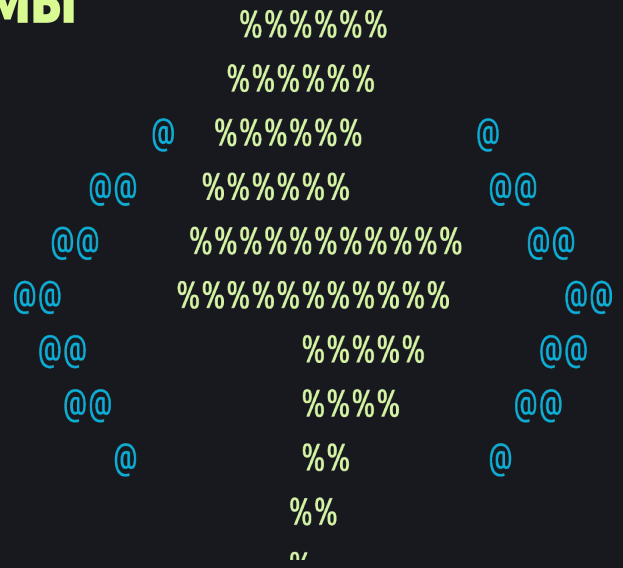
Решаем одну проблему с помощью Azure Functions. Теперь у нас две проблемы



DODO ENGINEERING

Юрий Пастушенко

Чем займёмся

- Немного расскажу о Serverless и об Azure Functions
- Создадим и задеплоим функцию (даже два раза)
- Покопаемся во внутренностях
- Поделюсь опытом использования Azure Functions (печальным и не очень)

Serverless – это когда ты

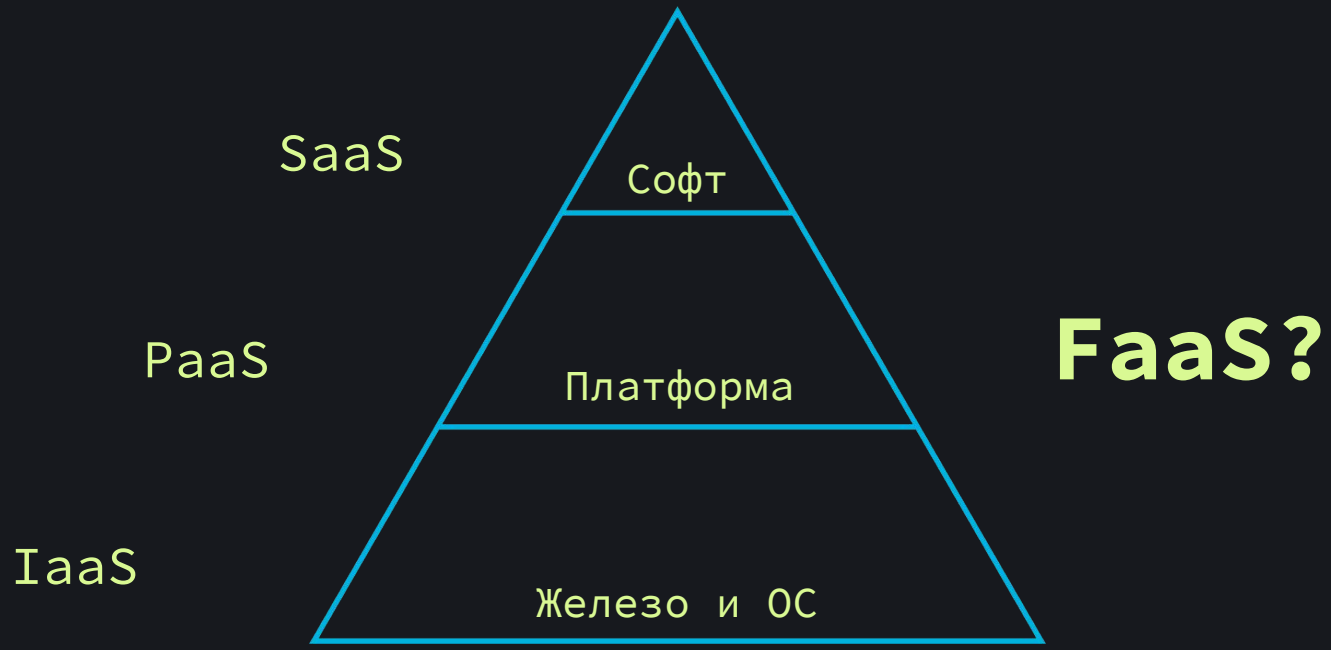

Не думаешь об инфраструктуре и масштабируемости


Платишь только за время, которое код работал


Испытываешь горечь от ограничений по памяти и CPU

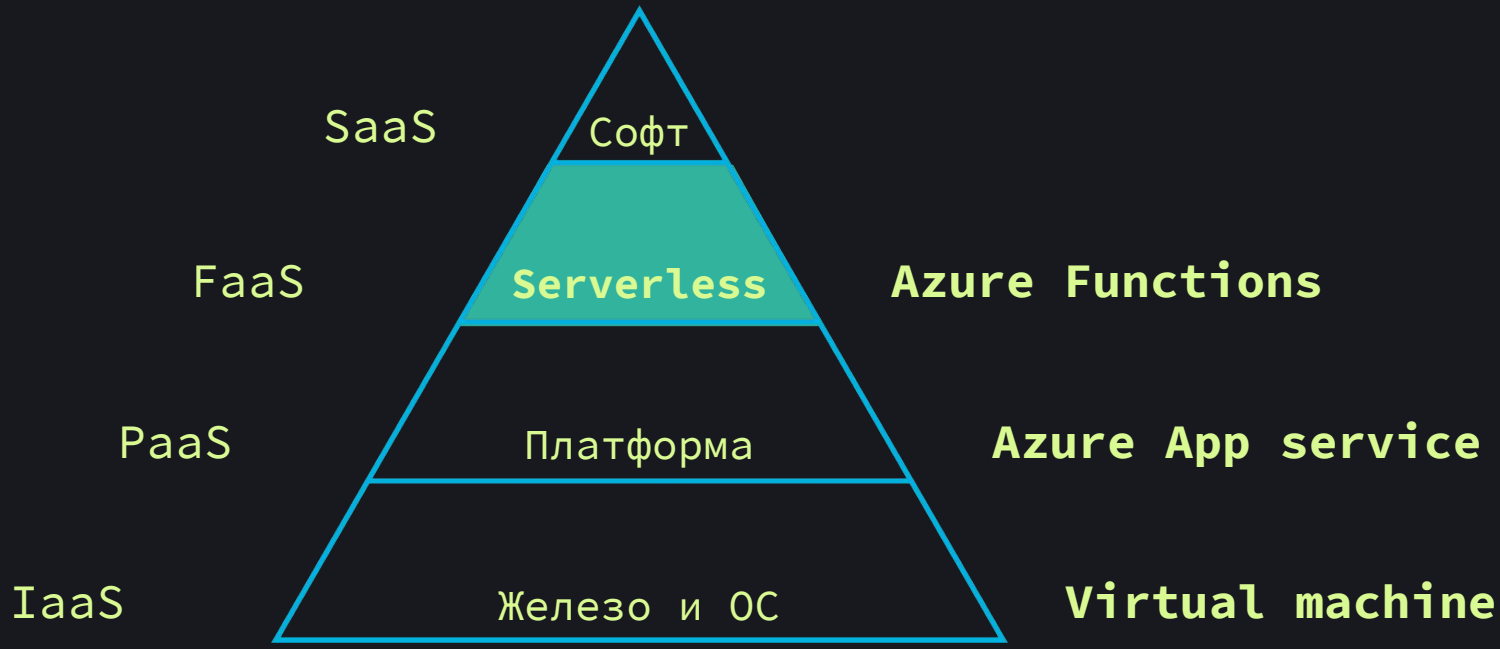

Борешься с проблемой холодного старта

Function as a service (FaaS)

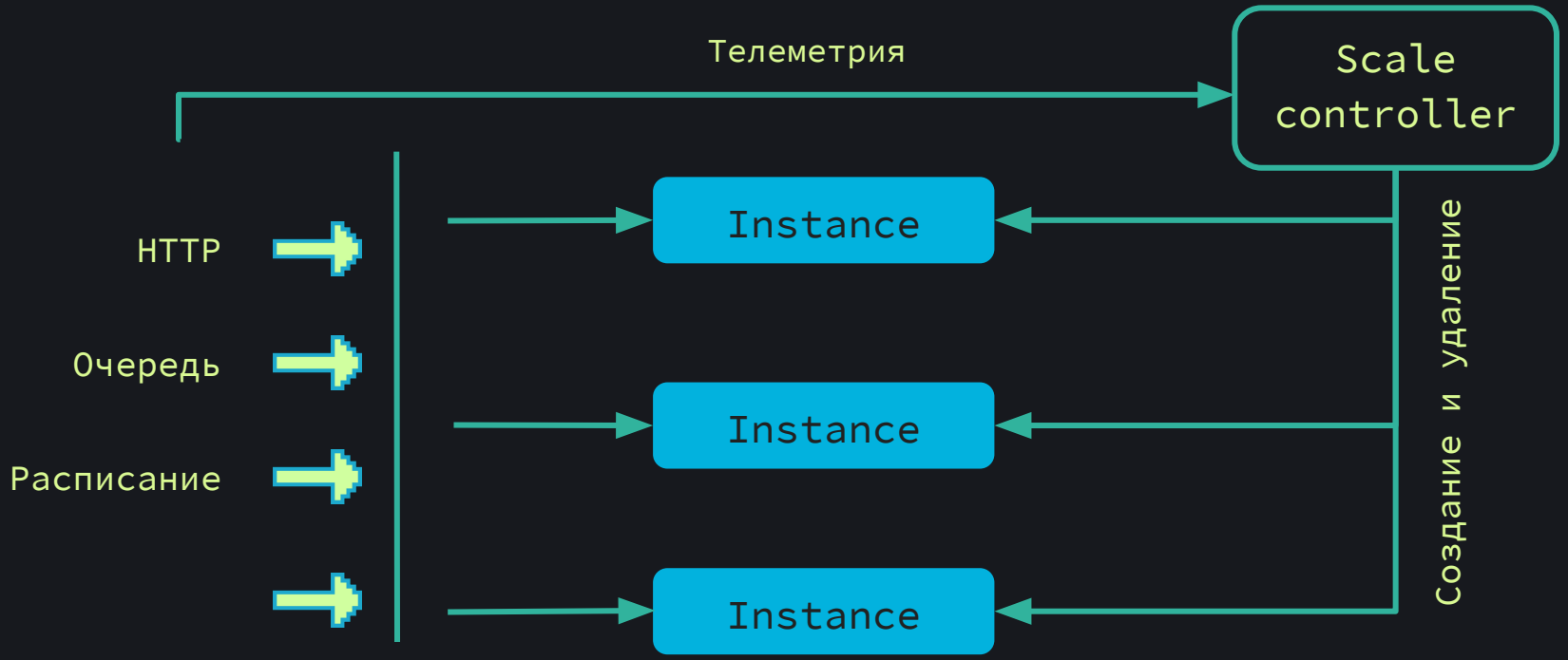


FaaS?

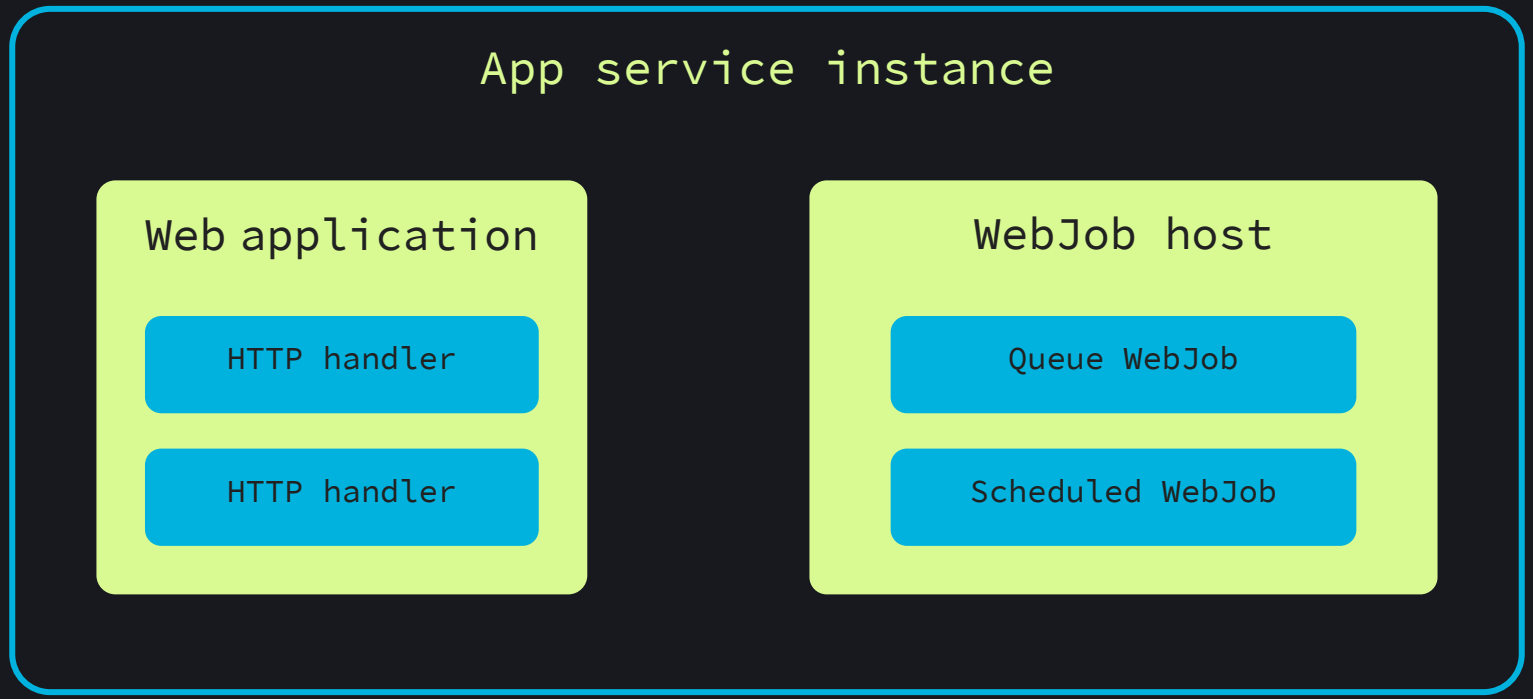
Function as a service



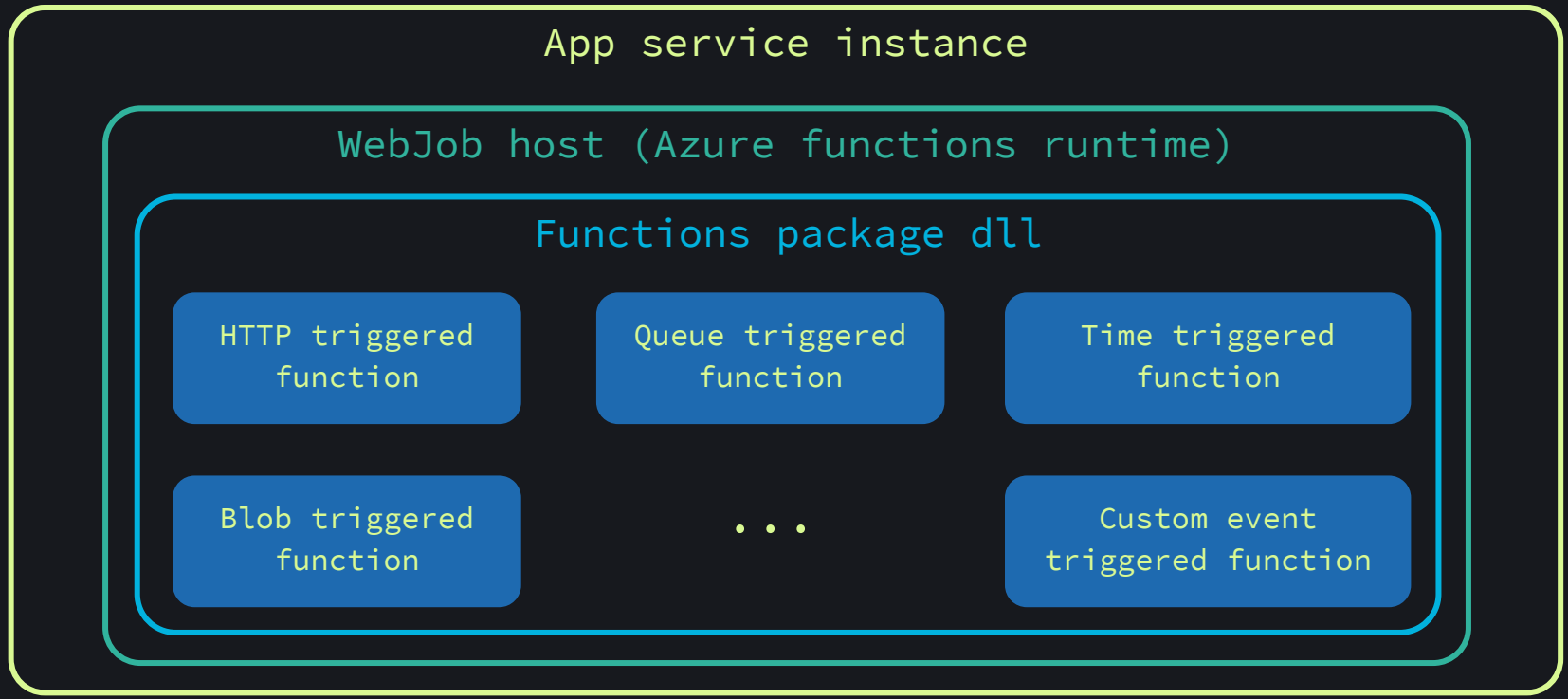
Azure App Service



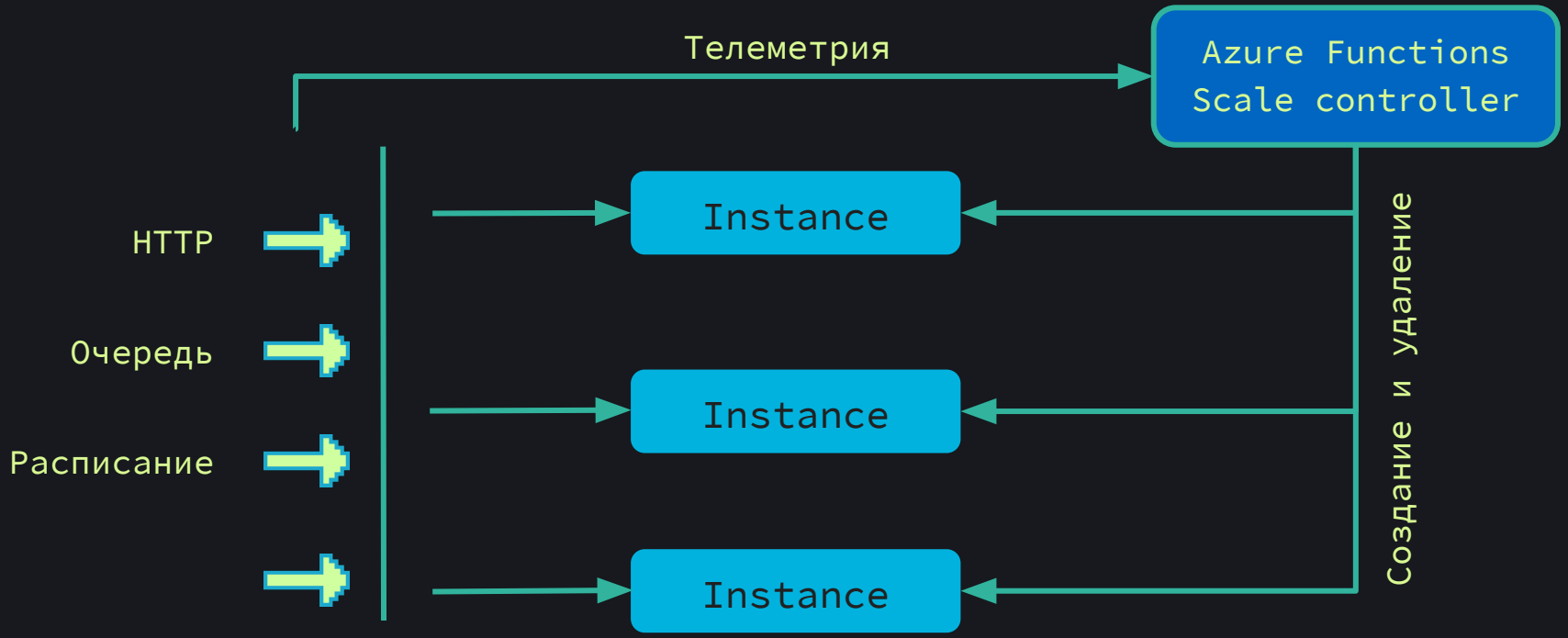
Azure App Service WebJobs



Azure App Service -> Azure Functions



Azure App Service -> Azure Functions



Azure function execution types (.NET)

Script

- Можно писать прямо в Azure portal
- “Свой” C# или F#
- Не стоит использовать в проде

In-process

- DLL с функциями загружается в процесс хоста
- Зависимости могут конфликтовать

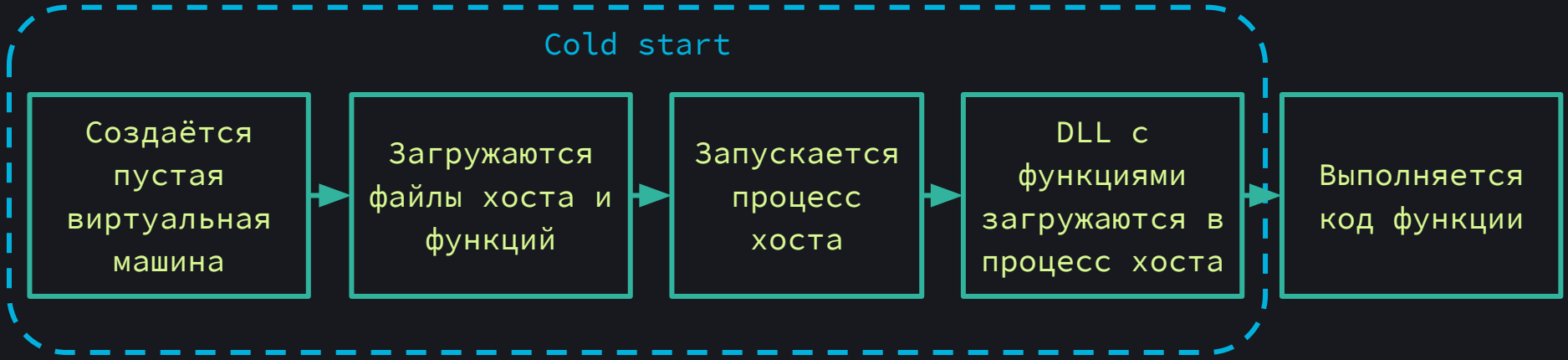
Isolated process

- Обычное консольное приложение с IHost
- Изолировано от хоста в другом процессе
- Начиная с .NET 5

Azure Functions pricing plans

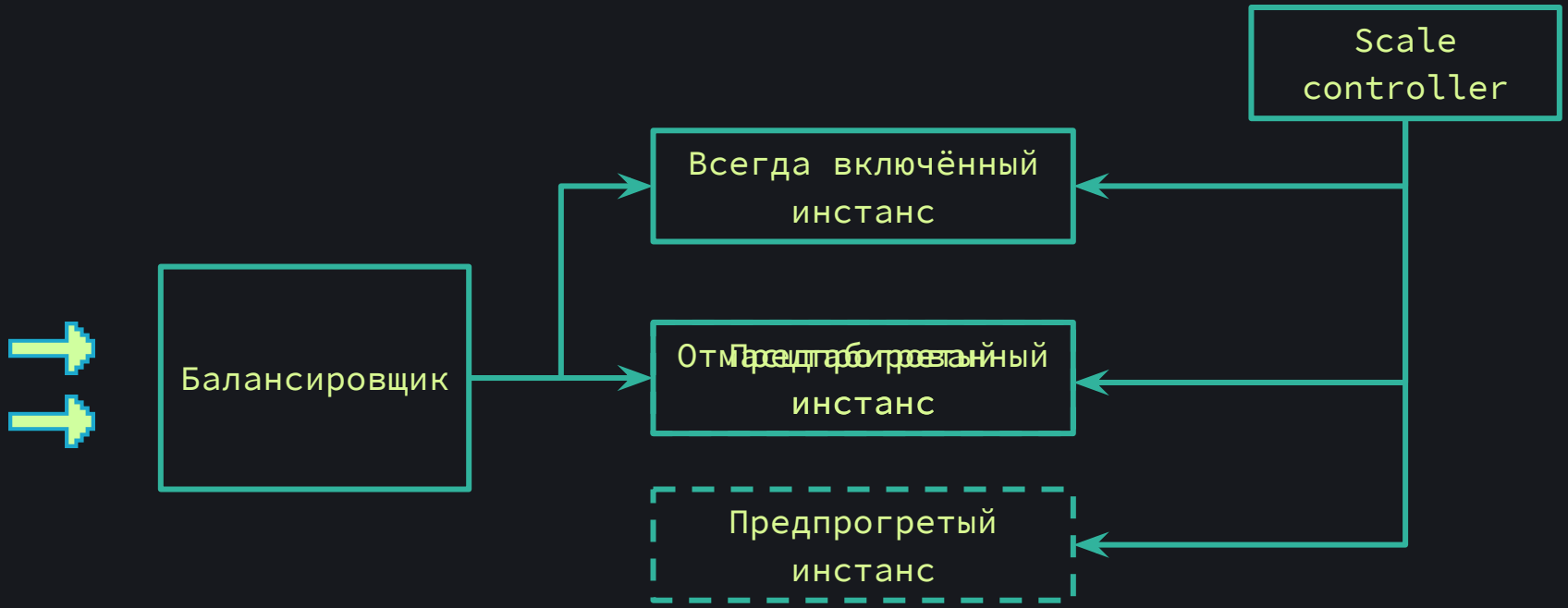
	Consumption	Premium	Dedicated
Максимальное количество инстансов	100	20 для Linux	30
Оплата	Только за фактическое использование	За фактическое использование + за всегда включенные инстансы	За время, когда инстансы включены
Максимальный размер инстанса	1,5 Gb памяти 100 ACU	14 Gb памяти 840 ACU	Ограничен максимальной виртуальной машиной
Проблема холодного старта	Есть	Решена предпрогретыми инстансами	Есть, но стоит не так остро

Cold start problem




```
public class DontSleep
{
    public async Task Run([TimerTrigger("0 */5 * * * *")] TimerInfo timerInfo, Logger<DontSleep> logger)
    {
        logger.LogInformation("Don't sleep");
    }
}
```

Cold start problem: Premium plan



Azure Functions core tools

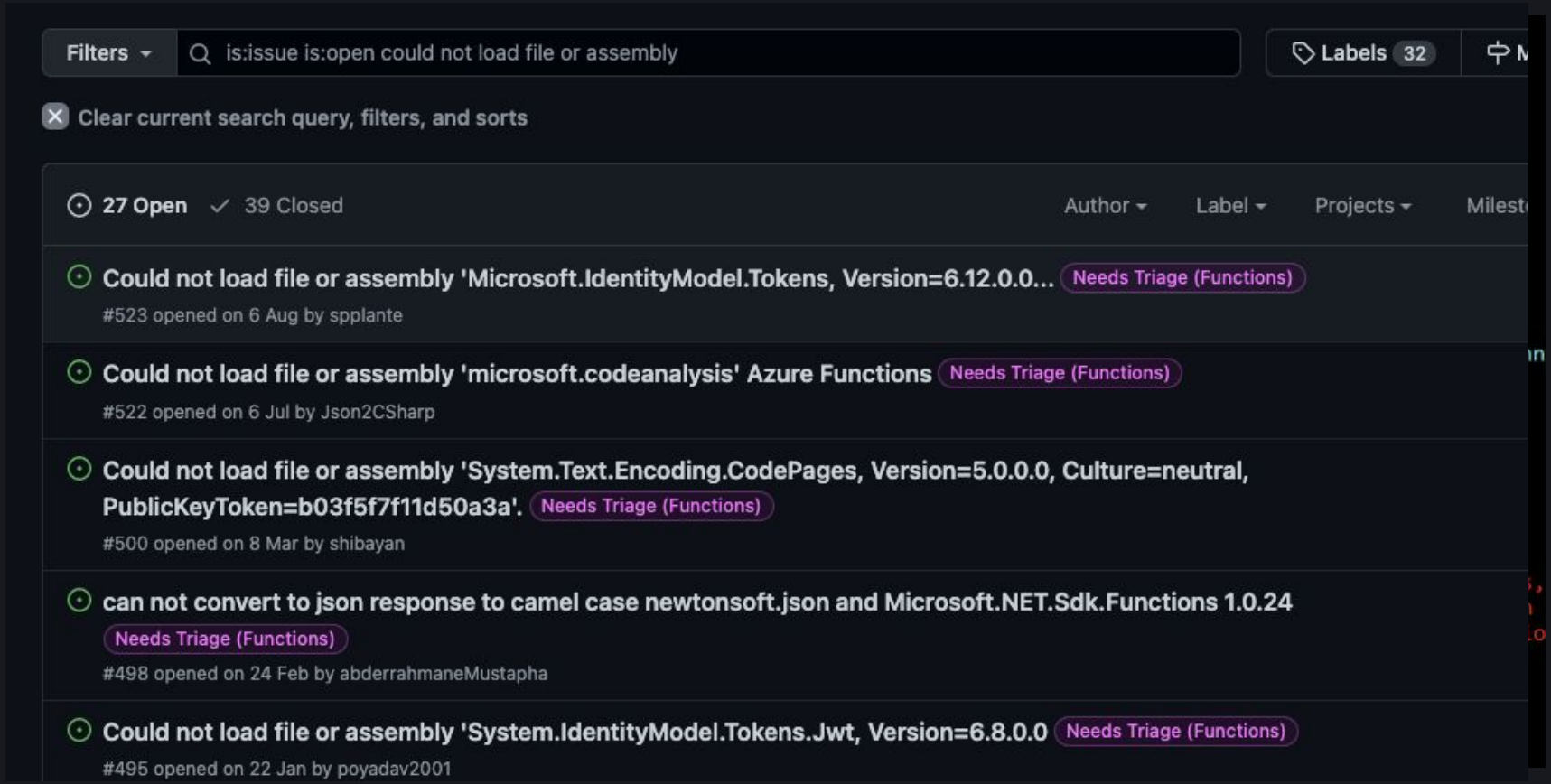
Создаёт проекты и сами функции

Запускает функции в отладке

Деплоит прямо в облако

Давайте создадим function app!

Проблемы in-process: Dependency hell



Проблемы in-process: Dependency hell

Версия хоста при локальном запуске не совпадает с версией хоста в облаке

При большом количестве зависимостей появляются несовместимости, которые приведут к ошибке

Проблемы in-process: directory structure

При использовании нативных библиотек есть риск получить после деплоя ошибку:

```
Error loading native library. Not found in any of the possible locations:
/home/site/wwwroot/bin/libgrpc_csharp_ext.x64.so,
/home/site/wwwroot/bin/runtimes/linux-x64/native/libgrpc_csharp_ext.x64.so,
/home/site/wwwroot/bin/../../runtimes/linux-x64/native/libgrpc_csharp_ext.x64.so
```

Лечится добавлением таргета:

```
<Target Name="CopyGrpcOnBuild" AfterTargets="Build">
      <Copy
            SourceFiles="$(OutputPath)/runtimes/linux-x64/native/libgrpc_csharp_ext.x64.so"
            DestinationFolder="$(OutputPath)/bin"
      />
</Target>
```

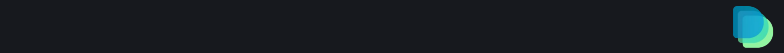
Проблемы in-process: конфигурация и DI

```
public class Startup : FunctionsStartup
{
    public override void Configure(IFunctionsHostBuilder builder)
    {
        var configurationBuilder = new ConfigurationBuilder();
        if (IsAzureFunctionsCloud())
        {
            configurationBuilder.SetBasePath("/home/site/wwwroot");
        }

        var configuration = configurationBuilder
            .AddJsonFile("appsettings.json")
            .AddJsonFile($"appsettings.{Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT")}.json")
            .AddEnvironmentVariables()
            .Build();
```


Решение -
isolated process

Isolated process

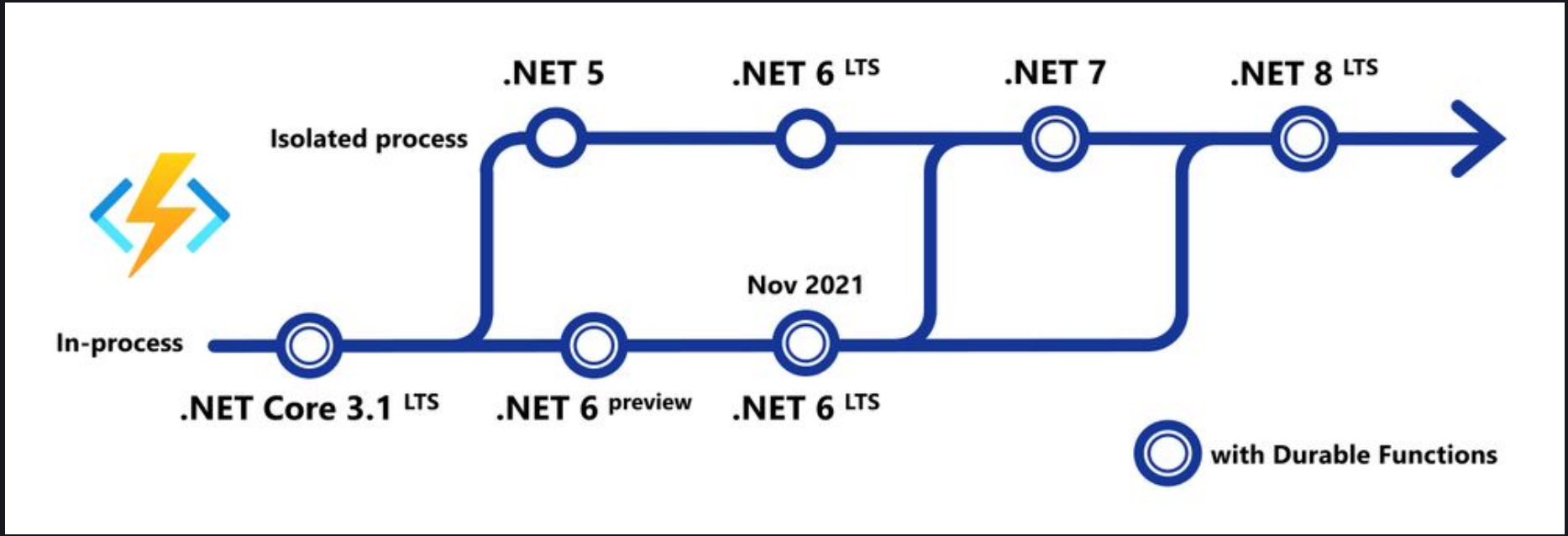


```
public class Program
{
    public static async Task Main(string[] args)
    {
        var host = Host
            .CreateDefaultBuilder()
            .ConfigureFunctionsWorkerDefaults()
            .ConfigureAppConfiguration((_, config) => config
                .AddJsonFile("appsettings.json", false, false)
                .AddJsonFile($"appsettings.{Variables.EnvironmentName}.json", true, false)
                .AddEnvironmentVariables()
            )
            .ConfigureServices(services => services.AddSingleton<MyAwesomeService>())
            .Build();

        await host.RunAsync();
    }
}
```

Azure Function isolated model

Azure Function roadmap



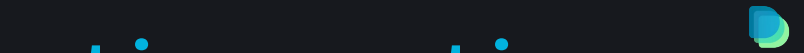
Создаём isolated process function app

In-process или isolated process?

- Все новые функции следует запускать как isolated process
- Старые функции рано или поздно придётся переписать
- Если важно время холодного старта, то лучше перейти на Premium plan
- Единственная причина использовать in-process - это durable functions

Версионирование Azure Functions

Версионирование Azure Functions runtime



Azure Functions version	FUNCTIONS_EXTENSION_VERSION	WebHost version
4.x (preview)	~4	4.x (.NET 6.0)
3.x	~3	3.x (.NET Core 3.1)
2.x	~2	3.x (.NET Core 3.1)
2.x	~2.0	2.x (.NET Core 2.2)
1.x	~1	1.x (.NET Framework 4.8)

Azure Functions versioning: app targets

Azure Functions version	.NET in-process	.NET isolated process	Node.js
4.x (preview)	.NET 6.0	.NET 6.0	Node 14
3.x	.NET Core 3.1	.NET 5.0	Node 14, 12, 10
2.x	.NET Core 2.1	-	Node 10, 8
1.x	.NET Framework 4.7.2	-	Node 6

Azure Functions versioning: WebHost

- Мажорная версия рантайма определяется значением переменной FUNCTIONS_EXTENSION_VERSION (~1,~2 ...)
- Azure стремится запустить функции в последней доступной минорной версии рантайма
- На Windows можно зафиксировать минорную версию, но она рано или поздно состарится
- Объявления о выводе минорных версий публикуются в репозитории app-service-announcements

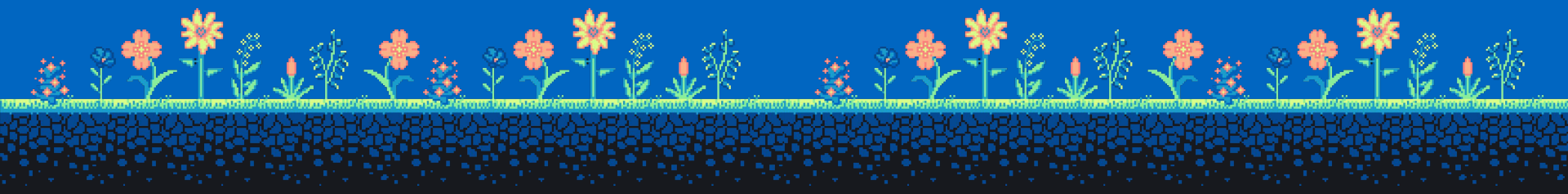
host.json version

```
{
  "version": "2.0",
  "functionTimeout": "00:10:00",
  "logging": {
    "logLevel": {
      "default": "Information"
    }
  }
}
```

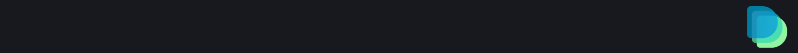
1.0 - для Azure Functions 1.x

2.0 - для всех остальных версий

Наши примеры



Кейс 1: сбор аналитики



- Функция принимает события аналитики со всех наших клиентов (iOS, Android, Web) по HTTP
- Конвертирует сообщение из одного формата json в другой и кладёт в очередь
- В пиках до 1000 rps и до 50ти инстансов
- SL 99.5 (гарантировано 99.95)

Сколько эта функция стоила в сентябре 2021?

Кейс 2: изменение размера фотографий

- Функция принимает на вход ссылку на большую картинку и выдаёт 2 уменьшенные копии
- Является частью бизнес-приложения с большим количеством зависимостей
- Использует нативную библиотеку Skia.Sharp

Кейс 2: проблемы

- Не помещалась в 1,5 Гб памяти из-за неэффективной библиотеки
  - Перешли на библиотеку Skia.Sharp
- Проблемы с зависимостями и со структурой директорий
  - Решилось переходом на isolated process
- Blob-trigger оказался ненадёжным
  - Перешли на queue trigger

Кейс 3: web-hook для создания промокодов

- Функция принимает вызовы от сторонней системы и вызывает нашу систему по WebApi.
  Код сторонней системы мы не контролируем

- Холодный старт приводил к ошибкам и промокод не создавался

  - Стали использовать Dedicated план

Ответ на вопрос: 213.39 рублей

Итоги

 Быстро  Дёшево  Удобно

 Используйте isolated-process

 Низкий SLA вынуждает думать о фоллбэках и ретраях

 План Consumption не стоит использовать
для фронтенд-сервисов и интеграций через HTTP

Ссылки

- Официальная документация
- https://mikhail.io/ - сайт Михаила Шилкова
- Доклад Михаила Шилкова на DotNext 2019
- Организация Azure на github
- Официальный Twitter
- Официальный youtube-канал

Вопросы