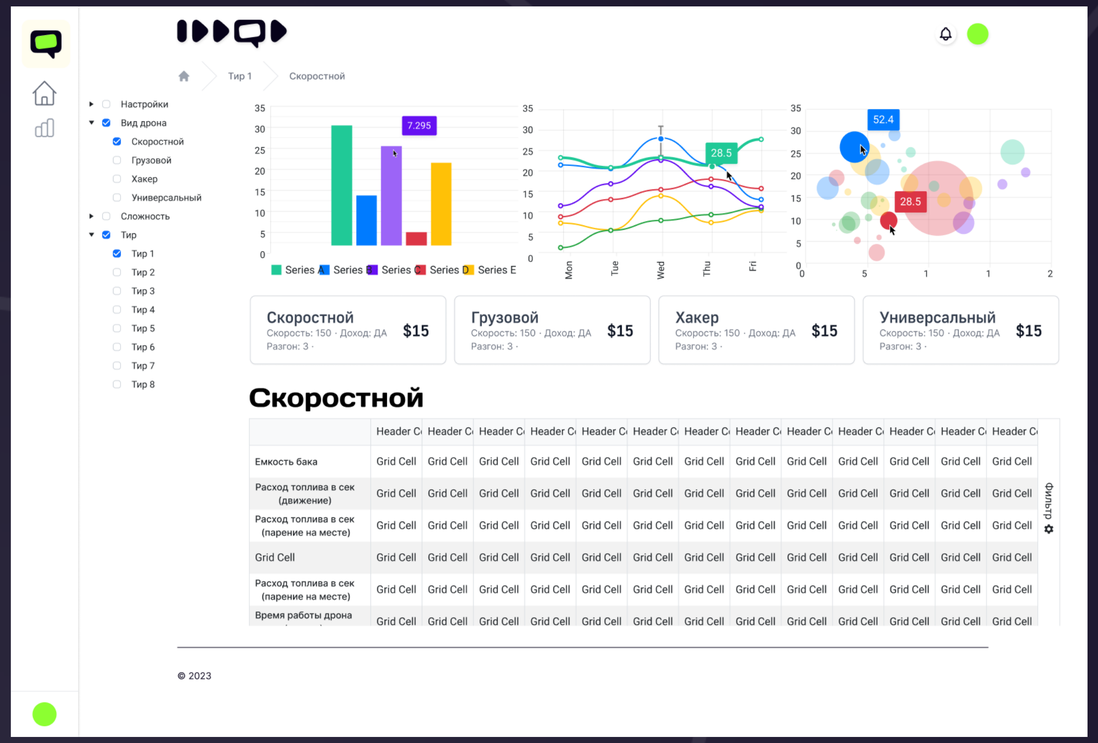
1. Научить геймдизайнера работе с Cocos Creator
2. Редактирование JSON на тестовом стенде при билде игры
3. Панель управления конфигурацией

Удаленная конфигурация

{ "Tier1": { "levelGenerationRules": { "levelBlocksStepsCount": 3, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 5, "damaging": 1 }, "inner": { "static": 4, "damaging": 2 }, "finish": { "static": 4, "damaging": 3 } }, "copterSafeZone": 0.1 }, "copterConfig": { "Speed": { "income": { "Gold": 15, "HackerToken": 0, "CargoToken": 15, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 15, "HackerToken": 15, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 15, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 15 } }, "Universal": { "income": { "Gold": 115, "HackerToken": 2, "CargoToken": 2, "SpeedToken": 2 } } } }, "Tier2": { "levelGenerationRules": { "levelBlocksStepsCount": 3, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 5, "damaging": 1 }, "inner": { "static": 4, "damaging": 2 }, "finish": { "static": 4, "damaging": 3 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 20, "fuelConsumptionInIdle": 40, "batteryEasy": 240, "batteryMedium": 200, "batteryHard": 160, "maxFuelQuantity": 400, "maxHealth": 3 }, "Speed": { "income": { "Gold": 40, "HackerToken": 0, "CargoToken": 40, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 40, "HackerToken": 40, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 40, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 40 } }, "Universal": { "income": { "Gold": 315, "HackerToken": 5, "CargoToken": 5, "SpeedToken": 5 } } } }, "Tier3": { "levelGenerationRules": { "levelBlocksStepsCount": 3, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 5, "damaging": 1 }, "inner": { "static": 4, "damaging": 2 }, "finish": { "static": 4, "damaging": 3 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 30, "fuelConsumptionInIdle": 60, "batteryEasy": 360, "batteryMedium": 300, "batteryHard": 240, "maxFuelQuantity": 600, "maxHealth": 3 }, "Speed": { "income": { "Gold": 75, "HackerToken": 0, "CargoToken": 80, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 75, "HackerToken": 80, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 75, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 80 } }, "Universal": { "income": { "Gold": 565, "HackerToken": 10, "CargoToken": 10, "SpeedToken": 10 } } } }, "Tier4": { "levelGenerationRules": { "levelBlocksStepsCount": 4, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 4, "damaging": 2 }, "inner": { "static": 4, "damaging": 3 }, "finish": { "static": 4, "damaging": 4 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 40, "fuelConsumptionInIdle": 80, "batteryEasy": 480, "batteryMedium": 400, "batteryHard": 320, "maxFuelQuantity": 800, "maxHealth": 3 }, "Speed": { "income": { "Gold": 120, "HackerToken": 0, "CargoToken": 125, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 120, "HackerToken": 125, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 120, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 125 } }, "Universal": { "income": { "Gold": 900, "HackerToken": 15, "CargoToken": 15, "SpeedToken": 15 } } } }, "Tier5": { "levelGenerationRules": { "levelBlocksStepsCount": 4, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 4, "damaging": 2 }, "inner": { "static": 4, "damaging": 3 }, "finish": { "static": 4, "damaging": 4 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 50, "fuelConsumptionInIdle": 100, "batteryEasy": 600, "batteryMedium": 500, "batteryHard": 400, "maxFuelQuantity": 1000, "maxHealth": 3 }, "Speed": { "income": { "Gold": 165, "HackerToken": 0, "CargoToken": 175, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 165, "HackerToken": 175, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 165, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 175 } }, "Universal": { "income": { "Gold": 1240, "HackerToken": 20, "CargoToken": 20, "SpeedToken": 20 } } } }, "Tier6": { "levelGenerationRules": { "levelBlocksStepsCount": 5, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 4, "damaging": 3 }, "inner": { "static": 4, "damaging": 4 }, "finish": { "static": 3, "damaging": 6 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 60, "fuelConsumptionInIdle": 120, "batteryEasy": 720, "batteryMedium": 600, "batteryHard": 480, "maxFuelQuantity": 1200, "maxHealth": 3 }, "Speed": { "income": { "Gold": 230, "HackerToken": 0, "CargoToken": 230, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 230, "HackerToken": 230, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 230, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 230 } }, "Universal": { "income": { "Gold": 1640, "HackerToken": 30, "CargoToken": 30, "SpeedToken": 30 } } } }, "Tier7": { "levelGenerationRules": { "levelBlocksStepsCount": 5, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 4, "damaging": 3 }, "inner": { "static": 4, "damaging": 4 }, "finish": { "static": 3, "damaging": 6 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 70, "fuelConsumptionInIdle": 140, "batteryEasy": 840, "batteryMedium": 700, "batteryHard": 560, "maxFuelQuantity": 1400, "maxHealth": 3 }, "Speed": { "income": { "Gold": 280, "HackerToken": 0, "CargoToken": 290, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 280, "HackerToken": 290, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 280, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 290 } }, "Universal": { "income": { "Gold": 2070, "HackerToken": 35, "CargoToken": 35, "SpeedToken": 35 } } } }, "Tier8": { "levelGenerationRules": { "levelBlocksStepsCount": 5, "safeZoneSize": 0.1, "obstacleQuantity": { "start": { "static": 4, "damaging": 3 }, "inner": { "static": 4, "damaging": 4 }, "finish": { "static": 3, "damaging": 6 } }, "copterSafeZone": 0.1 }, "copterConfig": { "common": { "maxMovementSpeed": 10000, "accelerationTime": 3, "fuelConsumptionInMotion": 80, "fuelConsumptionInIdle": 160, "batteryEasy": 960, "batteryMedium": 800, "batteryHard": 640, "maxFuelQuantity": 1600, "maxHealth": 3 }, "Speed": { "income": { "Gold": 340, "HackerToken": 0, "CargoToken": 360, "SpeedToken": 0 } }, "Cargo": { "income": { "Gold": 340, "HackerToken": 360, "CargoToken": 0, "SpeedToken": 0 } }, "Hacker": { "income": { "Gold": 340, "HackerToken": 0, "CargoToken": 0, "SpeedToken": 360 } }, "Universal": { "income": { "Gold": 2540, "HackerToken": 45, "CargoToken": 45, "SpeedToken": 45 } } } } }, "incomeMultipliers": { "hardSuccess": 1, "hardFailure": 0.4, "mediumSuccess": 0.8, "mediumFailure": 0.32, "easySuccess": 0.6, "easyFailure": 0.3, "lightCargo": 0.1, "mediumCargo": 0.15, "heavyCargo": 0.2, "terminalHack": 0.1, "winChest": 0.8, "supportChest": 1 }, "commonConfig": { "maxEnergy": 170, "levelStartEnergyCost": 10 } }

И еще 10 раз по столько

Удаленная конфигурация




```
provide(
  token: ‘GameConfig’,
  useFactory: () => {
    return env() === ‘staging’
    ? new RemoteConfiguration(SOME_URL)
    : new LocalConfiguration(LOCAL_PATH)
  }
```

Менеджер игровых событий

Как работает менеджер COCOS

```
import { EventTarget } from 'cc';
const eventTarget = new EventTarget();

function eventHandler( event ) {
  console.log(”some event fired”);
}
```

Подписка на событие:

```
eventTarget.on(SOME_EVENT, eventHandler, this);
```

Отписка обработчика:

```
eventTarget.off(SOME_EVENT, eventHandler, this);
```

А если хочется использовать анонимную функцию?

```
eventTarget.on(SOME_EVENT, () => {...code}, this);
```

Как отписаться от такой функции?

```
eventTarget.off(SOME_EVENT, ???, this);
```

Менеджер игровых событий

Как работает наш менеджер

```
import { EventManager } from ‘platform-tools';
const eventManager = new EventManager();
```

Подписка на событие:

```
const listener = eventManager.on(SOME_EVENT, () => { console.log(`some event fired`)});
```

Отписка обработчика:

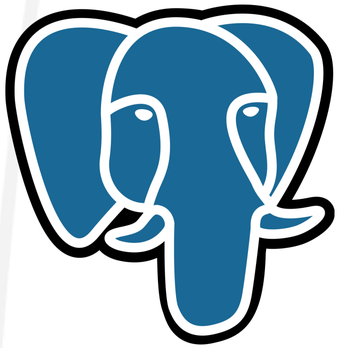
```
listener.off();
```

Инициализация события:

```
eventManager.emit(SOME_EVENT, ...args);
```

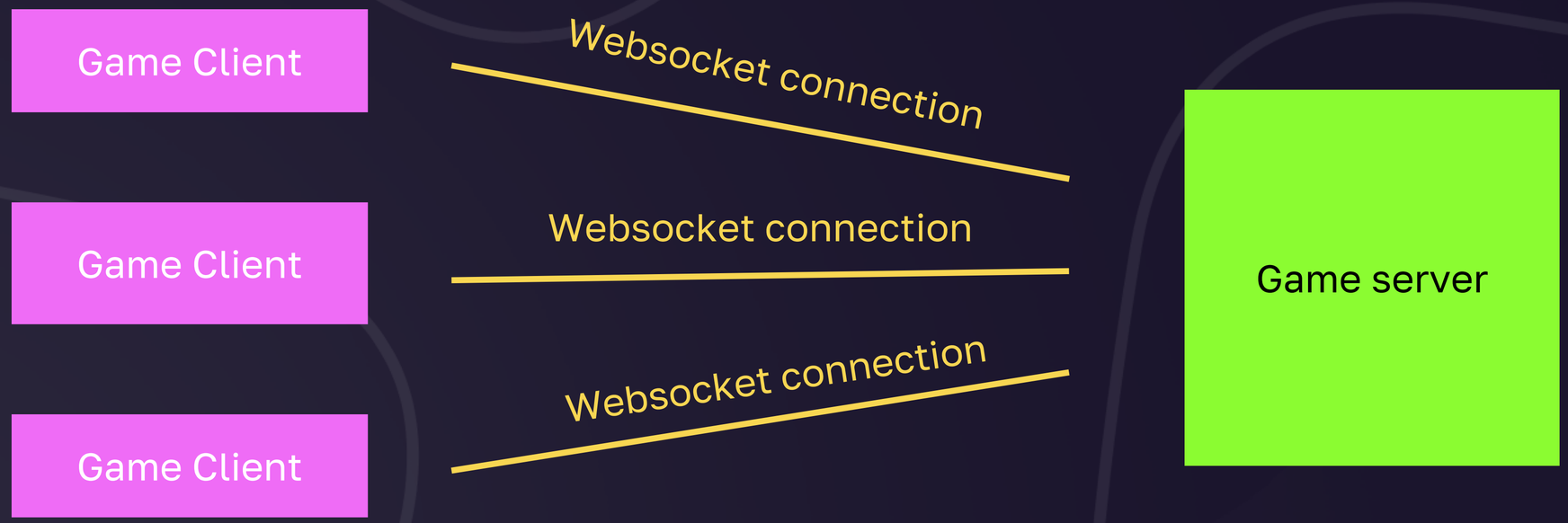
Главное отличие: поддержка анонимных функций, в том числе и стрелочных, а значит можно не указывать постоянно контекст this.

Как мы собираем бэкенд игры

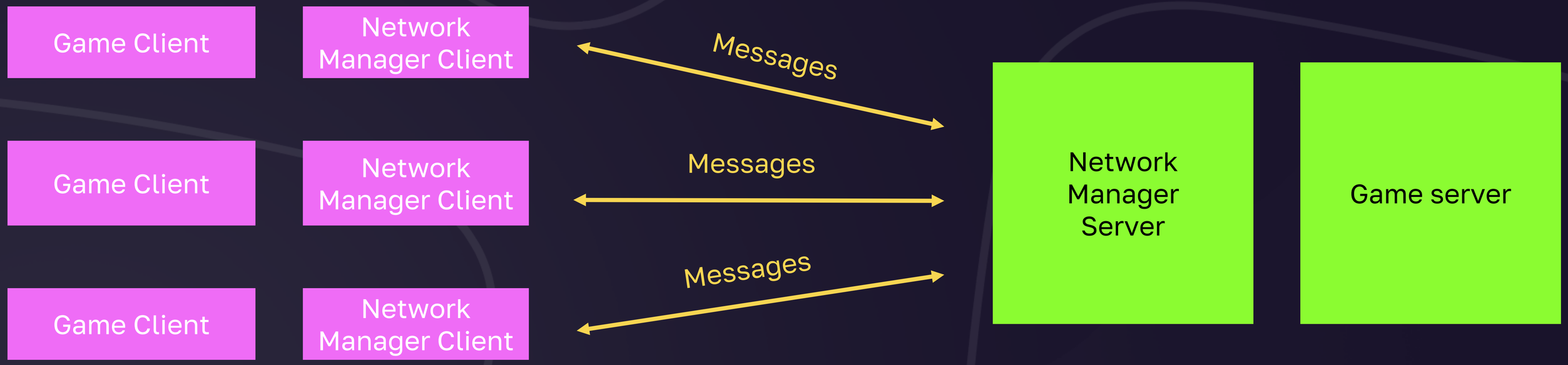


Как мы собираем бэкенд игры

Сетевой код


Game Client
Game Client
Game Client
Websocket connection
Websocket connection
Websocket connection
Game server

Сетевой код


Game Client
Network Manager Client
Game Client
Network Manager Client
Game Client
Network Manager Client
Messages
Messages
Messages
Network Manager Server
Game server

Сообщения

Пример сообщения:

```
Interface PositionMessageTransferData = {
  x: number,
  y: number
}

class PositionMessage extends Message {
  x: number;
  y: number;
}
```

Нужно добавить:

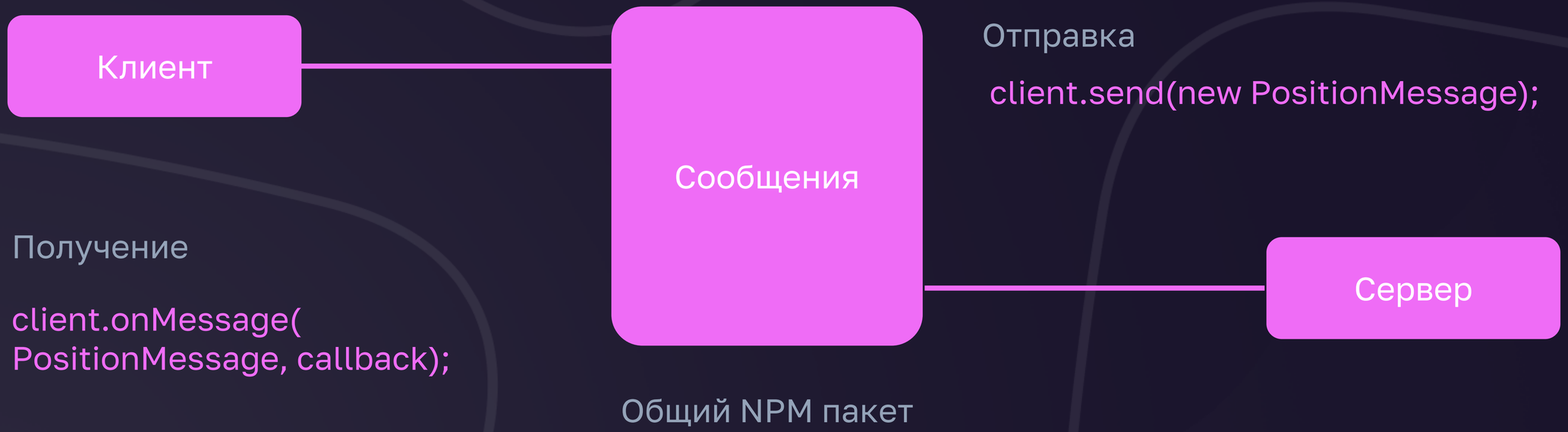
```
toJSON(): PositionMessageTransferData {
  return {
    x: this.x,
    y: this.y
  }
}

static fromJSON(data: PositionMessageTransferData)  {
  const message = new PositionMessage;
  message.x = data.x;
  message.y = data.y;
  return message;
}
```

Отправка

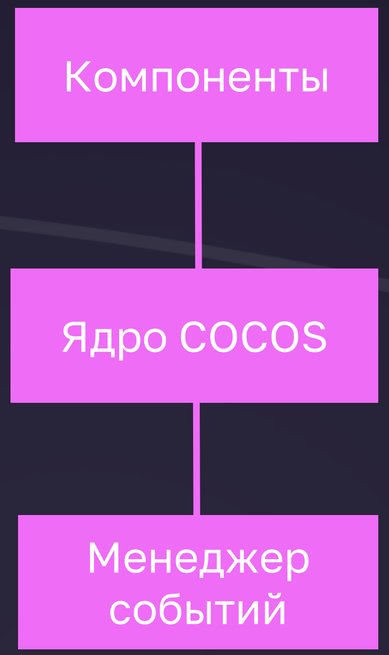
```
client.send(new PositionMessage);
```

Получение

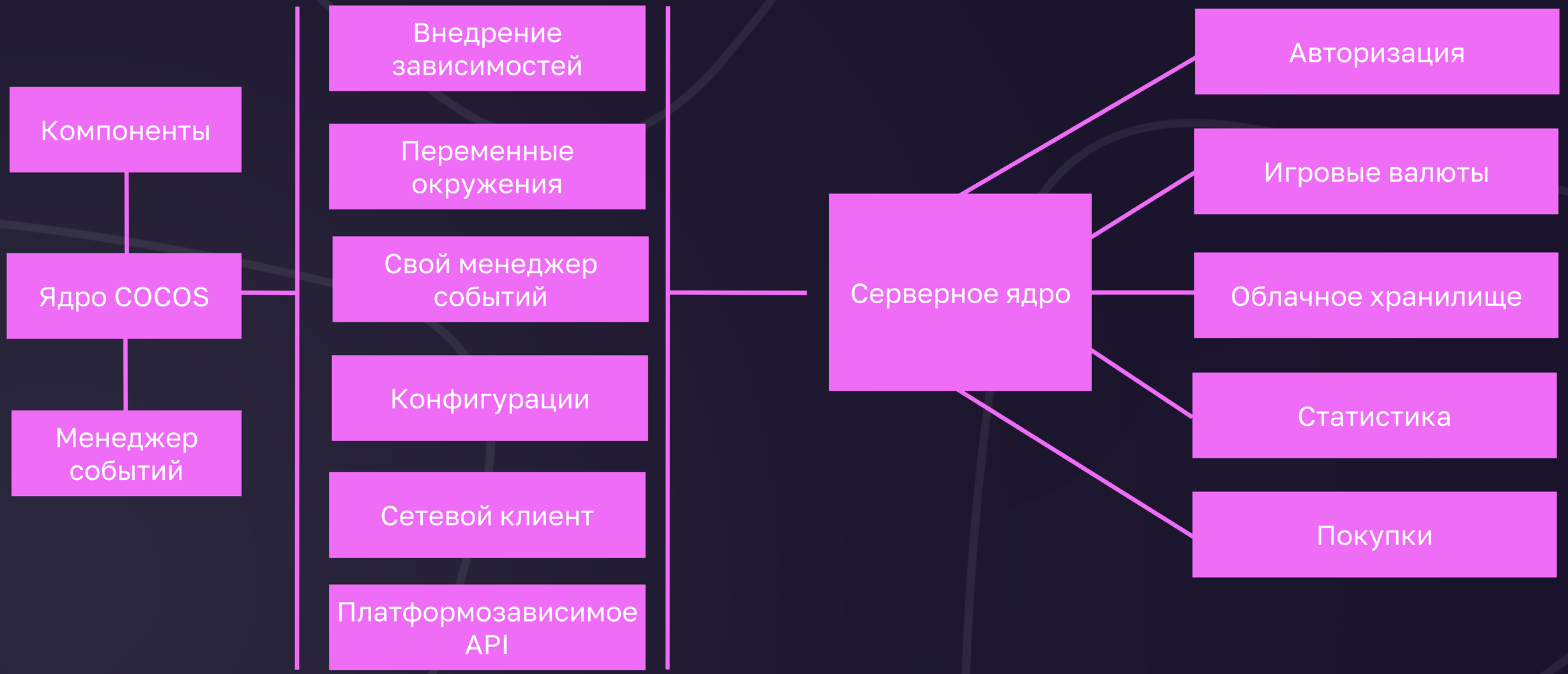
```
client.onMessage(
PositionMessage, callback);
```


Клиент
Сообщения
Сервер
Отправка
client.send(new PositionMessage);
Получение
client.onMessage(
PositionMessage, callback);
Общий NPM пакет

Итого: как было



Итого: как стало



Благодарю за внимание

Ваши вопросы?

@DmitriyTretyakov