


Проверить, идет ли запись!





Меня хорошо видно && слышно?

Ставьте , если все хорошо
Напишите в чат, если есть проблемы

Введение в параллелизм в .NET. Отличия процесса, потока, домена и задачи



Усманов Александр
Ведущий разработчик, стаж 10+ лет
<https://t.me/AlexanderUsmanov>

Правила вебинара



Активно участвуем



Задаем вопрос в чат или голосом



Off-topic обсуждаем в Slack #канал группы или #general



Вопросы вижу в чате, могу ответить не сразу. Сегодня много тем, не на все вопросы смогу дать точный ответ, можно поискать самим и написать в чат или ответу в Slack

Маршрут вебинара

Параллелизм



Процессы и домены



Потоки



Таски

Цели вебинара | После занятия вы сможете

1 Дать определение основным понятиям параллелизма

2 Сформулировать отличия процессов, потоков, доменов приложений и тасков

3 Использовать эти понятия в задачах для обработки данных на C# на основе примера задачи

Смысл | Зачем вам это уметь

1 Один из самых главных инструментов в работе программиста, задают вопросы на собеседованиях

2 Позволяет улучшить производительность обработки данных

3 Большая часть рабочих проектов явно или неявно используют параллелизм

Задача

Нам необходимо сделать маркетинговую рассылку пользователям в разные каналы, сообщения для рассылки уже сформированы и сохранены в базе данных.

У сообщения есть:

- Id сообщения
- Тип:
 - ☐ Email
 - ☐ Sms
 - ☐ Push
- Текст сообщения
- Id пользователя, которому нужно доставить сообщение
- Дата создания сообщения



Тайминг: 3-5 минут

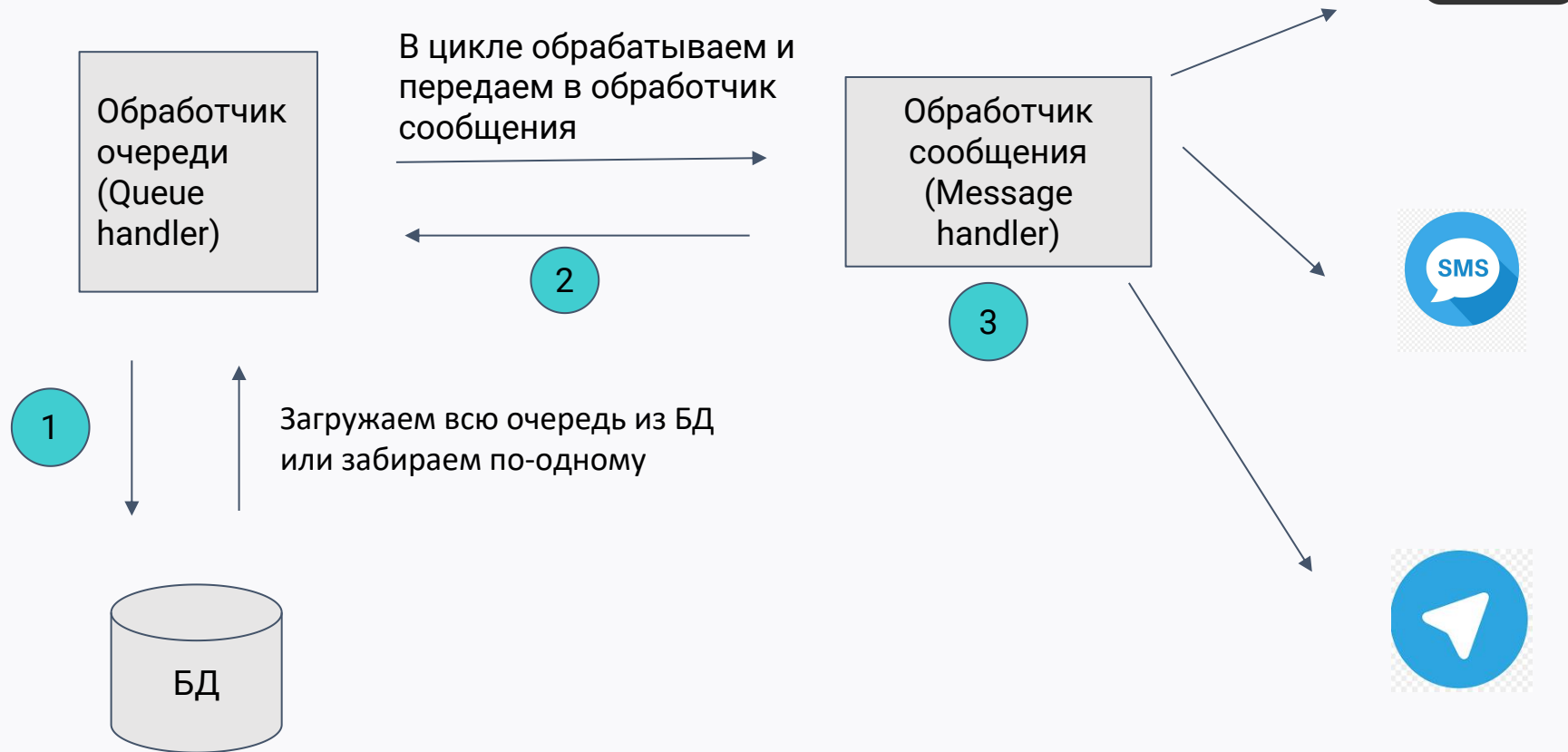
Сообщения отсортированы по возрастанию даты создания.

Нужно сделать приложение, которое будет обрабатывать сообщения из БД и передавать их другому компоненту на отправку.

Необходимо в чат описать общую логику работы алгоритма обработки сообщений из БД, то есть как бы вы решали эту задачу, общими словами (например, мы берем сообщения из БД и делаем ...)

Алгоритм последовательной обработки

Так как сообщения отсортированы по дате создания, то будем считать их очередью сообщений для отправки.



Пример реализации



Тайминг: 1-2
минуты

Кто хочет посмотреть код примера параллельно, клонируем
к себе из репозитория

<https://gitlab.com/devgrav/Otus.Teaching.Concurrency.Queue>

The image features a high-angle, aerial view of a dense urban landscape, likely New York City, characterized by numerous skyscrapers and buildings. The entire image is rendered in a monochromatic blue and green color scheme. A central horizontal band, transitioning from a vibrant green on the left to a deep blue on the right, serves as a background for the word "CODE". This band is overlaid with a subtle, white geometric pattern of interconnected lines and dots, resembling a network or data structure. The word "CODE" is written in a large, bold, white, sans-serif typeface, centered within this band.

CODE

Вопрос



Тайминг: 2-3
минуты

Какие могут быть проблемы при простой
последовательной обработке сообщений?

Напишите в чат плюсы и минусы такого решения

Основная проблема: медленная обработка, если сообщений становится много или если они постоянно поступают

Решение: надо подумать, как увеличить количество обработчиков



The background of the slide is a blue-tinted aerial photograph of a dense city skyline, likely New York City, featuring numerous skyscrapers. Overlaid on this image is a semi-transparent network diagram consisting of a grid of points connected by thin lines, creating a mesh-like pattern across the center of the slide.

Параллелизм

Маршрут вебинара

Параллелизм



Процессы и домены



Потоки



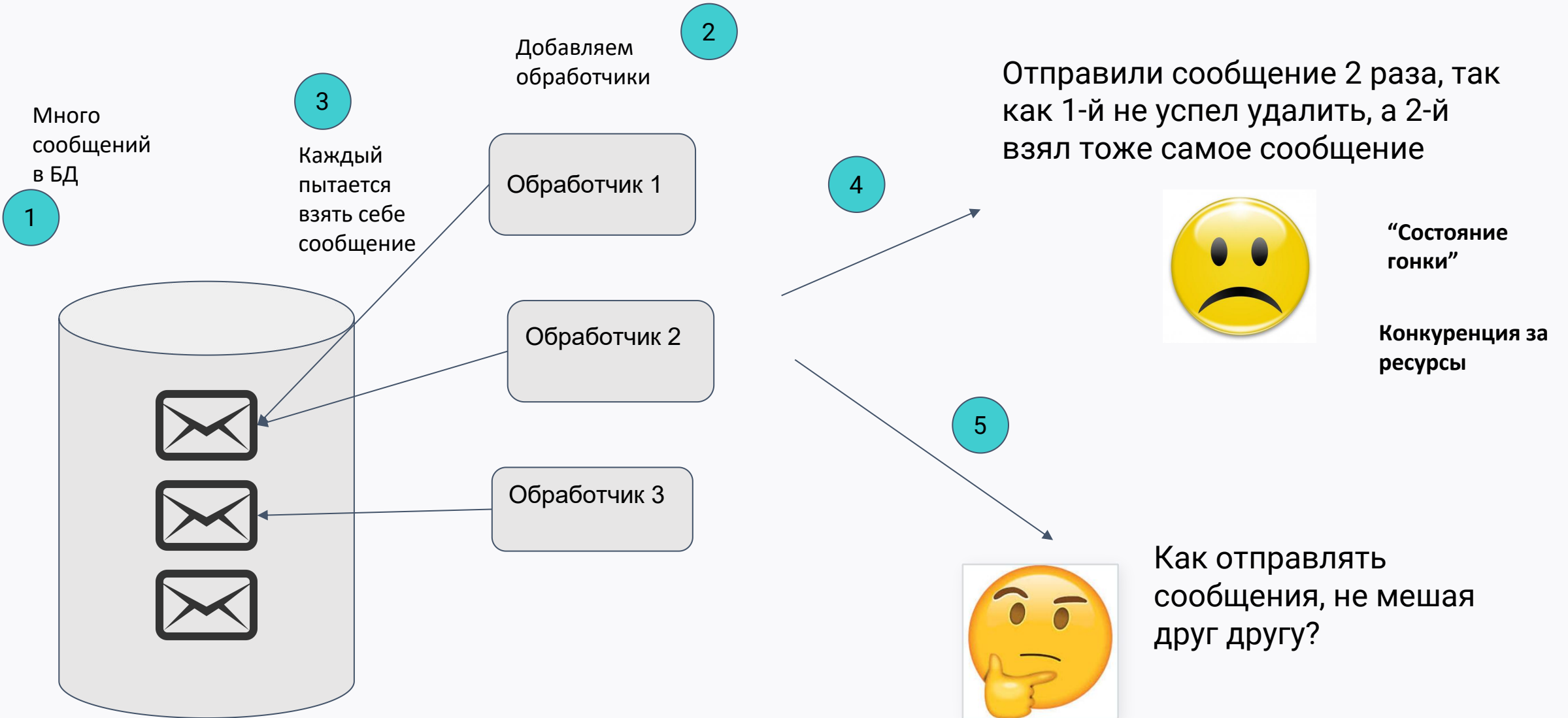
Таски

Параллелизм

Параллелизм — выполнение нескольких действий физически одновременно в единицу времени.

Попробуем реализовать...

Простой параллельный вариант



Вопрос



Тайминг: 2-3
минуты

Как мы можем реализовать параллельный вариант, чтобы
обработчики не мешали друг другу?
Напишите в чат или минус

Ответ

Разбиваем очередь сообщений на три по типу сообщения, делаем по обработчику на каждый тип.

Каждый забирает сообщения только своего типа и не мешает другим.

Задача

Нам необходимо сделать маркетинговую рассылку пользователям в разные каналы, сообщения для рассылки уже сформированы и сохранены в базе данных.

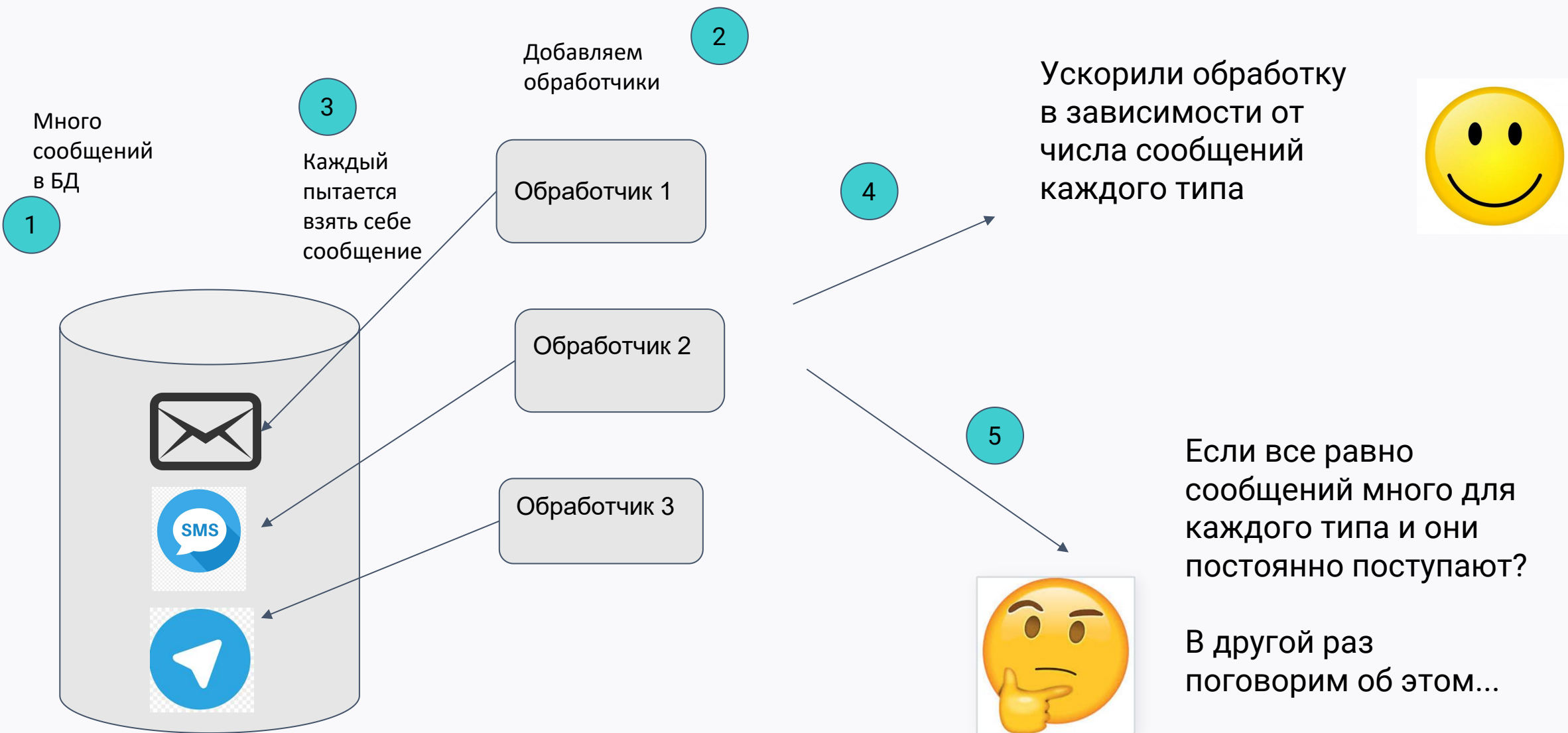
У сообщения есть:

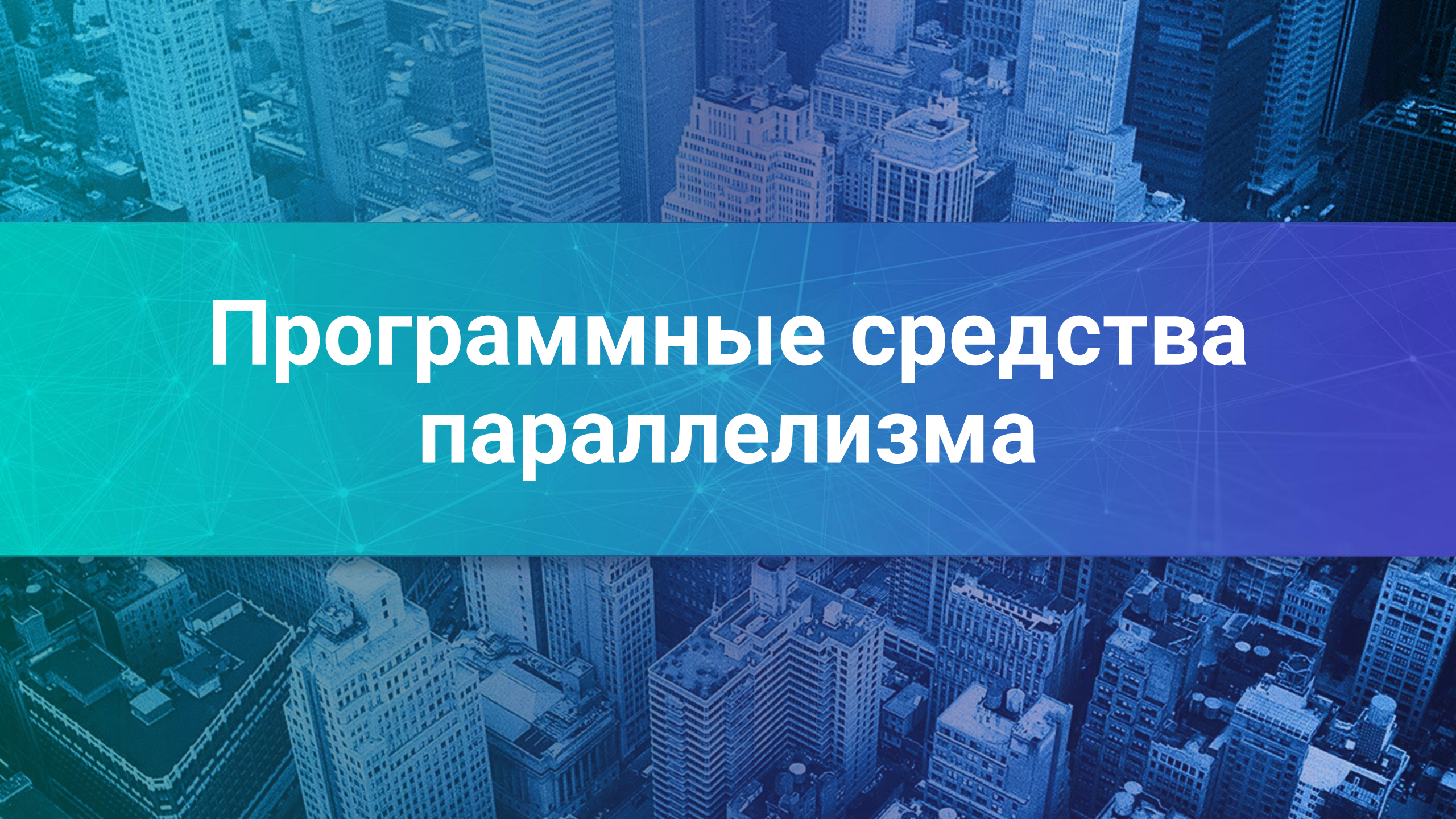
- Id сообщения
- Тип:
 - Email
 - Sms
 - Push
- Текст сообщения
- Id пользователя, которому нужно доставить сообщение
- Дата создания сообщения

Сообщения отсортированы по возрастанию даты создания.

Нужно сделать приложение, которое будет обрабатывать сообщения из БД параллельно обработчиками на каждый тип и передавать их другому компоненту на отправку.

Более параллельный вариант





Программные средства параллелизма

Вопрос

Кто участвовал в семинаре CLRium #6?
Напишите в чат + или -

Некоторые темы подготовлены по материалам этого семинара, рекомендуется для более глубокой проработки всего, что связано с параллелизмом



<https://clrium.ru/index.html>

Основные понятия и их уровни

Процесс (Process) — объект ОС, контейнер для кода, имеет изолированное адресное пространство в памяти, содержит потоки.

Поток (Thread) — объект ОС, наименьшая единица выполнения кода, часть процесса, потоки делят память и другие ресурсы между собой в рамках процесса.

Многозадачность — свойство ОС, возможность выполнять несколько процессов почти одновременно

Домен приложения (App Domain) - объект .NET, нужен для изоляции разного кода в одном процессе, аналогия процесса

Основные понятия и их уровни

Операционная система Windows

Процесс 1

Домен приложения 1

Поток
(Id 1)

Поток
(Id 4)

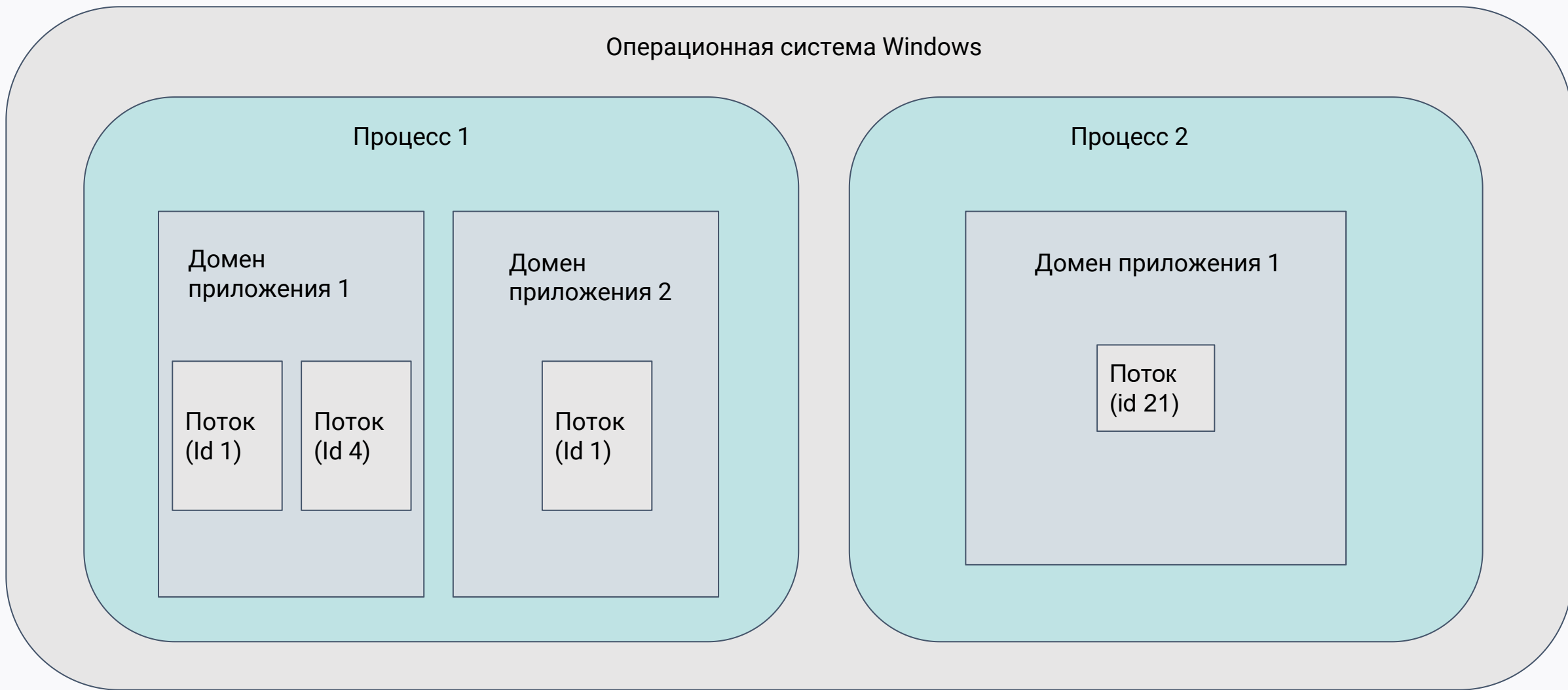
Домен приложения 2

Поток
(Id 1)

Процесс 2

Домен приложения 1

Поток
(id 21)



The background of the slide is a blue-tinted aerial photograph of a dense city skyline, likely New York City. Overlaid on this is a semi-transparent network pattern consisting of light blue lines connecting various points, creating a web-like structure. The title text is centered within this pattern.

Процессы и домены

Маршрут вебинара

Параллелизм



Процессы и домены



Потоки



Таски

The background of the slide features an aerial view of a dense city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer that contains a white network pattern of interconnected dots and lines, suggesting a digital or technological theme. The word "Процессы" is centered in the middle of the slide in a large, white, sans-serif font.

Процессы

Что такое процесс?

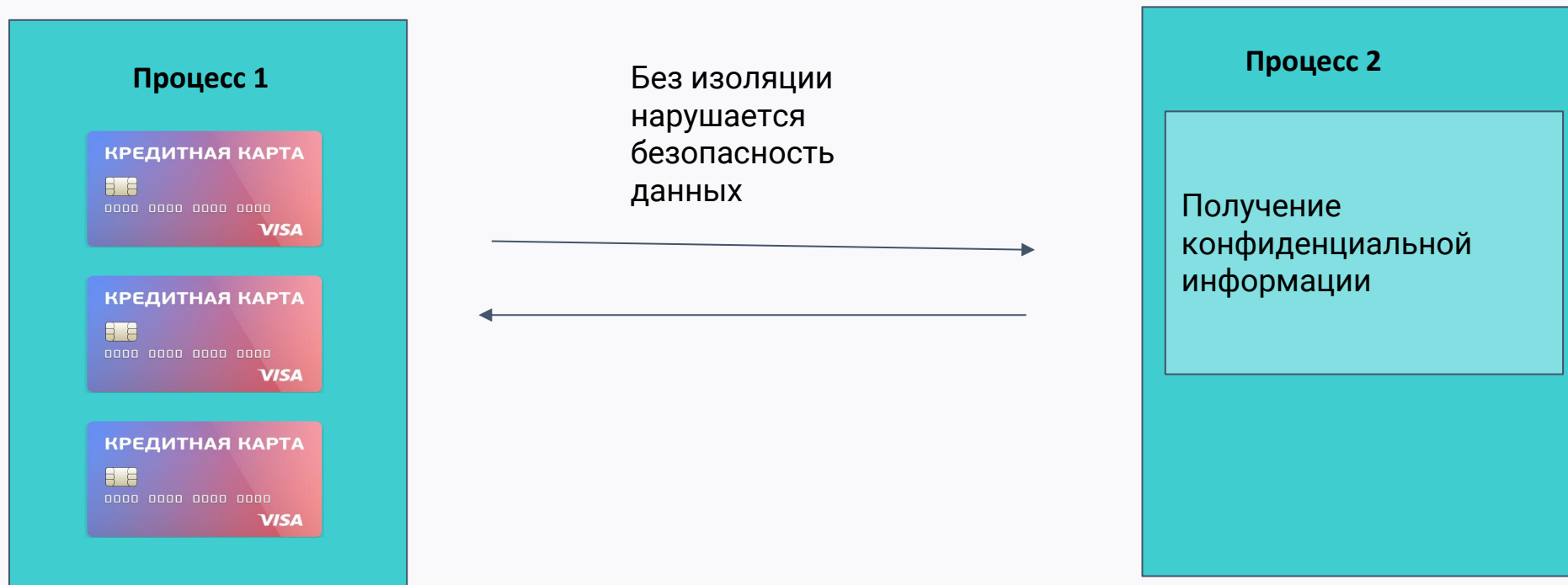
Процесс — это выполняемая в ОС программа и все, что ей необходимо для исполнения.

Процесс не имеет доступа к памяти другого процесса - это **механизм изоляции процессов**.

За обеспечение изоляции и за управление процессами отвечает ОС

Зачем нужна изоляция процессов?

Чтобы один процесс не получил доступ к конфиденциальной информации в памяти другого процесса или не повредил данные.



Что содержит процесс?

В Windows процесс включает в себя:

1. Закрытое виртуальное адресное пространство памяти, нужно для изоляции.
2. Исполняемая программа, то есть машинный код
3. Список открытых дескрипторов для I/O Streams (сеть, диски) - нужен, чтобы ОС могла закрыть порты ввода/вывода.
4. Маркер доступа для пользователя или группы - для уровней доступа процесса
5. Id процесса
6. Как минимум один главный поток исполнения - в нем выполняется код.

Процесс в .NET

Процессы с .NET кодом - это обычные процессы ОС, но в их память сначала загружается CLR, которая и выполняет IL код.

Вызывается функция
ОС для старта нового
процесса

```
static void Main(string[] args)
{
    var startInfo = new ProcessStartInfo()
    {
        ArgumentList = {"SomeInfo"},
        FileName = "/path/process.exe"
    };

    var process = Process.Start(startInfo);

    Console.WriteLine($"Process started with id: {process.Id}");
}
```


Задача

Нам необходимо сделать маркетинговую рассылку пользователям в разные каналы, сообщения для рассылки уже сформированы и сохранены в базе данных.

У сообщения есть:

- Id сообщения
- Тип:
 - Email
 - Sms
 - Push
- Текст сообщения
- Id пользователя, которому нужно доставить сообщение
- Дата создания сообщения

Сообщения отсортированы по возрастанию даты создания.

Нужно сделать приложение, которое будет обрабатывать сообщения из БД параллельно обработчиками на каждый тип и передавать их другому компоненту на отправку.

Вопрос

Можем ли мы решить задачу на основе процессов и как это сделать?

Ответ

Для каждого типа сообщений создаем по процессу и в них последовательно обрабатываем свой набор сообщений

The image features a high-angle, aerial view of a dense urban landscape, likely New York City, characterized by numerous skyscrapers and buildings. The entire image is rendered in a monochromatic blue and green color scheme. A central horizontal band, transitioning from a vibrant green on the left to a deep blue on the right, serves as a background for the word "CODE". This band is overlaid with a subtle, white geometric pattern of interconnected lines and dots, resembling a network or data structure. The word "CODE" is written in a large, bold, white, sans-serif typeface, centered within this band.

CODE

Вопрос



Тайминг: 1-2
минуты

Какие есть проблемы в решении задачи на процессах?

Напишите в чат плюсы и минусы такого решения

Ответ

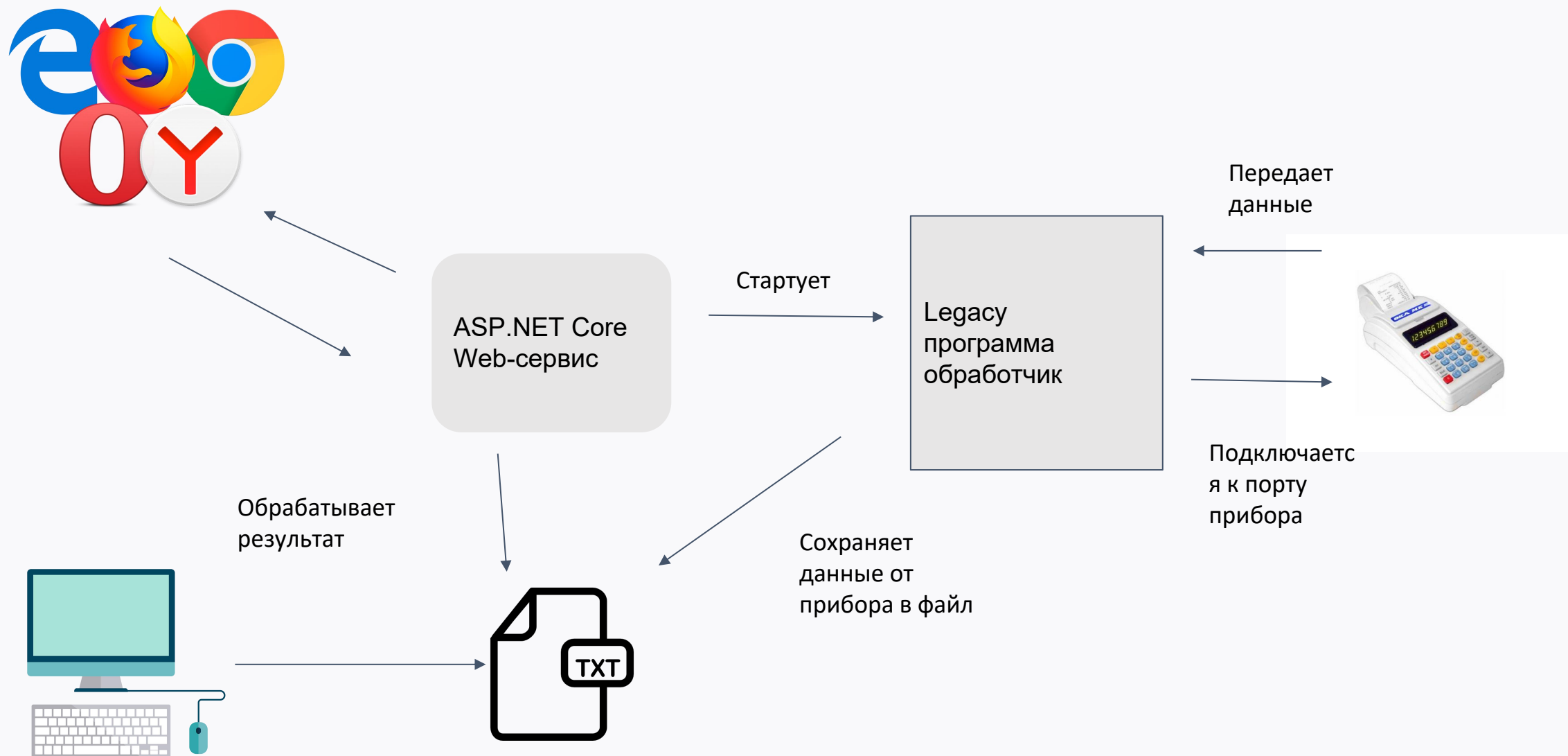
Плюсы:

1. Независимость обработчиков;
2. Можно независимо увеличивать число обработчиков для типа внутри одного процесса (через потоки);

Минусы:

1. Сложность управления процессами;
2. Порождение процессов является трудоемкой операцией, так как это функция ядра ОС;
3. Если умрет процесс-планировщик, то процессы-обработчики могут остаться работать дальше или их никто не запустит снова;

Пример для создания процесса



The background of the image is a high-angle, blue-tinted aerial photograph of a dense urban skyline, likely New York City. The image is divided into three horizontal sections. The top and bottom sections show the city buildings. The middle section is a solid blue band with a white, glowing network pattern of lines and dots. The text "Домены приложений" is centered in this middle band.

Домены приложений

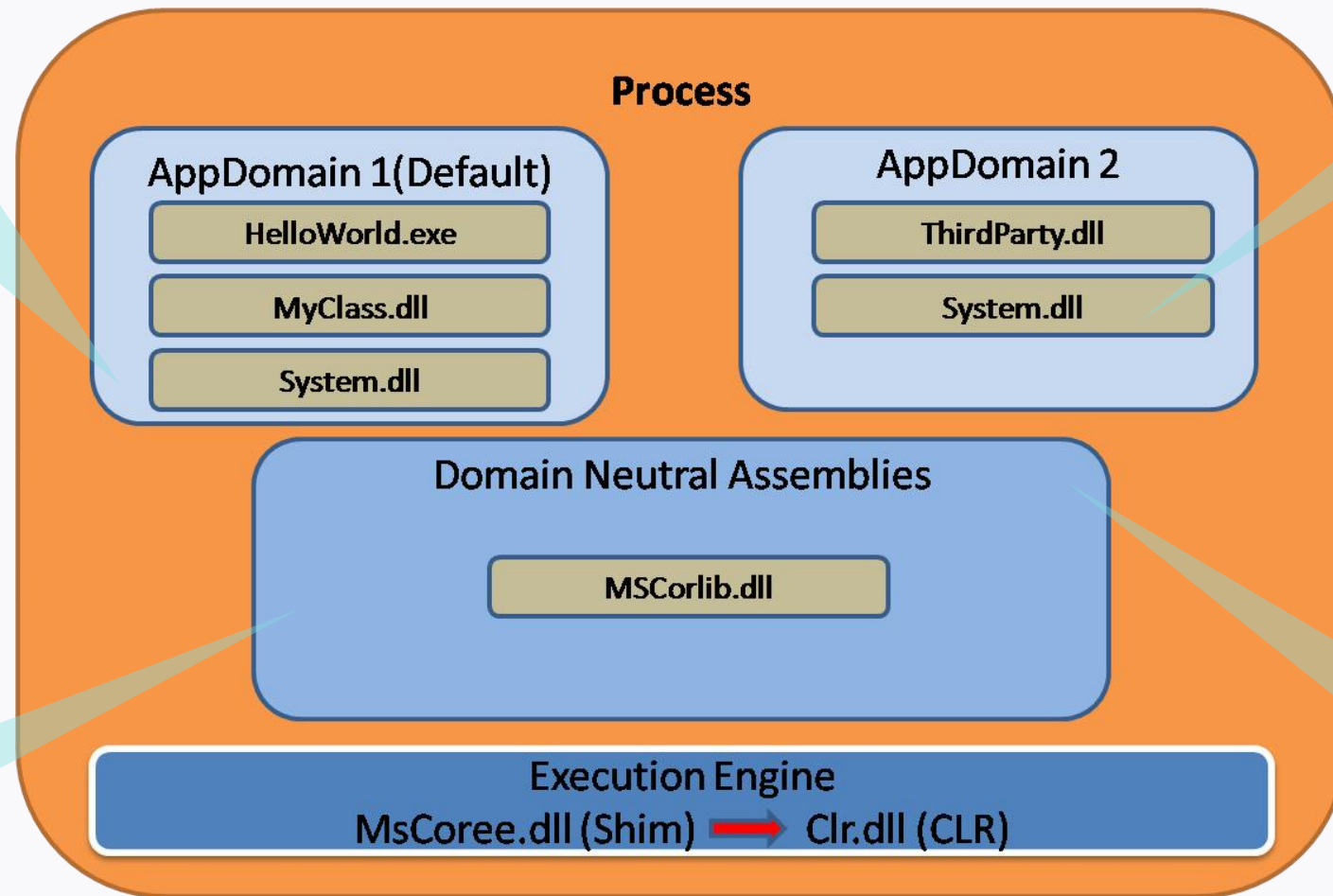
Что такое домен приложения?

1. Единица изоляции программного кода в .NET Framework
2. Аналогия процесса ОС
3. Контейнер для загрузки сборок .NET в память процесса.
4. Процесс может в своей памяти содержать несколько доменов/приложений

Домены приложений в процессе .NET Framework

В домене все сборки для работы конкретного приложения, несколько доменов в процессе, один из доменов Default

Сборки CLR загружаются один раз в память процесса



Даже общие сборки загружаются в каждый домен для изоляции

Это общий домен (Shared Domain) Он создается через системный домен (System Domain) при старте CLR

Вопрос



Тайминг: 1-2
минуты

Отличается ли работа доменов приложений в .NET Framework и .NET Core?

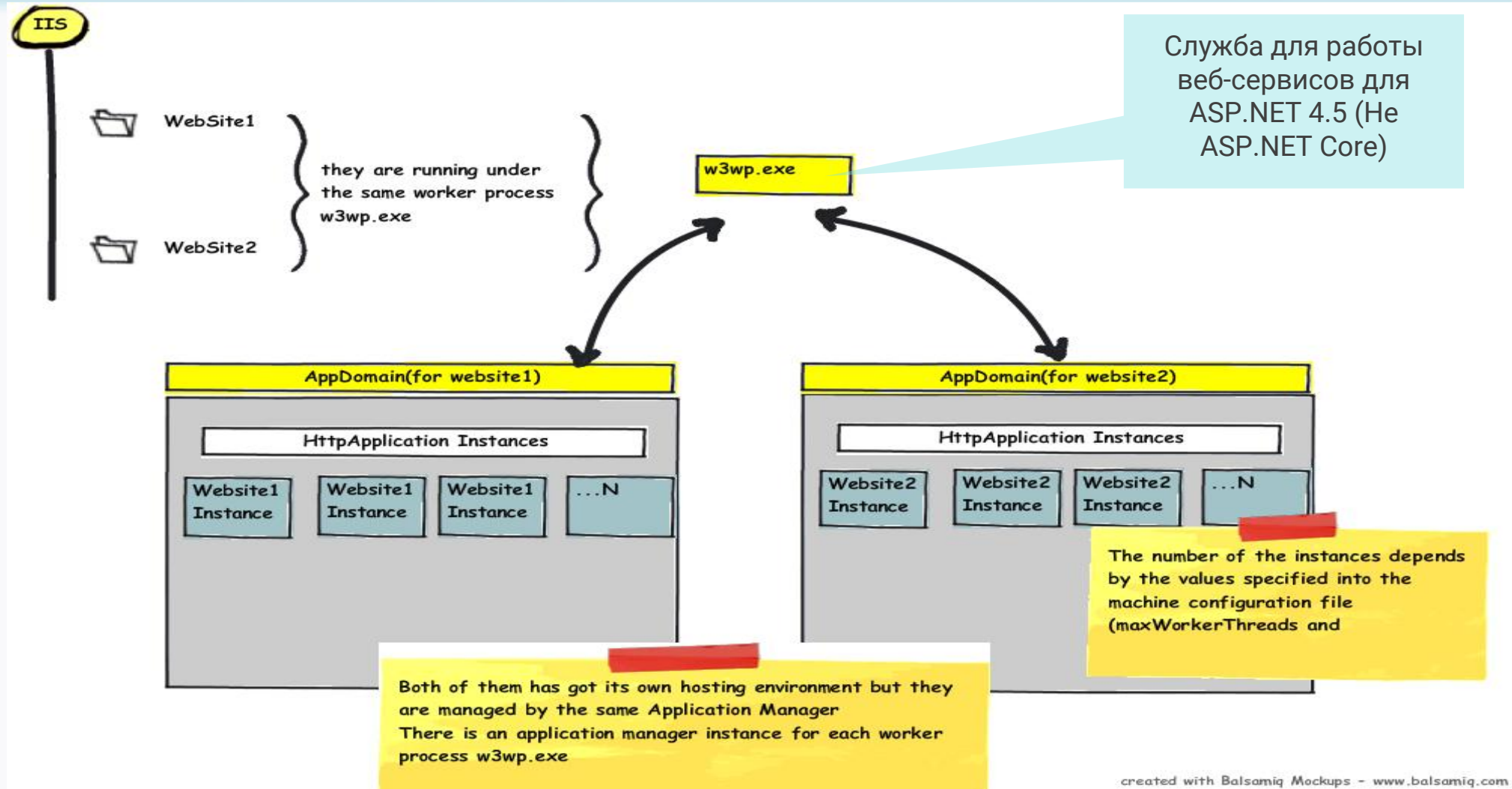
Напишите в чат ответ или минус, если надо сразу
рассказать

Домены приложений в процессе .NET Core

1. *AppDomain* оставлен для обратной совместимости и не будет поддерживаться;
2. Не обеспечивает изоляцию. Загрузка/выгрузка сборок сделана через класс *AssemblyLoadContext*;
3. По умолчанию при старте создается один *AppDomain*, новые создавать нельзя;
4. Изоляция за счет создания отдельных процессов, а не доменов;

<https://docs.microsoft.com/ru-ru/dotnet/core/porting/net-framework-tech-unavailable>

Использование доменов в .NET Framework



The background of the image is an aerial view of a dense city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue and green gradient. A network of white lines and dots, resembling a data or communication network, is visible across the center. The word "Потоки" is written in large, white, bold Cyrillic letters in the center.

Потоки

Маршрут вебинара

Параллелизм



Процессы и домены



Потоки



Таски

Что такое поток?

Поток - это механизм операционной системы, который позволяет выполнять на одном процессоре почти одновременно несколько задач, за счет того, что каждой задаче выдается небольшой промежуток времени, также он имеет доступ ко всем ресурсам процесса.

Для многоядерных систем можно говорить о полностью параллельном исполнении.

Процессор не знает про потоки, он просто выполняет какой-то машинный код, это инструмент ОС.

Поток в .NET

Потоки в .NET - это обычные потоки ОС, объект Thread хранит общую информацию о созданном в ОС потоке выполнения

Иногда .NET потоки называют управляемыми (Managed), но на самом деле Thread это объект для лучшего контроля за приложением для CLR. CLR хранит ссылки на все потоки приложения.

```
static void Main(string[] args)
{
    Console.WriteLine($"Started in thread: {Thread.CurrentThread.ManagedThreadId}");

    var thread = new Thread( start: () => Console.WriteLine($"Logged in thread: " +
                                                            $"{Thread.CurrentThread.ManagedThreadId}"));
    thread.Start();
    thread.Join();
}
```

Вызывается
метод ОС -
CreateThread

Вопрос

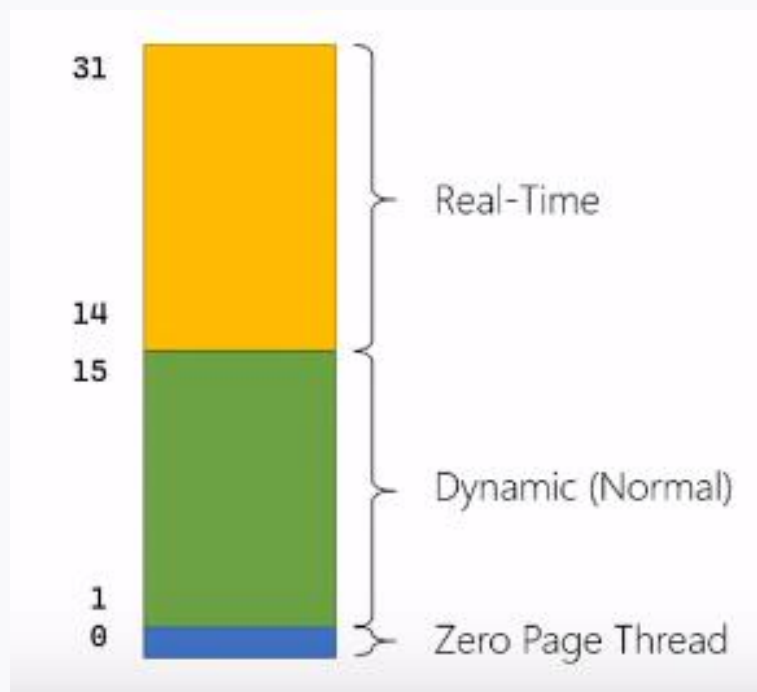


Тайминг: 1-2
минуты

Кто в практике использовал явно Thread класс и зачем?
Напишите в чат или минус

Приоритеты потоков

Приоритеты потоков



Класс приоритета процесса, влияет на базовый приоритет потока

	Имя	Класс	Базовый Приоритет
1.	Real-Time	4	24
2.	High	3	13
3.	Above Normal	6	10
4.	Normal	2	8
5.	Below Normal	5	6
6.	Idle	1	4

Вопрос



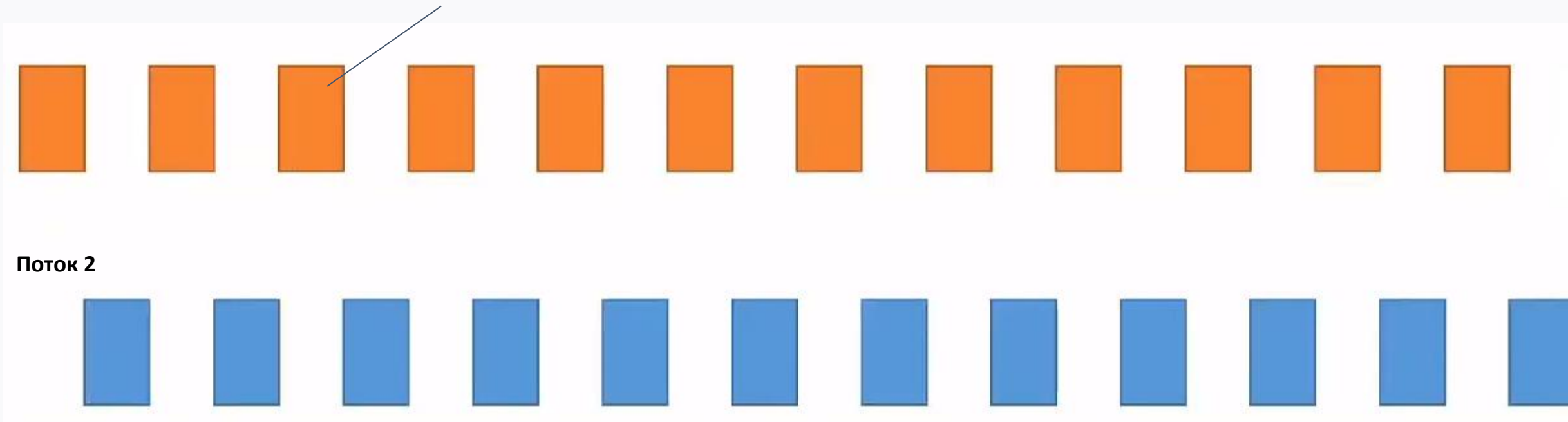
Тайминг: 1-2
минуты

Нужно ли рассказать про кванты времени для потоков?
Напишите в чат о чем это

Кванты времени для потоков

Поток 1

Квант времени



Увеличение числа потоков ведет к уменьшению квантов и в итоге к тому, что они будут ждать ядра, то есть увеличение числа потоков не значит увеличение производительности

Задача

Нам необходимо сделать маркетинговую рассылку пользователям в разные каналы, сообщения для рассылки уже сформированы и сохранены в базе данных.

У сообщения есть:

- Id сообщения
- Тип:
 - Email
 - Sms
 - Push
- Текст сообщения
- Id пользователя, которому нужно доставить сообщение
- Дата создания сообщения

Сообщения отсортированы по возрастанию даты создания.

Нужно сделать приложение, которое будет обрабатывать сообщения из БД параллельно обработчиками на каждый тип и передавать их другому компоненту на отправку.

Вопрос

Как решить задачу на основе потоков?

Ответ

Для каждого типа сообщений создаем по потоку и в них последовательно обрабатываем свой набор сообщений

The image features a high-angle, aerial view of a dense urban landscape, likely New York City, characterized by numerous skyscrapers and buildings. The entire image is rendered in a monochromatic blue and green color scheme. A central horizontal band, transitioning from a vibrant green on the left to a deep blue on the right, serves as a background for the word "CODE". This band is overlaid with a subtle, white geometric pattern of interconnected lines and dots, resembling a network or data structure. The word "CODE" is written in a large, bold, white, sans-serif typeface, centered within this band.

CODE

Вопрос



Тайминг: 1-2
минуты

Какие есть плюсы и минусы в решении задачи на потоках?

Напишите ответ в чат

Ответ

Плюсы:

1. Удобство управления потоками;
2. Можно сделать динамическое увеличение числа потоков, если бы решили проблему конкуренции;
3. Более оптимальный вариант;

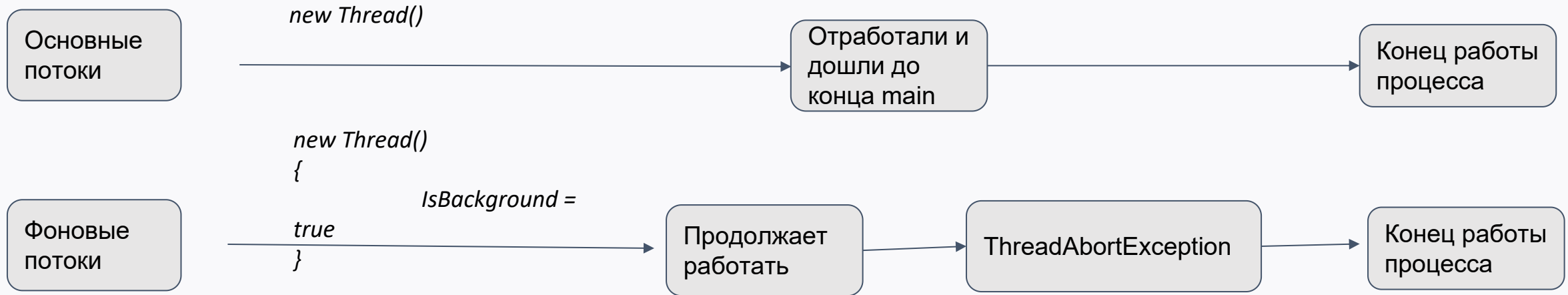
Минусы:

1. В случае с процессами у нас полная изоляция по памяти;
2. Если умирает процесс, то обработка не закончится, но мы можем ее просто перезапустить.

Основной поток и фоновые потоки

Основные и фоновые потоки - это понятия .NET и нужны CLR, нам они помогают разделить задачи на важные и неважные. При создании процесса создается один основной поток

Если все основные потоки завершены, то мы выходим из main, CLR выгружает домен и все фоновые потоки принудительно завершаются через `ThreadAbortException`



Для фоновых потоков нужно вызывать `Join()`, чтобы остановить поток выполнения и подождать завершения их работы

`Thread` - основной поток по умолчанию

`IsBackground` - признак фонового потока

The image features a high-angle, aerial view of a dense urban landscape, likely New York City, characterized by numerous skyscrapers and buildings. The entire image is rendered in a monochromatic blue and green color scheme. A central horizontal band, transitioning from a vibrant green on the left to a deep blue on the right, serves as a background for the word "CODE". This band is overlaid with a subtle, white geometric pattern of interconnected lines and dots, resembling a network or data structure. The word "CODE" is written in a large, bold, white, sans-serif font, centered within this band.

CODE

Как упростить создание потоков?

Хорошо, если не будем каждый раз тратить ресурсы ОС на создание потока, а будем иметь готовый, когда он нужен.



Создадим пул потоков

Пул потоков

1. Реализован классом ThreadPool;
2. Хранит набор изначально созданных для процесса потоков;
3. Все потоки фоновые;
4. Подходит для небольших вычислительных задач и участвует в другом важном механизме;
5. После обработки задачи поток возвращается в пул;
6. Может создавать новые потоки, если нужно;

```
static void Main(string[] args)
{
    ThreadPool.QueueUserWorkItem(HandleInThreadPool);
}

static void HandleInThreadPool(object item)
{
    //Some calculation
}
```


The image features a high-angle, aerial view of a dense urban landscape, likely New York City, characterized by numerous skyscrapers and buildings. The entire image is rendered in a monochromatic blue and green color scheme. A central horizontal band, transitioning from a vibrant green on the left to a deep blue on the right, serves as a background for the word "CODE". This band is overlaid with a subtle, white geometric pattern of interconnected lines and dots, resembling a network or data structure. The word "CODE" is written in a large, bold, white, sans-serif typeface, centered within this band.

CODE

Вопрос

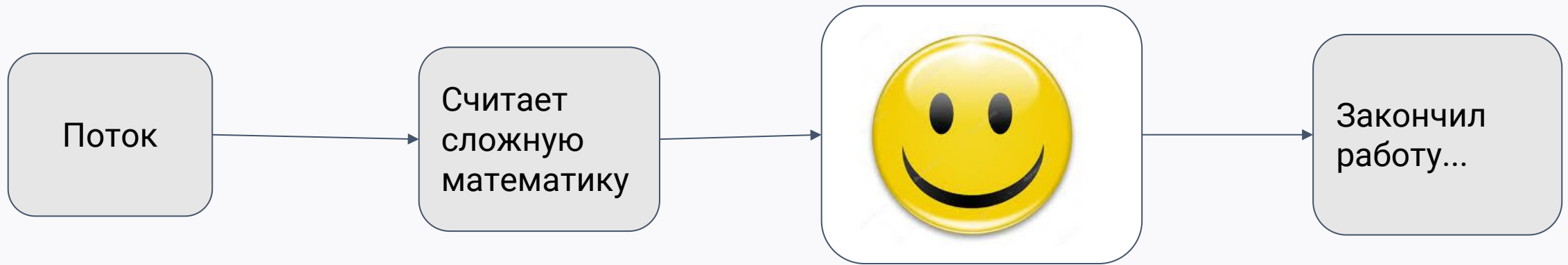


Тайминг: 1-2
минуты

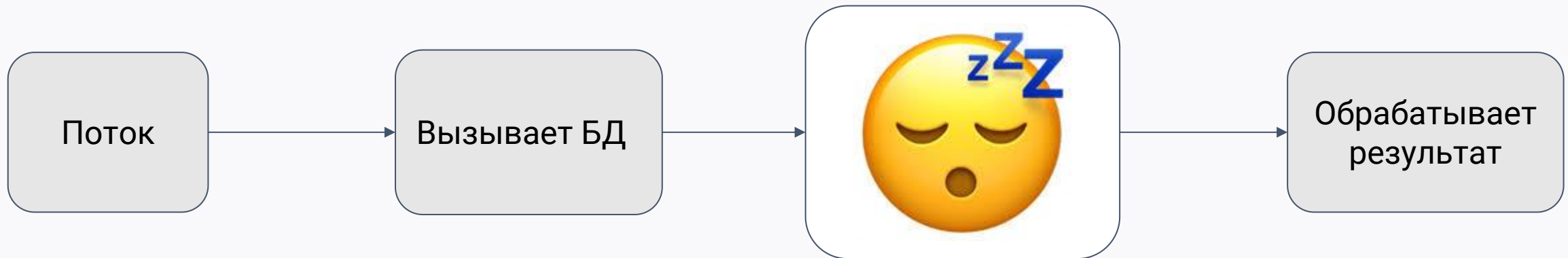
Как вы считаете потоки в наших приложениях больше
работают или больше ждут?
Напишите в чат или минус

Потоки в CPU-операциях и в I/O операциях

CPU



I/O



The background of the entire image is an aerial view of a dense city skyline, likely New York City, with numerous skyscrapers. A semi-transparent blue overlay covers the entire image. In the center, there is a network pattern of white lines connecting various points, resembling a digital or social network. The text "Внезапный вопрос" is written in white, bold, sans-serif font across the middle of the image.

Внезапный вопрос

Вопрос



Тайминг: 1-2
минуты

Многопоточный ли язык JavaScript?
Напишите в чат плюс или минус

Вопрос



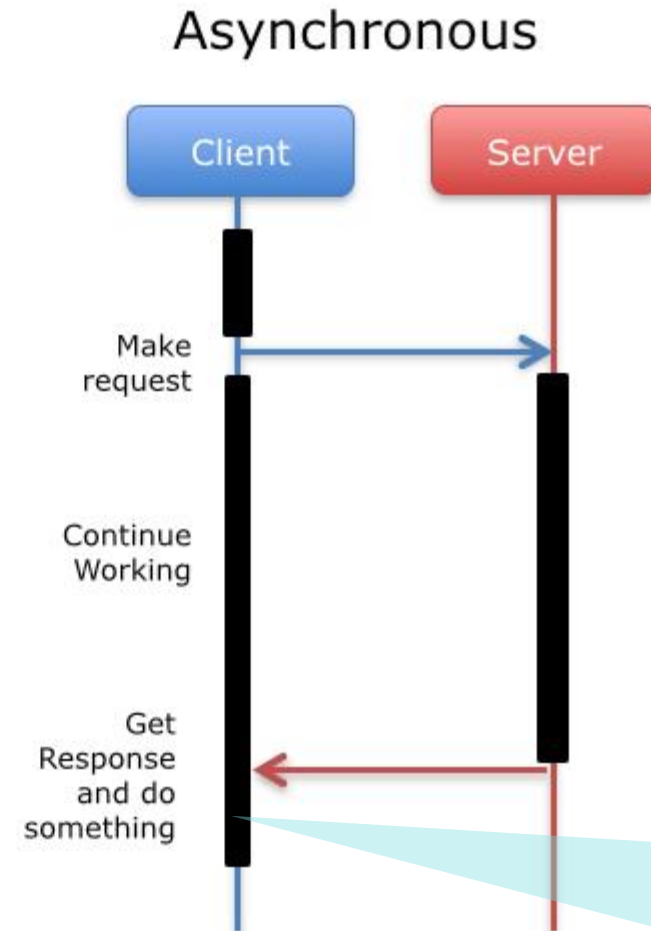
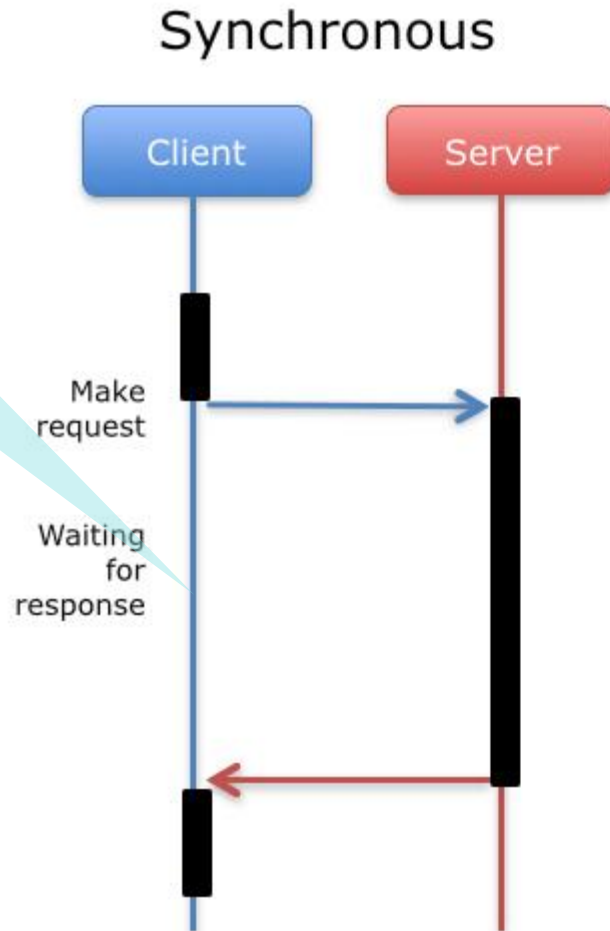
Тайминг: 1-2
минуты

Почему если JavaScript однопоточный у нас не блокируется интерфейс браузера, когда скачиваются файлы и делаются аjax запросы?

Напишите в чат ответ

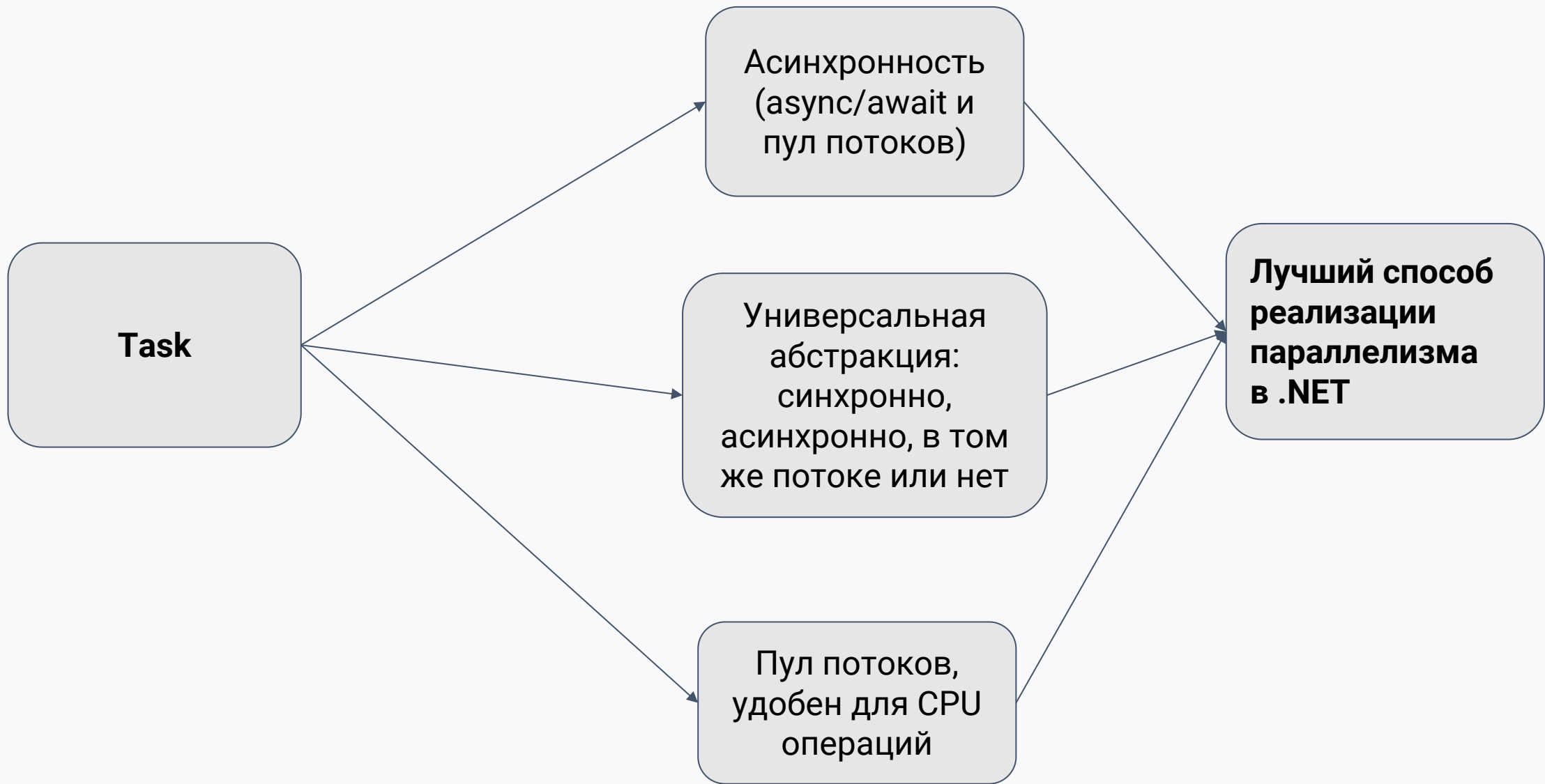
Синхронность и асинхронность

Основной поток ждет окончания запроса, он в блокировке и простаивает



Основной поток не ждет, а продолжает работу, просто мы его потом попросим вернуться и обработать результат, когда результат будет готов

Как объединили асинхронность и многопоточность в C#?



The background of the slide is a high-angle, aerial photograph of a dense urban skyline, likely New York City, featuring numerous skyscrapers and buildings. The image is overlaid with a semi-transparent blue and green gradient. A network of thin, light blue lines connects various points across the gradient, creating a digital or network-like pattern. The word "Task" is centered in the middle of the slide in a large, white, sans-serif font.

Task

Маршрут вебинара

Параллелизм



Процессы и домены



Потоки



Таски

Что такое Task?

Task - универсальная абстракция для выполнения кода в любом количестве потоков, в синхронном или асинхронном режиме.

Особенности

1. По умолчанию Task выполняется в потоке из пула, гораздо легче, чем работать с ним напрямую;
2. Представляет собой объект-задачу, которая имеет статусы выполнения и позволяет сделать абстракцию над потоками, задачами управлять удобнее и проще, чем потоками;
3. В сочетании с `async/await` дает удобный доступ к асинхронным операциям, которые не блокируют вызывающий поток;

Как выглядит использование Task?

Запуск CPU-based задачи в пуле потоков через Task

```
static void Main(string[] args)
{
    var task = Task.Run(() => Console.WriteLine("Some calculation in thread from pool"));

    task.Wait();
    Console.WriteLine("Program cancelled...!");
}
```

Вызов асинхронной операции через Task

```
static async Task Main(string[] args)
{
    await using DataContext dataContext = new DataContext();
    await dataContext.MessageQueue.ToListAsync();
}
```

Задача

Нам необходимо сделать маркетинговую рассылку пользователям в разные каналы, сообщения для рассылки уже сформированы и сохранены в базе данных.

У сообщения есть:

- Id сообщения
- Тип:
 - Email
 - Sms
 - Push
- Текст сообщения
- Id пользователя, которому нужно доставить сообщение
- Дата создания сообщения

Сообщения отсортированы по возрастанию даты создания.

Нужно сделать приложение, которое будет обрабатывать сообщения из БД параллельно обработчиками на каждый тип и передавать их другому компоненту на отправку.

The image features a high-angle, aerial view of a dense urban landscape, likely New York City, characterized by numerous skyscrapers and buildings. The entire image is rendered in a monochromatic blue and green color scheme. A central horizontal band, transitioning from a vibrant green on the left to a deep blue on the right, serves as a background for the word "CODE". This band is overlaid with a subtle, white geometric pattern of interconnected lines and dots, resembling a network or data structure. The word "CODE" is written in a large, bold, white, sans-serif font, centered within this band.

CODE

Вопрос

Какие есть плюсы и минусы в решении задачи на тасках?

Ответ

Плюсы:

1. Удобство управления задачами еще выше, чем потоками;
2. Если использовать асинхронность, то не будет лишних блокировок;
3. Не создаем потоки в ОС;

Минусы:

1. В случае с процессами у нас полная изоляция по памяти;
2. Если умирает процесс, то обработка не закончится, но мы можем ее просто перезапустить.
3. Если не используем асинхронность, то поток пула может быть заблокирован очень долго, это не критично в этой задаче, но нужно учитывать, если много потоков.

Task to be continued...

Больше про Task и асинхронность в следующих занятиях...

Выводы

1 Дали определения основным понятиям параллелизма

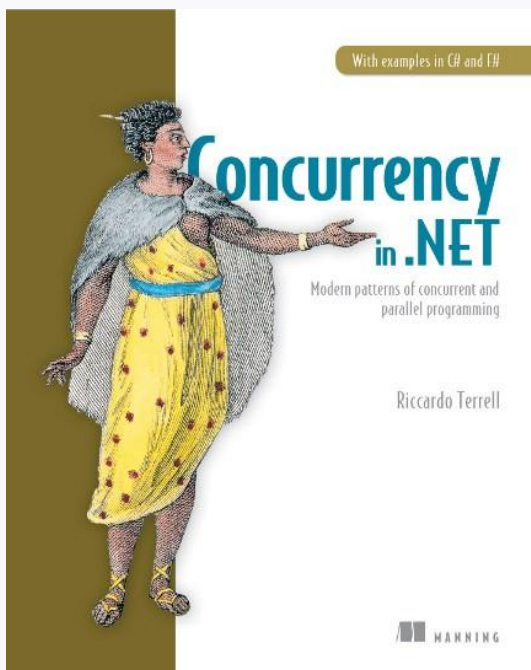
2 Изучили отличия процесса, домена, потока и таска

3 Посмотрели, как их использовать на C#

4 Посмотрели и обдумали ход решения задачи для параллельной обработки данных



Список материалов для изучения




<https://www.manning.com/books/concurrency-in-dot-net>



<https://www.ozon.ru/context/detail/id/21236101/>

MSDN для .NET Core

<https://docs.microsoft.com/ru-ru/dotnet/core/index>



Заполните, пожалуйста,
опрос о занятии по ссылке
<https://otus.ru/polls/111418/>
Лучше всего написать что-то текстом!)