Yandex Cloud

Serverless. Под капотом Cloud Functions

Максим Шошин

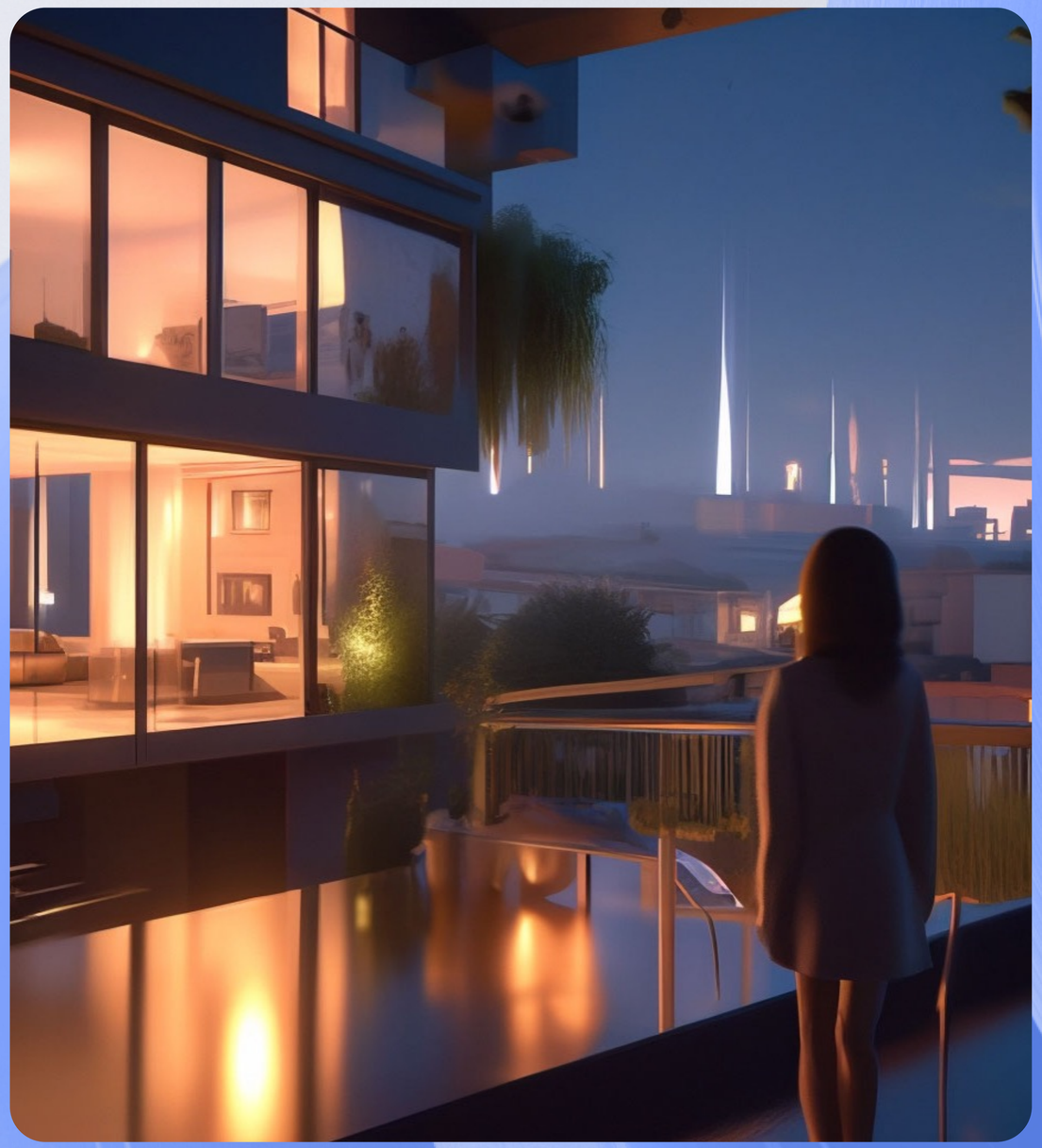
Yandex Cloud

Serverless. Под капотом Cloud Functions

Максим Шошин

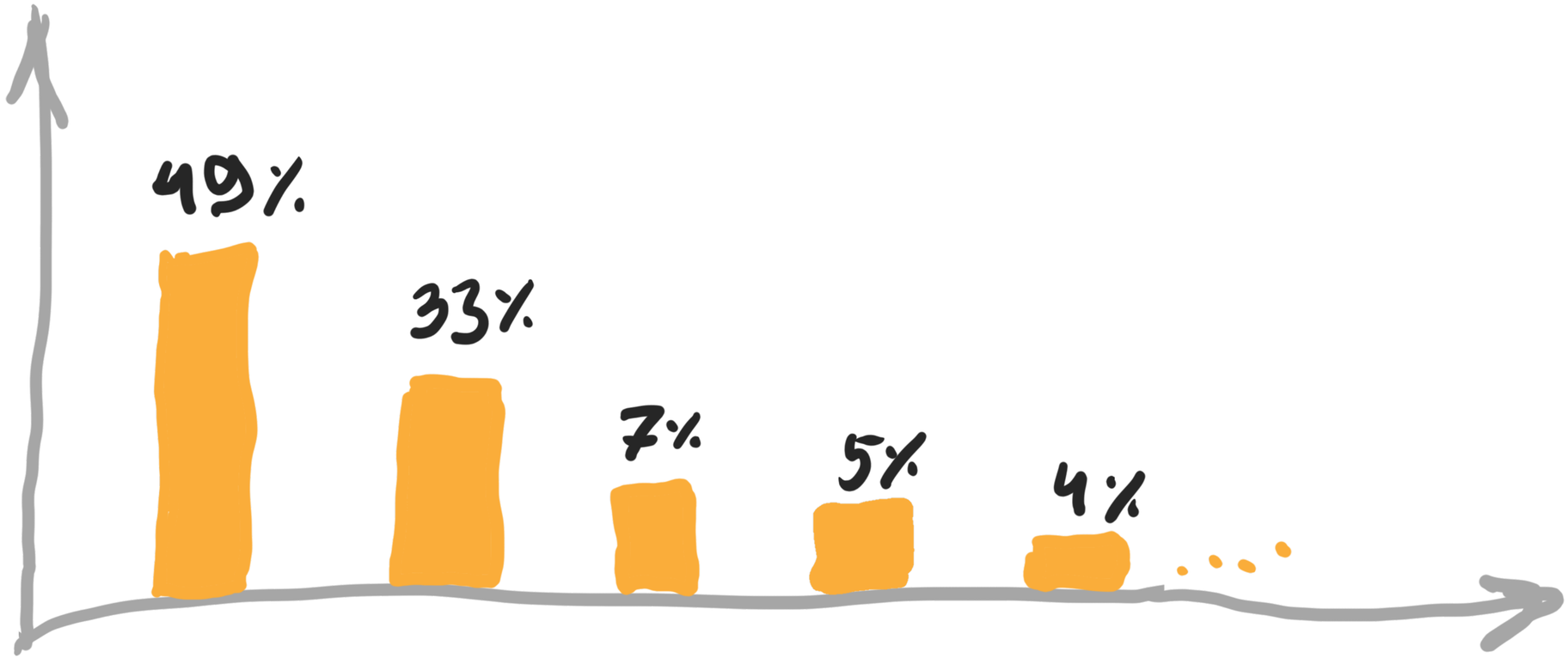
Yandex Cloud

Serverless. Под капотом Cloud Functions

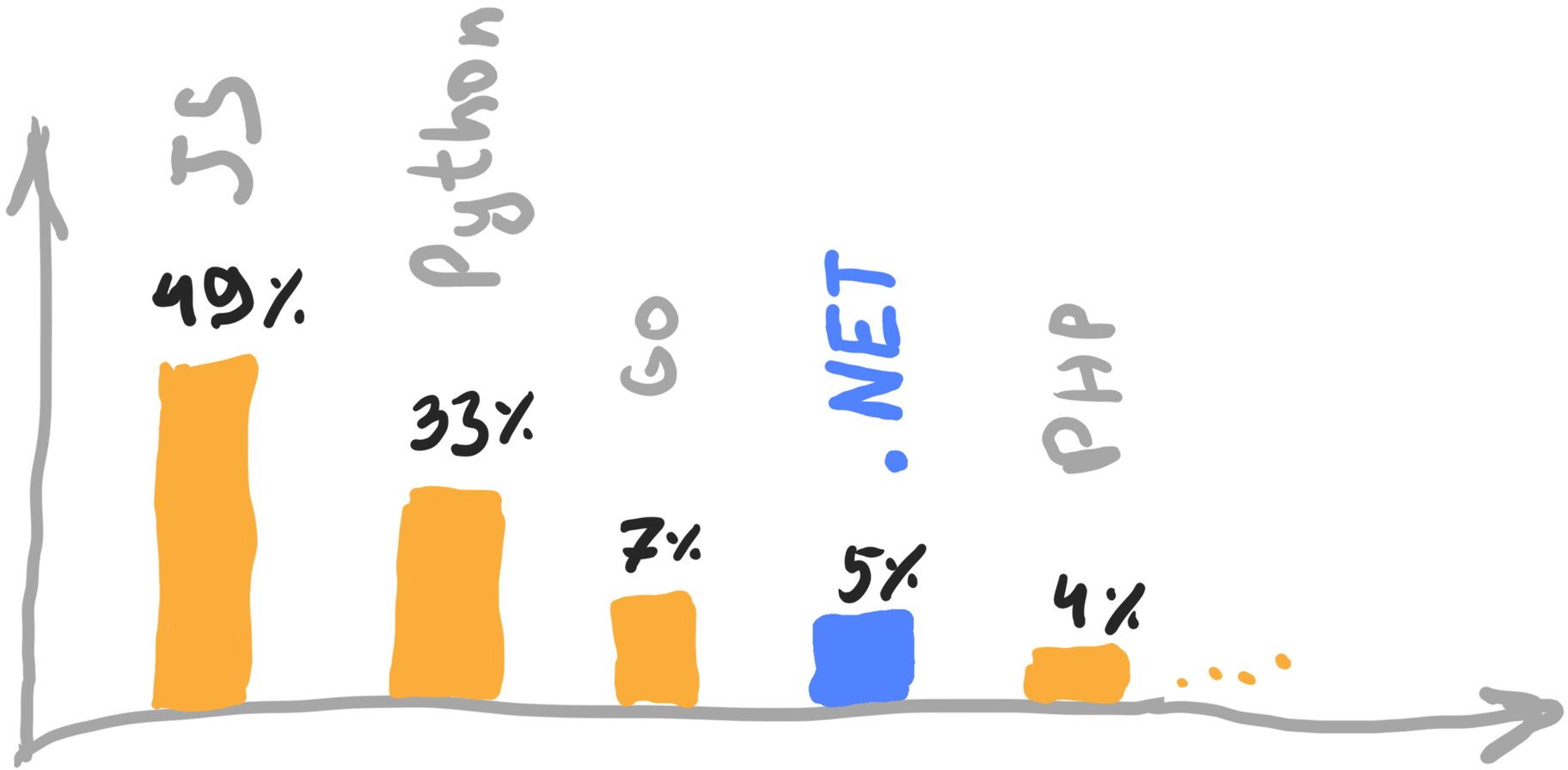


Максим Шошин

Использование рантаймов



Использование рантаймов



1. Что такое Serverless

2. Как работают
   Cloud Functions

3. Как устроены
   Cloud Functions

Всё, о чем говорю дальше, — верно сегодня

Ссылки

- Картинки сгенерированы с помощью shedevrum.ai
- Схемы нарисованы мной

https://maxshoshin.ru/dotnetru2023.html

Ссылки

- Картинки сгенерированы с помощью shedevrum.ai
- Схемы нарисованы мной

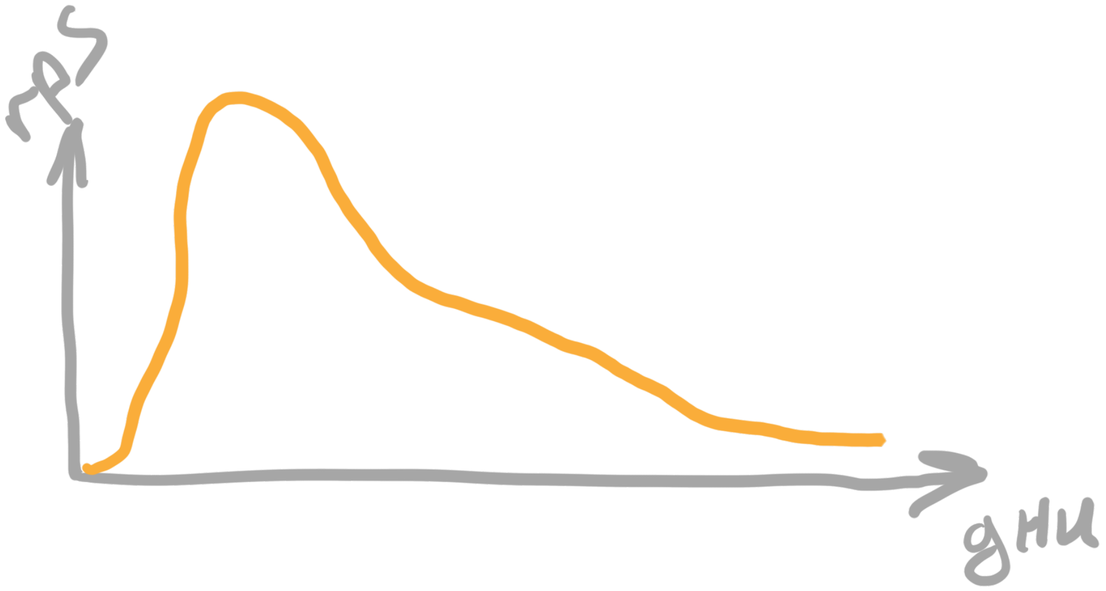
https://maxshoshin.ru/dotnetru2023.html

1. Что такое Serverless

2. Как работают Cloud Functions

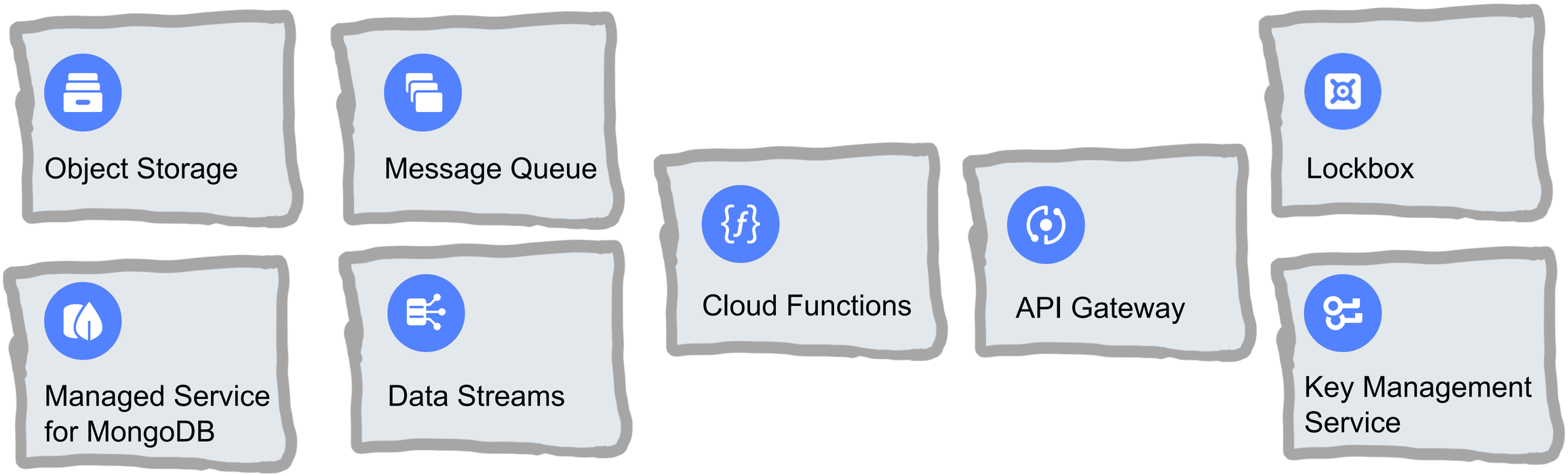
3. Как устроены Cloud Functions

Экзамен ПДД от Авто.ру

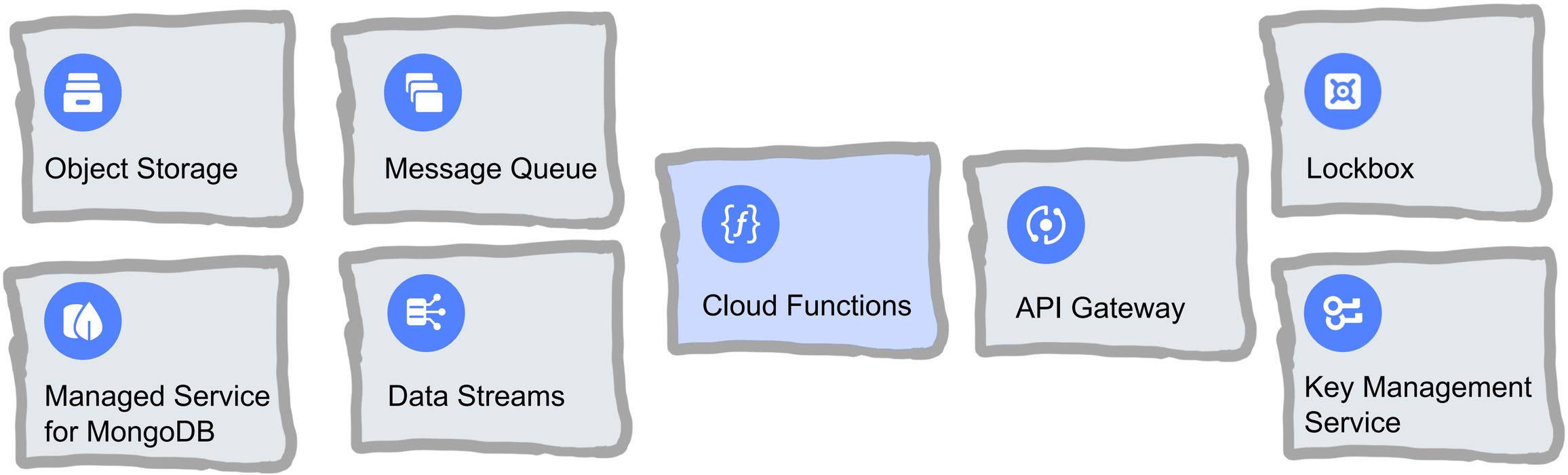


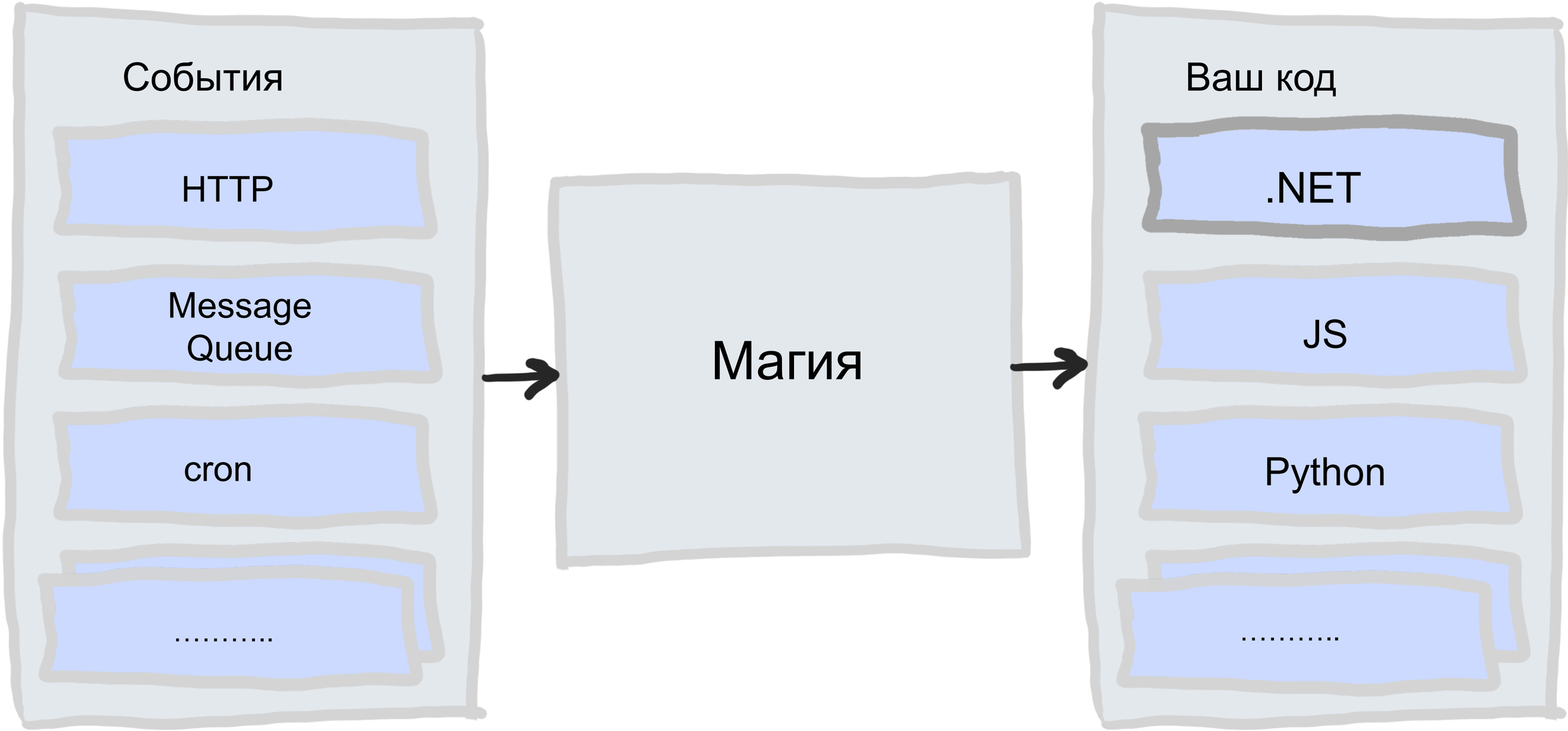
https://pdd.auto.ru/#stats

Serverless


Object Storage
Message Queue
Lockbox
Cloud Functions
API Gateway
Managed Service for MongoDB
Data Streams
Key Management Service

Serverless


Object Storage
Message Queue
Lockbox
Cloud Functions
API Gateway
Managed Service for MongoDB
Data Streams
Key Management Service


События
HTTP
Message Queue
cron
.....
Магия
Ваш код
.NET
JS
Python
.....

Создаём функцию

```
public class Handler
{
  public async Task<Response> FunctionHandler(HttpRequest req)
  {
      ...
  }
}
```

Функция. Вход и выход

```
public class Handler
{
  public async Task<Response> FunctionHandler(HttpRequest req)
  {
     ...
  }
}
```

Функция. Вход и выход

```
public class HttpRequest
{
    [JsonPropertyName("headers")]
    public Dictionary<string, string> Headers;

    [JsonPropertyName("body")]
    public string Body;
}
```

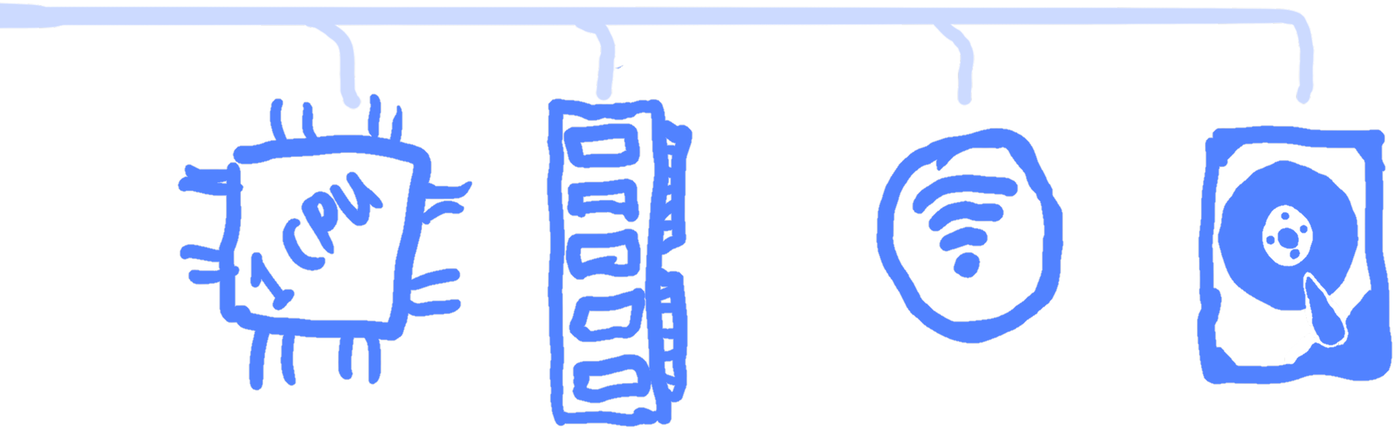
Функция. Вход и выход

```
public class Response
{
    [JsonPropertyName("statusCode")]
    public int StatusCode;

    [JsonPropertyName("body")]
    public string Body;
}
```

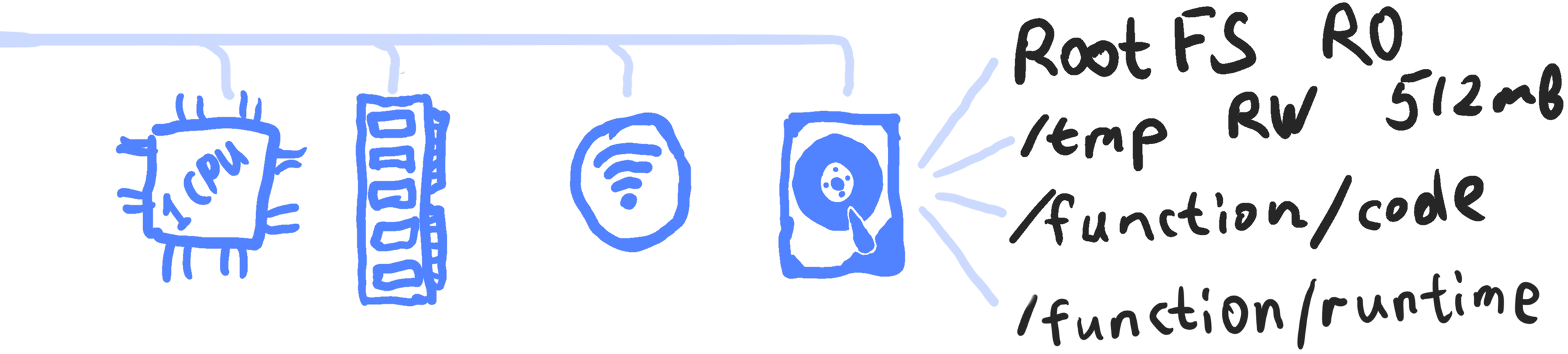
Функция работает, пока исполняется метод

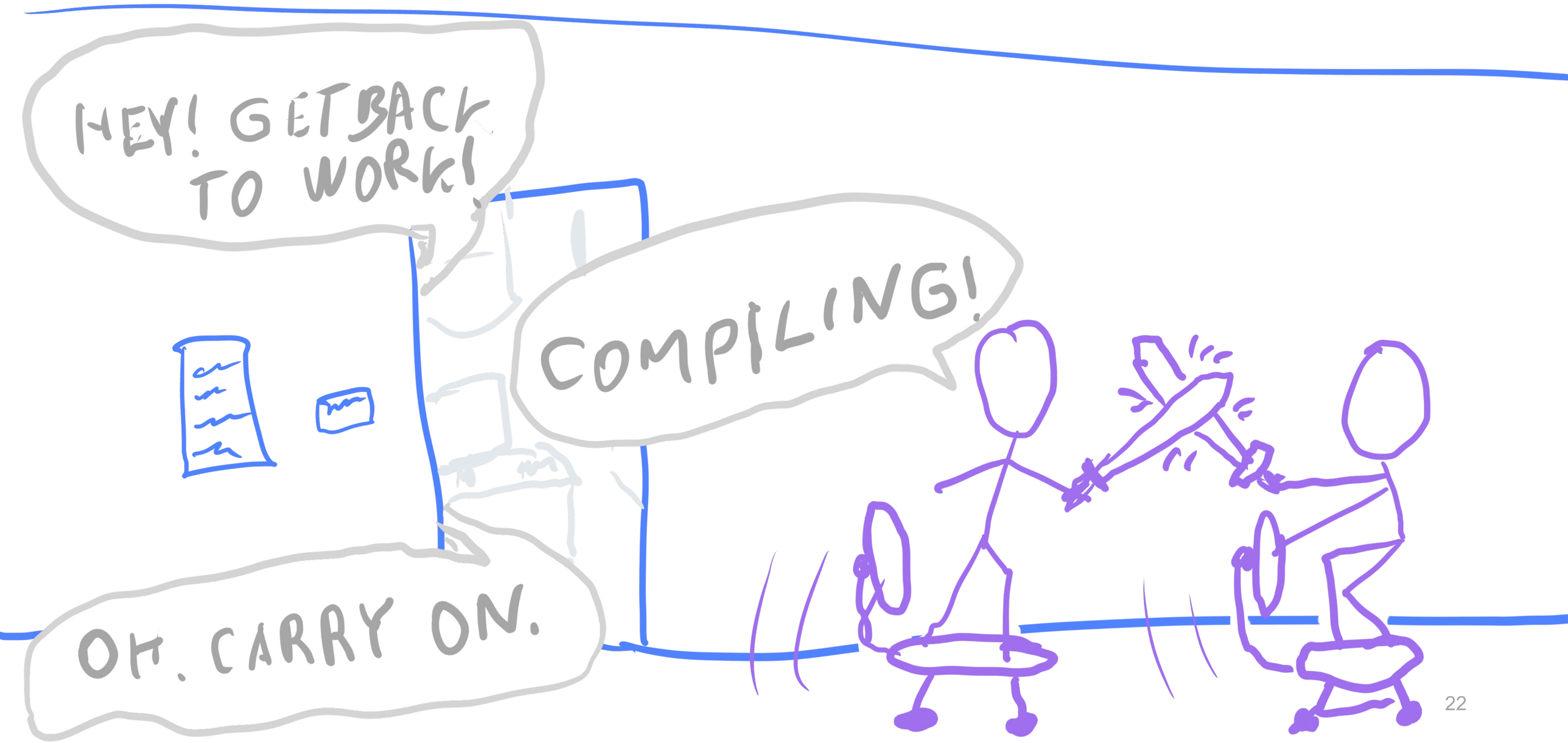
```
public class Handler
{
    public async Task<Response> FunctionHandler(HttpRequest req)
    {
        ...
    }
}
```

Что есть у функции

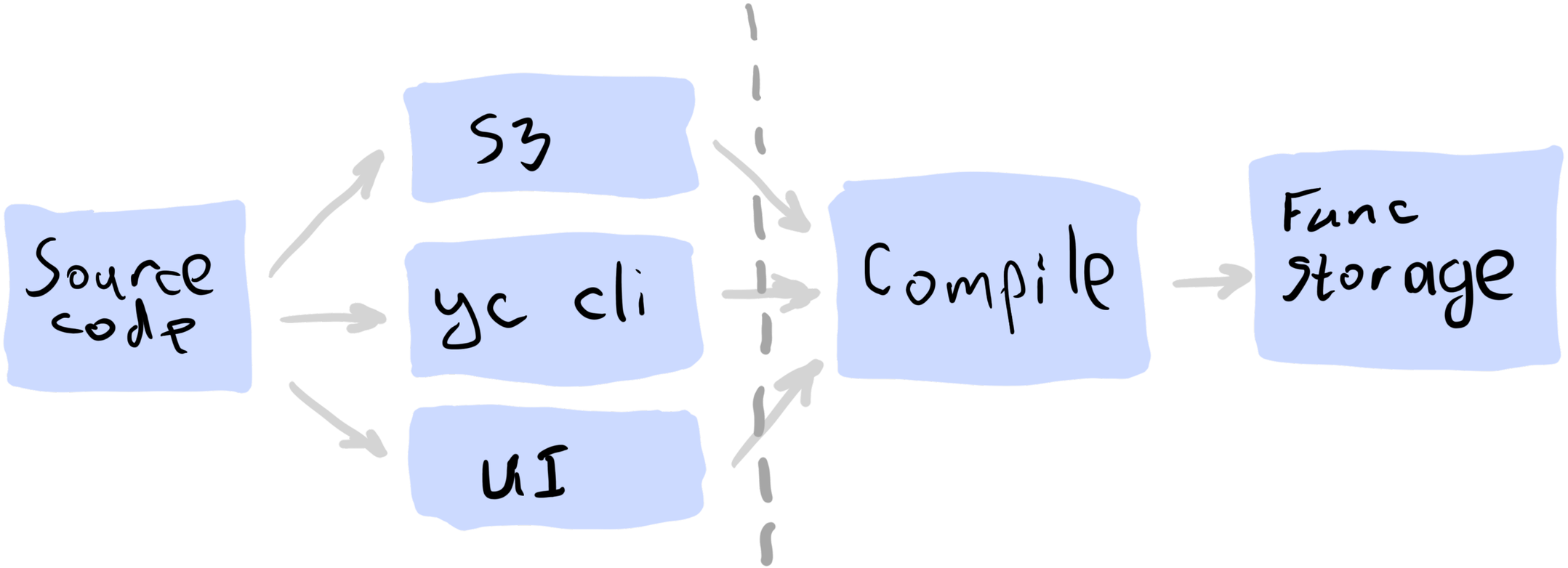


Что есть у функции

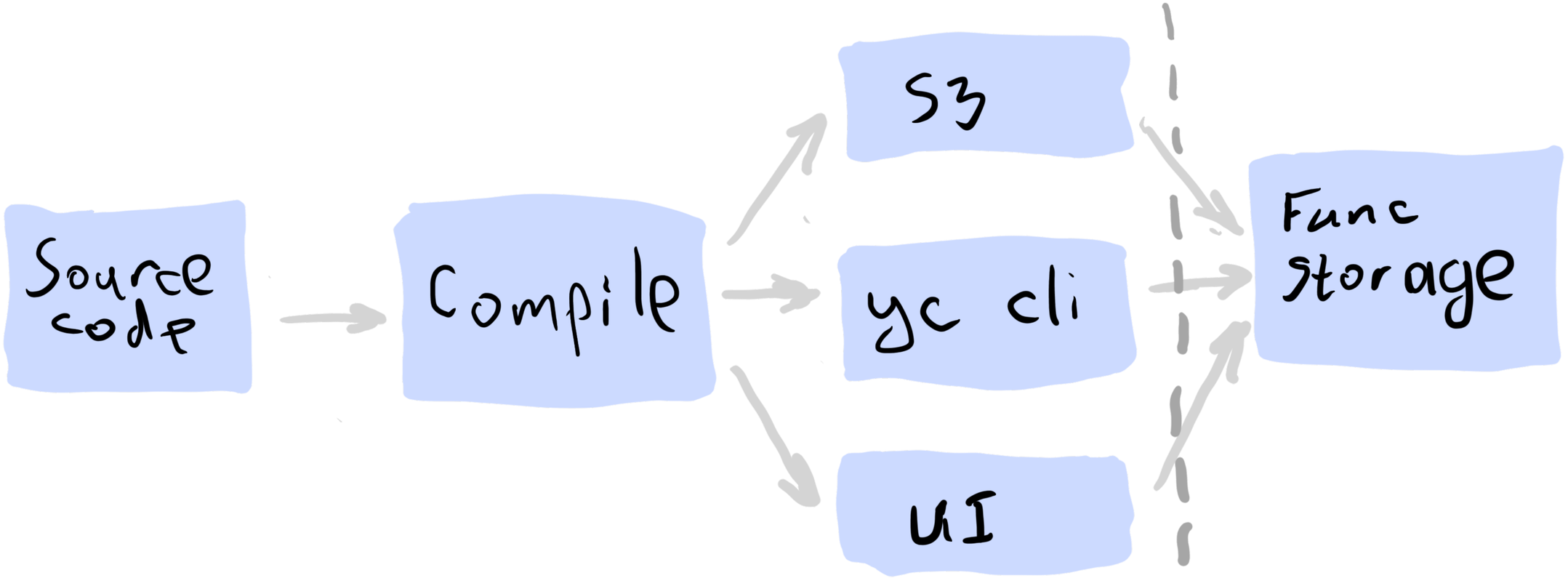


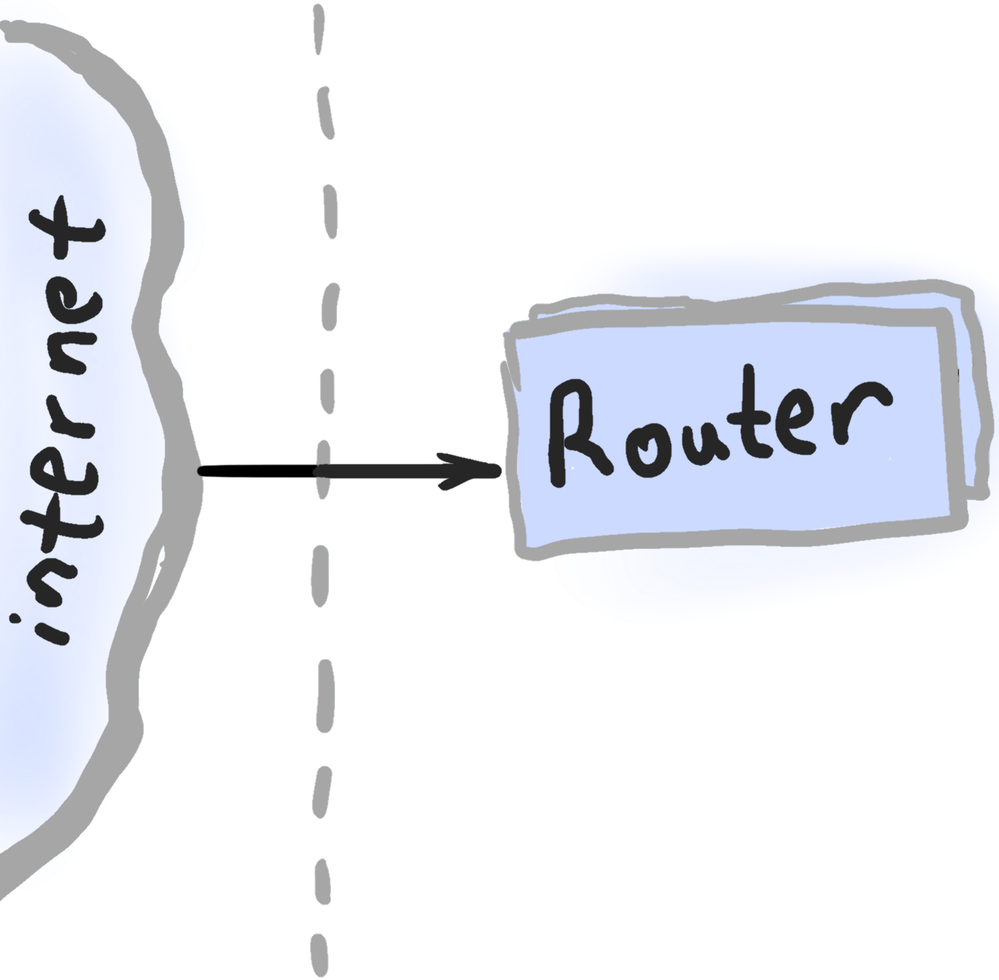

HEY! GET BACK TO WORK!
COMPILING!
OH. CARRY ON.

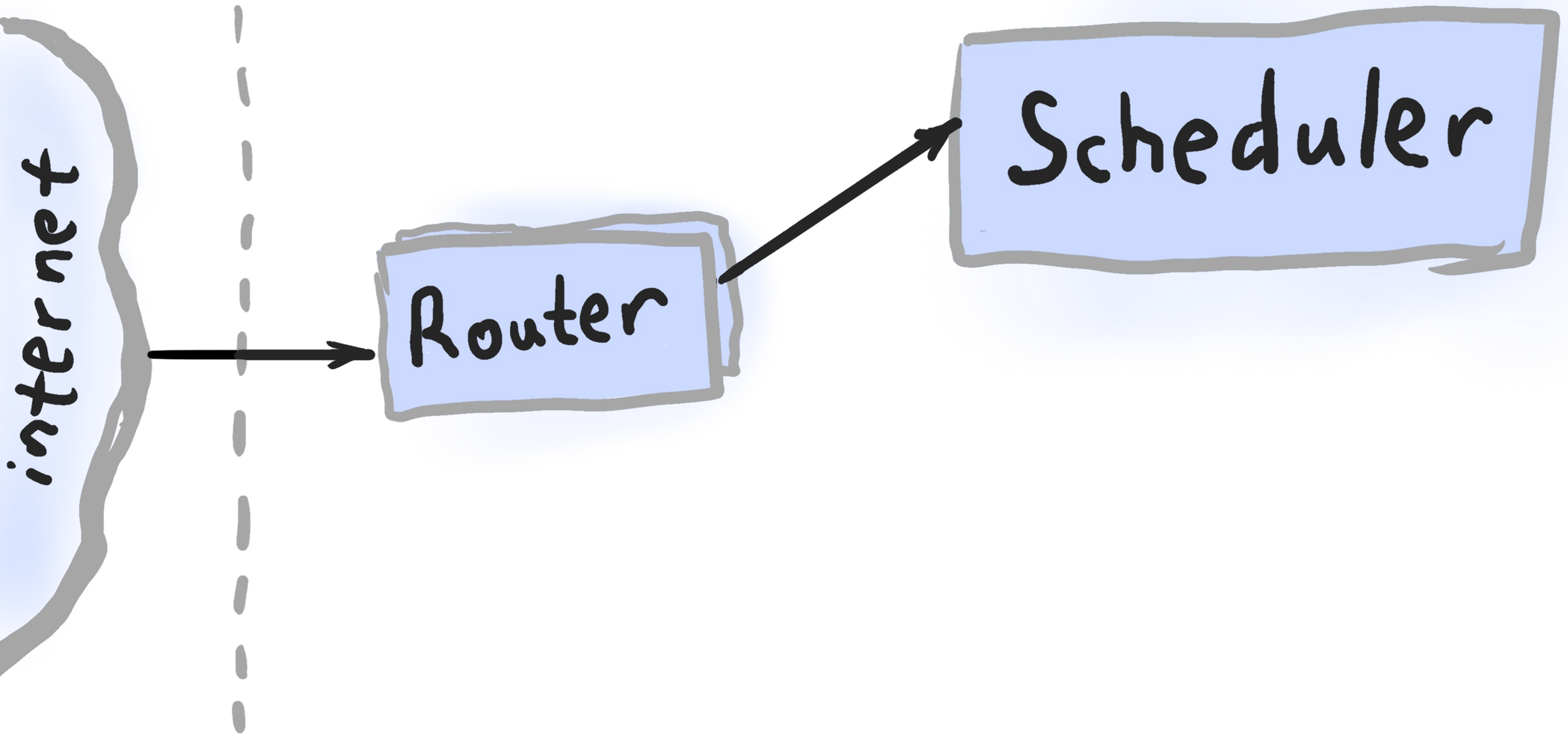
Деплой

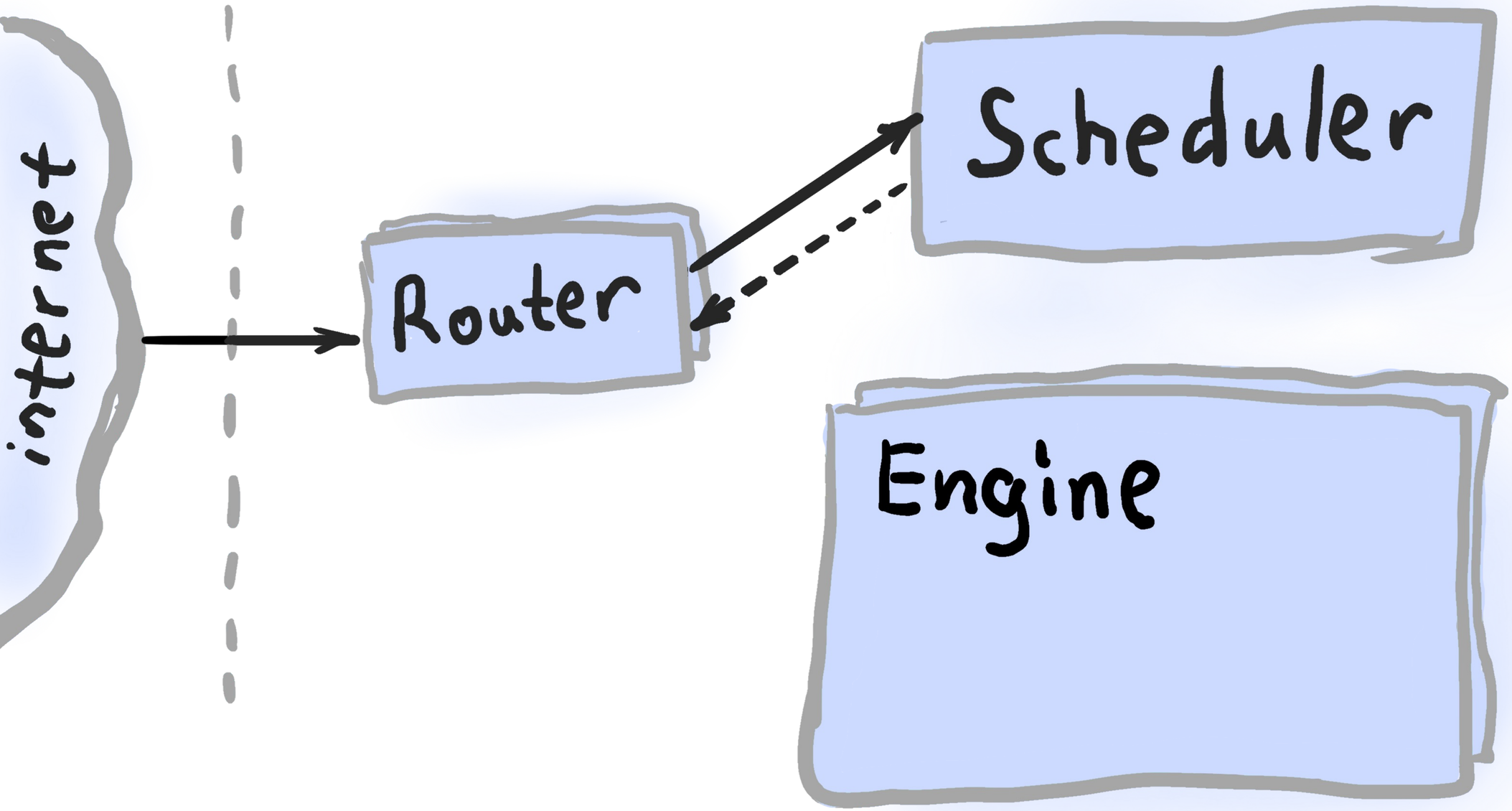


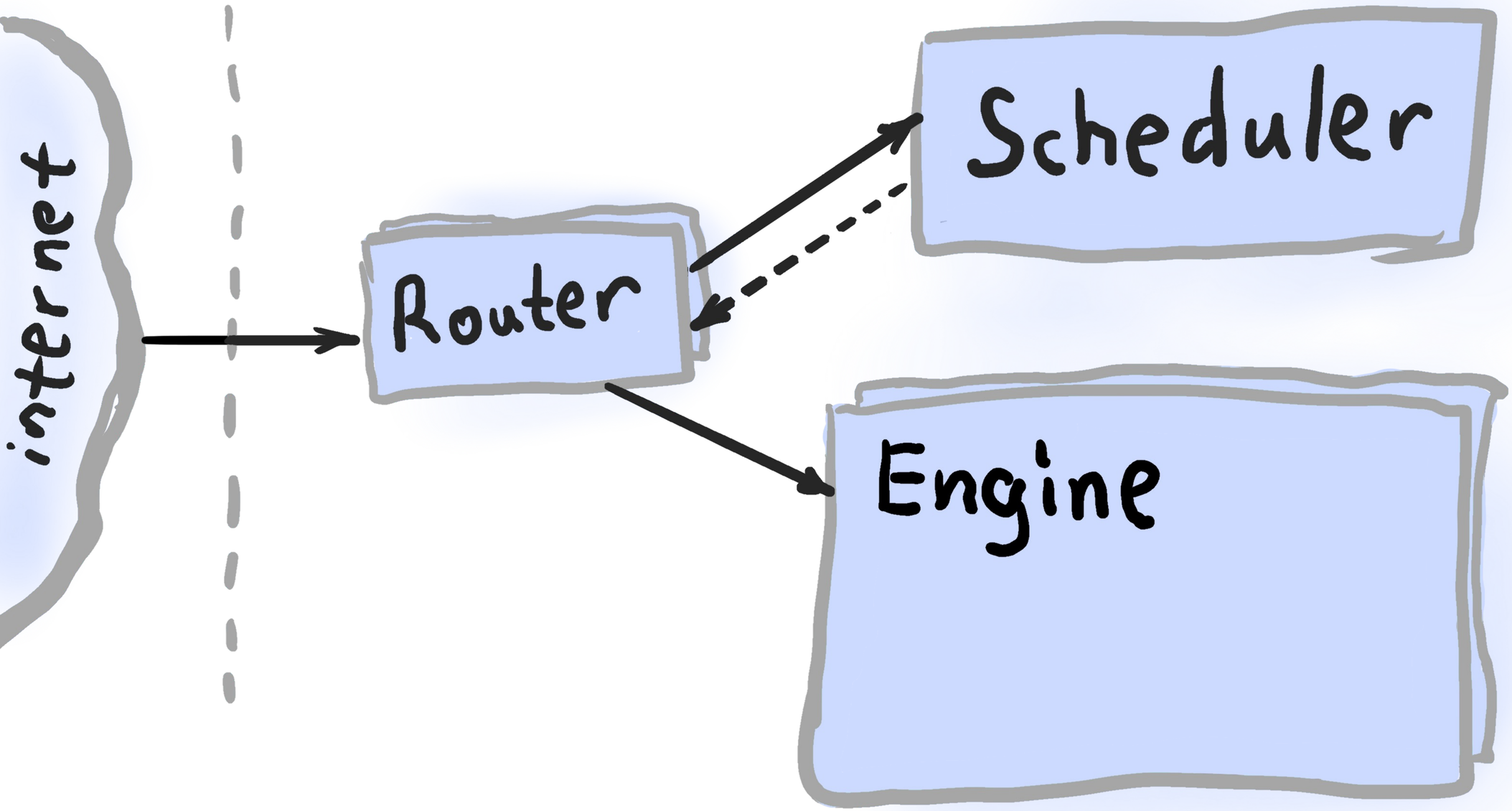
Деплой

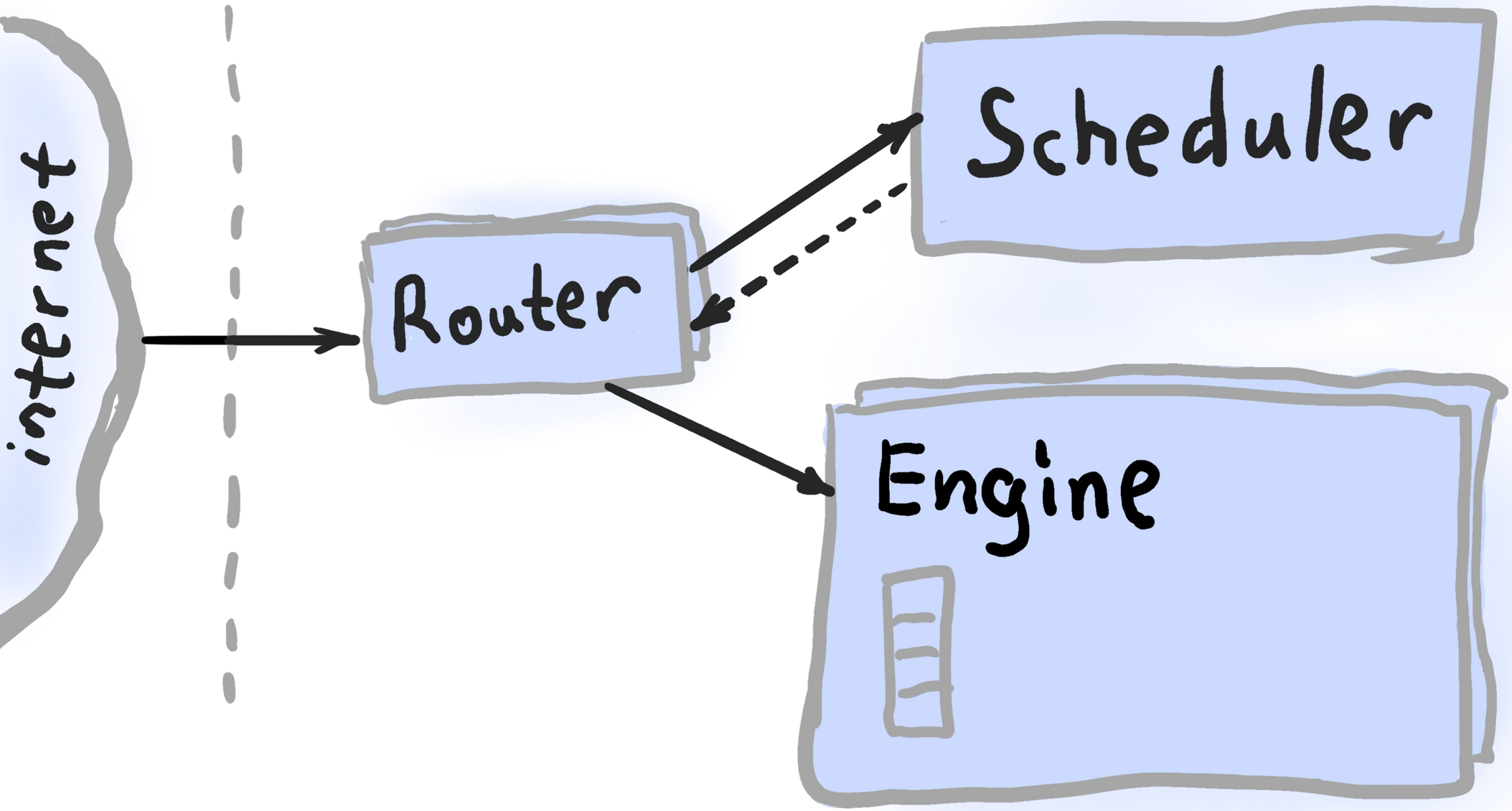


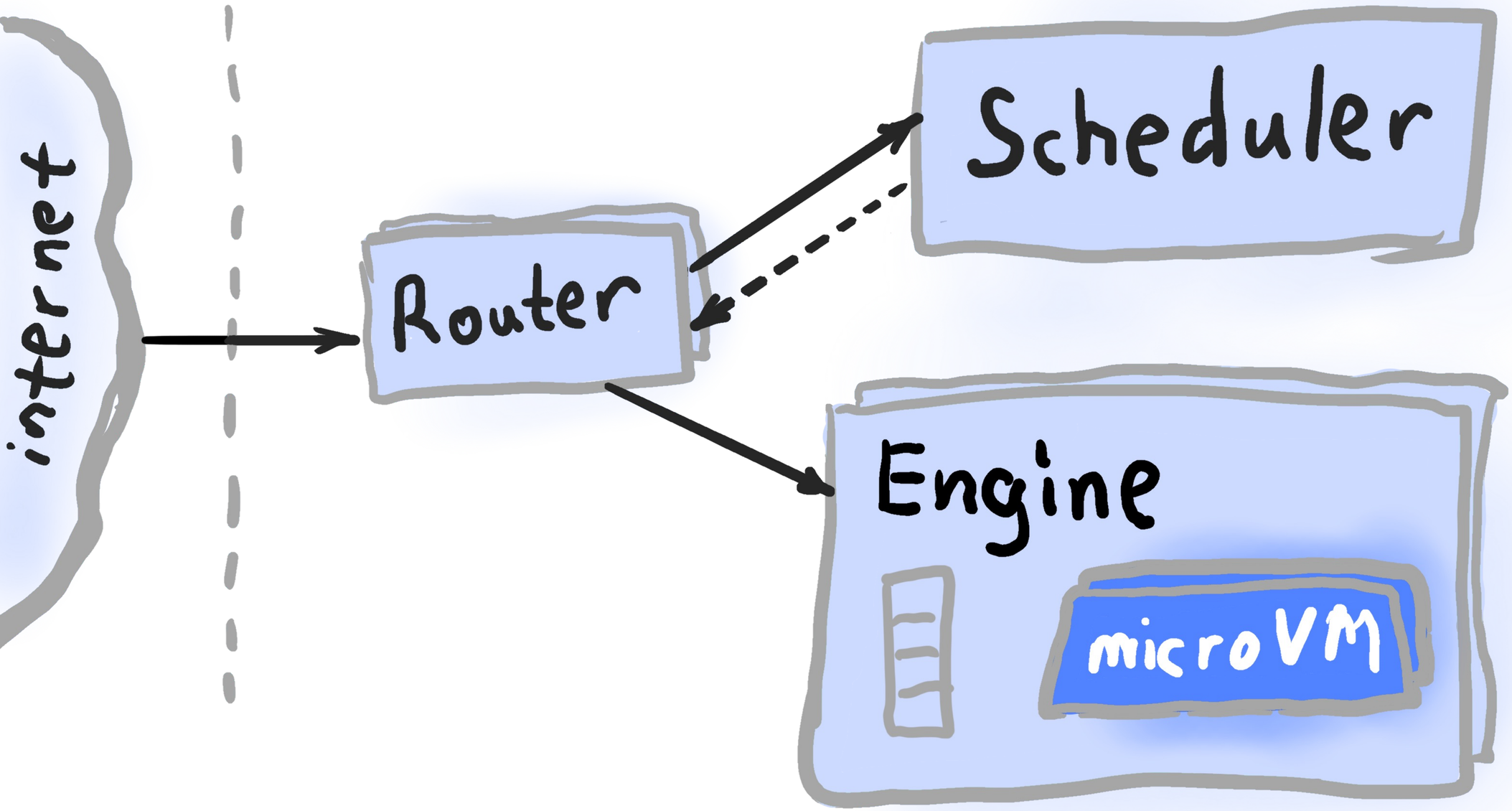

internet
Router

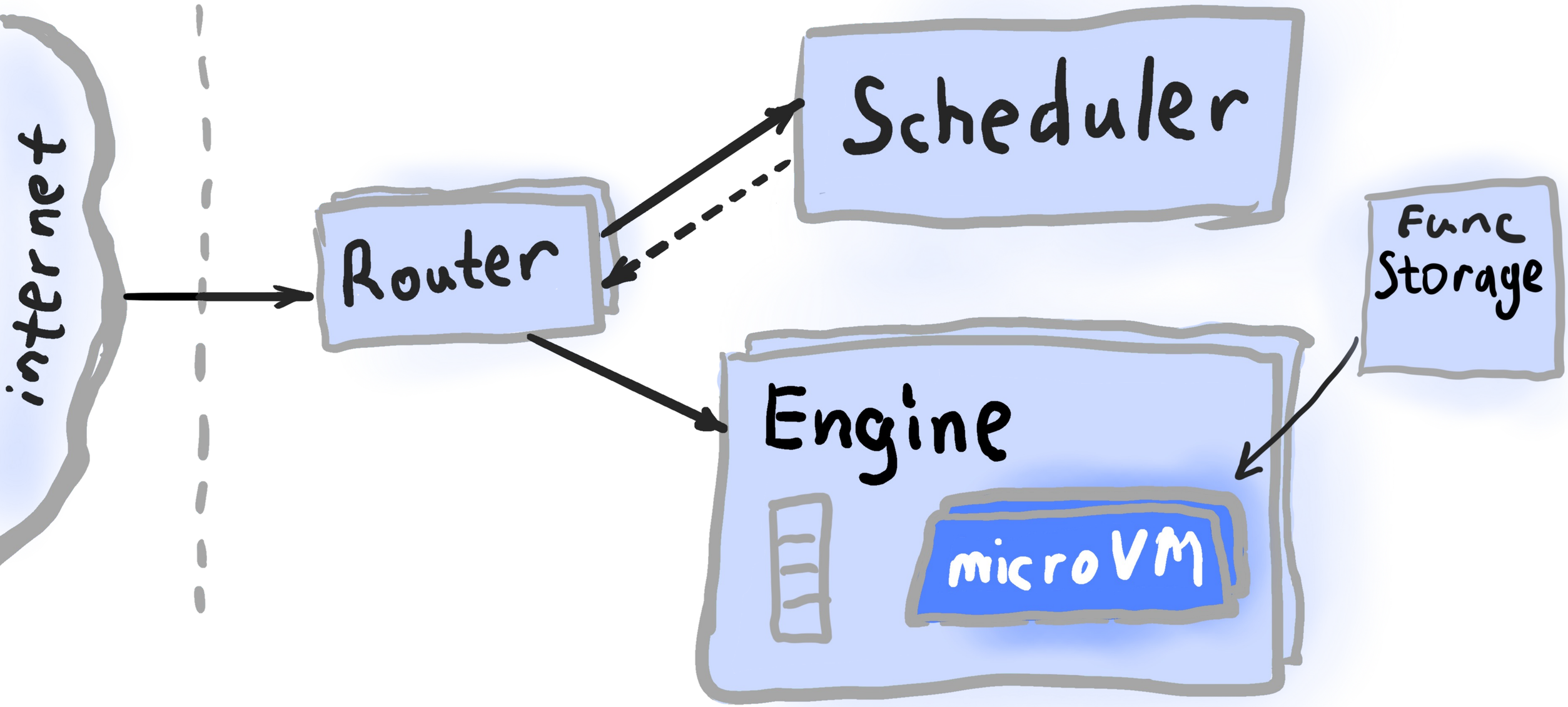

internet
Router
Scheduler

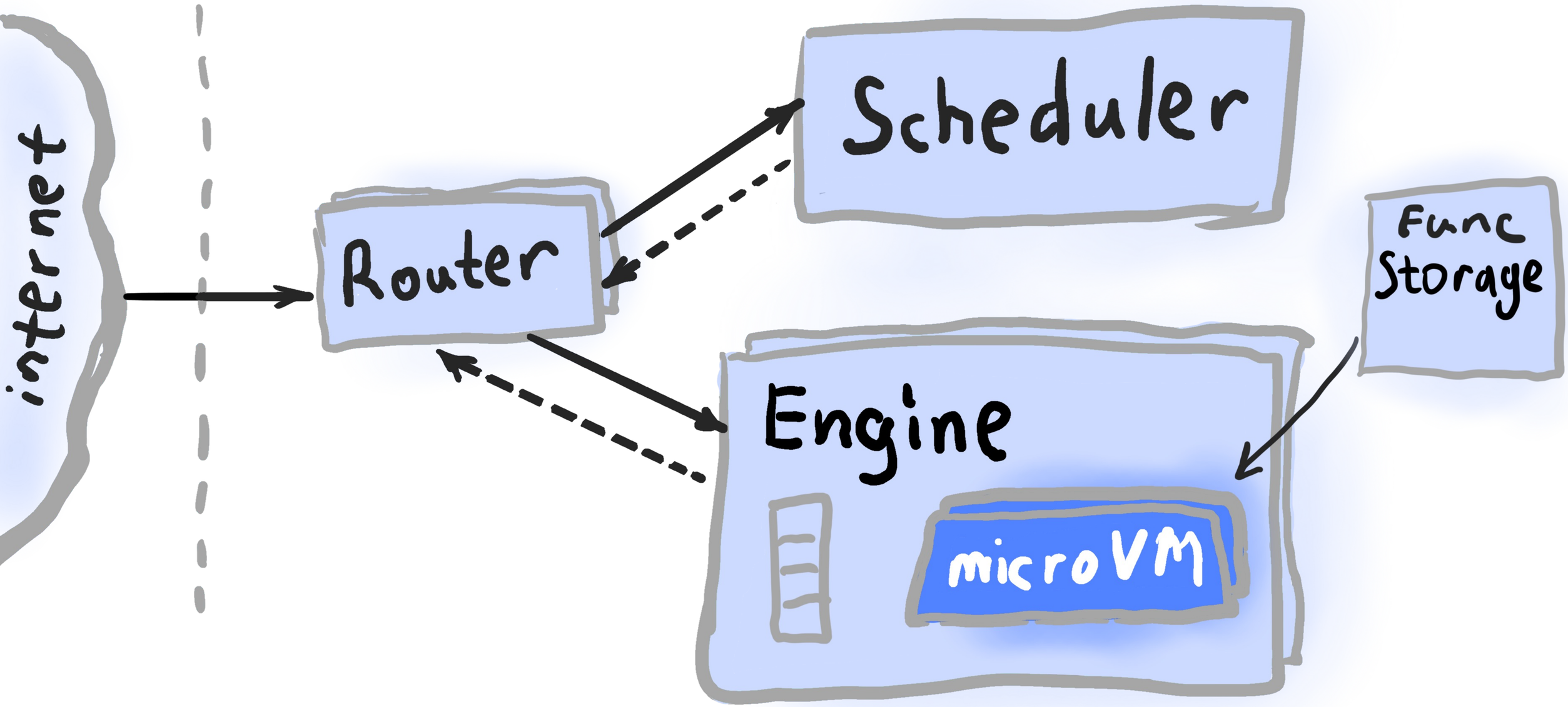

internet
Router
Scheduler
Engine

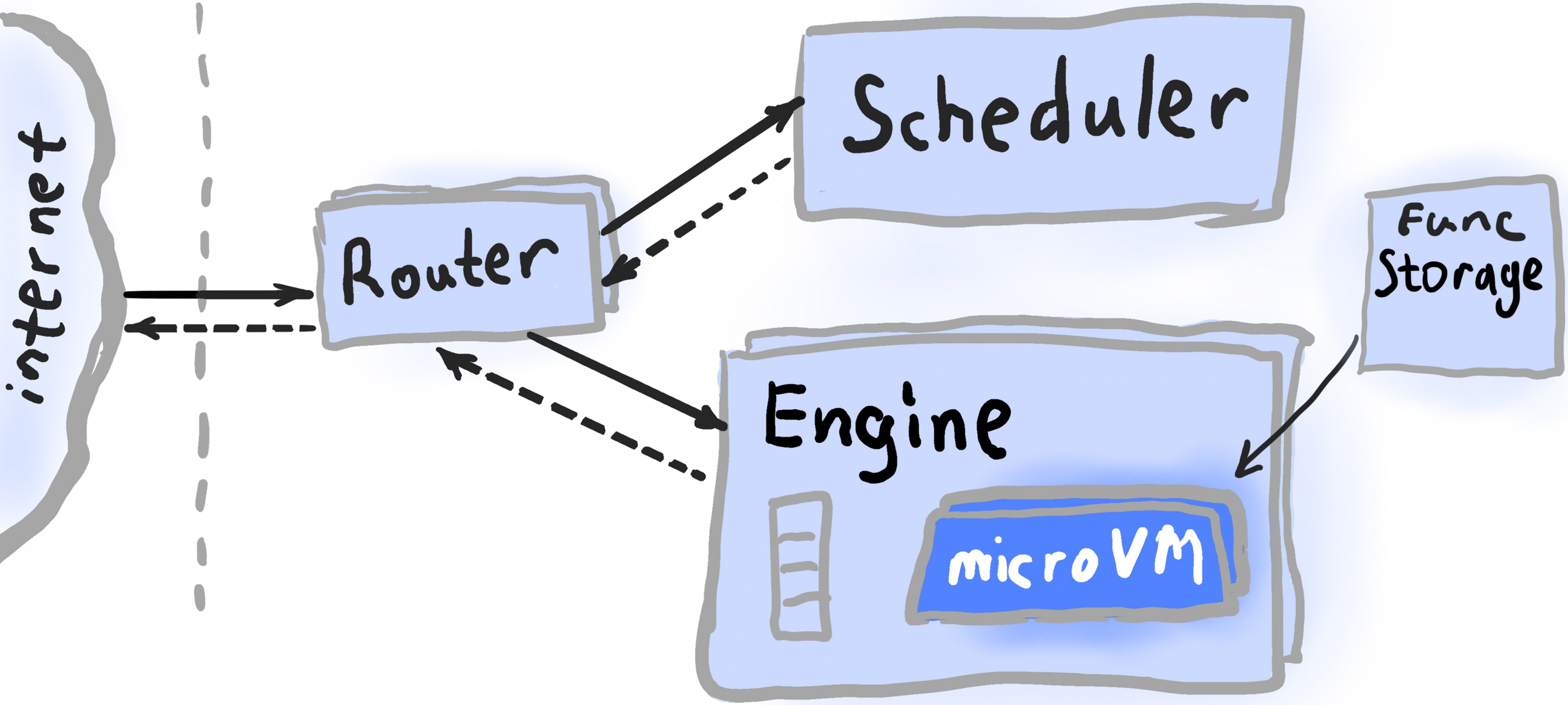

internet
Router
Scheduler
Engine


internet
Router
Scheduler
Engine


internet
Router
Scheduler
Engine
microVM


internet
Router
Scheduler
Func
Storage
Engine
microVM


internet
Router
Scheduler
Func
Storage
Engine
microVM


internet
Router
Scheduler
Func
Storage
Engine
microVM

Ограничения

- **Нет background-работ**

Ограничения

- **Нет background-работ**

  Асинхронная задача

  Периодическая задача

Ограничения

- **Нет background-работ**

  Асинхронная задача — Message Queue

  Периодическая задача

Ограничения

- **Нет background-работ**

  Асинхронная задача — Message Queue

  Периодическая задача — Cron Trigger

Ограничения

- **Нет background-работ**

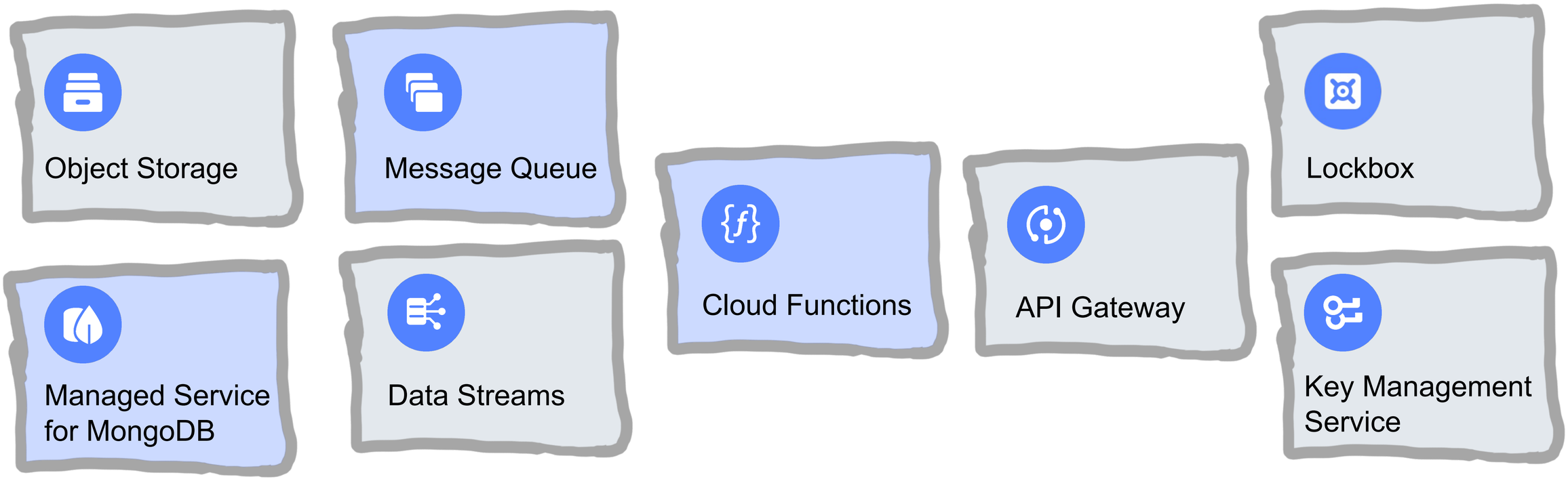
  Асинхронная задача — Message Queue

  Периодическая задача — Cron Trigger

- **Нет состояния**

  Храним в отдельных сервисах

Ограничения



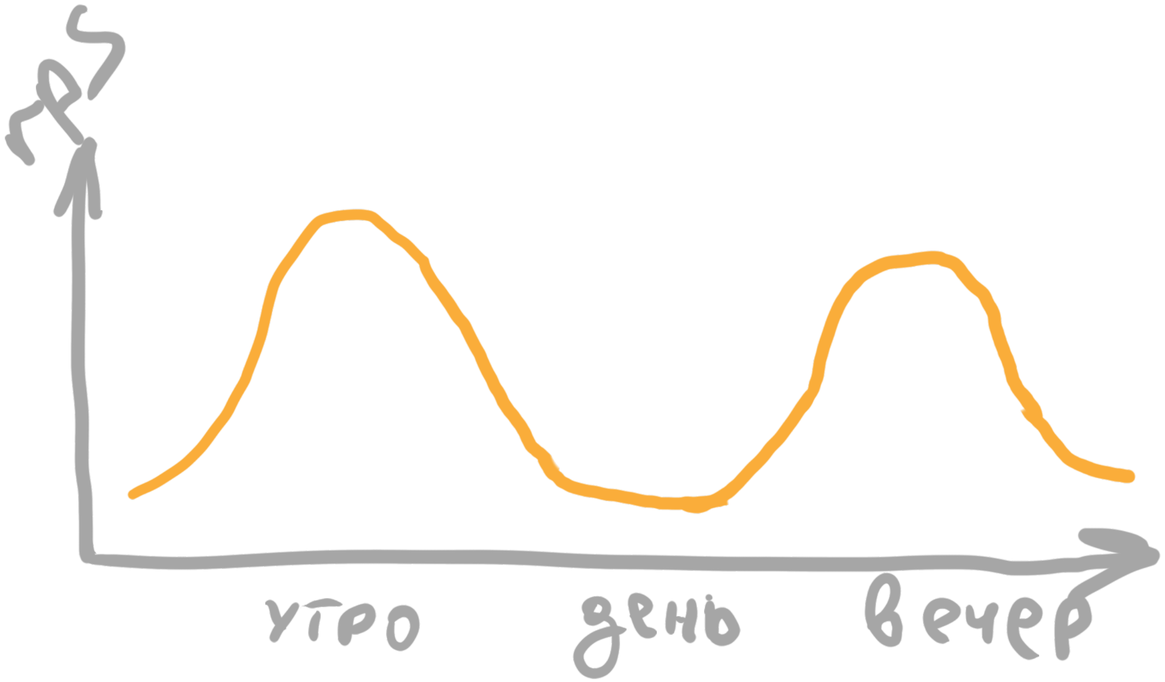
1. Что такое Serverless
2. Как работают Cloud Functions
3. Как устроены Cloud Functions

В этом году Yandex Cloud 5 лет

2019 — появились первые Cloud Functions

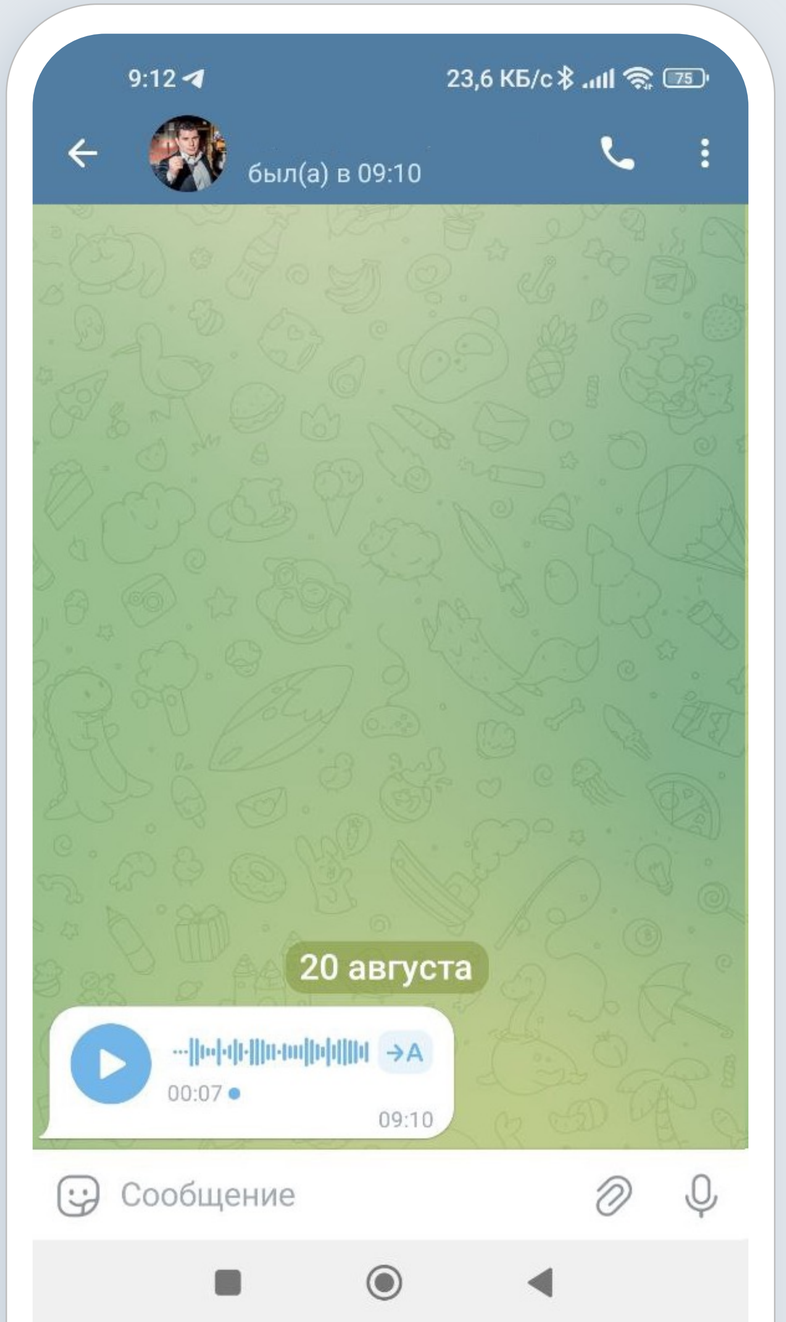
Вначале самым массовым сценарием были навыки Алисы

Навыки Алисы

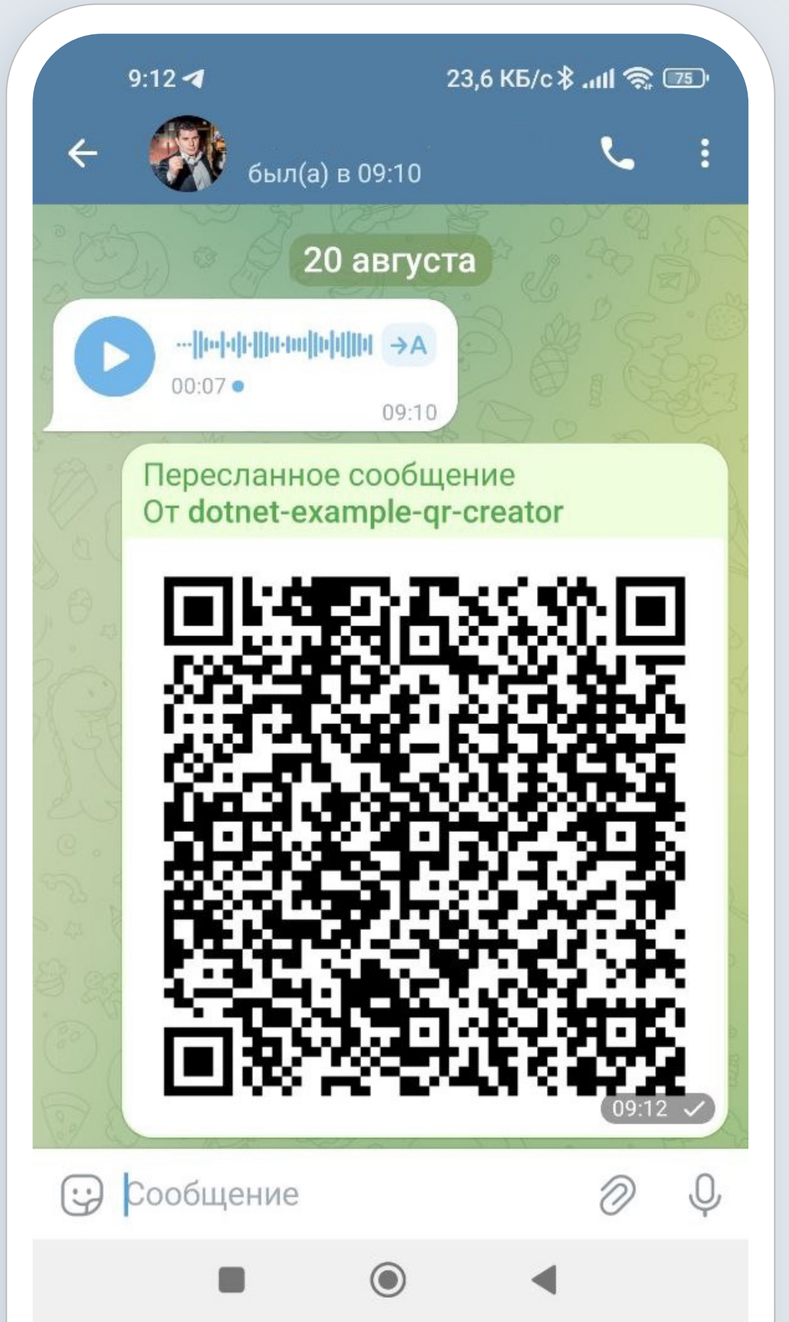


Кому проиграла Алиса?

Чат-боты

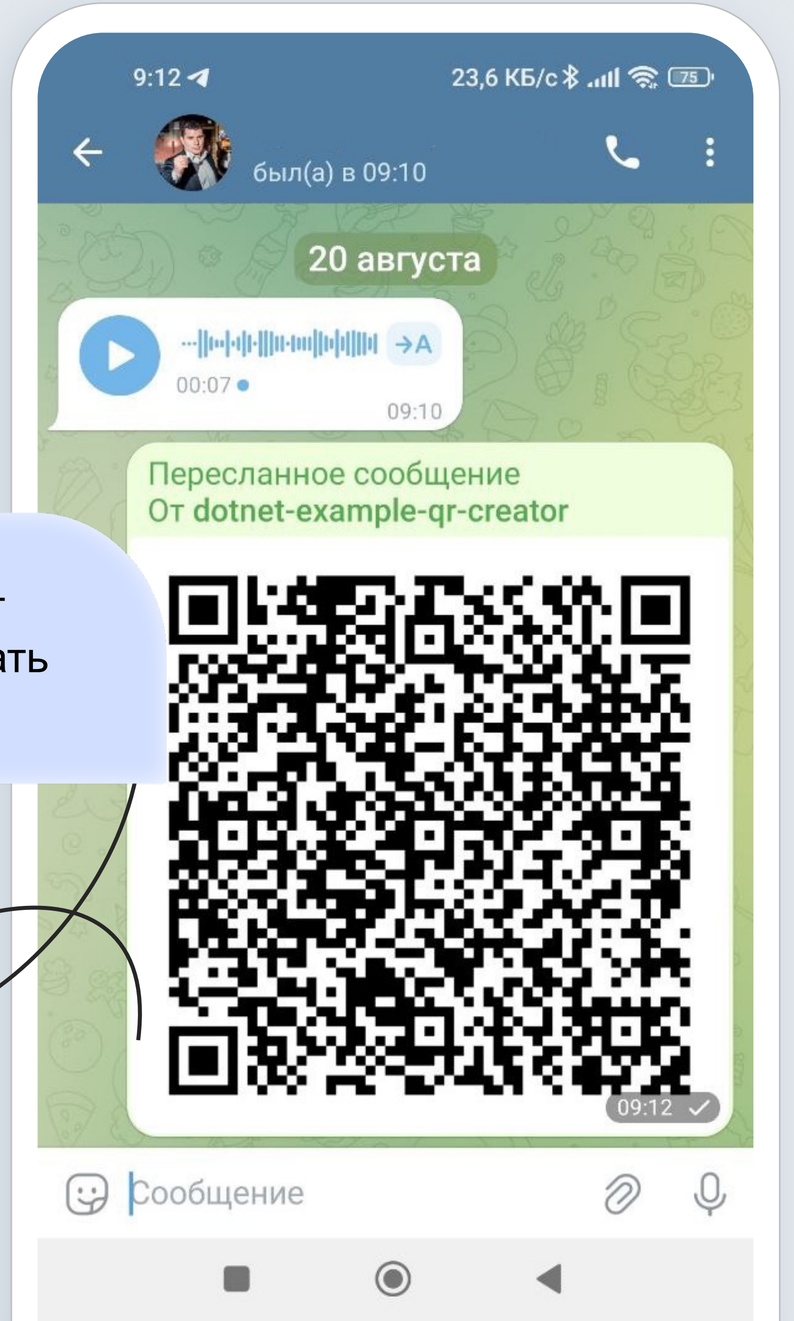


Чат-боты


9:12
23,6 КБ/с
был(а) в 09:10
20 августа
00:07
09:10
Пересланное сообщение
От dotnet-example-qr-creator
09:12
Сообщение

Чат-боты

Извините, у меня нет возможности прослушать ваше сообщение



Функция. Десериализация. Telegram

```
public async Task<Response> FunctionHandler(HttpRequest req)
{
    var msg = serializer.Deserialize<MessageUpdate>(req.Body);
    if (msg.Data is TextMessage txtMsg)
    {
        ...
    }

    ...
}
```

Функция. Десериализация. Telegram

```
public async Task<Response> FunctionHandler(HttpRequest req)
{
    var msg = serializer.Deserialize<MessageUpdate>(req.Body);
    if (msg.Data is TextMessage txtMsg)
    {
        ...
    }
    ...
}
```

Функция. Генерация QR

```
if (msg.Data is TextMessage txtMsg)
{
  var qrGenerator = new QRCodeGenerator();
  var data = qrGenerator
     .CreateQrCode(txtMsg.Text, QRCodeGenerator.ECCLevel.Q);

  var code = new PngByteQRCode(data);
  var stream = new MemoryStream(code.GetGraphic(20));

  await s_client.SendPhotoAsync(
      chatId, new InputFileStream(stream));
}
```

Функция. Генерация QR

```
if (msg.Data is TextMessage txtMsg)
{
  var qrGenerator = new QRCodeGenerator();
  var data = qrGenerator
     .CreateQrCode(txtMsg.Text, QRCodeGenerator.ECCLevel.Q);

  var code = new PngByteQRCode(data);
  var stream = new MemoryStream(code.GetGraphic(20));

  await s_client.SendPhotoAsync(
      chatId, new InputFileStream(stream));
}
```

Функция. Генерация QR

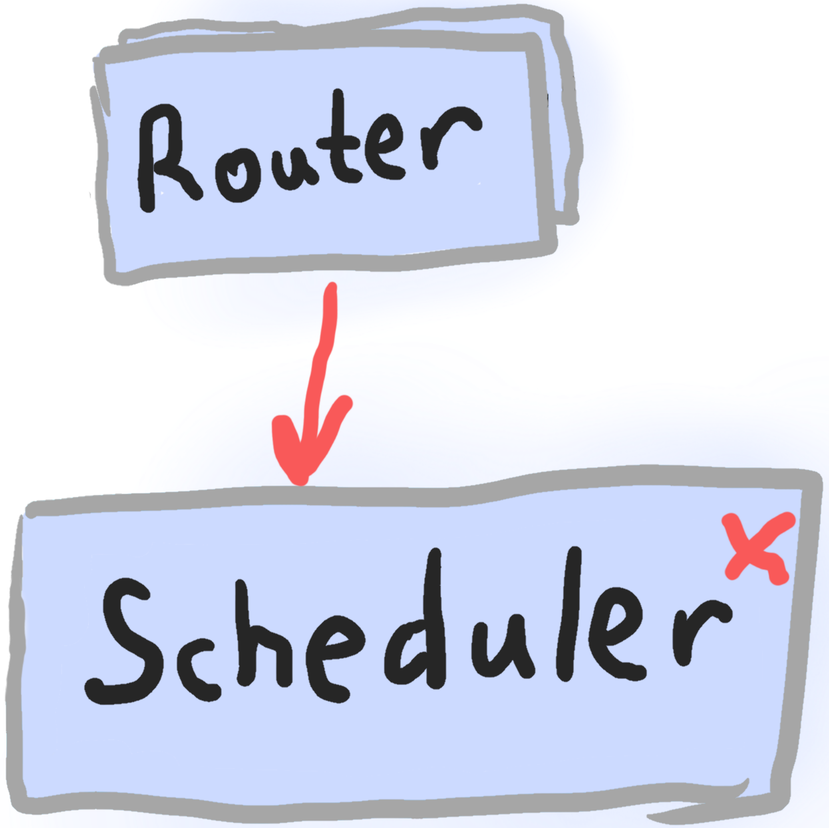
```
if (msg.Data is TextMessage txtMsg)
{
  var qrGenerator = new QRCodeGenerator();
  var data = qrGenerator
     .CreateQrCode(txtMsg.Text, QRCodeGenerator.ECCLevel.Q);

  var code = new PngByteQRCode(data);
  var stream = new MemoryStream(code.GetGraphic(20));

  await s_client.SendPhotoAsync(
      chatId, new InputFileStream(stream));
}
```

Примеры на GitHub

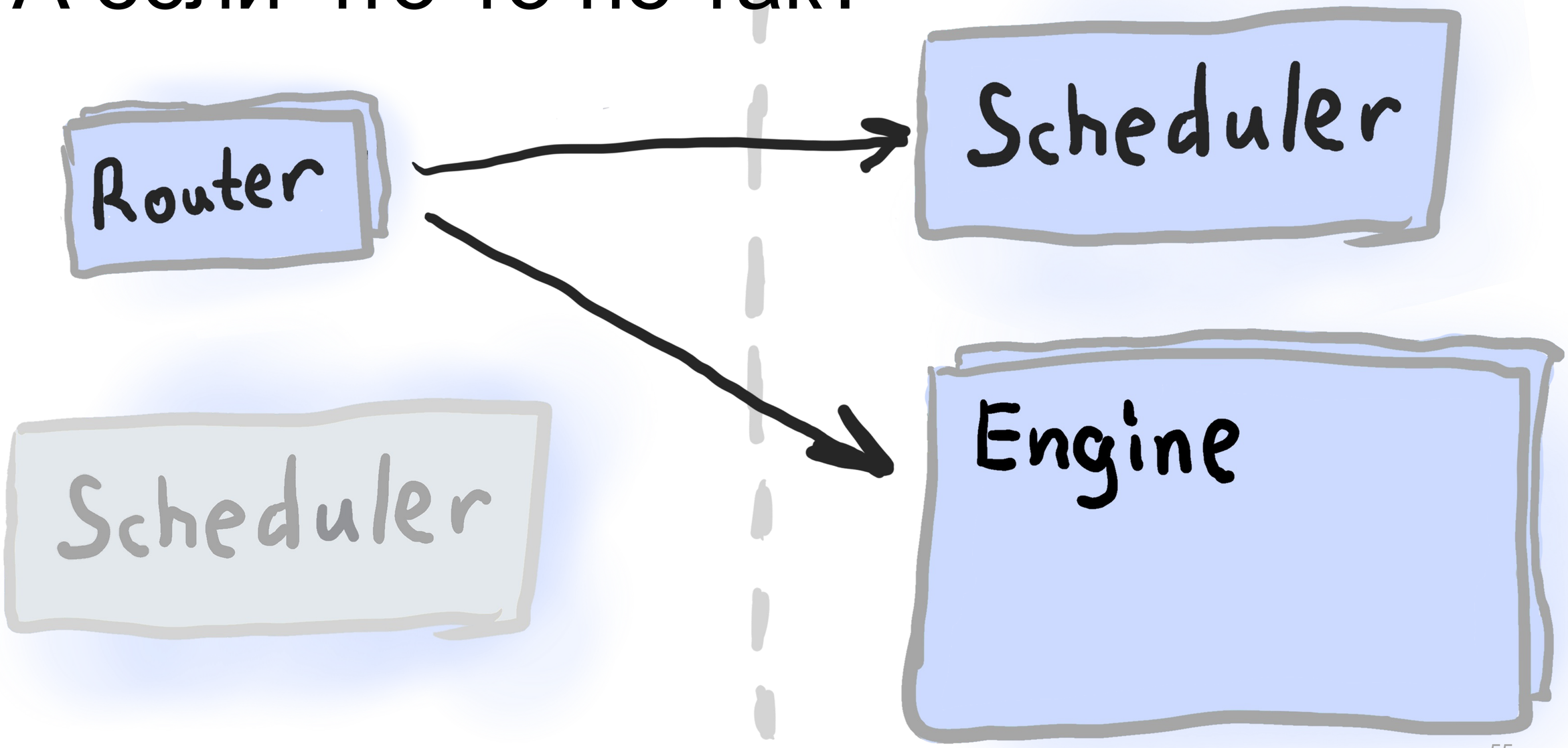
https://github.com/MaxShoshin/serverless-examples

А что если что-то пошло не так?

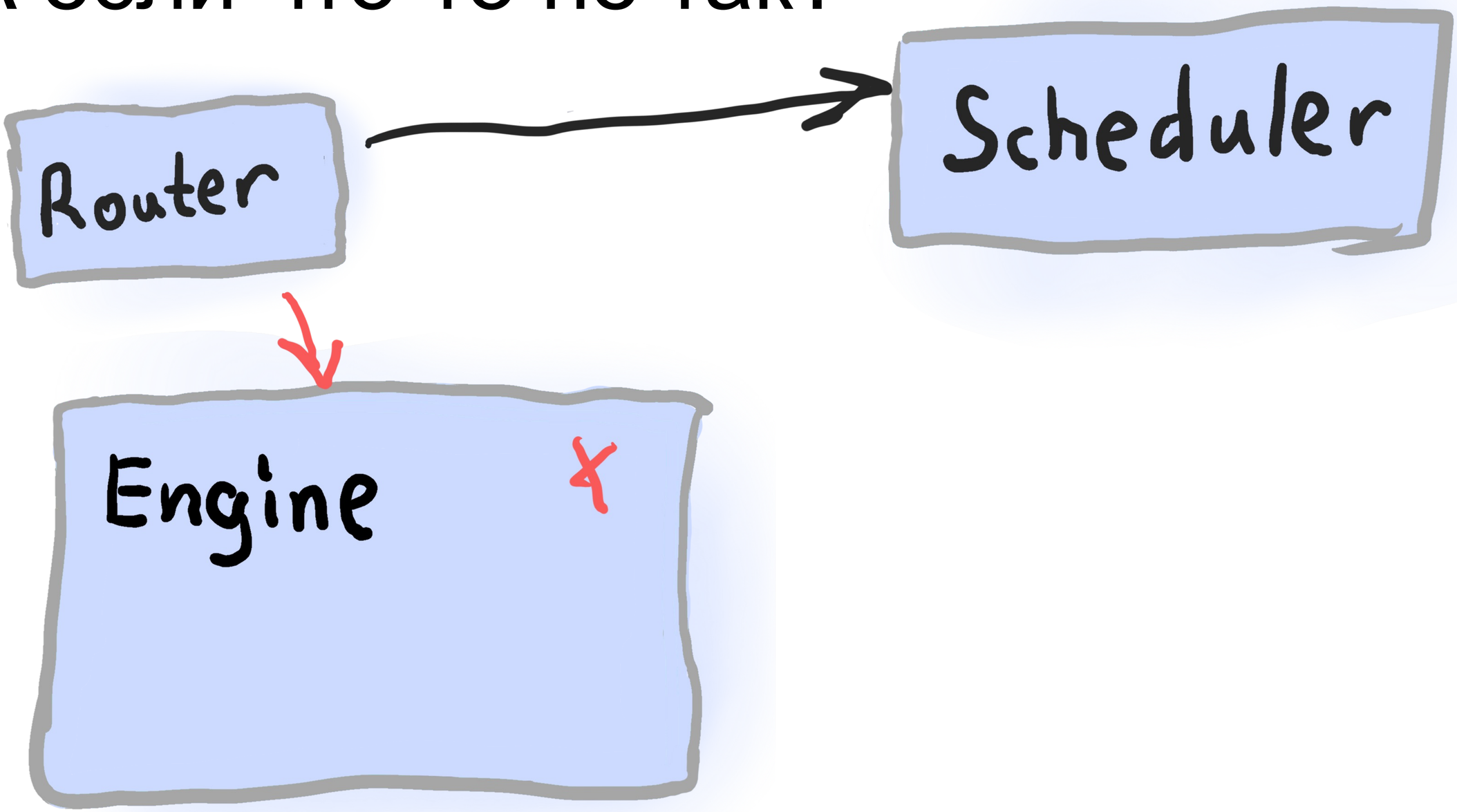
А если что-то не так?



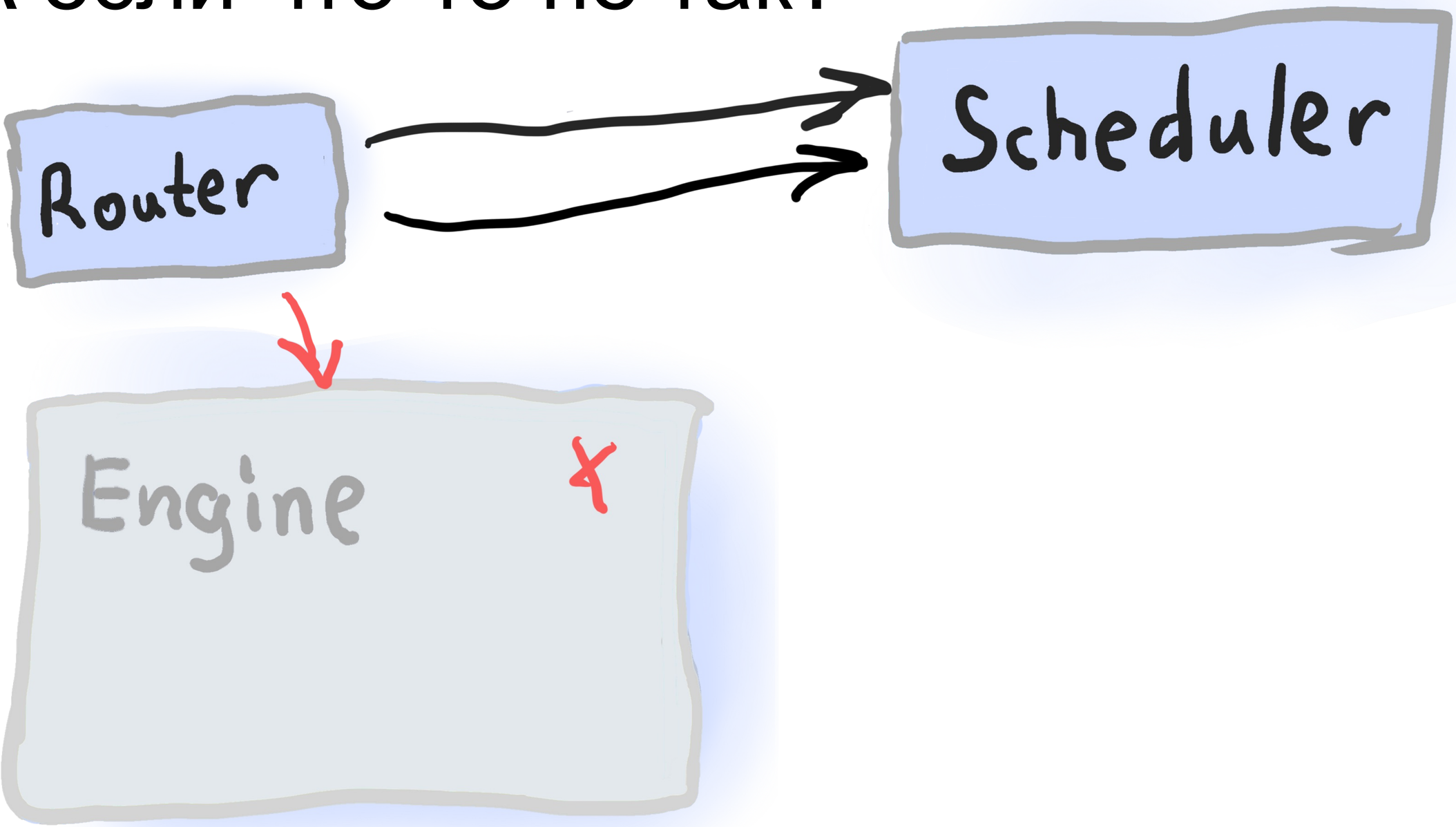
А если что-то не так?



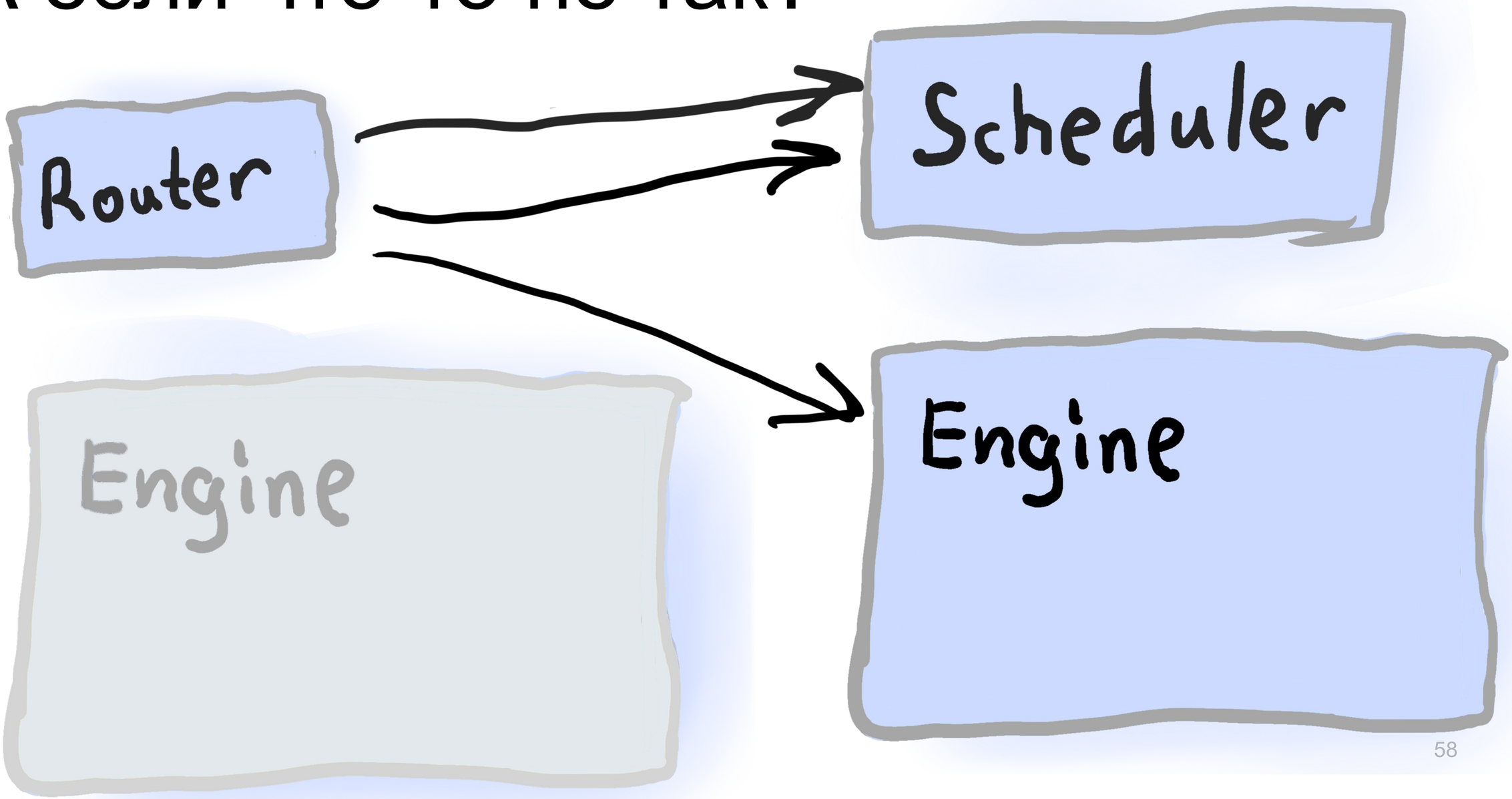
А если что-то не так?



А если что-то не так?



А если что-то не так?

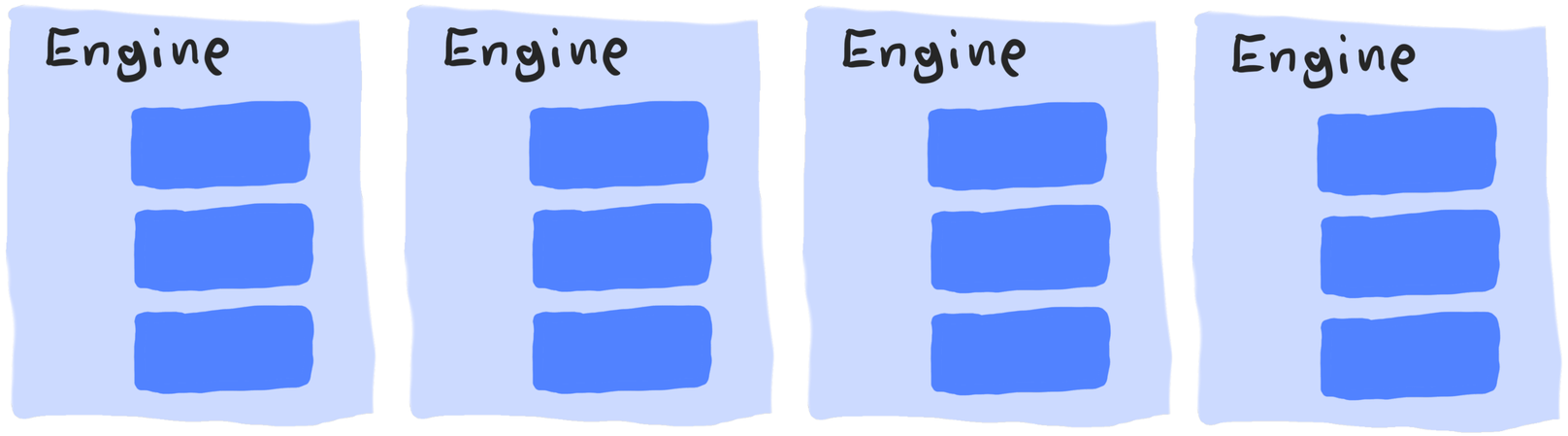


Выбираем Engine
для запуска нового экземпляра

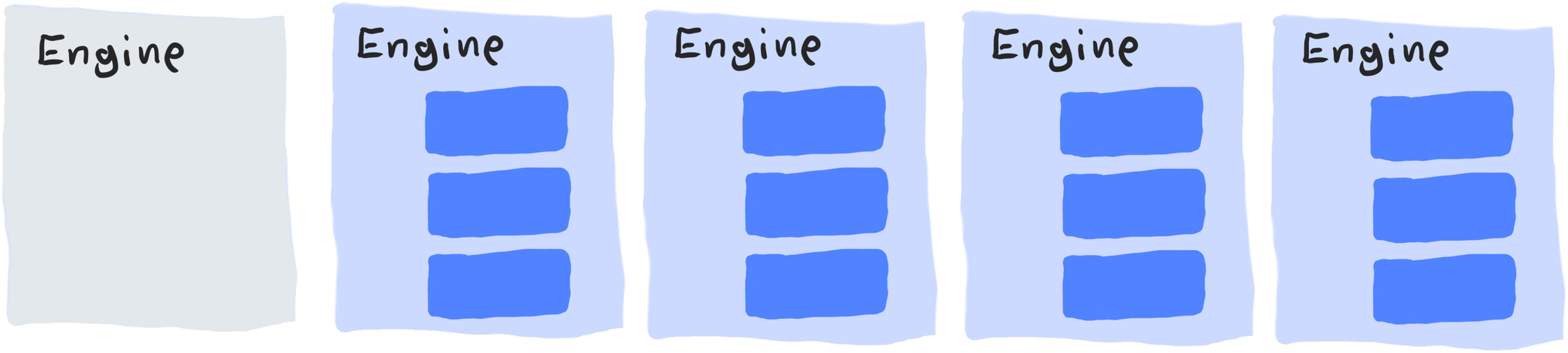
Смотрим
на ресурсы

Смотрим
на сети

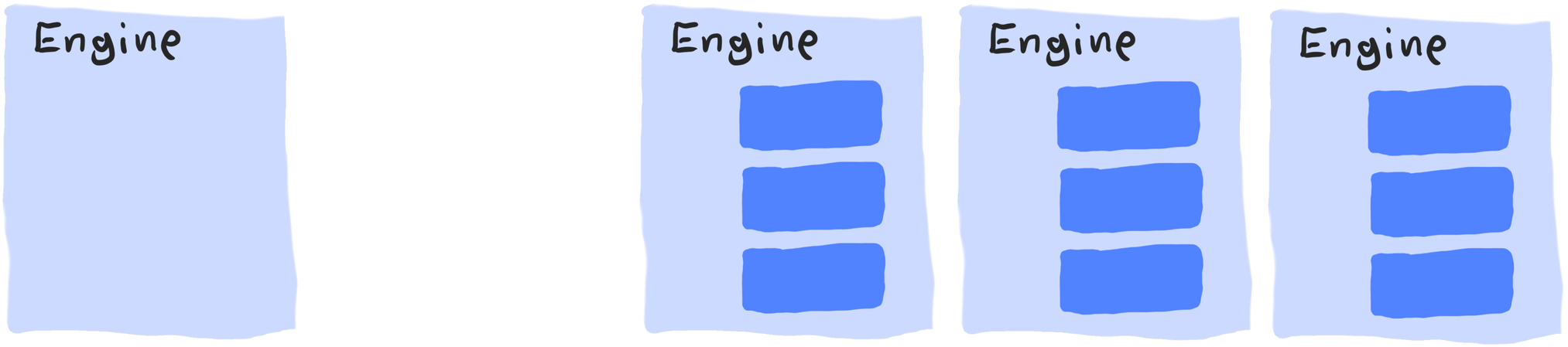
Обновление Engines



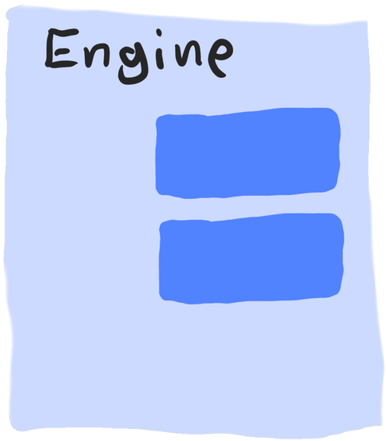
Обновление Engines

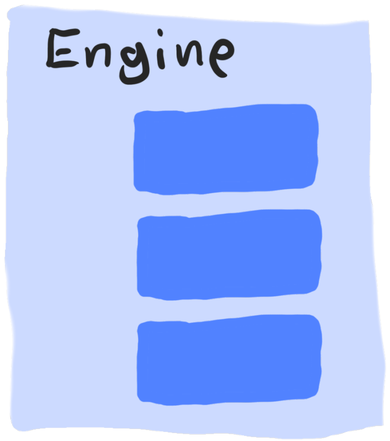


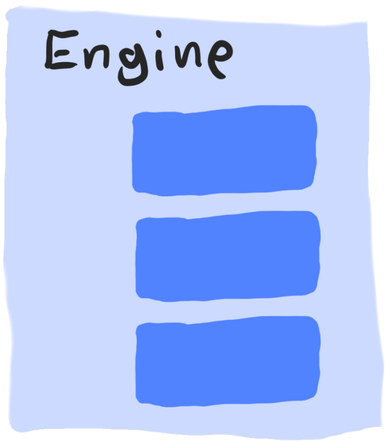
Обновление Engines

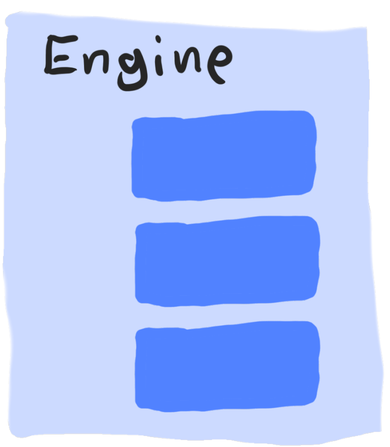


Обновление Engines

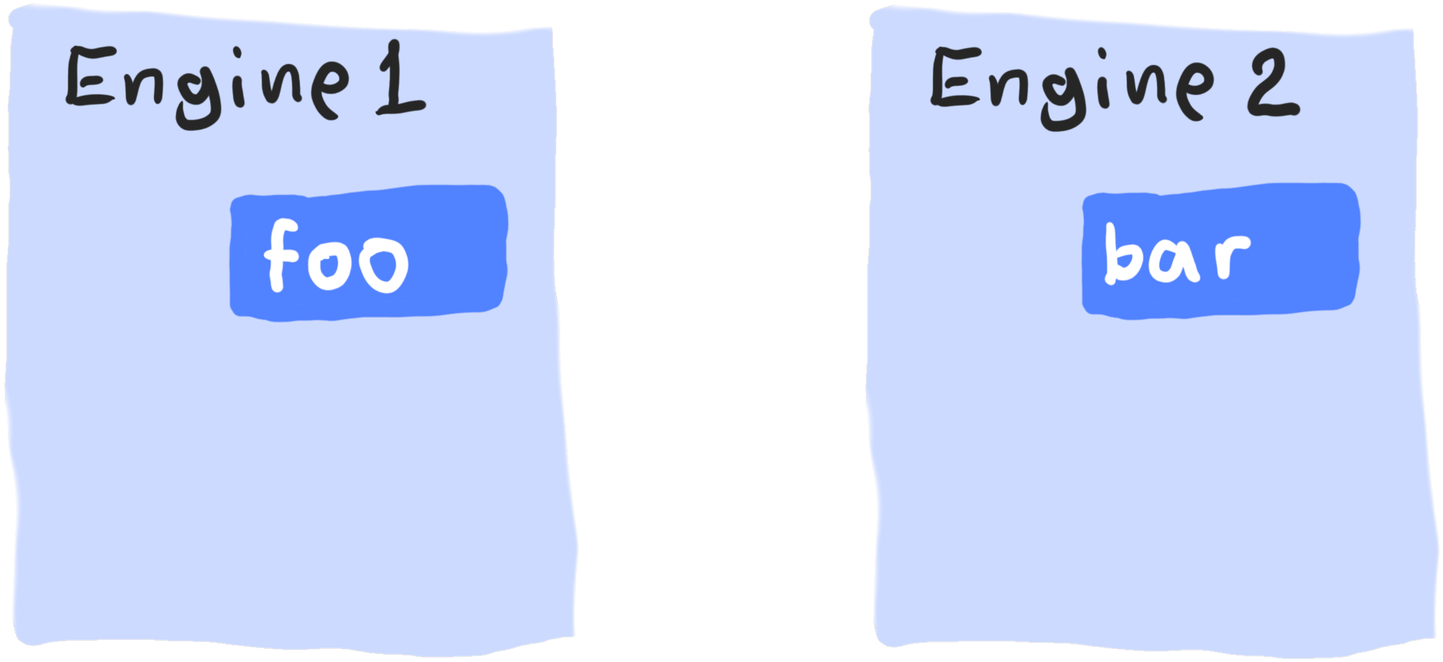




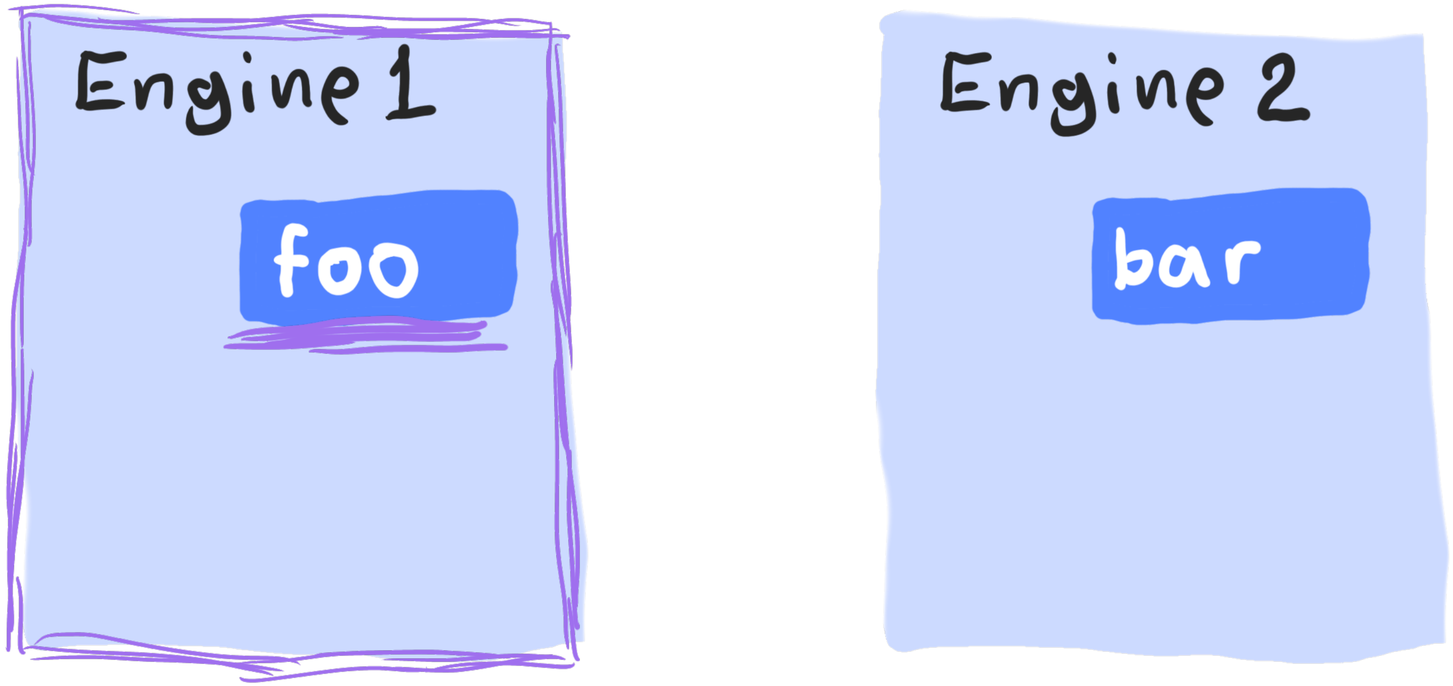




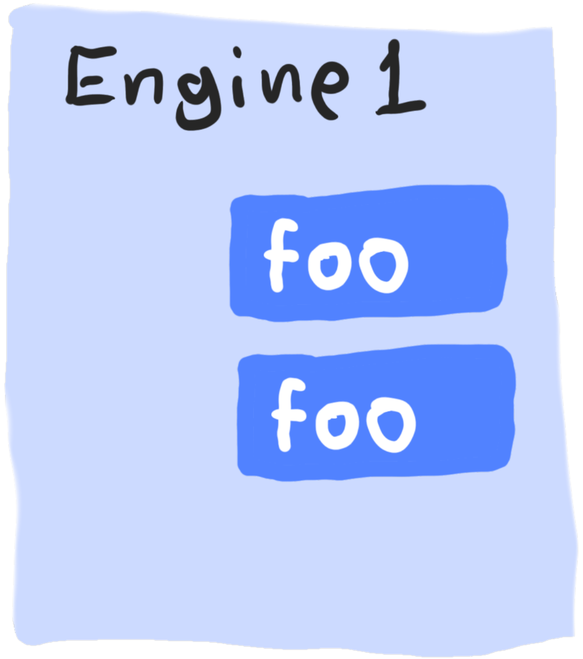
Как выбираем Engine?



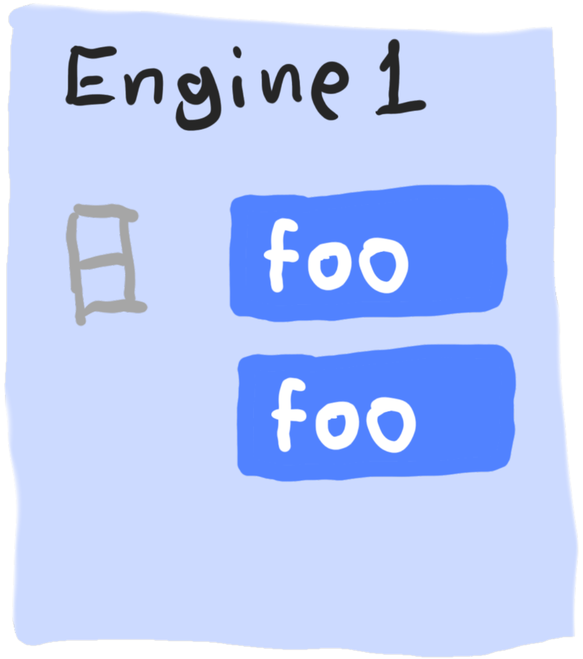
Как выбираем Engine?

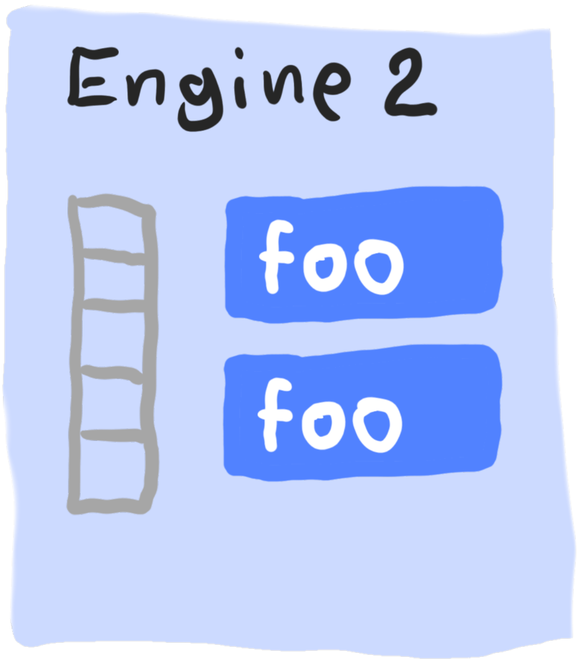


Как выбираем Engine?

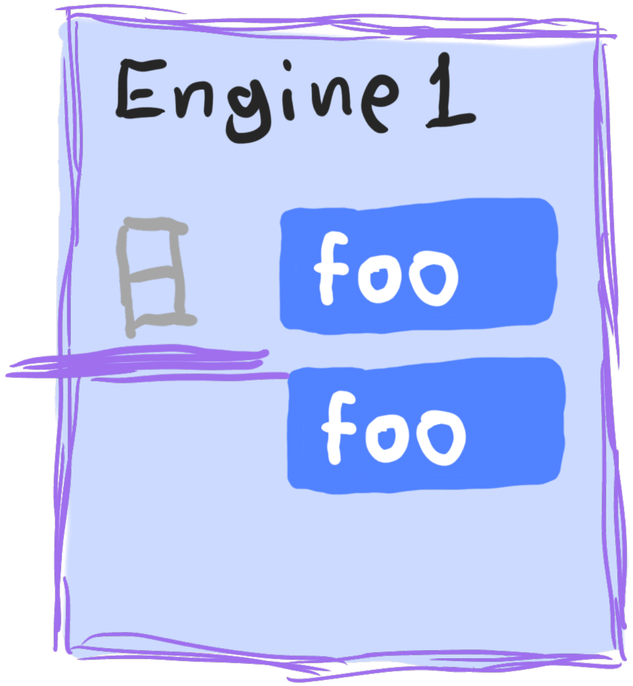


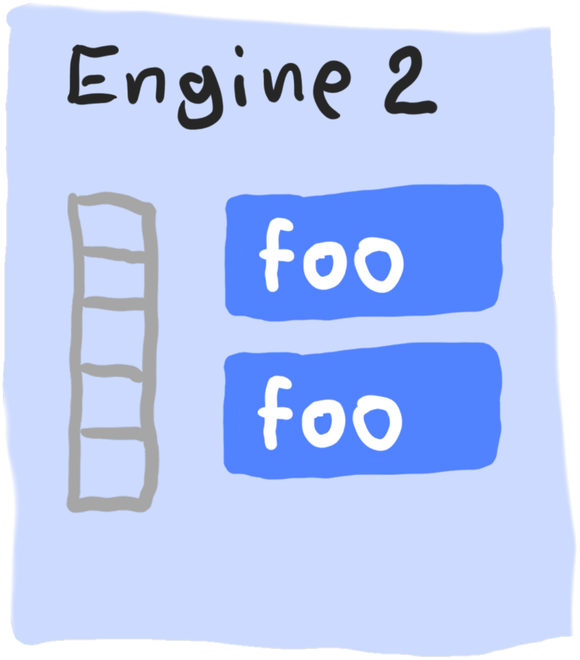
Как выбираем Engine?





Как выбираем Engine?





Как выбираем Engine?

В **10%** случаев идём создавать новый экземпляр

Функция. Логи

20 авг. 19:56:12.597

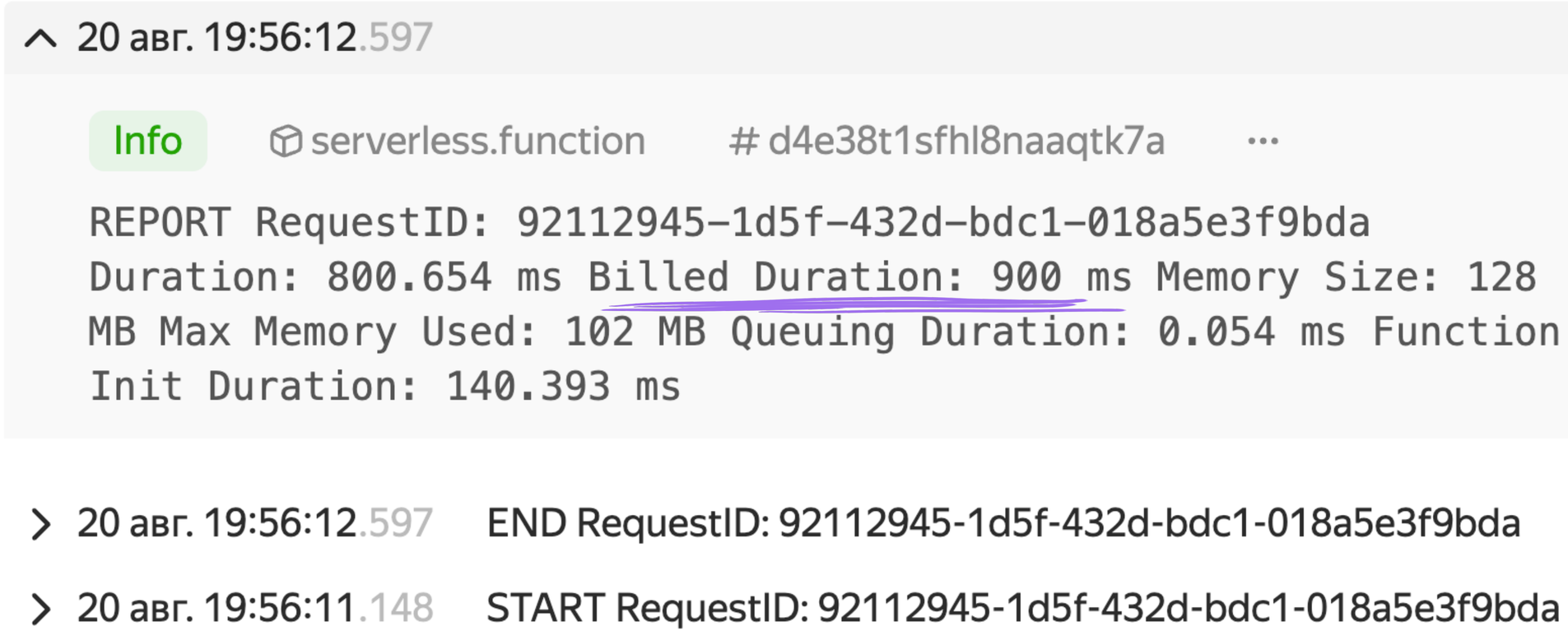
Info serverless.function # d4e38t1sfhl8naaqtk7a ···

```
REPORT RequestID: 92112945-1d5f-432d-bdc1-018a5e3f9bda
Duration: 800.654 ms Billed Duration: 900 ms Memory Size: 128
MB Max Memory Used: 102 MB Queuing Duration: 0.054 ms Function
Init Duration: 140.393 ms
```

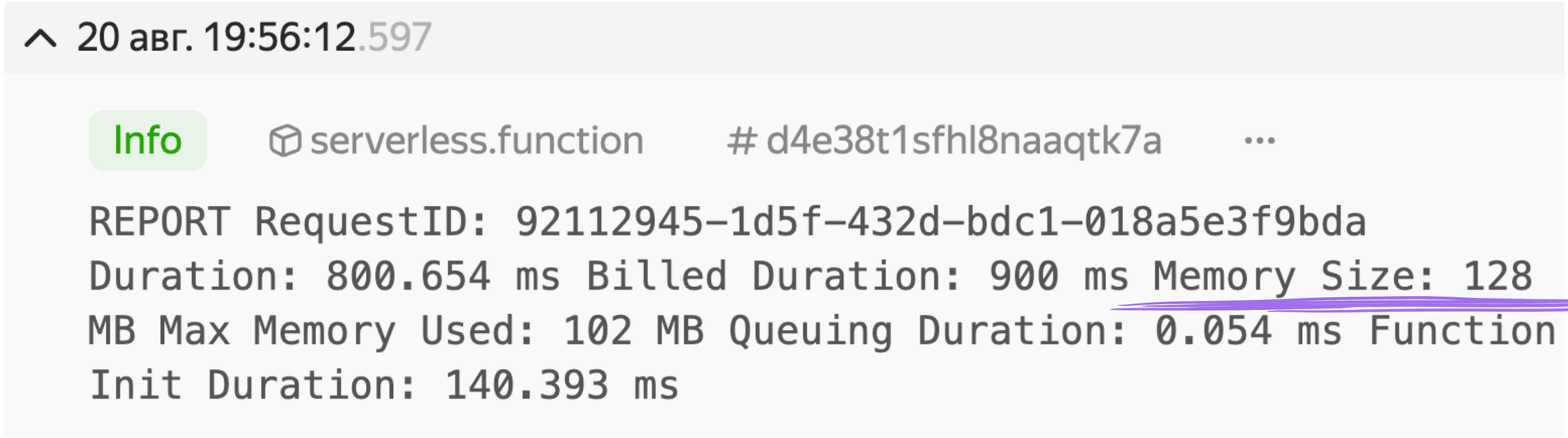
20 авг. 19:56:12.597 END RequestID: 92112945-1d5f-432d-bdc1-018a5e3f9bda

20 авг. 19:56:11.148 START RequestID: 92112945-1d5f-432d-bdc1-018a5e3f9bda

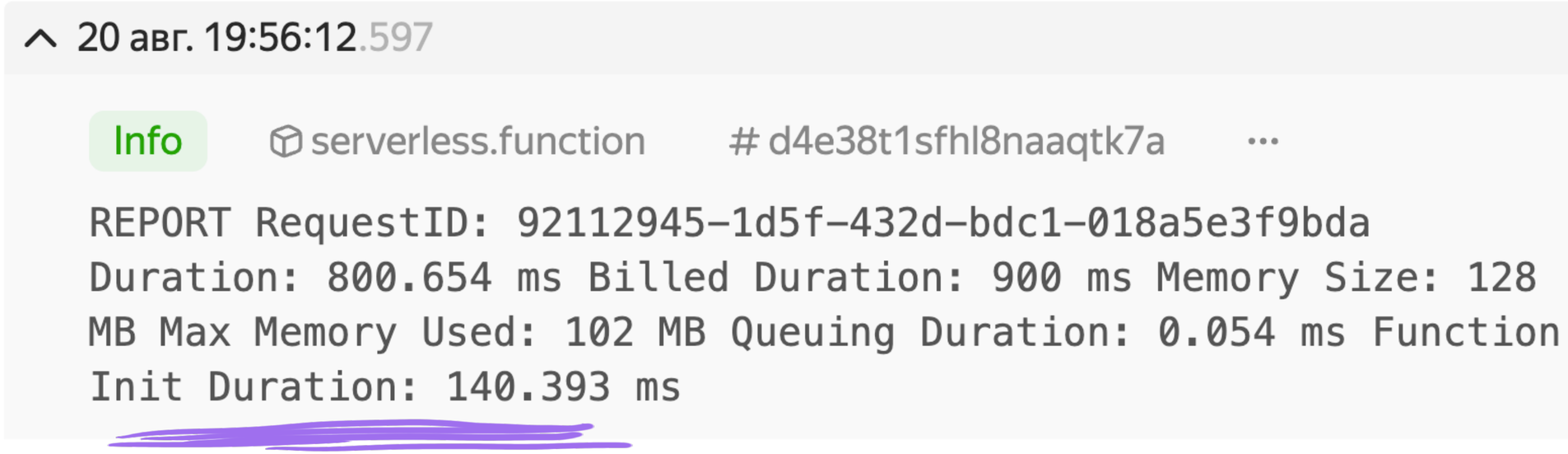
Функция. Логи



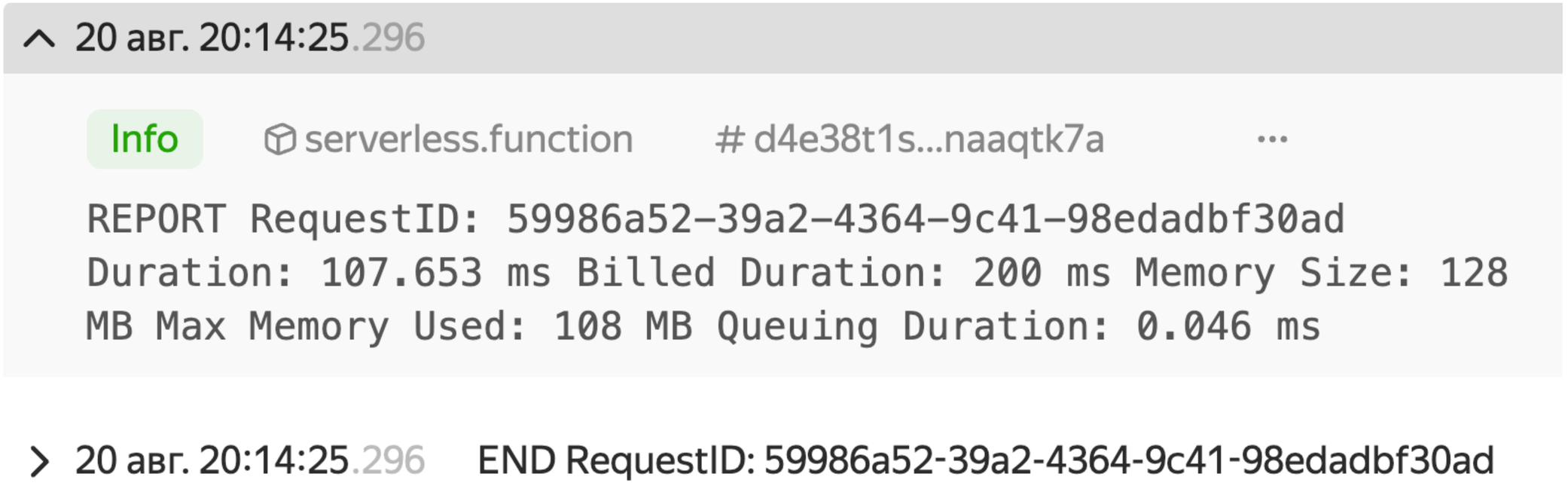
Функция. Логи



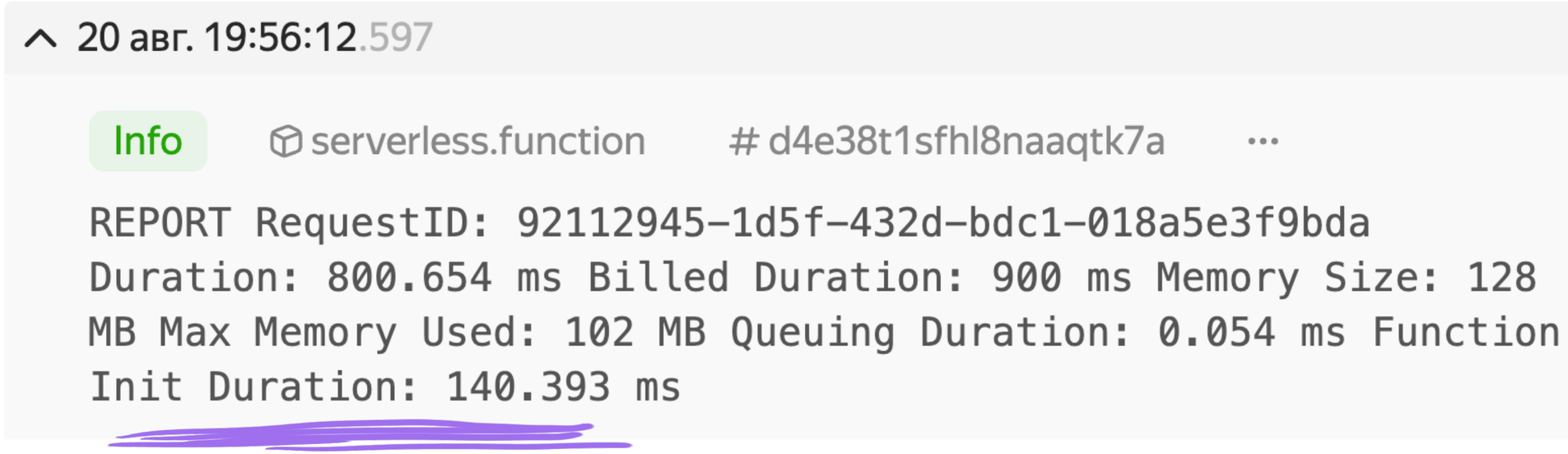
Функция. Логи



Функция. Логи



Функция. Логи



Статика включается в холодный старт

```
class Handler
{
    static Handler() { ... }

    private static Foo foo = CreateFoo();
}
```

Холодный старт — время загрузки виртуалки?

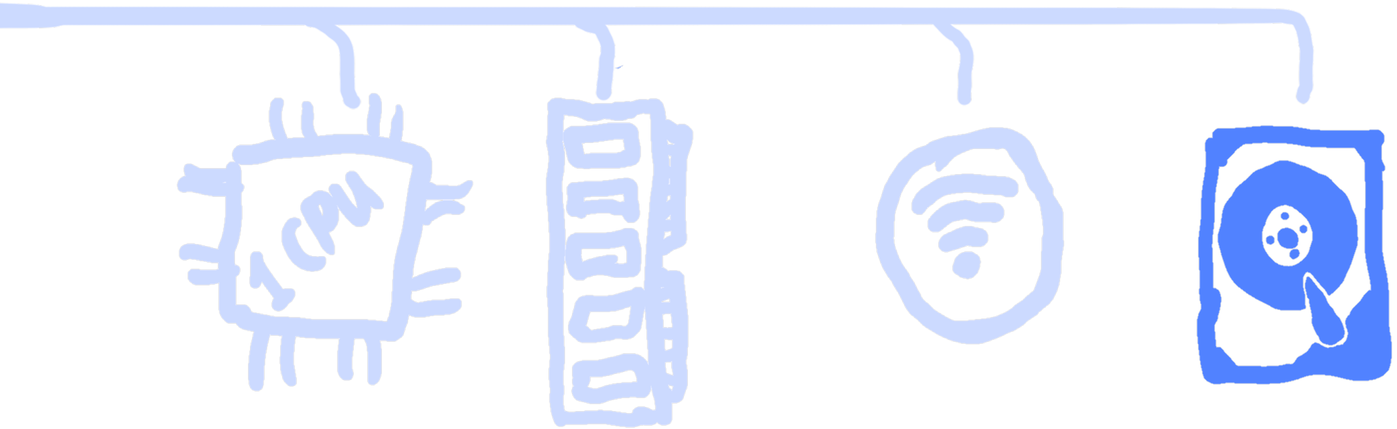
Задачи microVM

{f}

Изоляция функций

Быстрый старт

Как выглядит microVM



Как выглядит microVM

Как выглядит microVM

Запуск ОС

- Свой Bootstrap
- Запускаем без виртуального BIOS

Linux ядро

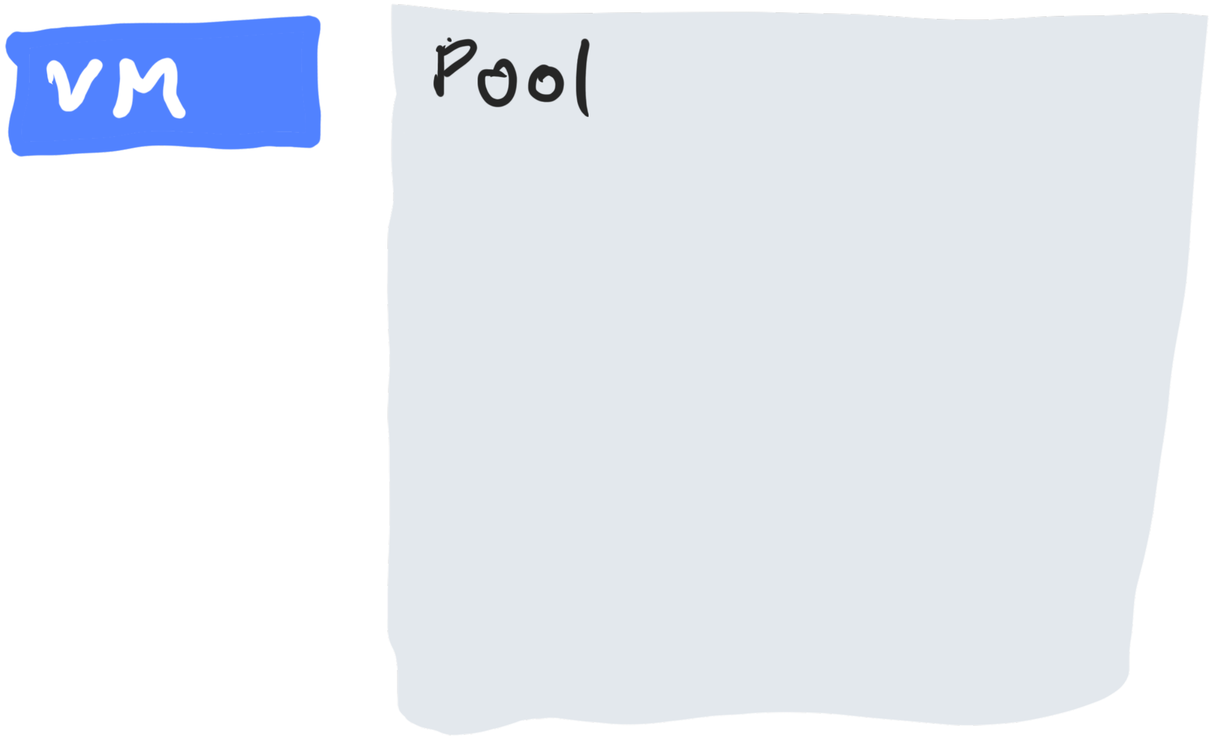
- Запуск ядра занимает 60–160 мс
- Собрано без ненужных модулей

Боремся с Cold Start

Поддерживаем пул запущенных виртуальных машин

Пулы виртуалок

- Запустили виртуалку с 64 MB
- Загрузили ОС

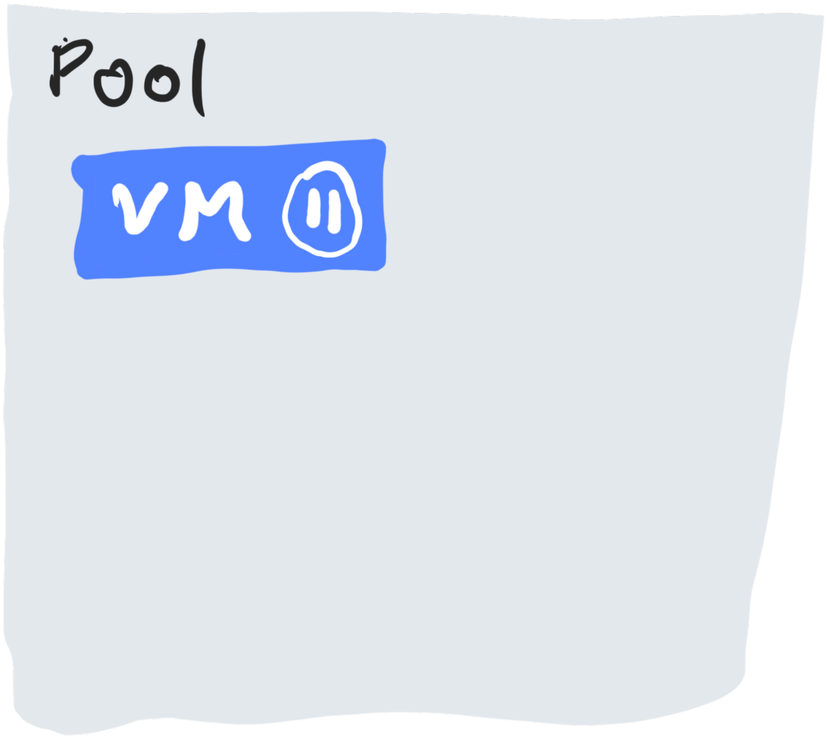


Пулы виртуалок

- Запустили виртуалку с 64 MB
- Загрузили ОС
- Suspend

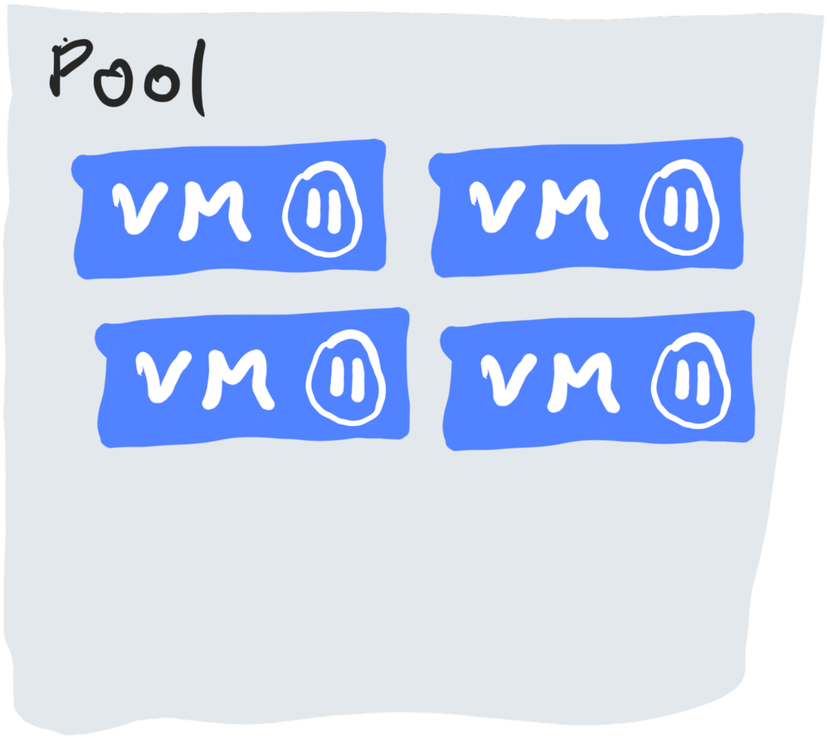
Пулы виртуалок

- Запустили виртуалку с 64 MB
- Загрузили ОС
- Suspend
- Положили в пул



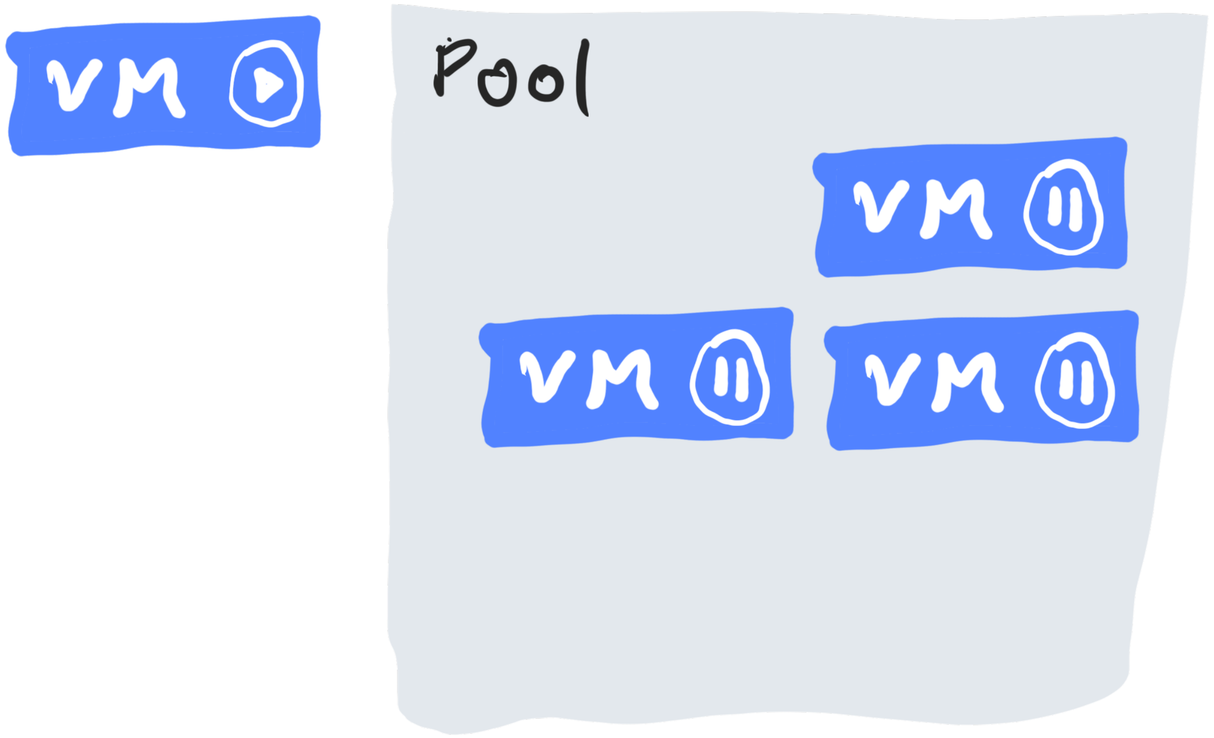
Пулы виртуалок

- Запустили виртуалку с 64 MB
- Загрузили ОС
- Suspend
- Положили в пул



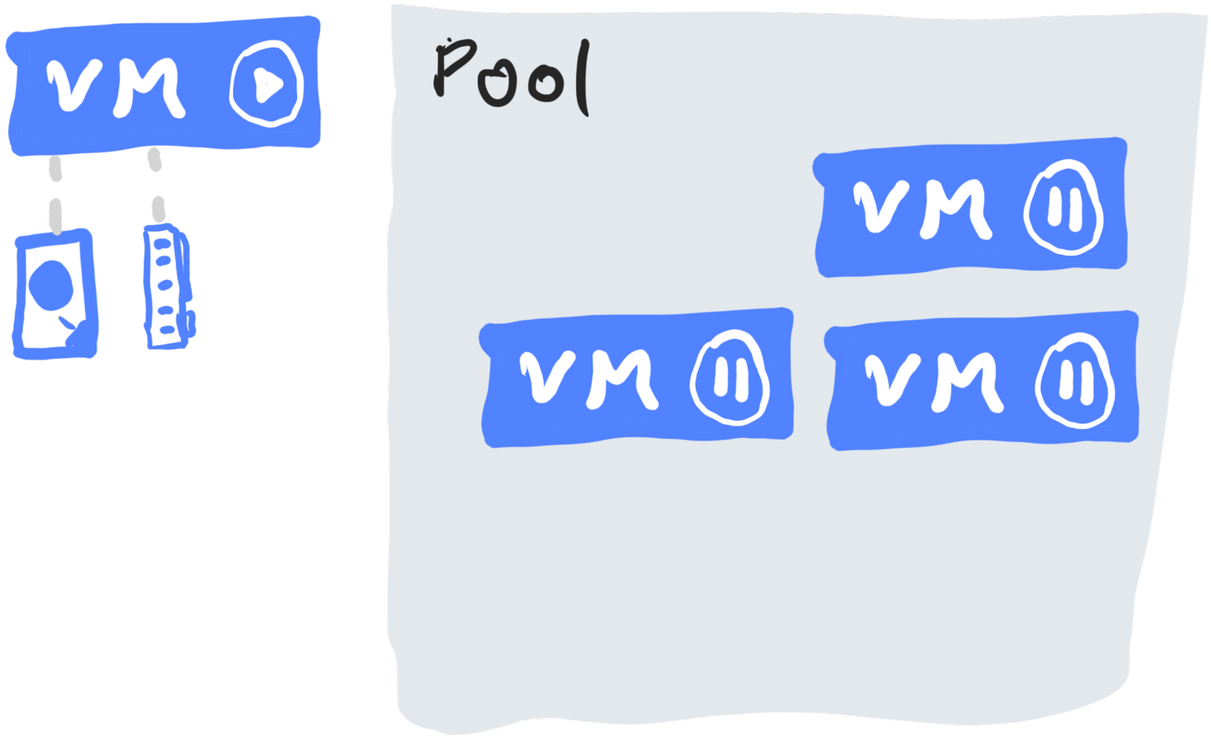
Новый экземпляр функции

- Взяли из pool
- Resume



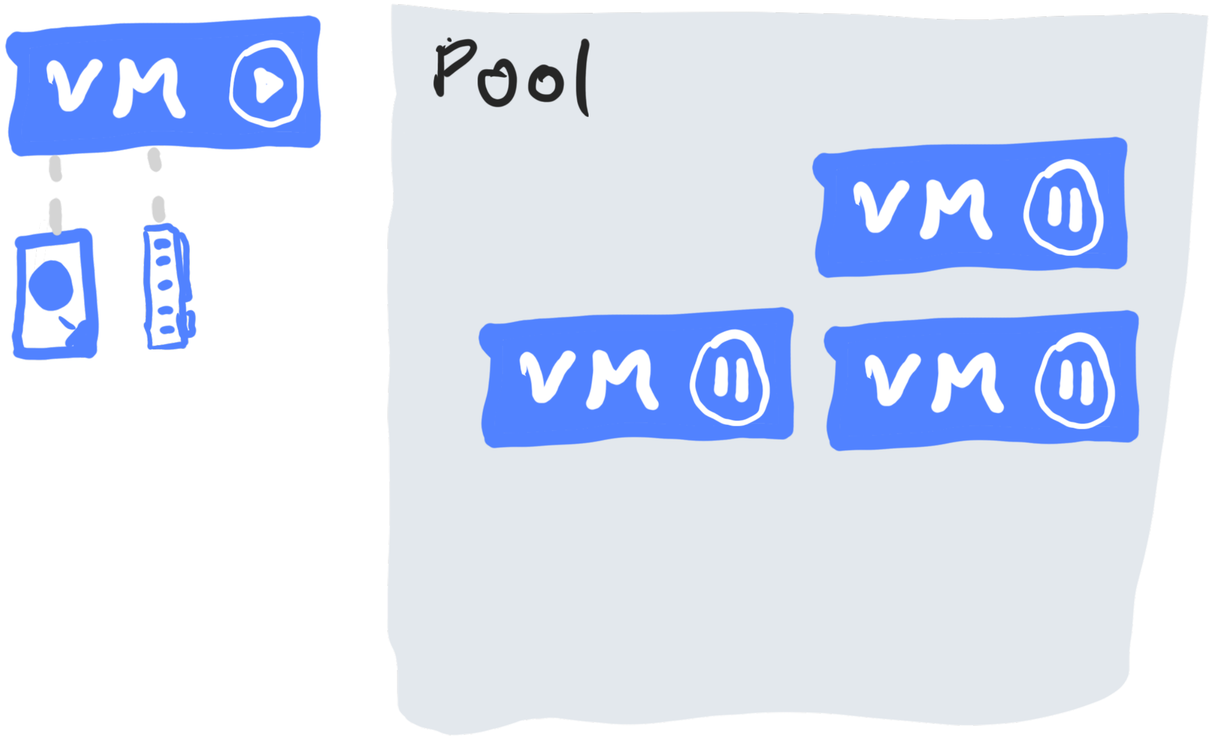
Новый экземпляр функции

- Взяли из pool
- Resume
- Смонтировали FileSystem
- Добавили памяти



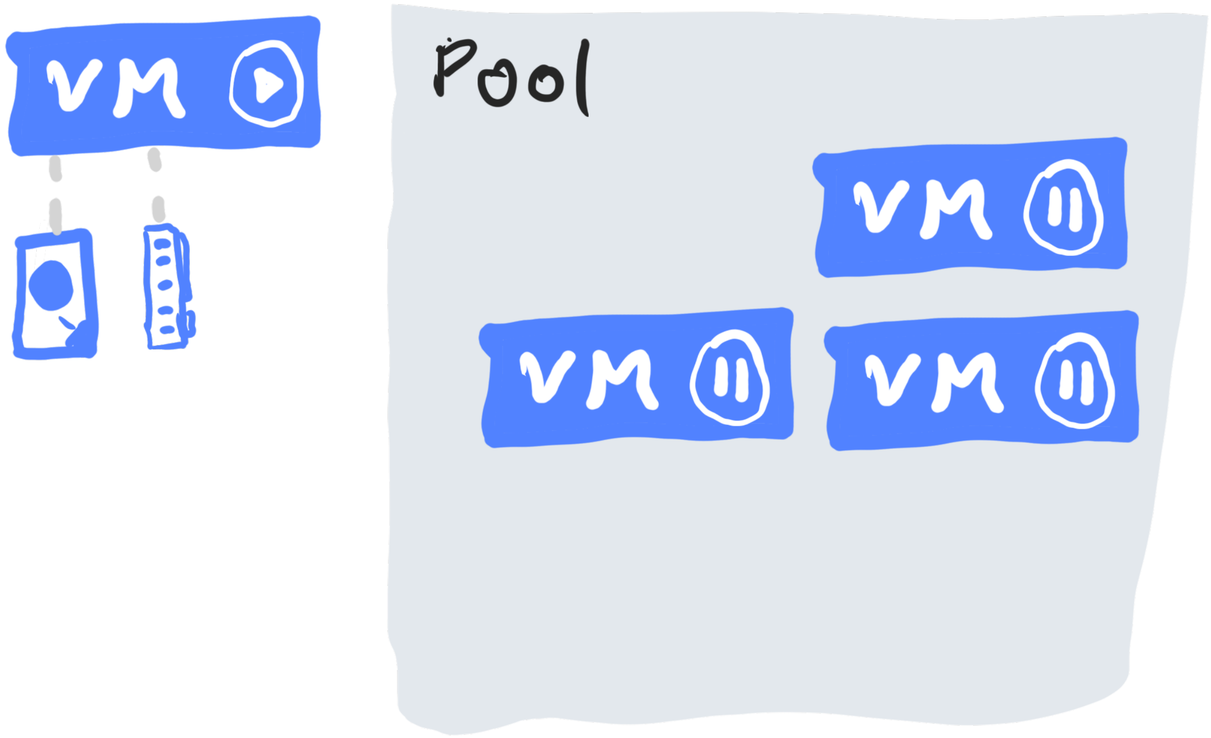
Новый экземпляр функции

- Взяли из pool
- Resume
- Смонтировали FileSystem
- Добавили памяти
- Установили переменные окружения
- Запустили Runtime



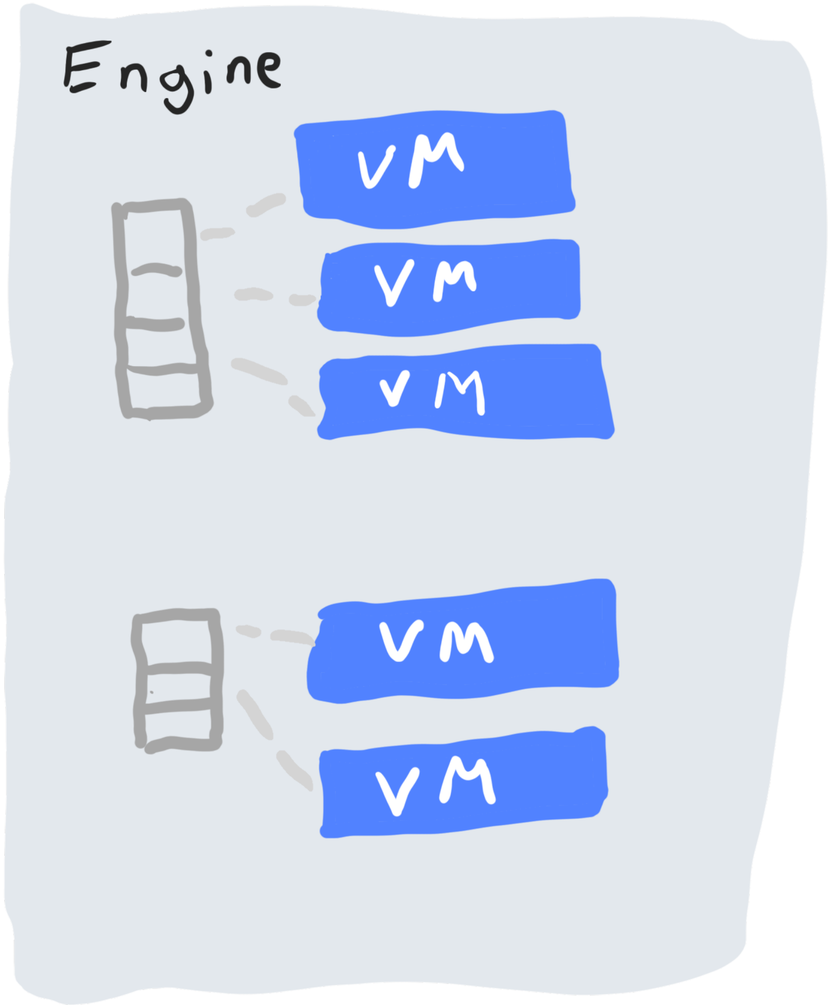
Новый экземпляр функции

- Взяли задачу из очереди



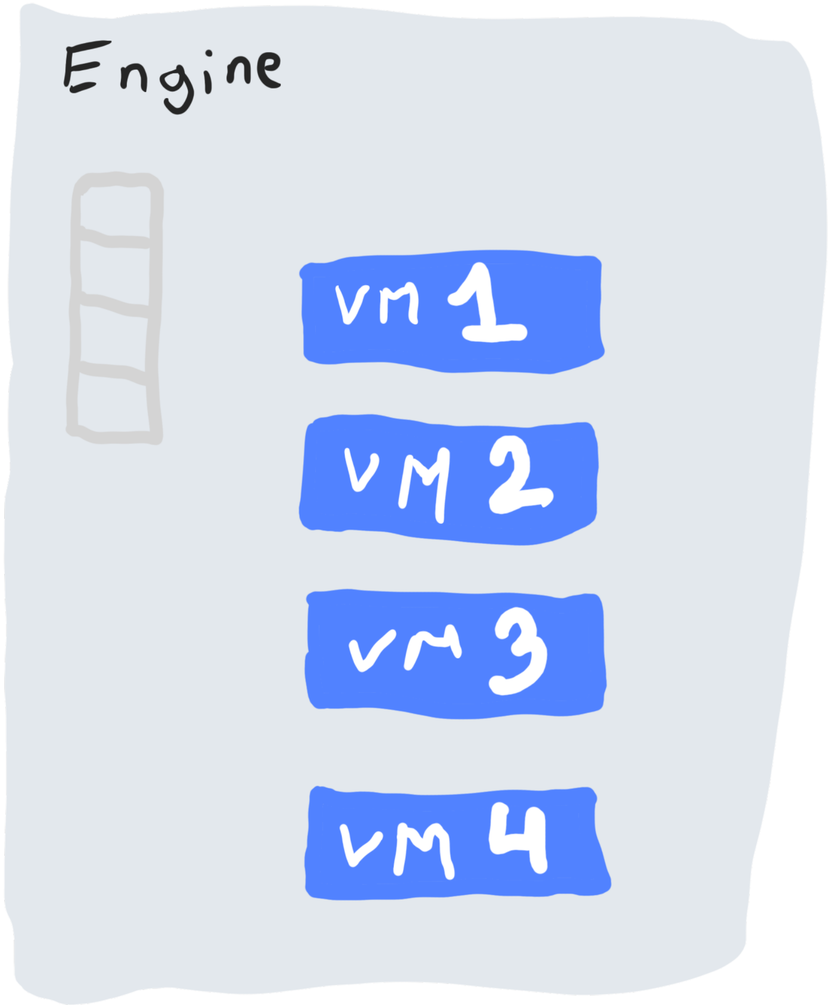
Очередь задач

- Запрос попадает в очередь для экземпляров функций
- Каждый экземпляр вытаскивает задачу из очереди



Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся

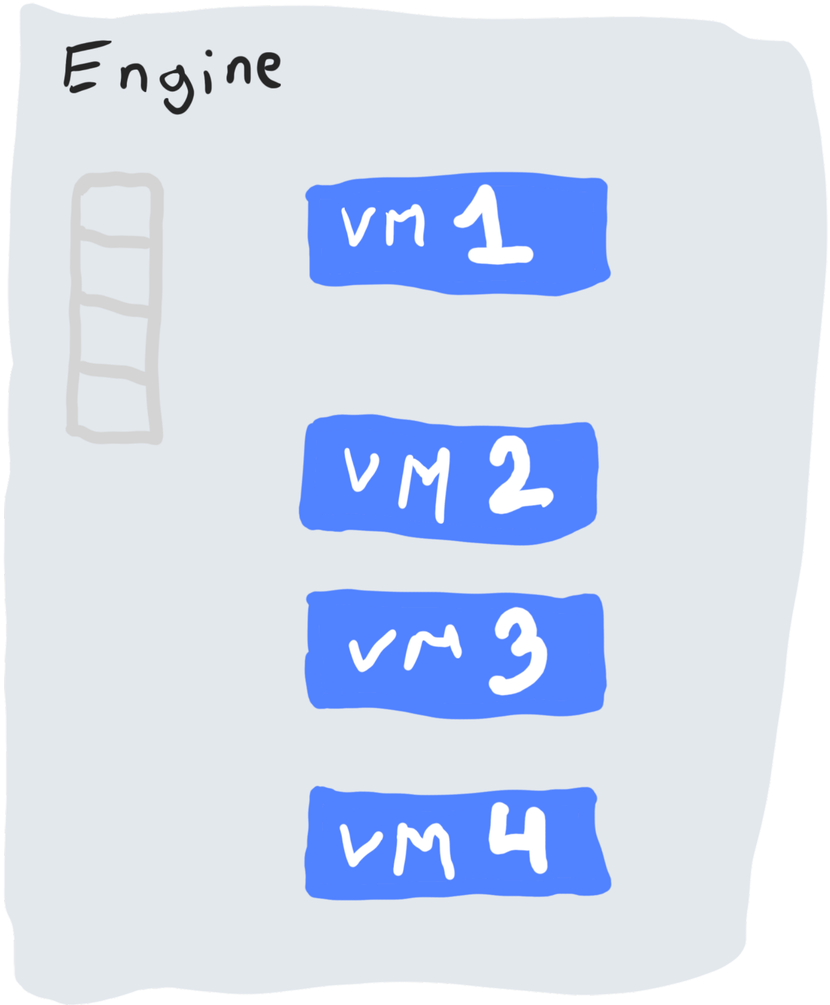


Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся

Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся

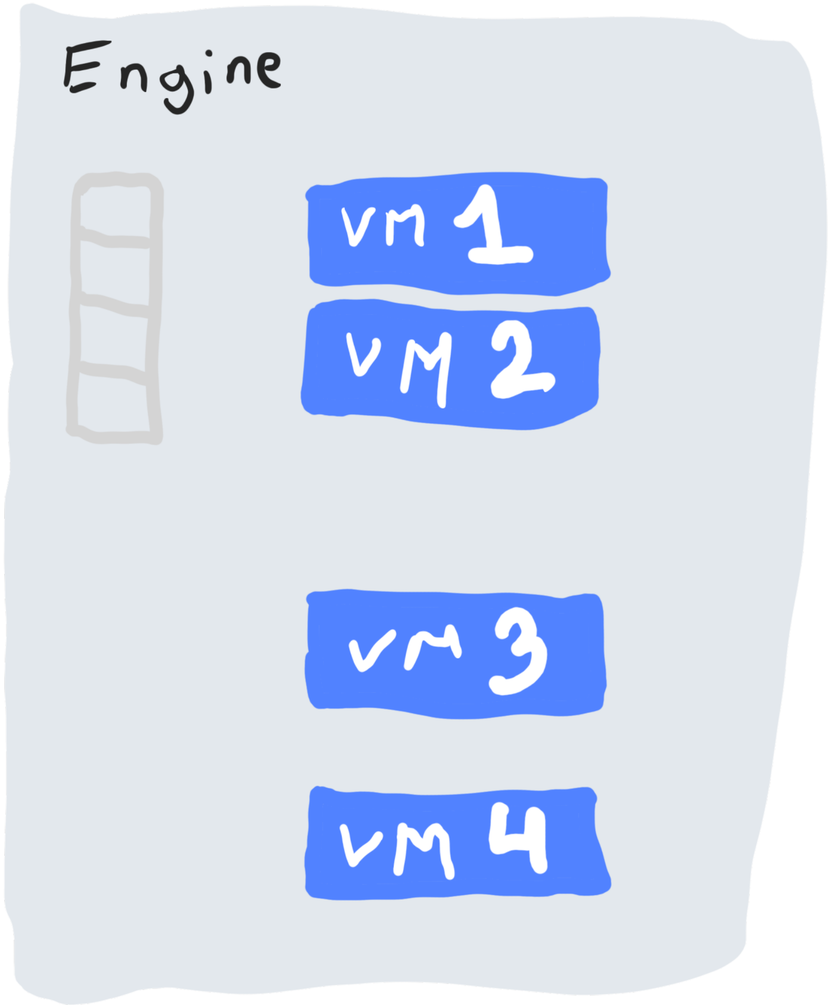


Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся

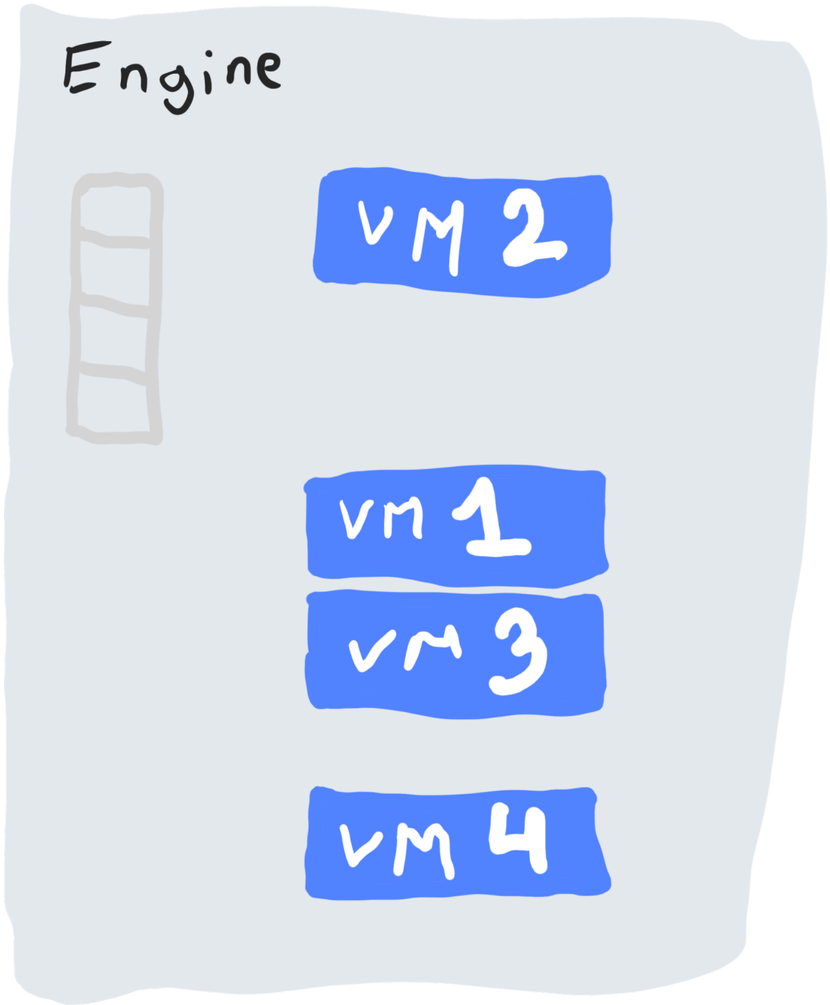
Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



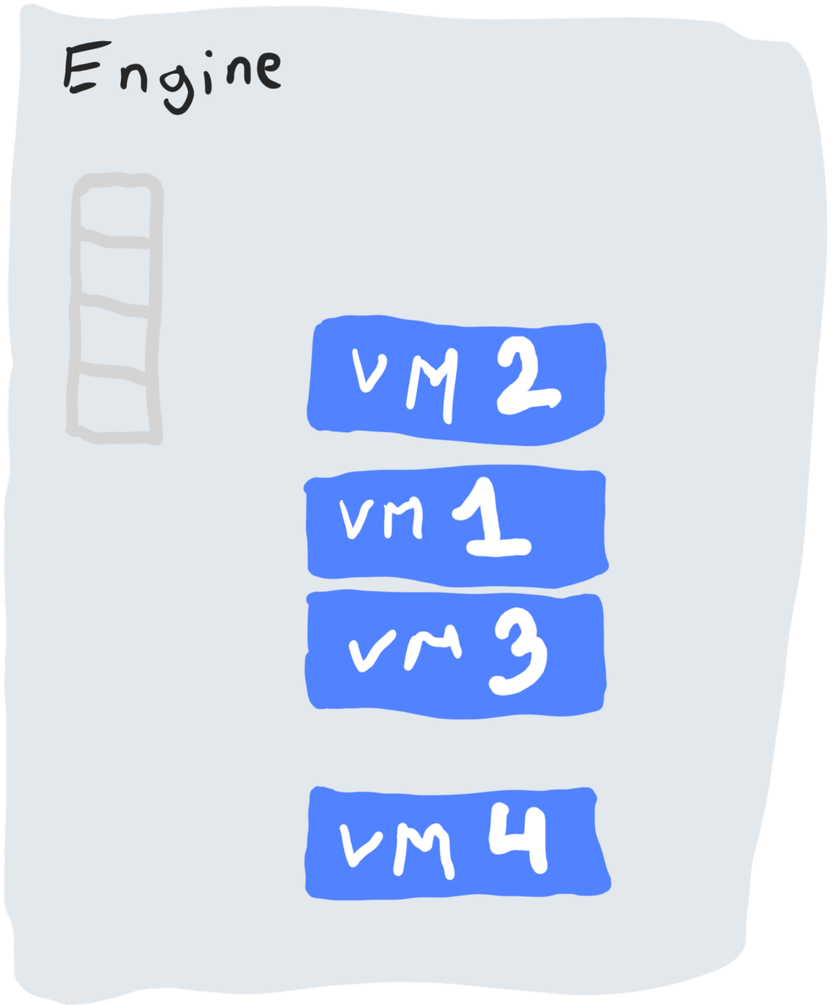
Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



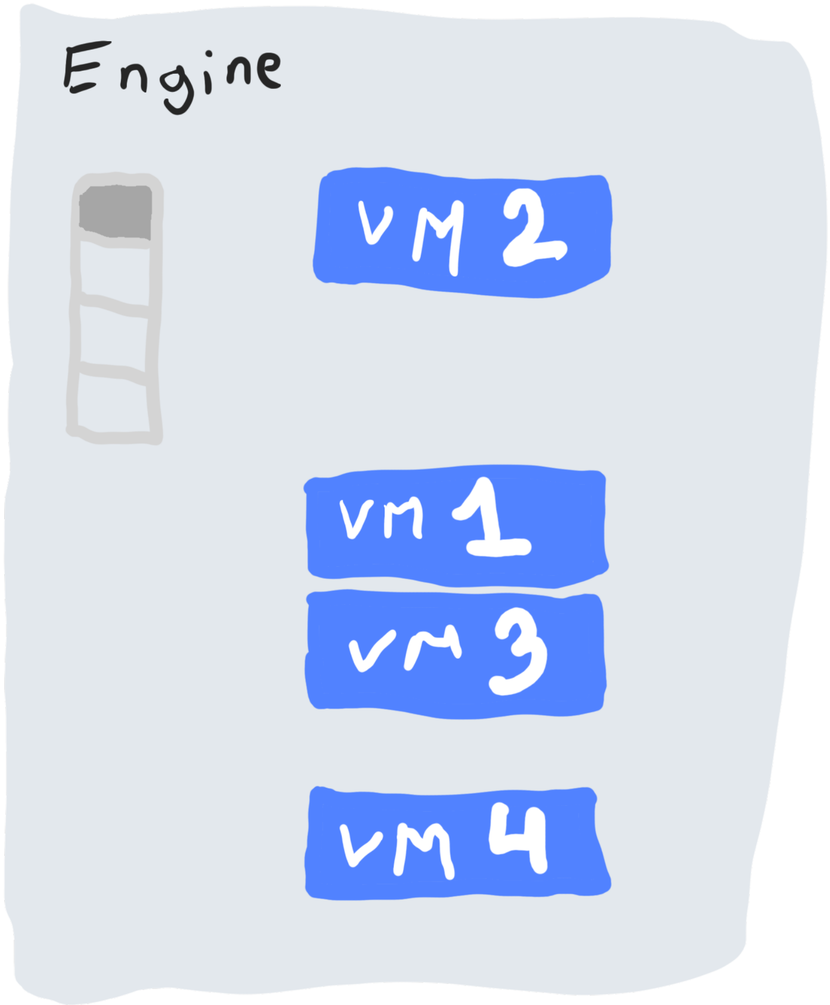
Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



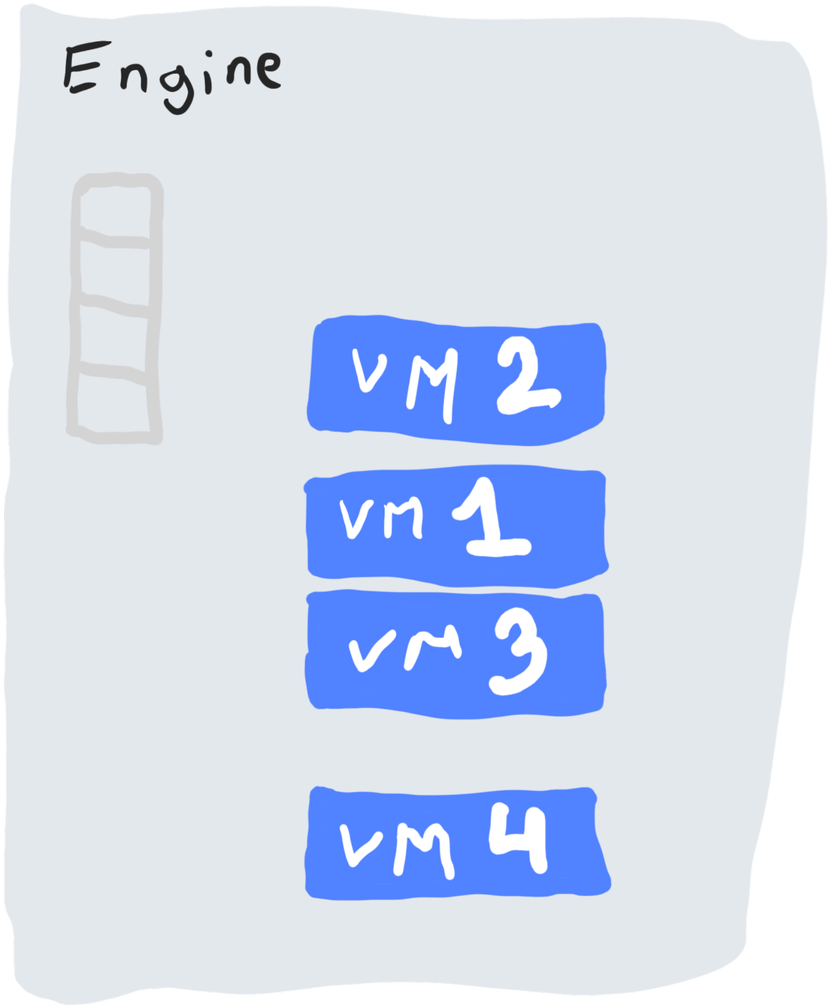
Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



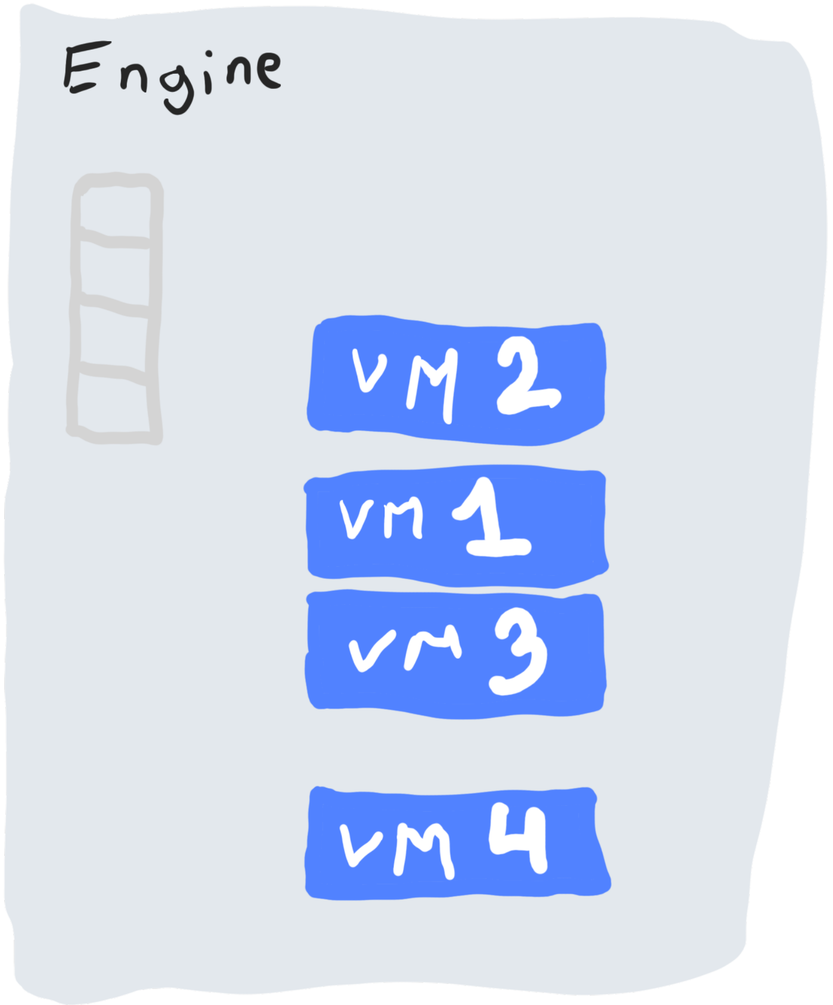
Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



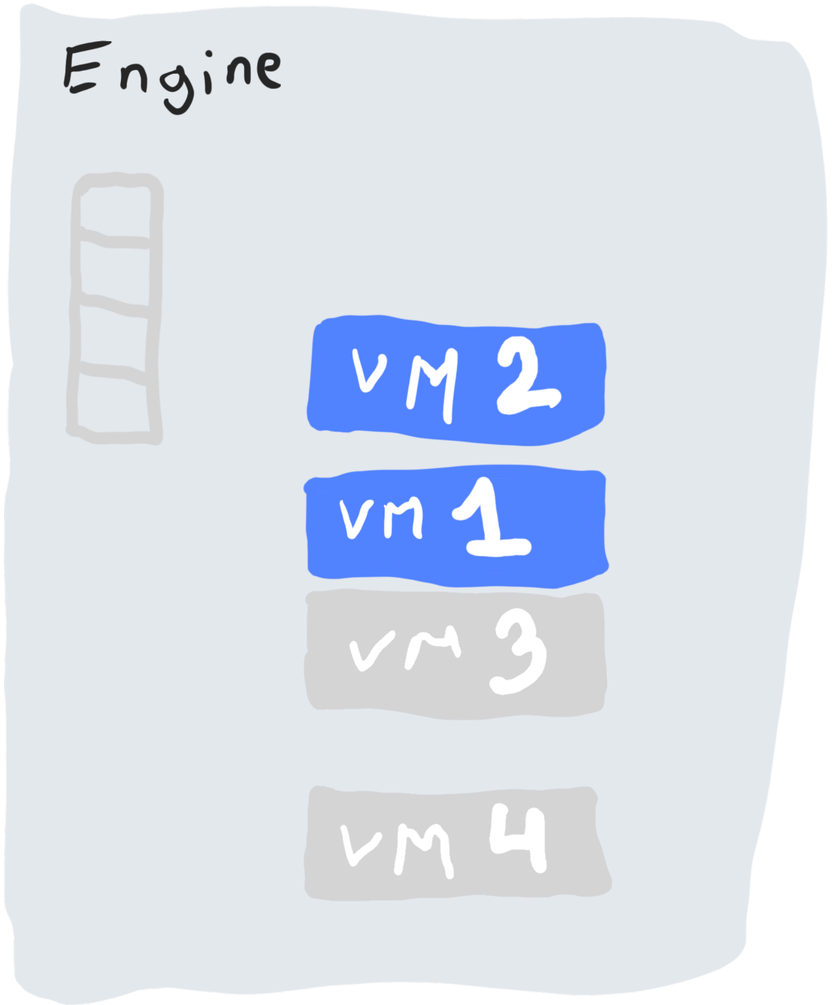
Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



Очередь задач пустая

- Несколько экземпляров функций
- Новую задачу берёт последний освободившийся



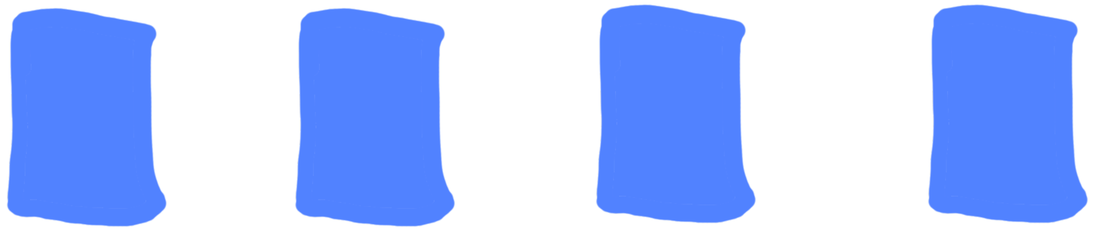
А что если запросов будет много?

Квоты

- Одновременно запущенные экземпляры функций
- Потребляемая память
- Одновременно выполняемые запросы

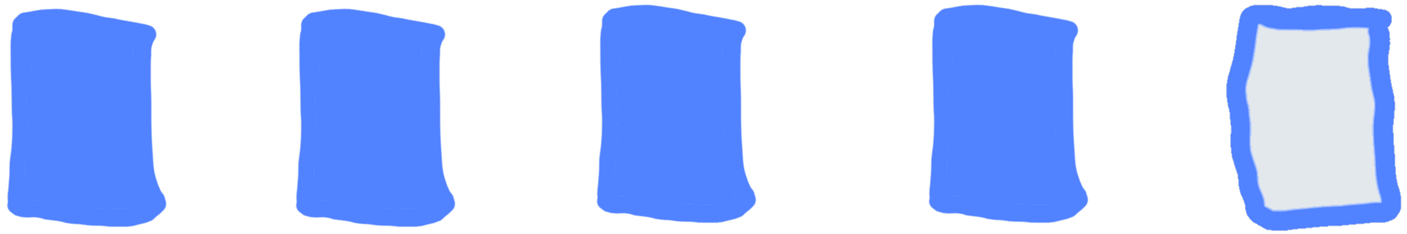
Превысили — HTTP 429

Проблема деплоя новой версии

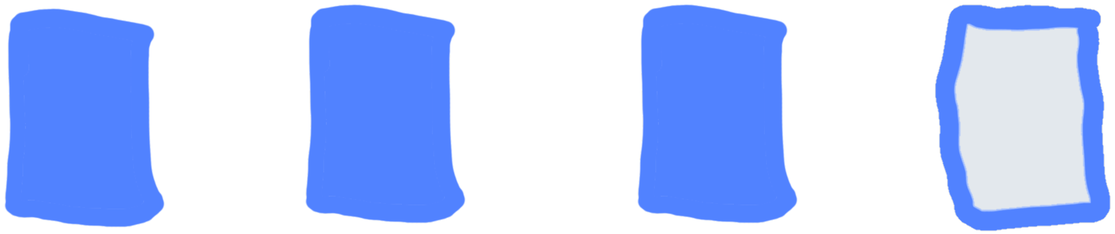


Проблема деплоя новой версии

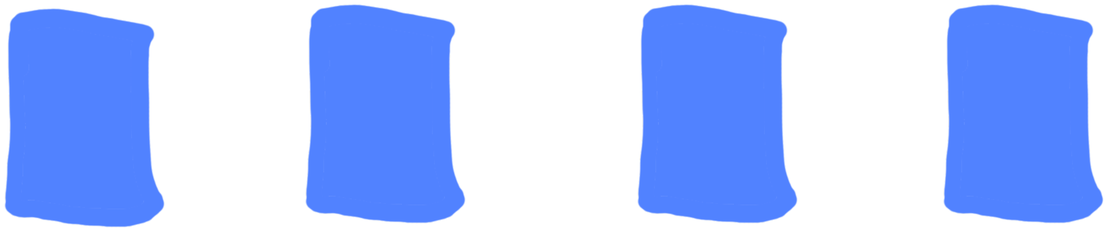
Проблема деплоя новой версии



Проблема деплоя новой версии

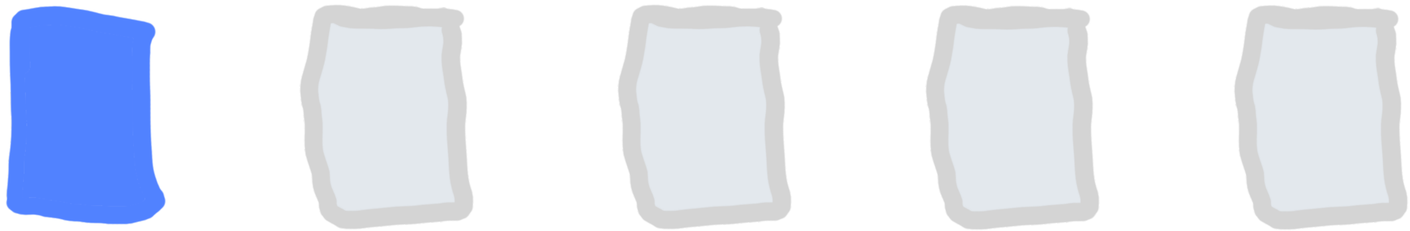


Проблема деплоя новой версии

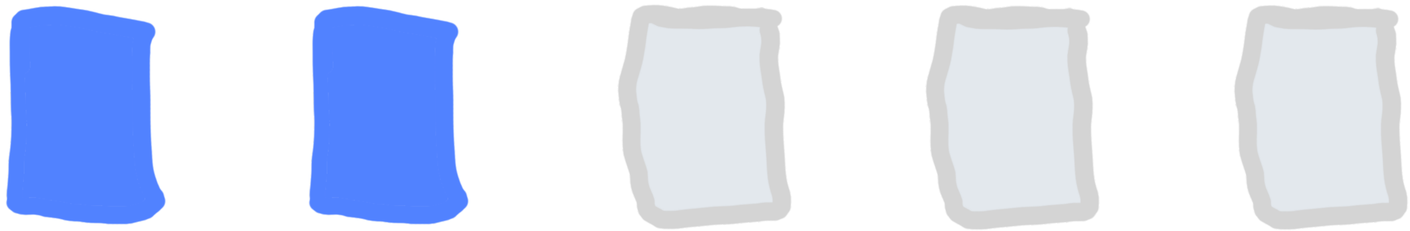


Pipelines

Pipelines



Pipelines



Pipelines

Pipelines

Pipelines

Pipelines

Pipelines

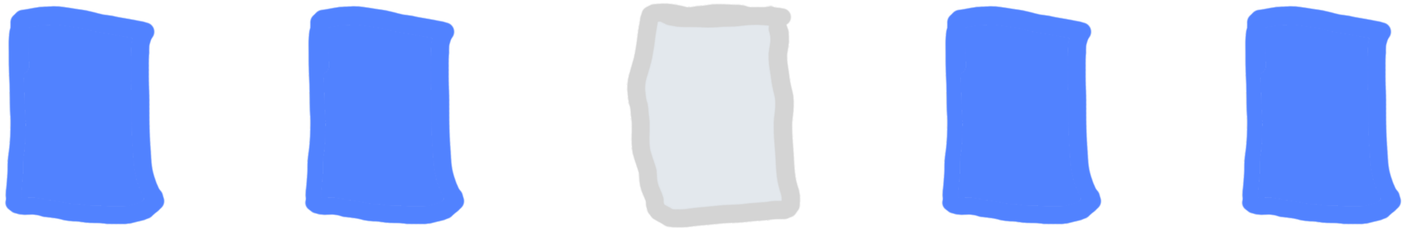
Pipelines

Pipelines

Pipelines

Pipelines

Pipelines



1. Что такое Serverless

2. Как работают Cloud Functions

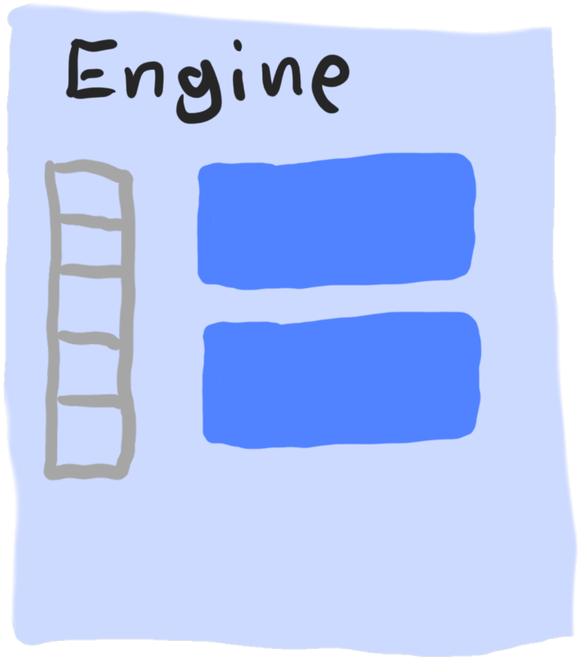
3. Внутреннее устройство Cloud Functions

Autoscaler

- Фоновый процесс
- Запускается не чаще чем раз в 500 мс

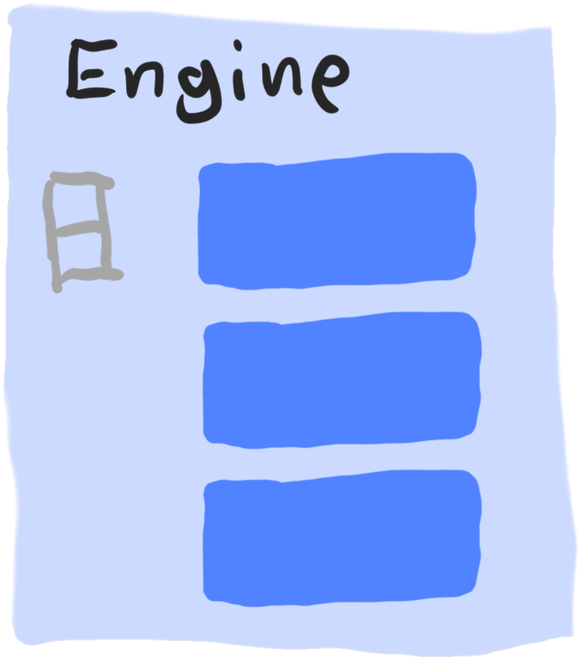
Up Scale

- Очередь больше, чем количество экземпляров, на 30%



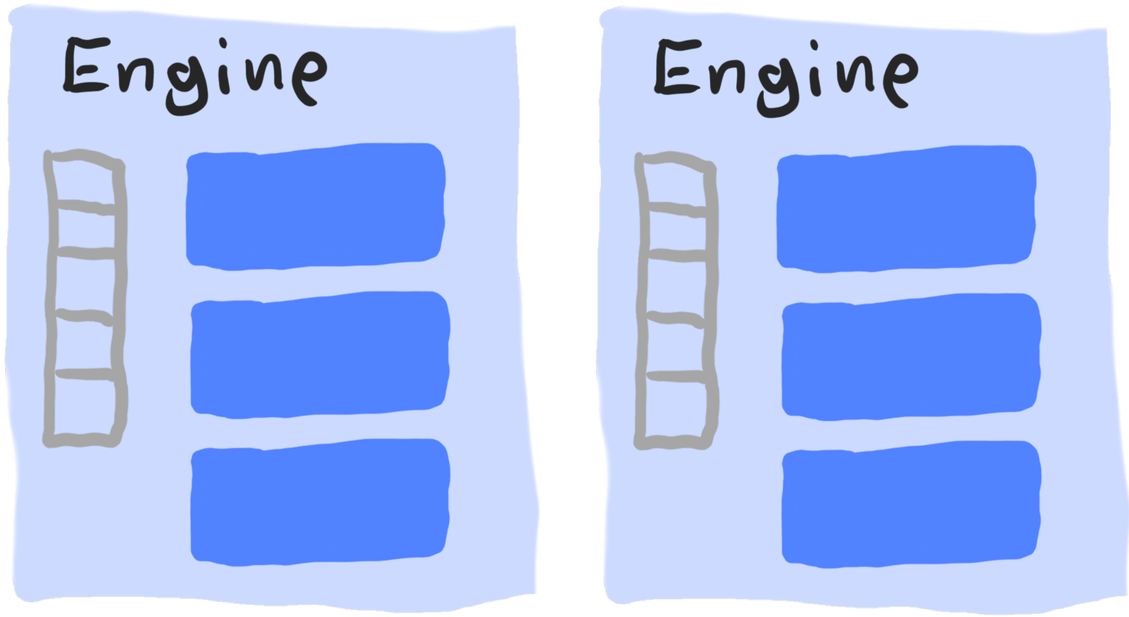
Up Scale

- Очередь больше, чем количество экземпляров, на 30%
- Создаём экземпляр тут же



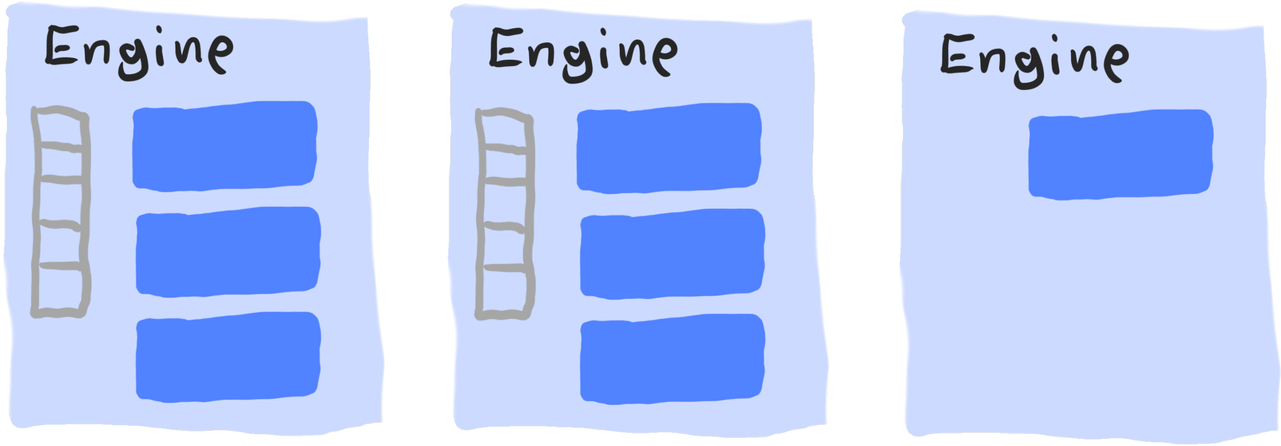
Up Scale

- Сумма очередей больше, чем сумма экземпляров, на 30%



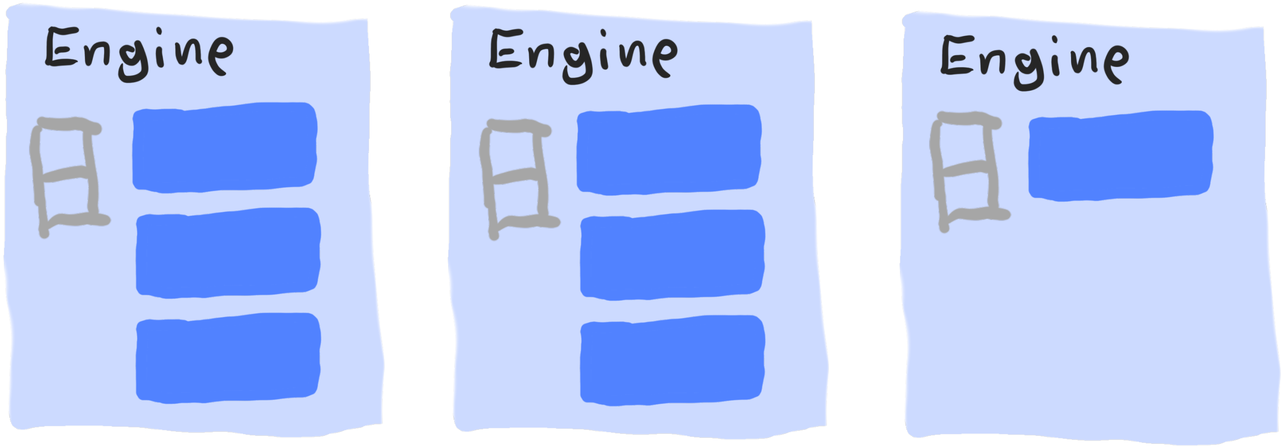
Up Scale

- Сумма очередей больше, чем сумма экземпляров, на 30%
- Создаём экземпляр на новом Engine



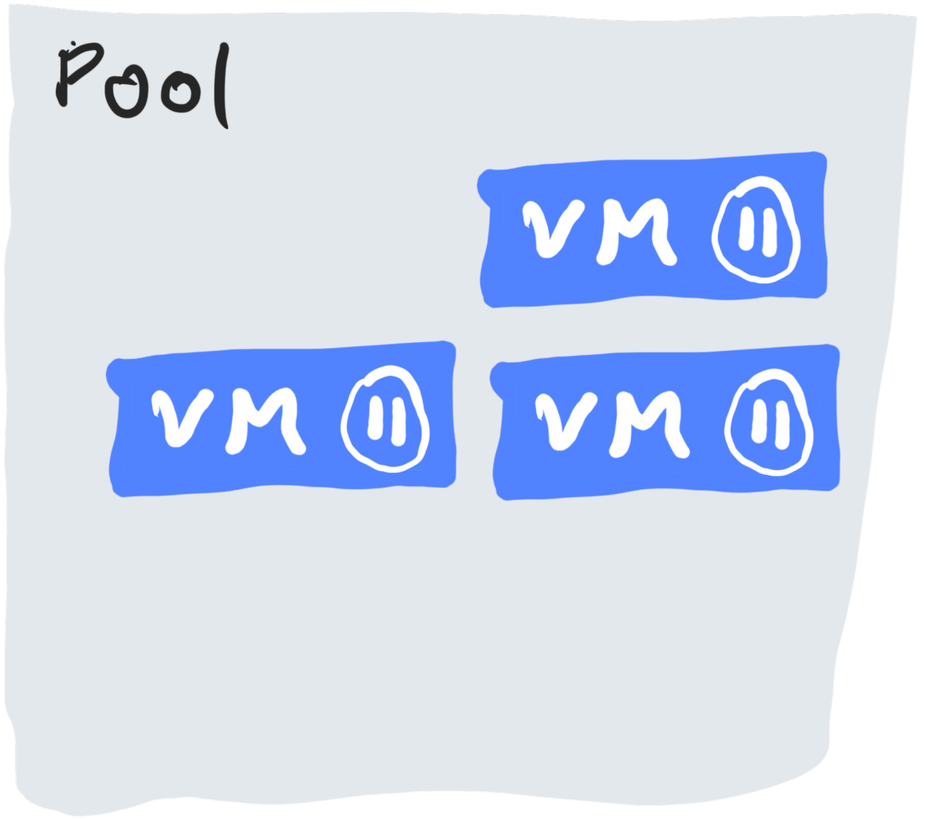
Up Scale

- Сумма очередей больше, чем сумма экземпляров, на 30%
- Создаём экземпляр на новом Engine



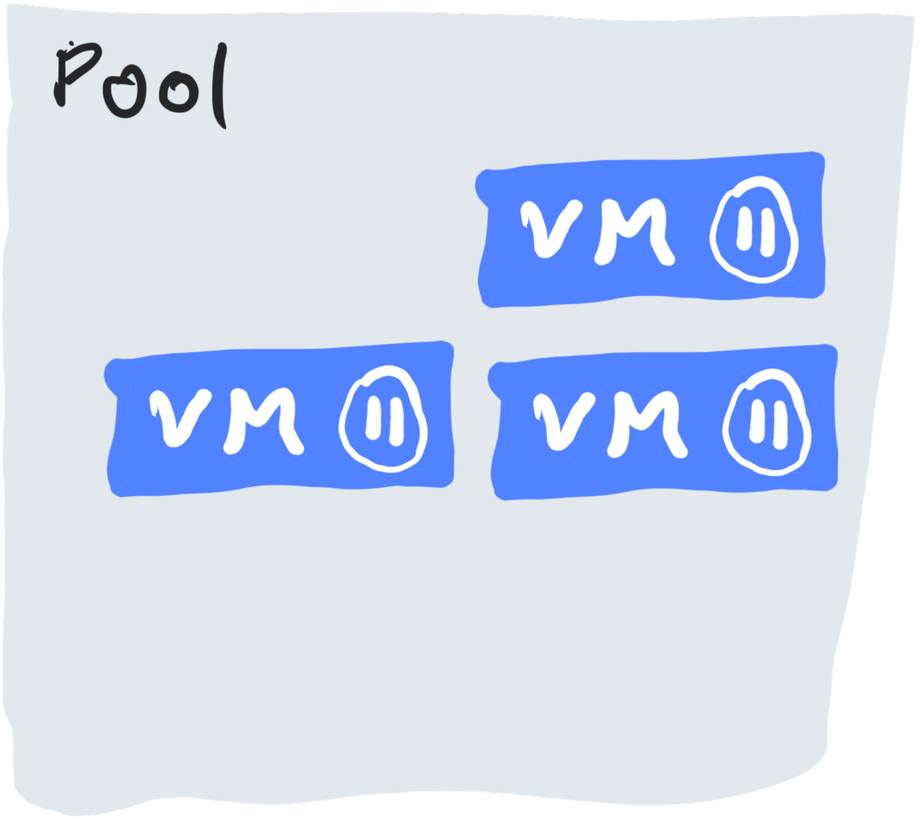
Создаём экземпляр

- Взяли виртуалку



Создаём экземпляр

- Взяли виртуалку
- Смонтировали диск
- Добавили память



Memory hot plug

- Добавляем 128 MB — Ok
- Добавляем 512 MB — Fail

Для добавления памяти нужна память

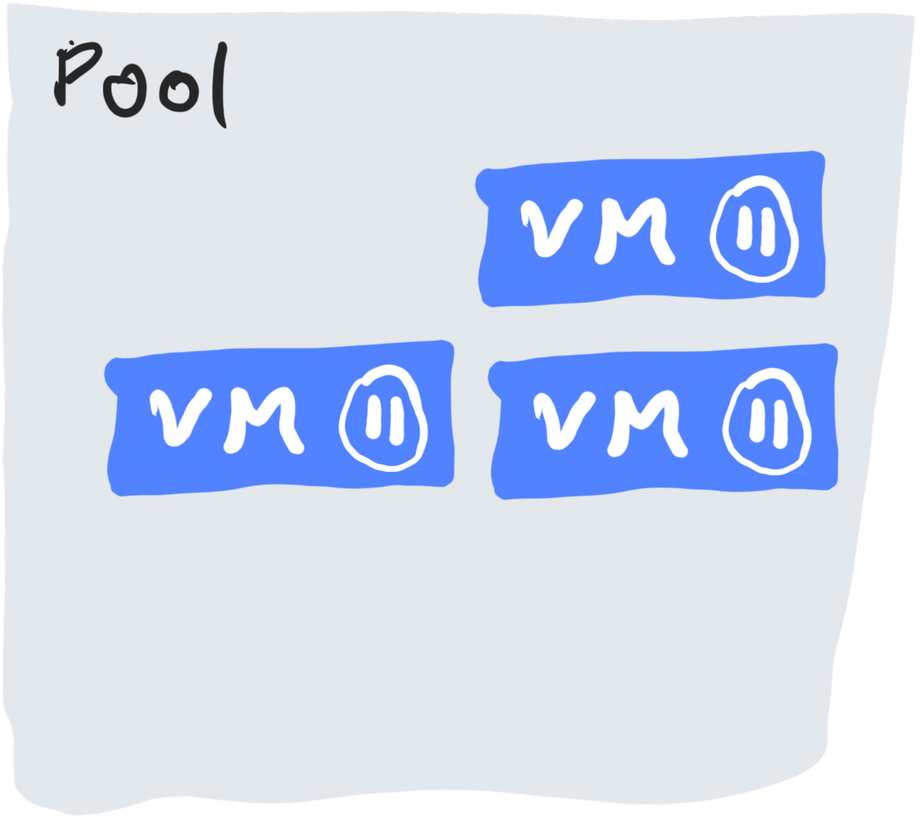
- Добавляем 128 MB — Ok
- Добавляем 512 MB — Fail

Двухэтапное добавление памяти

- Добавляем 128 MB
- Добавляем всё остальное

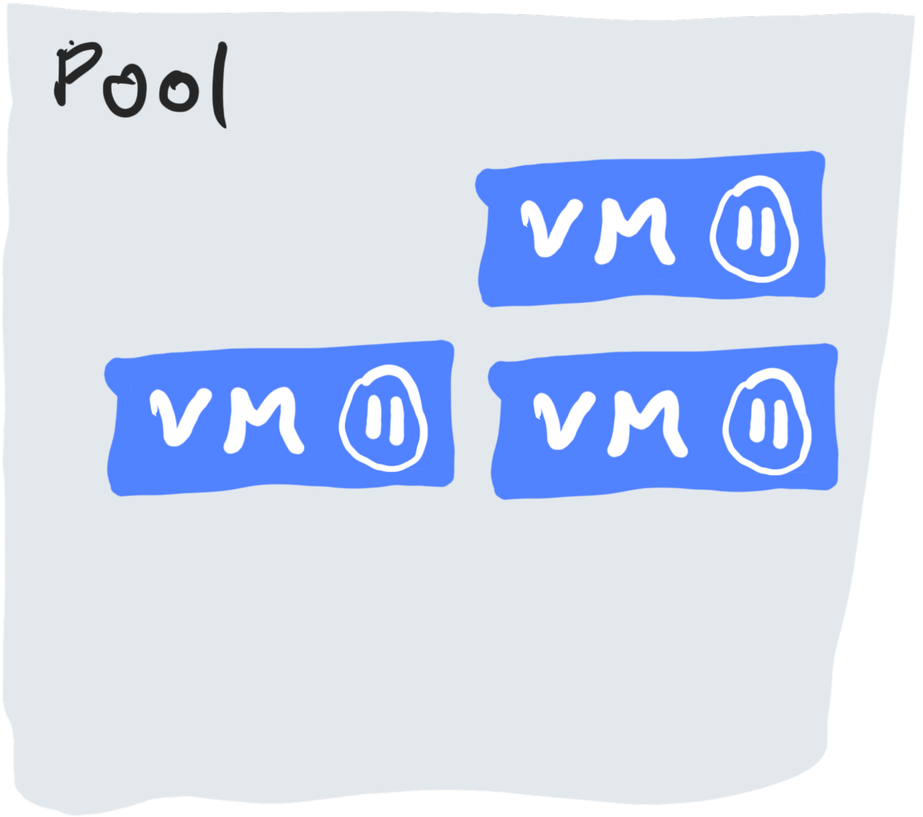
Создаём экземпляр

- Взяли виртуалку
- Смонтировали диск
- Добавили память



Создаём экземпляр

- Взяли виртуалку
- Смонтировали диск
- Добавили память
- Запускаем Serverless Runtime



Запуск Runtime

Проставляем переменные окружения
и запускаем Runtime с помощью виртуальной
консоли QEMU

Проблемы
с Environment variables

- Команды — Ok
- Переменные окружения — не всегда

Проблемы с Environment variables

- Консольный буфер 1,5 KB
- Всё, что больше, — в /dev/null

Проблемы с Environment variables

- Консольный буфер 1,5 KB
- Всё, что больше, — в /dev/null

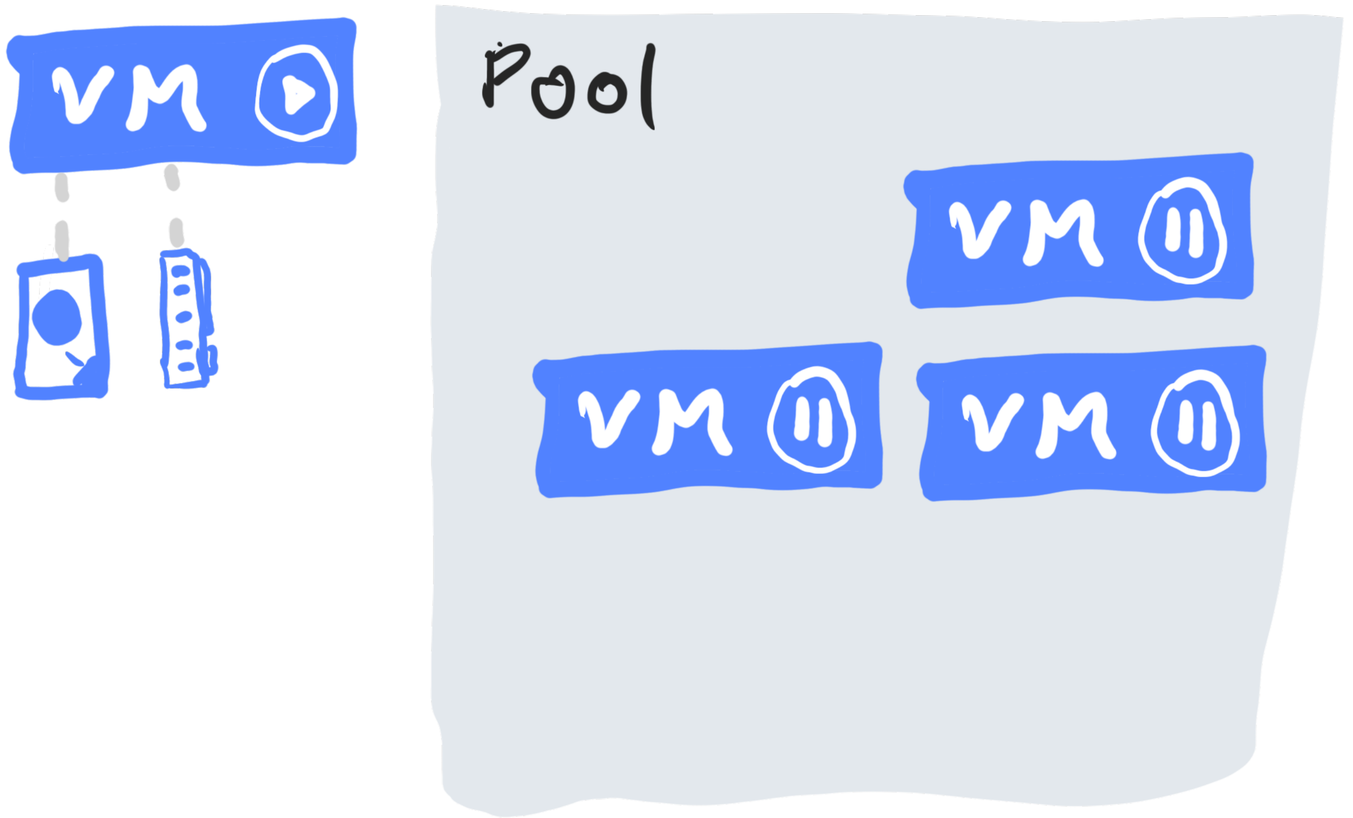
- Отправляем команды кусочками

Неправильное время

Перед выполнением функции синхронизируем время

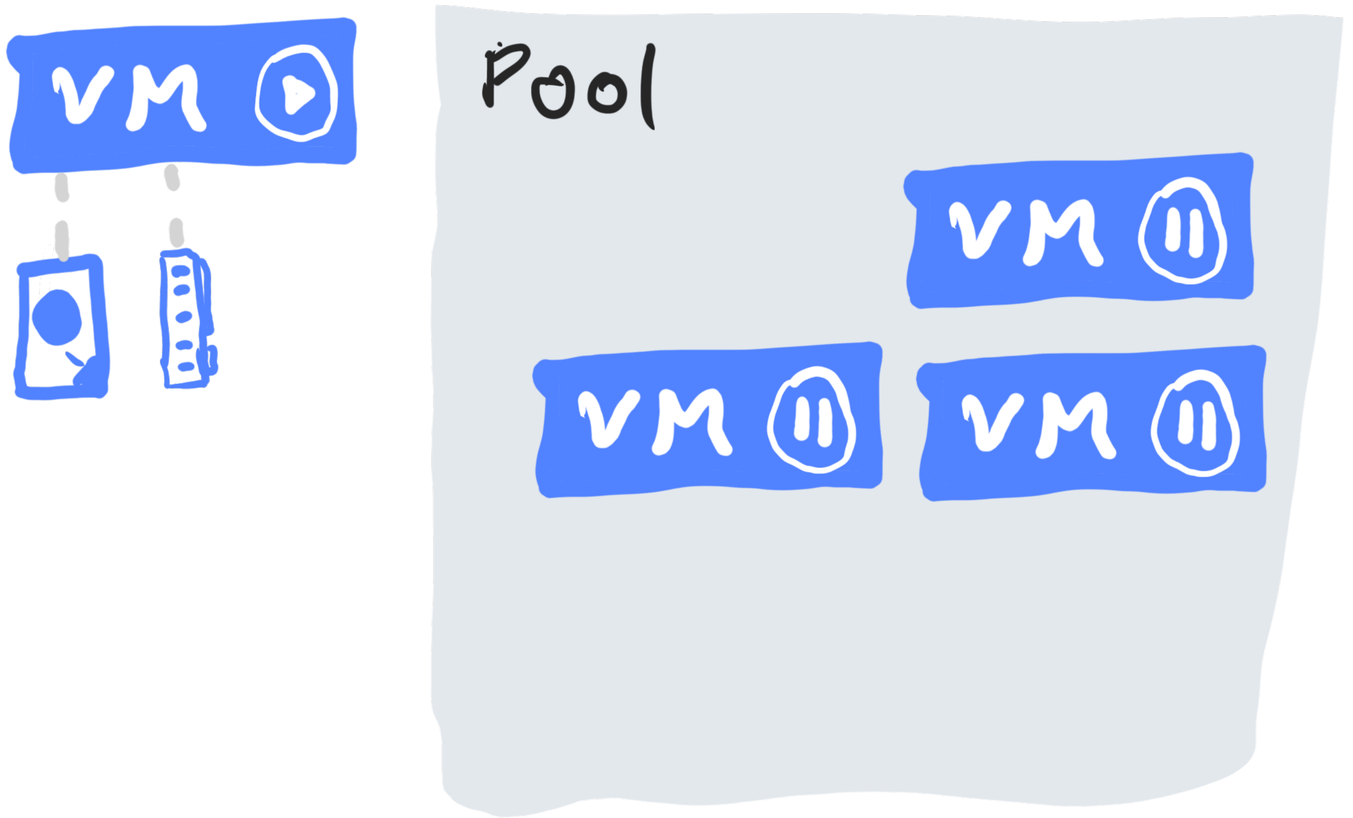
Обработка запроса

- Взяли задачу



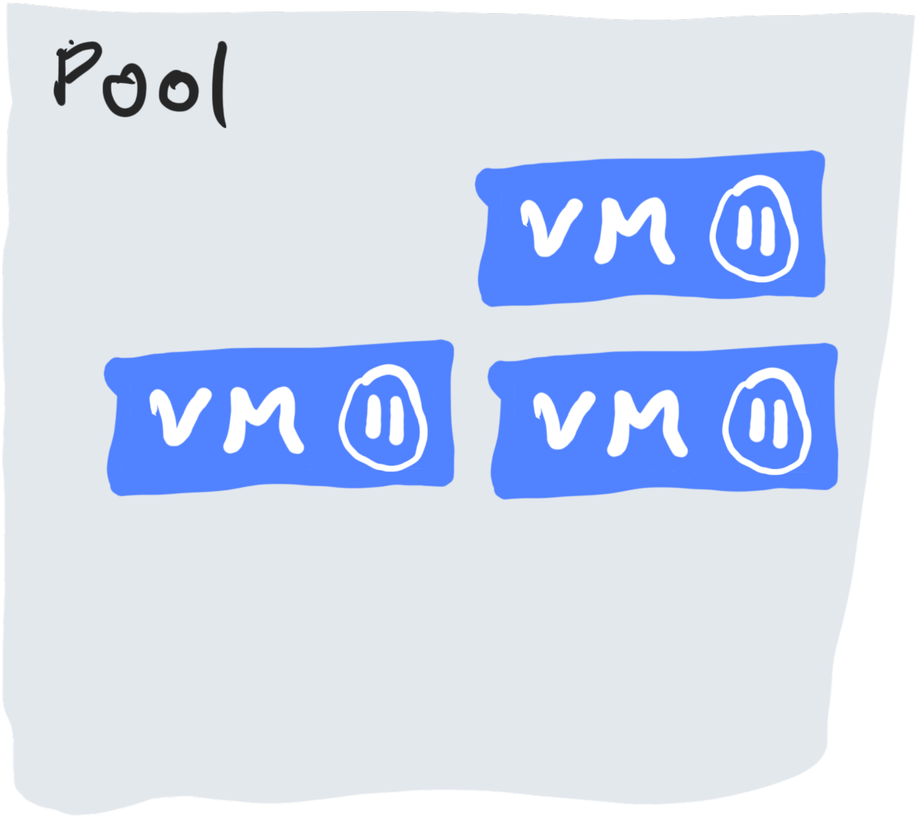
Обработка запроса

- Взяли задачу
- Выполнили её



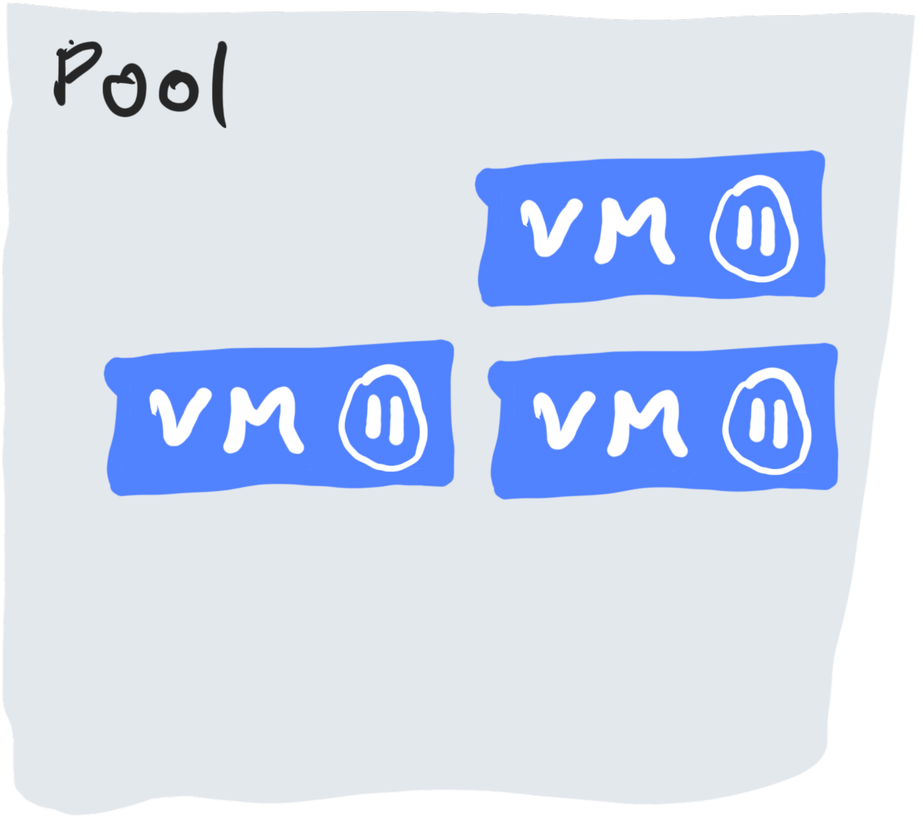
Обработка запроса

- Взяли задачу
- Выполнили её
- Suspend



Обработка запроса

- Взяли задачу
- Выполнили её
- Suspend
- Планировщик может решить потушить экземпляр



Down Scaling

«Скушаем»
все ресурсы

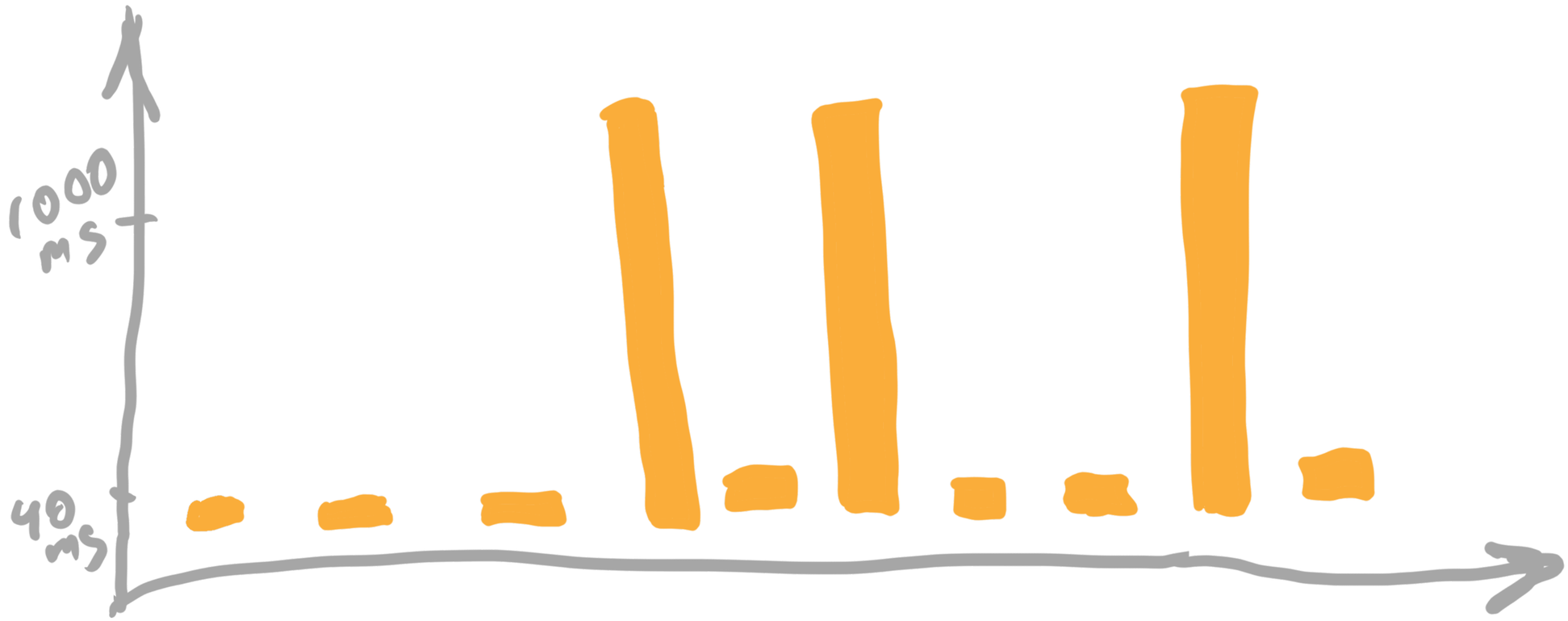
Много
Cold Start

Down Scaling

- Если экземпляр долго не работал — прибиваем
- Насколько долго — зависит

А могут ли быть проблемы, если microVM стартует быстро?

Холодный старт больше секунды



Холодный старт больше секунды

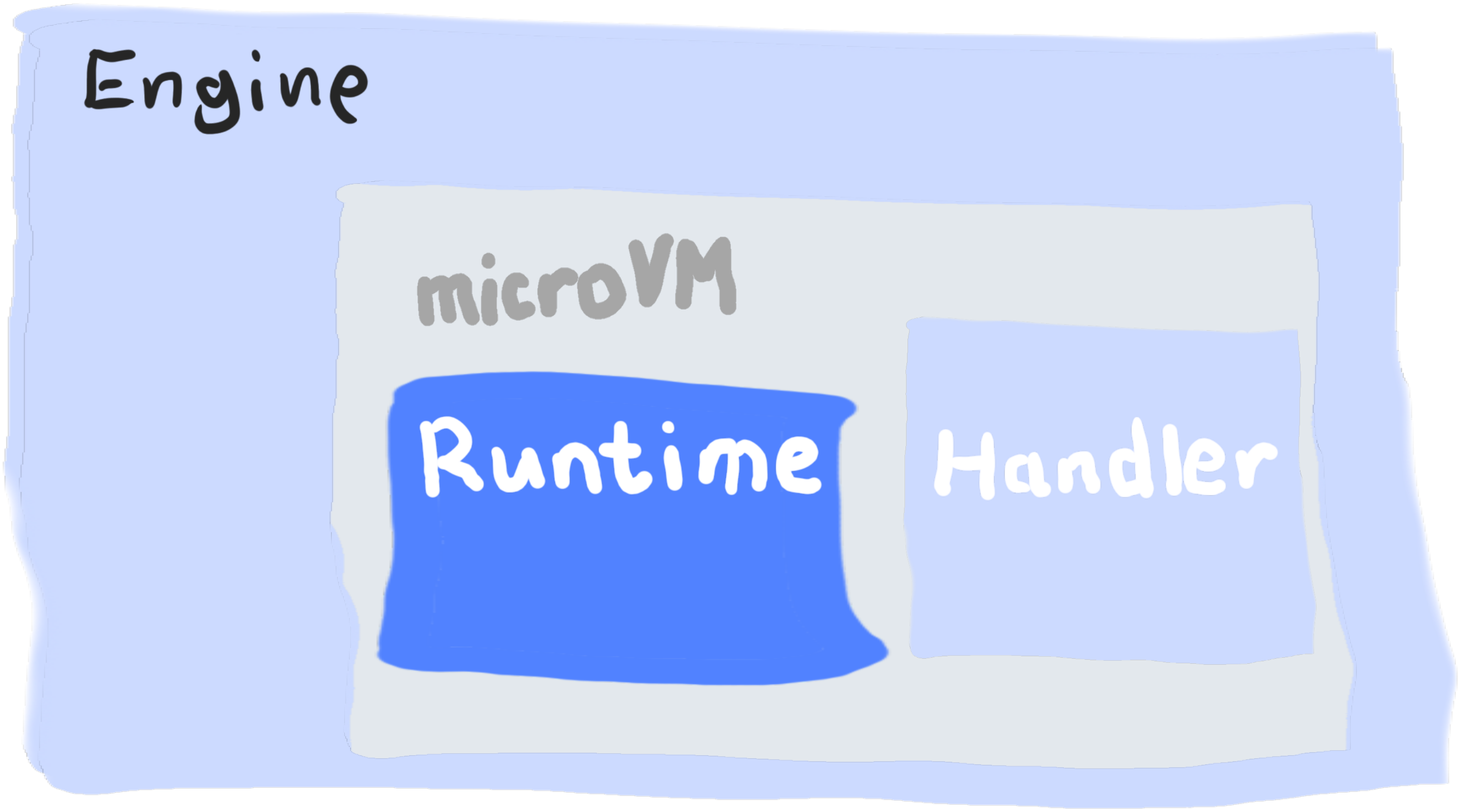
- На DNS-запрос не получили ответ
- Retry timeout настраивается в секундах

Холодный старт больше секунды

- На DNS-запрос не получили ответ
- Retry timeout настраивается в секундах

- Исправили libc
- Поправили работу с сетью

Serverless Runtime



Задачи Serverless Runtime

- Загрузить обработчик
- Взять из очереди задачу
- Десериализовать
- Вызвать обработчик
- Сериализовать ответ / обработать ошибки

Получение задач

- Ходим к Engine через HttpClient
- Десериализуем через System.Text.Json

DLL hell

- Runtime ссылается на сборку System.Text.Json
- Функция тоже ссылается на System.Text.Json,
  но новой версии

- Используем AssemblyLoadContext

Вызов Handler

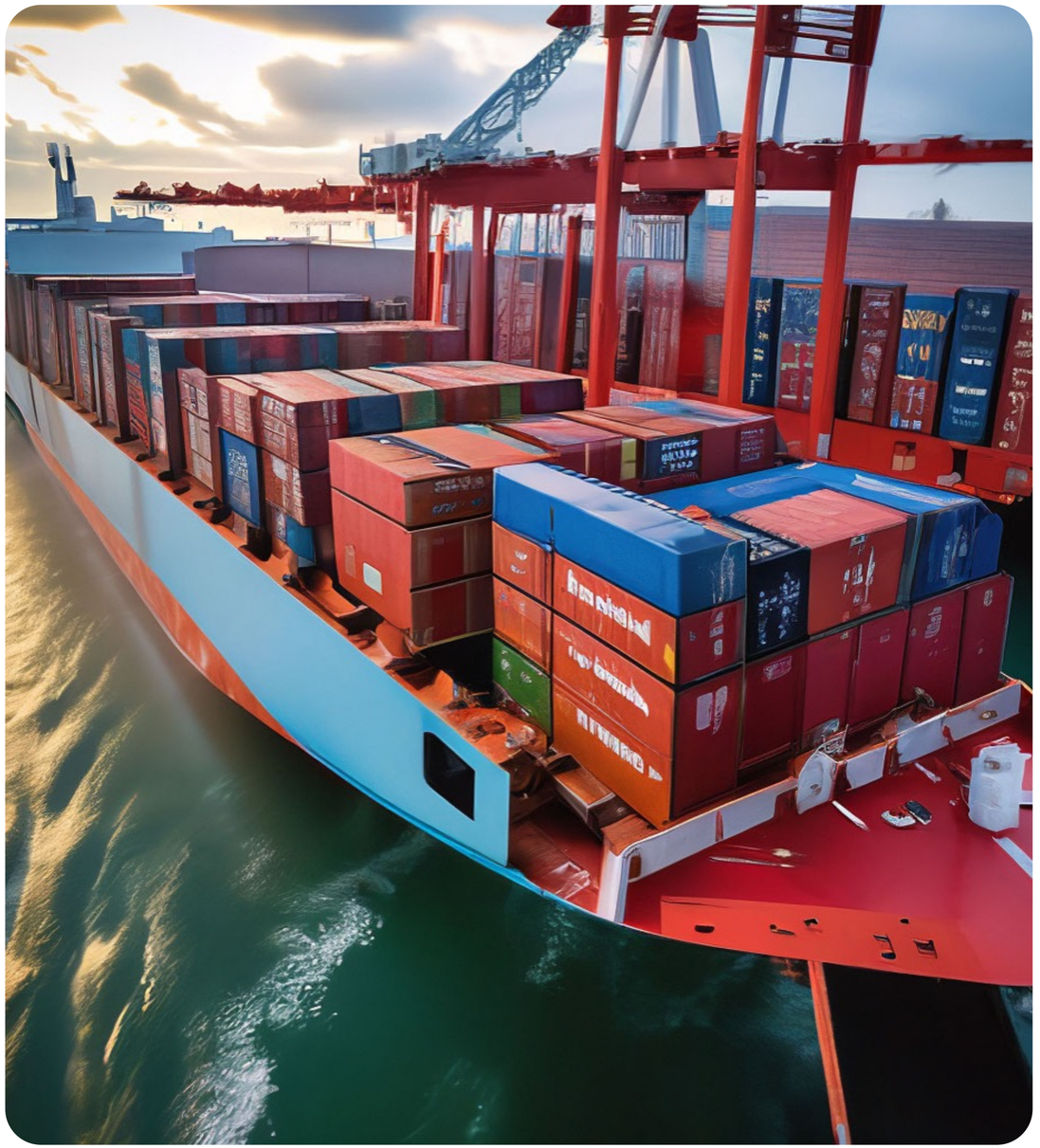
```
public class Call : Handler
{
  private dynamic _next;

  public virtual object FunctionHandler(Request param)
  {
    dynamic arg = PrepareParam(param);
    return next.FunctionHandler(arg);
  }

  private dynamic PrepareParam(Request param) => { ... }
}
```

Уменьшаем время Cold Start

- Скомпилированы с ReadyToRun

Ограниченные ресурсы

- Напоминает работу в ограниченном контейнере
- Выставлена переменная окружения DOTNET_RUNNING_IN_CONTAINER



WorkstationGC

- Минимальные GC паузы
- Потребляет меньше памяти

Настройки через переменные окружения

- Отключить/включить Tired PGO: DOTNET_TieredPGO=1

https://learn.microsoft.com/ru-ru/dotnet/core/runtime-config/

Настройки через переменные окружения

- Отключить/включить Tired PGO: DOTNET_TieredPGO=1

- Thread Pool — не настраивается

https://learn.microsoft.com/ru-ru/dotnet/core/runtime-config/

Будущее

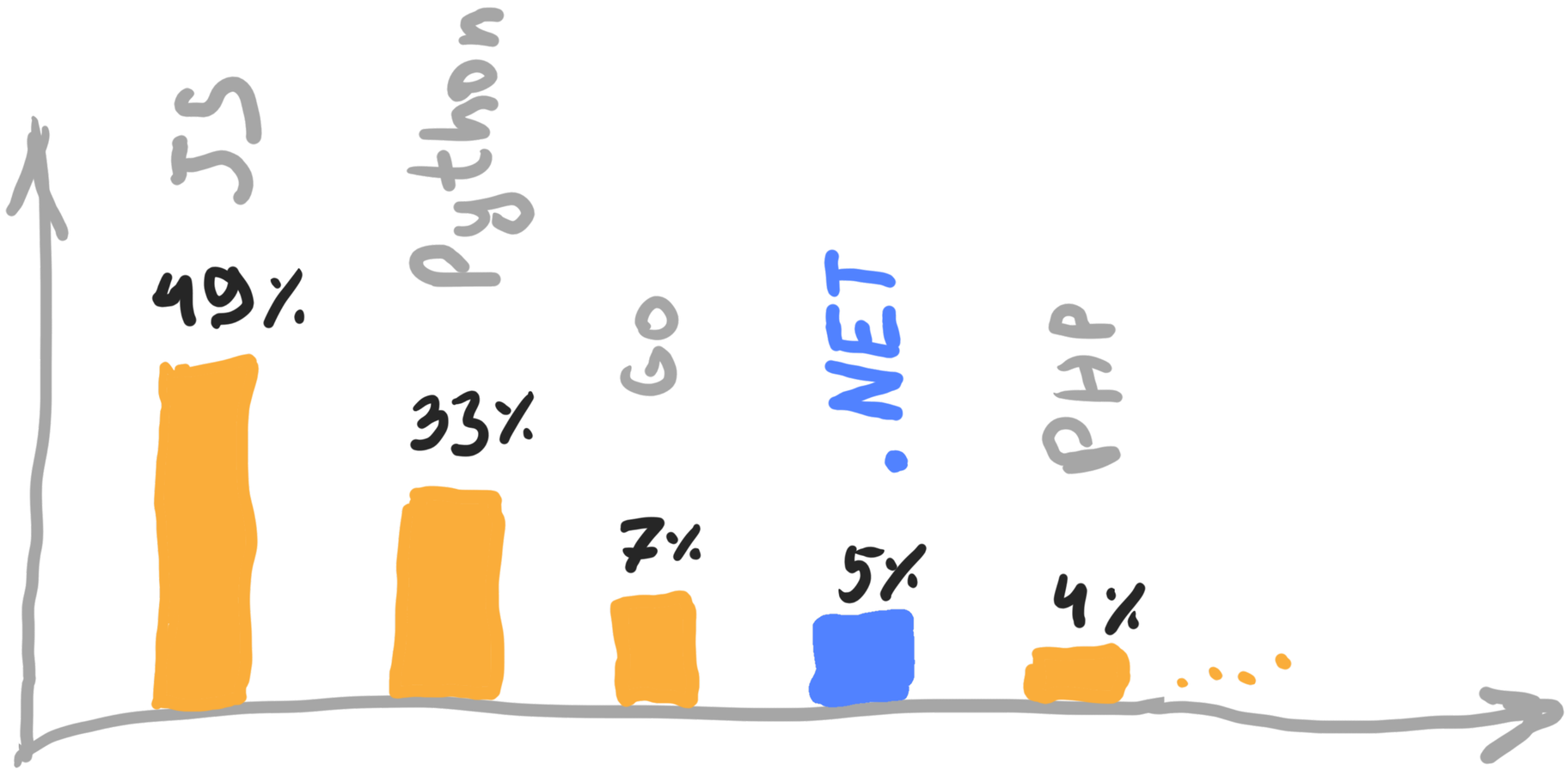
- Компиляция, AOT
- .NET 8 Runtime

1. Что такое Serverless

2. Как работают Cloud Functions

3. Внутреннее устройство Cloud Functions

Использование рантаймов



Если не пользовались — попробуйте.
Если уже пользовались — приходите поговорить

Что дальше:

- Чат с комьюнити: https://t.me/YandexCloudFunctions
- Примеры: https://github.com/yandex-cloud-examples
- Поиграть в Serverless Game: https://game.serverless.yandexcloud.net/login.html
  Исходники GitHub: https://clck.ru/zLrZ5

https://maxshoshin.ru/dotnetru2023.html