

# Карта знаний .NET Web программиста

Детализированная карта знаний для .NET Web программистов от Trainee до Senior. Используйте для самообучения, поиска пробелов в знаниях, создания программ обучения, подготовки к собеседованиям и продвижениям по карьерной лестнице.

📅 Обновлено 25 декабря 2019    Поделиться: [f](#) [🐦](#) [VK](#) [in](#)

**YOUIT**

➔ Войти

	Trainee	Junior	Middle	Senior
Язык программирования C#	★	★	★★	★★★★
.NET Framework / .NET Core	★	★	★★	★★★★
Контроль качества кода / Рефакторинг	★	★	★★	★★★★
Многопоточность и Асинхронное программирование		★	★	★★
Модульное и интеграционное тестирование	★	★	★★	★★★★
Алгоритмы и Структуры Данных	★	★	★	★★
Принципы объектно-ориентированного программирования (ООП)	★	★★	★★	★★
Шаблоны и принципы проектирования, SOLID		★	★★	★★
Предметно-ориентированное проектирование (Domain-driven design, DDD)		★	★★	★★★★
Шаблоны уровня доступа к данным		★	★★	★★★★

	Trainee	Junior	Middle	Senior
Фреймворки доступа к данным: Entity Framework, NHibernate, Dapper	★	★	★★	★★★★
Архитектурные шаблоны и Атрибуты качества (Quality Attributes)		★	★★	★★★★
HTTP, REST, Веб-фреймворки: ASP.NET MVC / Web API / Core	★	★★	★★	★★★★
Фронтенд разработка: JavaScript / TypeScript / SPA фреймворки	★	★★	★★★★	★★★★
Веб-безопасность, Шифрование данных, OWASP	★	★	★	★★
Распределенные системы и Микросервисная архитектура			★	★★
Базы данных: T-SQL, MS SQL, NoSQL	★	★	★★	★★★★
Системы контроля версий: Git, Mercurial, TFS	★	★	★★	★★★★
Методологии разработки: Scrum, Kanban и Scrumban		★	★★	★★

## Язык программирования C#

Уровень	Темы	Материалы
★	Примитивные конструкции языка C#: Типы данных, Переменные, Методы, Операторы, Символы, Строки, Массивы, Циклы, Условия, Комментарии, Пространства имен, Перегрузка операторов	Книга C# 7.0 in a Nutshell, Глава 2 ● Introduction to C# ● Типы и переменные ● Методы ● Строки ● Операторы C# ● Массивы ● for ● namespace ● if-else

Уровень	Темы	Материалы
★	<p>Ссылочные и значимые типы данных   Stack vs Heap, Классы и структуры   Класс Object   Модификаторы доступа: public/private/protected/internal/protected internal   Интерфейсы и Абстрактные классы   Перечисления   Операторы const и readonly   Упаковка и Распаковка</p>	<p>Книга C# 7.0 in a Nutshell, Глава 2-3 ●  Ссылочные и значимые типы данных в CLR via C# ● Stack vs Heap ● Object Класс ●  Интерфейсы или абстрактные классы ●  Перечисления ●  Модификаторы доступа ●  Обобщения</p>
★	<p>Обобщения для классов, методов, интерфейсов, структур и делегатов   Ограничения</p>	<p>Книга C# 7.0 in a Nutshell, Глава 2-3 ●  Обобщения</p>
★	<p>Статические классы/свойства/конструкторы   Правила вызовов статических конструкторов</p>	<p>Книга C# 7.0 in a Nutshell, Глава 3 ●  Статические конструкторы ●  Статические члены и модификатор static</p>
★	<p>Наследование в C#   Правила вызова конструкторов в иерархиях наследования   Операторы is и as   constructor chaining   Использование ключевых слов virtual, override, new</p>	<p>Наследование в C# и .NET ● C# 7.0 in a Nutshell ● Операторы приведения и тестирования типов ●  Constructor chaining in C#</p>
★	<p>Делегаты и События, Func vs Action   Лямбда Выражения/Анонимные методы   Обработка исключений try/catch/finally/throw, Фильтры исключений   Nullable типы   Методы расширения   Tuples</p>	<p>Книга C# 7.0 in a Nutshell, Глава 4 ●  Делегаты и Лямбда выражения в C# .Net — Шпаргалка или коротко о главном ● Лучшие методики обработки исключений ● Фильтры исключений в C# 6.0 ●  Методы расширения ●  Func vs. Action vs. Predicate</p>

Уровень	Темы	Материалы
★	Классы string и StringBuilder, Неизменяемость строк, Интернирование строк, ASCII и Unicode	Книга C# 7.0 in a Nutshell, Глава 6 ● String vs StringBuilder ● Особенности строк в .NET
★	Структуры DateTime/TimeSpan/DateTimeOffset   Класс Random   Структура Guid   Интерфейсы IComparable, IEquatable, IComparer	Книга C# 7.0 in a Nutshell, Глава 6 ● String vs StringBuilder ● Выбор между типами DateTime, DateTimeOffset, TimeSpan и TimeZoneInfo ● Делегаты и события в .NET ● Лямбда-выражения
★	Коллекции ArrayList, List, Dictionary, Hashtable, HashSet, LinkedList, Stack, Queue, Класс Array   Интерфейсы IEnumerable и IEnumerator, Утиная типизация (duck typing), Цикл foreach   Интерфейсы ICollection, IList, IReadOnlyList	Книга C# 7.0 in a Nutshell, Глава 7 ● Коллекции и структуры данных ● Итераторы
★	Механизм работы Сборщика Мусора, Финализаторы, Интерфейс IDisposable, using	Книга C# 7.0 in a Nutshell, Глава 12
★	Работа с потоками ввода/вывода, Пространство имен System.IO, Классы File, Directory, FileInfo, DirectoryInfo, Path, FileStream, MemoryStream, StreamReader, StreamWriter, BinaryReader, BinaryWriter...	Книга C# 7.0 in a Nutshell, Глава 15
★	Механизм сериализации, Бинарная сериализация, XML сериализация, Атрибуты [DataContract], [DataMember], [OnSerializing], [Serializable], [OnSerialized], [OnDeserializing], [OnDeserialized]	Книга C# 7.0 in a Nutshell, Глава 17
★★	Pattern Matching	Pattern matching в C# 7
★★	Шаблон Dispose, Слабые Материалы (Weak References), Утечки памяти в управляемом коде и способы их предотвращения	Книга C# 7.0 in a Nutshell, Главы 8-9 ● Learning .NET memory management
★★	Работа с классом Regex, Написание регулярных выражений	C# 7.0 in a Nutshell ● C# Regex в примерах

Уровень	Темы	Материалы
★★	Диагностика: Stopwatch, Классы Debug и Trace, Классы StackTrace и StackFrame, Использование CPU и Memory профайлеров	Книга C# 7.0 in a Nutshell, Глава 13
★★★★	Работа со структурами Memory<T> и Span<T>	Рекомендации по использованию структур Memory<T> и Span<T>
★★★★	Работа со структурой Деревьями выражений (Expression Trees)	Книга C# 7.0 in a Nutshell, Глава 8
★★★★	Рефлексия, Метаданные, Атрибуты, Позднее связывание, Класс Activator, Класс Type Ключевое слово dynamic, Классы ExpandableObject и DynamicObject	Книга C# 7.0 in a Nutshell, Главы 19-20

### Контрольные вопросы:

- В чем разница между ссылочными и значимыми типами данных?
- В чем разница между делегатами и событиями?
- В чем разница между операторами const и readonly?
- Может ли структура реализовывать интерфейс
- Что такое Duck typing? В чем отличия интерфейсов IEnumerable и IEnumerator?
- В чем заключается необходимость неизменяемости строк? (или какие преимущества неизменяемых типов над изменяемыми?)
- Почему StringBuilder значительно опережает string по производительности при большом количестве конкатенаций?
- Когда вызывается статический конструктор экземплярного класса?
- В чем разница между коллекциями: Dictionary vs Hashtable, Hashtable vs HashSet, Dictionary vs Lookup?
- Для чего механизм сборки мусора использует поколения?
- Существует ли связь между механизмом рефлексии и метаданными сборки?
- Как реализовать механизм Замыкания (Closure) в C#?

## .NET Framework / .NET Core

Уровень	Темы	Материалы
★	Основы .NET Framework: CLR, CTS, JIT	C# 7.0 in a Nutshell ● Общие сведения о платформе .NET Framework ● Книга CLR via C# by Jeffrey Richter, Глава 1

Уровень	Темы	Материалы
★	Модель выполнения кода в среде CLR	Книга CLR via C# by Jeffrey Richter, Глава 1
★	.NET Framework vs .NET Core	.NET Core vs .NET Framework: How to Pick a .NET Runtime for an Application
★	Понимание .NET Standard	What is .NET Standard?
★★	Архитектура .NET Framework и .NET Core	Книга CLR via C# by Jeffrey Richter, Глава 1 ● Книга C# 7.0 in a Nutshell, Глава 1 ● .Net Core Architecture
★★	Сборки: Содержимое сборок, Манифест, Управление версиями, GAC, Загрузка сборок...	Сборки в .NET ● Книга CLR via C# by Jeffrey Richter, Главы 2, 3, 23
★★★	Домены приложения, Процесс vs Домен vs Поток, Класс AppDomain	Книга CLR via C# by Jeffrey Richter, Глава 22 ● Книга C# 7.0 in a Nutshell, Глава 24
★★★	Куча больших объектов (Large Object Heap), Конфигурация сборщика мусора, GC триггеры	Книга C# 7.0 in a Nutshell, Глава 12 ● Книга CLR via C# by Jeffrey Richter, Глава 21
★★★	Работа со счетчиками производительности (Performance Counters), Performance Monitor	Книга C# 7.0 in a Nutshell, Глава 13 ● Unsafe Code in C# ● Книга Pro .NET Performance: Optimize Your C# Applications by Sasha Goldshtein, Главы 1-2
★★★	Работа с небезопасным кодом (Unsafe code)   Маршалинг	Книга C# 7.0 in a Nutshell, Глава 25 ● Unsafe Code in C#
★★★	Roslyn компилятор	Книга C# 7.0 in a Nutshell, Глава 27

**Контрольные вопросы:**

- Что такое JIT-компиляция?
- В чем разница между Процессом и Доменом приложения?
- Зачем может понадобиться явно создавать дополнительный Домен приложения?
- Почему лучше избегать попадания в Large Object Heap и как?
- Какие режимы сборки мусора существуют?
- Какие конкретные счетчики производительности существуют в Windows?
- Какое предназначение у механизма Маршалинг?
- Что такое Roslyn?

## Контроль качества кода / Рефакторинг

Уровень	Темы	Материалы
★	Понимание целей Рефакторинга	Рефакторинг: основные принципы и правила ● Цена рефакторинга
★	Понимание распространенных Code Smells: Длинный метод, Длинный список параметров, Большой класс, Временное поле, Дублирование кода, Мёртвый код, Цепочка вызовов	Martin Fowler, CodeSmell ● Запахи кода ● CodeSmell
★	Владение стандартами кодирования и правилами именования	Соглашения о написании кода на C# ● Рекомендации по написанию кода на C# от Aviva Solutions ● Правила именования ● Книга Code Complete by Steve McConnell
★	Владение распространенными техниками рефакторинга: Извлечение метода, Встраивание метода, Замена алгоритма, Расщепление переменной, Переименование метода, Разбиение условного оператора, Замена магического числа символьной константой и других.	Книга Refactoring Improving the Design of Existing Code by Martin Fowler ● Приёмы рефакторинга

Уровень	Темы	Материалы
★★	Владение продвинутыми техниками рефакторинга: Замена делегирования наследованием, Создание шаблонного метода, Свёртывание иерархии, Замена конструктора фабричным методом, Введение Null-объекта, Замена наследования делегированием и других.	Книга Refactoring Improving the Design of Existing Code by Martin Fowler ● Приёмы рефакторинга
★★	Работа со статическими анализаторами кода: StyleCop, Resharper, SonarCube	Статический анализ кода ● StyleCop: A Detailed Guide to Starting and Using It
★★	Код-ревью: понимание процесса, как проводить, лучшие практики, инструменты	Code review: вы делаете это неправильно ● Лучшие и проверенные практики ревью: советы для авторов кода
★★	Документирование кода с помощью комментариев	Документирование кода с помощью XML-комментариев ● Книга Code Complete by Steve McConnell
★★★★	Сбор и анализ метрик качества кода: Цикломатическая сложность, Количество строк кода, Глубина наследования, Количество входящих/исходящих зависимостей класса	Значения метрик кода ● Метрики программного обеспечения в Visual Studio
★★★★	Работа с унаследованным кодом: рефакторинг, сопровождение, повторное использование	Книга Working Effectively with Legacy Code by Michael Feathers
★★★★	Работа с техническим долгом	Технический долг на проекте или выбраться из черной дыры ● Как научиться не накапливать технический долг — отвечают эксперты



Уровень	Темы	Материалы
★★★	Выполнение large-scale рефакторинга	<a href="#">How To Do Large Scale Refactoring</a>

### Контрольные вопросы:

- Какая цель Рефакторинга?
- Почему модульные тесты важны при рефакторинге?
- Что для Вас есть Code Smells?
- Что могут показать статические анализаторы кода?
- Как рефакторить длинный список параметров метода?
- Нужно ли оставлять в коде комментарии?
- На что обращать внимание при проведении код-ревью?
- Какие инструменты код-ревью существуют?
- Нужно ли собирать метрики кода при выполнении large-scale рефакторинга?

## Многопоточность и Асинхронное программирование

Уровень	Темы	Материалы
★	Понимание терминов Процесс, Поток, Пулл потоков, Мертвая блокировка	<a href="#">В чем разница между потоком и процессом?</a> ● <a href="#">Взаимная блокировка</a> ● <a href="#">Книга CLR via C# by Jeffrey Richter, Глава 26</a> ● <a href="#">Основы многопоточности в .NET Framework</a>
★	Понимание концепций Многопоточного программирования, Асинхронного программирования и Параллелизма	<a href="#">Книга CLR via C# by Jeffrey Richter, Глава 26</a> ● <a href="#">Многопоточное vs асинхронное программирование</a> ● <a href="#">Книга C# 7.0 in a Nutshell, Глава 14</a>
★	Класс Thread: Start, Join, Abort, Sleep   Фоновые и активные потоки	<a href="#">Книга C# 7.0 in a Nutshell, Глава 14</a> ●

Уровень	Темы	Материалы
★	Ключевые слова async и await, Synchronization Context	Книга C# 7.0 in a Nutshell, Глава 14 ● Асинхронное программирование с использованием ключевых слов async и await ● C# Async / Await - Make your app more responsive and faster with asynchronous programming
★	Класс Task, Использование методов Run, Delay, WhenAny, WhenAll, ContinueWith, Обработка исключений, AggregateException	Книга C# 7.0 in a Nutshell, Глава 17 ● Книга Concurrency in C# Cookbook 1st Edition by Stephen Cleary
★	Необходимость использования синхронизации потоков   Простая синхронизация потоков: ключевое слово lock, Monitor.Enter, Monitor.Exit, Interlocked	Книга C# 7.0 in a Nutshell, Глава 17 ● Книга CLR via C# by Jeffrey Richter, Глава 29 ● Инструкция lock
★★	PLINQ: Parallel.For, Parallel.ForEach, AsParallel, AsOrdered, WithCancellation	Книга C# 7.0 in a Nutshell, Глава 24 ● Введение в PLINQ
★★	Синхронизация потоков: Mutex, Semaphore, AutoResetEvent, ManualResetEvent, Reader/Writer Locks, volatile	Книга C# 7.0 in a Nutshell, Глава 22 ● Книга CLR via C# by Jeffrey Richter, Главы 29,30
★★	Использование потокобезопасных коллекций: ConcurrentBag<T>, IProducerConsumerCollection<T>, ConcurrentDictionary<T>	Книга C# 7.0 in a Nutshell, Глава 23 ● Книга Concurrency in C# Cookbook 1st Edition by Stephen Cleary
★★	Тестирование асинхронных методов	Книга Concurrency in C# Cookbook 1st Edition by Stephen Cleary
★★	Task Scheduling	Книга Concurrency in C# Cookbook 1st Edition, Глава 12

Уровень	Темы	Материалы
☆☆☆	Неизменяемые коллекции: <code>ImmutableList&lt;T&gt;</code> , <code>ImmutableDictionary&lt;T&gt;</code> , <code>ImmutableHashSet&lt;T&gt;</code>	Книга <a href="#">Concurrency in C# Cookbook 1st Edition</a> , Глава 8

### Контрольные вопросы:

- Что такое Виртуальное адресное пространство?
- Почему большое количество параллельных потоков могут ухудшить производительность приложения?
- В чем разница между Поток и Процессом?
- Когда возникает Мертвая блокировка?
- В чем разница между `Thread.Sleep` и `Task.Delay`?
- В чем разница между `Lock` и `Monitor`?
- В чем разница между `Thread` и `Task`?
- Почему неизменяемые типы являются потокобезопасными?
- В чем разница между Многопоточным и Асинхронным программированием?
- Когда нужно думать о синхронизации потоков?
- В чем разница между механизмами синхронизации `Mutex` и `Semaphore`?
- Почему примитивные операции инкремента или присваивания нужно синхронизировать классом `Interlocked`?
- Что такое `ReaderWriterLockSlim`?
- Как создать потокобезопасный Синглтон?
- Является ли класс `Lazy<T>` потокобезопасным?

## Модульное и интеграционное тестирование

Уровень	Темы	Материалы
☆	Цели юнит-тестирования	Юнит-тестирование для чайников ● Зачем нужны юнит-тесты ● Книга <a href="#">The Art of Unit Testing by Roy Oshero</a> , Глава 1-2
☆	Методика тестирования черного, белого, серого ящиков	Общая картина модульного тестирования ● <a href="#">White/Black/Grey Box-тестирование</a>
☆	Понятия Test Plan, Test Suite, Test case	<a href="#">Test Case, Test Suite, Test Run, what's the difference?</a> ● <a href="#">Test cases and test suites</a>

Уровень	Темы	Материалы
★	Шаблон Arrange-Act-Assert (AAA)	Unit Testing and the Arrange, Act and Assert (AAA) Pattern
★	Фреймворки модульного тестирования: NUnit или xUnit	NUnit Tutorial ● xUnit Tutorial
★	Изоляционные фреймворки: Moq, NSubstitute, FakeItEasy	Moq Tutorial ● NSubstitute Tutorial ● FakeItEasy Tutorial
★	Тестовые заглушки: Dummy, Stub, Mock..	Книга The Art of Unit Testing by Roy Oshero, Глава 4
★	Параметризованные тесты	Книга The Art of Unit Testing by Roy Oshero, Глава 2
★	Метрики покрытия кода тестами, NCover	Проблемы тестирования: почему 100% покрытие кода это плохо ● An introduction to code coverage
★★	Причины недетерминированного поведения юнит-тестов	Non-determinism in tests ● Eradicating Non-Determinism in Tests
★★	Принципы модульного тестирования F.I.R.S.T.	Пять принципов чистых тестов (F.I.R.S.T. Principles)
★★	Тестовая пирамида: модульные, интеграционные, E2E тесты	Пирамида тестов на практике
★★	Test smells: Conditional test logic, Hard-to-test code, Test code duplication, Assertion roulette, Fragile test, Test logic in production, Frequent Debugging, Slow Tests...	● Книга xUnit Test Patterns: Refactoring Test Code by Gerard Meszaros, Главы 2, 15-17
★★	Стили юнит-тестирования: верификация результата (value-based), верификация состояния (state-based), верификация взаимодействия (interaction testing)	Книга The Art of Unit Testing by Roy Oshero, Глава 4 ● Styles of unit testing

Уровень	Темы	Материалы
★★	Data-driven Tests	Книга xUnit Test Patterns: Refactoring Test Code by Gerard Meszaros, Глава 18
★★	AutoFixture	AutoFixture
★★	Интеграционные тесты	Integration testing in .NET ● Integration tests in ASP.NET Core ● Entity Framework Core - Isolation of Integration Tests ● Testing with InMemory
★★★	Шаблоны модульного тестирования	Книга xUnit Test Patterns: Refactoring Test Code by Gerard Meszaros
★★★	Модульные тесты для унаследованного кода	Книга The Art of Unit Testing by Roy Oshero, Глава 10 ● Книга Working Effectively with Legacy Code by Michael Feathers
★★★	Разработка через тестирование (Test-driven development, TDD)	Книга Test Driven Development by Kent Beck

### Контрольные вопросы:

- Что именно тестируют модульные тесты?
- В чем разница между Модульными и Интеграционными тестами?
- Что такое AAA шаблон?
- Чем Stub отличается от Mock?
- С какими Code Smells в модульных тестах Вы сталкивались?
- Как измерить процент покрытия кода модульными тестами?
- Почему тяжело тестировать код, использующий шаблон Синглтон?
- Какие шаблоны модульного тестирования Вам знакомы?
- Нужно ли тестировать приватные методы класса?
- Опишите процесс разработки через тестирование?
- Что такое Data-driven тесты?

# Алгоритмы и Структуры Данных

Уровень	Темы	Материалы
★	Понимание Big O нотации, Big O(1), Big O(n), Big O(log n)...	Big O Нотация
★	Простые алгоритмы поиска: линейный поиск и бинарный поиск	Обзор алгоритмов линейного и бинарного поиска ● Алгоритм бинарного/двоичного поиска
★	Понимание основных алгоритмов сортировки: пузырьковая сортировка (Bubble Sort), сортировка вставками (Insertion Sort), быстрая сортировка (Quick Sort), сортировка слиянием (Merge Sort)	Описание алгоритмов сортировки и сравнение их производительности ● Книга Алгоритмы. Руководство по разработке, Стивен С. Скиена
★	Работа с основными структурами данных и понимание их внутреннего устройства: массив, стек, очереди, списки односвязные/двусвязные, хеш-таблица   Принципы LIFO и FIFO	Важнейшие структуры данных, которые вам следует знать к своему собеседованию по программированию ● Основные структуры данных. Матчасть. Азы ● Книга Алгоритмы. Руководство по разработке, Стивен С. Скиена ● FIFO и LIFO
★	Значения Big O для операций со структурами данных и алгоритмов сортировок	Know Thy Complexities!
★	Внутреннее устройство хеш-таблиц, Хеш-функции, Коллизии	Хеш-таблица ● Коллизия хеш-функции ● Что такое хеш-таблицы и как они работают

Уровень	Темы	Материалы
★	Деревья и Бинарные деревья: Структура, Построение, Обход (Pre-Order, In-Order, Post-Order), Поиск (DFS, BFS), Вставка элементов	Все что нужно знать о древовидных структурах данных ● Визуализация алгоритмов ● Книга Алгоритмы. Руководство по разработке, Стивен С. Скиена
★★	Оценка сложности алгоритмов	Оценка сложности алгоритмов
★★	Виды графов: Однонаправленные и Двухнаправленные, Алгоритмы поиска на графах: DFS, BFS, Дейкстра	Алгоритмы на графах — Часть 0: Базовые понятия ● Алгоритмы на графах — Часть 1: Поиск в глубину и проблема взаимоблокировок ● Алгоритм Дейкстры. Поиск оптимальных маршрутов на графе

### Контрольные вопросы:

- Что показывает Big O нотация?
- По каким двум параметрам можно оценивать сложность алгоритма?
- Чему равна Big O для бинарного поиска?
- Чему равна Big O для операции обращения к ячейке массива по индексу?
- Опишите алгоритм Сортировки слиянием (Merge Sort)?
- Чему равна Big O для операции извлечения значения по ключу из хеш-таблицы?
- Опишите реализацию алгоритма поиска в ширину (BFS) для дерева?
- В каких случаях для деревьев лучше использовать DFS, а в каких BFS?
- Почему может возникнуть коллизия в хеш-таблицах?
- Как оценить сложность алгоритма?
- Опишите принцип работы алгоритма Дейкстры.

## Принципы объектно-ориентированного программирования (ООП)



Уровень	Темы	Материалы
★	Классы и Объекты	Классы и объекты
★	Понимание принципов ООП: Абстракция, Инкапсуляция, Наследование, Полиморфизм	Книга Объектно-ориентированный анализ и проектирование, Гради Буч ● ООП с примерами
★★	Понимание принципов Сильного сцеплений (High Cohesion) и Слабой связанности (Low Coupling)	Книга Объектно-ориентированный анализ и проектирование, Гради Буч ● Cohesion and Coupling: the difference
★★	Понимание связей Композиции, Агрегации, Наследования. Плюсы и минусы.	Книга Объектно-ориентированный анализ и проектирование, Гради Буч ● Наследование vs Композиция vs Агрегация

### Контрольные вопросы:

- В чем разница между Классом и Объектом?
- В чем разница между Абстракцией и Инкапсуляцией?
- В чем недостатки наследования?
- Что такое Полиморфизм?
- В чем разница между Композицией и Агрегацией?
- О чем говорит принцип Сильного сцеплений (High Cohesion)?
- О чем говорит принцип Слабой связанности (Low Coupling)?

## Шаблоны и принципы проектирования, SOLID

Уровень	Темы	Материалы
★	Принципы SOLID	Книга Паттерны проектирования на платформе .NET, Главы 17-22 ● Принципы SOLID, о которых должен знать каждый разработчик



Уровень	Темы	Материалы
★	Принципы DRY, KISS, YAGNI	10 принципов объектно-ориентированного программирования, о которых должен знать каждый разработчик ● Принципы DRY, KISS, YAGNI
★	Принцип Inversion of Control (IoC)	Inversion Of Control ● IoC, DI, IoC-контейнер — Просто о простом
★	Ключевые GoF шаблоны: Синглтон, Фабрики, Фасад, Стратегия, Декоратор, Адаптер, Наблюдатель, Состояние	Книга Паттерны проектирования на платформе .NET, Главы 1-16 ● Шаблоны проектирования с человеческим лицом
★★	Другие GoF шаблоны: Строитель, Прототип, Приспособленец, Компоновщик, Прокси, Цепочка обязанностей, Медиатор, Интерпретатор, Посетитель, Шаблонный метод	Книга Паттерны проектирования на платформе .NET, Главы 1-16 ● Шаблоны проектирования с человеческим лицом
★★	Не GoF шаблоны: Null-Object, Rules, Event Aggregator, Lazy Load pattern, Шаблоны внедрения зависимостей: внедрений через конструктор/свойство/метод,	Паттерны внедрения зависимостей. Часть 1 ● Паттерны внедрения зависимостей. Часть 2 ● Введение Null-объекта ● Event Aggregator ● Шаблоны проектирования: Rules
★★	Закон Деметры	Закон Деметры

### Контрольные вопросы:

- Как расшифровывается аббревиатура SOLID?
- В чем идея принципа замещения Лисков?
- Какие шаблоны проектирования помогают придерживаться принципа Открыт/Закрит?
- Какие шаблоны проектирования помогают придерживаться принципа Единой ответственности?

- В чем сходства и отличия принципов Единой ответственности и Разделения интерфейсов?
- Опишите принципы DRY, KISS, YAGNI.
- В чем разница между шаблонами Фасад и Адаптер?
- Как реализовать Синглтон? Можно ли реализовать Синглтон с помощью DI-контейнеров?
- В чем смысл шаблона Декоратор?
- В чем разница между шаблонами Стратегия и Шаблонный метод?
- В чем суть принципа Inversion of Control (IoC)?
- Какую проблему решает шаблон проектирования Посетитель?
- Опишите реализацию шаблона Медиатор.
- Когда нужно использовать шаблон проектирования Rules?

## Предметно-ориентированное проектирование (Domain-driven design, DDD)



Уровень	Темы	Материалы
★	Необходимость применения DDD	Книга Предметно-ориентированное проектирование, Эрик Эванс, Глава 1 ● Domain-driven design: рецепт для прагматика
★★	Понимание терминов Поддомен (Subdomain) и Ограниченный контекст (Bounded Context), Карта контекстов (Context Map)	Книга Предметно-ориентированное проектирование, Эрик Эванс, Глава 14 ● Ubiquitous Language и Bounded Context в DDD ● Domain-driven design: рецепт для прагматика
★★	Разработка Единого языка (Ubiquitous Language)	Книга Предметно-ориентированное проектирование, Эрик Эванс, Глава 2 ● Ubiquitous Language и Bounded Context в DDD
★★	Шаблоны DDD: Сущности, Объекты-значения, Доменные сервисы, Агрегаты, Корни Агрегатов, Репозитории, Доменные события	Книга Предметно-ориентированное проектирование, Эрик Эванс, Главы 5-7

Уровень	Темы	Материалы
★★	Anemic Model vs Rich Model	<a href="#">Anemic Domain Model vs Rich Domain Model with Examples</a>
★★★★	Построение Anti-Corruption Layer	<a href="#">Книга Предметно-ориентированное проектирование, Эрик Эванс, Глава 14</a> ● <a href="#">Wrapping your business logic with anti-corruption layers – NET Core</a>
★★★★	Имплементация DDD в .NET	<a href="#">ddd-guestbook</a> ● <a href="#">Full ASP.NET Core 2.2 application with DDD, CQRS and Event Sourcing</a>

### Контрольные вопросы:

- Когда нужно применять DDD?
- В чем разница между Поддоменом и Ограниченным контекстом?
- В чем смысл Ubiquitous Language и как его разработать?
- В чем разница между Сущностями и Объектами-значениями?
- Что означает Анемичная доменная модель?
- Как реализовать Доменный события?
- В каких случаях нужно размещать бизнес-логику в Доменных сервисах?
- Зачем может понадобиться создавать Anti-Corruption слой?

## Шаблоны уровня доступа к данным

Уровень	Темы	Материалы
★	Объектно-реляционное отображение (ORM)	<a href="#">ORM (Object-Relational Mapping)</a> ● <a href="#">Введение в ORM</a>

Уровень	Темы	Материалы
★	Шаблон Репозиторий (Repository)	<p>Паттерн «Репозиторий». Основы и разъяснения ●</p> <p>EntityFramework: (анти)паттерн Repository ●</p> <p>Domain-Driven Design: Repository ●</p> <p>Книга Patterns of Enterprise Application Architecture by Martin Fowler, Глава 13</p>
★	Шаблон Спецификация (Specification)	
★★	Шаблон Единица работы (Unit Of Work)	<p>Описание Unit of Work ●</p> <p>CRUD Operations Using the Generic Repository Pattern and Unit of Work in MVC ●</p> <p>Книга Patterns of Enterprise Application Architecture, Глава 11</p>
★★	Шаблон Объект запроса (Query Object)	<p>Query Object (Объект-запрос) ●</p> <p>Работа с Dapper + Query Object ●</p> <p>Dapper + QueryObject, как замена ORM ●</p> <p>Книга Patterns of Enterprise Application Architecture by Martin Fowler, Глава 13</p>
★★★	Шаблон Активная запись (Active Record)	<p>Active Record (Активная запись) ●</p> <p>Active Record Pattern ●</p> <p>Книга Patterns of Enterprise Application Architecture by Martin Fowler, Глава 10</p>
★★★★	Шаблон Репозиторий в DDD	<p>Domain-Driven Design: Repository ●</p> <p>Проектирование уровня сохраняемости инфраструктуры</p>

Уровень	Темы	Материалы
★★★	Оптимистическая блокировка, Пессимистическая блокировка	Книга <a href="#">Patterns of Enterprise Application Architecture</a> , Главы 5, 16

### Контрольные вопросы:

- Какую проблему решает технология ORM?
- Как сочетаются вместе шаблоны Unit of Work и Repository?
- Можно ли совмещать шаблоны Repository и Specification?
- В чем смысл шаблона Specification?
- Нужно ли создавать репозиторий для моделей, которые не являются корнями агрегатов?
- Является ли шаблон Active Record анти-шаблоном?
- Как работает механизм Оптимистической блокировки?

## Фреймворки доступа к данным: Entity Framework, Dapper



Уровень	Темы	Материалы
★	ADO.NET, Подключенный и автономный режимы, Классы SqlCommand, SqlDataReader, SqlDataAdapter, DataSet, DataTable	<a href="#">Руководство по ADO.NET и работе с базами данных</a>
★	Entity Framework 6 vs Entity Framework Core	<a href="#">Сравнение EF Core и EF6</a>
★	Подходы Code First и Database First	<a href="#">Code-First vs Model-First vs Database-First: Pros and Cons</a>
★	Создание моделей, Работа с классами DbContext, DbSet	<a href="#">The Entity Framework Core DbContext</a> ● <a href="#">The Entity Framework Core DbSet</a> ● <a href="#">Creating and configuring a model</a>
★	Чтение данных при помощи LINQ, Интерфейс IQueryable vs IEnumerable, Tracking vs No-Tracking	<a href="#">Querying Data</a> ● <a href="#">IEnumerable</a> и <a href="#">IQueryable</a> ● <a href="#">Отслеживание объектов и AsNoTracking</a>

Уровень	Темы	Материалы
★	Ленивая загрузка зависимых данных, Проблема N+1	<a href="#">Lazy Loading Related Data In Entity Framework Core</a> ● <a href="#">Common Entity Framework Problems: N + 1</a>
★	Сохранение данных	<a href="#">Saving Data</a>
★★	Архитектура Entity Framework	<a href="#">Entity Framework Architecture</a>
★★	Моделирование отношений один ко многим и многие ко многим в Entity Framework Core	<a href="#">Configuring Many To Many Relationships in Entity Framework Core</a>
★★	Работа с состояниями сущностей: Added, Unchanged, Modified, Deleted, Detached	<a href="#">Working with entity states</a>
★★	Работа с Fluent API	<a href="#">Fluent API Configuration</a>
★★	Работа с Dapper	<a href="#">Dapper</a>
★★★	Работа с миграциями	<a href="#">Entity Framework Core Migrations</a>
★★★★	Транзакции в Entity Framework Core	<a href="#">Использование транзакций</a>
★★★★	Наследование в EF Core: Table Per Hierarchy, Table Per Type, Table Per Concrete Type	<a href="#">Inheritance in Entity Framework Core</a>

### Контрольные вопросы:

- В чем разница между Code First и Database First в Entity Framework?
- Что такое N+1 проблема?
- В чем разница между интерфейсами IQueryable и IEnumerable?
- В фундаментальная разница между Entity Framework и Dapper?
- Какими классами в Entity Framework представлены шаблоны Unit Of Work и Repository?
- Как моделируется отношение "многие ко многим" в Entity Framework Core?

## Архитектурные шаблоны и Атрибуты качества (Quality Attributes)



Уровень	Темы	Материалы
---------	------	-----------

Уровень	Темы	Материалы
★	Трёхуровневая архитектура, Монолитная архитектура	Трёхуровневая архитектура ● Монолитная vs Микросервисная архитектура
★	Шаблон Model-View-Controller (MVC)	Design Patterns - MVC Pattern
★	Клиент-серверная архитектура	Клиент — сервер
★★	Микросервисная архитектура	Микросервисы — за и против ● Переход от монолита к микросервисам
★★	Шаблон Command Query Responsibility Segregation (CQRS)	Command and Query Responsibility Segregation (CQRS) на практике ● CQRS ● Types of CQRS ● CQRS. Факты и заблуждения
★★	Событийно-ориентированная архитектура (Event-driven architecture)	What is an Event-Driven Architecture?
★★★	Шаблон Event Sourcing	Введение в CQRS + Event Sourcing: Часть 1. Основы ● Pattern: Event sourcing
★★★★	Луковая архитектура (Onion architecture)	Луковая архитектура. Часть 1 ● Onion Architecture
★★★★	Совмещение CQRS и DDD	Simplified CQRS and DDD
★★★★	Архитектурные анти-шаблоны	Software Architecture AntiPatterns
★★★★	Serverless архитектура	Serverless Architectures
★★★★	Шаблоны корпоративных программных приложений	Catalog of Patterns of Enterprise Application Architecture
★★★★	Неизменяемая архитектура (Immutable architecture)	Immutable architecture

Уровень	Темы	Материалы
★★★★	Шаблоны интеграции корпоративных приложений	Шаблоны интеграции корпоративных приложений
★★★★	Атрибуты качества ПО (Software Quality Attributes)	Quality attributes in Software Architecture ● Software Architecture for Developers

### Контрольные вопросы:

- Какими конкретно уровнями представлена трехуровневая архитектура?
- Какие недостатки имеет монолитная архитектура?
- Какие преимущества микросервисной архитектуры над монолитной?
- Как взаимодействуют компоненты Модель, Контроллер и Представление в шаблоне MVC?
- В чем смысл шаблона CQRS?
- Опишите элементы Onion архитектуры и их взаимодействие?
- Как реализуется Serverless архитектура?
- К какому типу Software Requirements относятся атрибуты качества ПО?
- Какие атрибуты качества ПО вы знаете?
- В чем недостатки шаблоны интеграции Shared Database?
- Как два микросервиса могут взаимодействовать друг с другом/обмениваться данными?

## HTTP, REST, Веб-фреймворки: ASP.NET MVC / Web API / Core



Уровень	Темы	Материалы
★	Клиент-серверная архитектура, REST	Синхронизируем понимание REST
★	HTTP протокол, Структура HTTP запросов и ответов, HTTP методы: GET, POST, PUT, DELETE	Общая структура HTTP-запросов и ответов. ● Общая структура HTTP-запросов и ответов. ● Типы HTTP-запросов и философия REST
★	Жизненный цикл запроса в ASP.NET MVC	Жизненный цикл приложения ASP.NET MVC 5



Уровень	Темы	Материалы
★	ASP.NET MVC vs ASP.NET Web API	<a href="#">Difference Between MVC and Web API</a>
★★	Спецификация OWIN, Katana	<a href="#">OWIN и Katana: первый взгляд</a> ● <a href="#">Начало работы с OWIN и Katana</a>
★★	Веб-сервер Kestrel	<a href="#">Kestrel web server implementation in ASP.NET Core</a>
★★	Концепции ASP.NET Core: Startup, Middleware, Routing, Controller, Action, Action Filters, Model Binding, Dependency Injection, View, Partial View, Layout, Razor, Exception filters, Bundling & Minification...	<a href="#">Get started with ASP.NET Core MVC</a> ● <a href="#">ASP.NET Core Tutorial</a>
★★	Внедрение зависимостей в ASP.NET Core Использование DI-фрейворков: Autofac, Simple Injector, Ninject	<a href="#">ASP.NET Core Dependency Injection Deep Dive</a>
★★	Способы хранения состояния приложения: сессии веб-сервера, базы данных, куки, local/session хранилище, hidden поля	<a href="#">ASP.NET Core Blazor state management</a> ● <a href="#">Session and app state in ASP.NET Core</a>
★★	Библиотека SignalR	<a href="#">ASP.NET Core SignalR Chat with Angular 5</a>
★★★	ASP.NET Identity	<a href="#">Introduction to Identity on ASP.NET Core</a>
★★★★	Кэширование в ASP.NET, Использование E-Tag	<a href="#">Кэширование в памяти в ASP.NET Core</a> ● <a href="#">HTTP ETag</a>
★★★★	Версионирование API	<a href="#">API Versioning in Asp.Net Core 2.0</a> ● <a href="#">Four REST API Versioning Strategies</a>
★★★★	gRPC в ASP.NET Core	<a href="#">Введение в gRPC</a> ● <a href="#">Why gRPC?</a>
★★★★	Identity Server	<a href="#">IdentityServer</a>

### Контрольные вопросы:

- Что такое REST?

- В чем разница между HTTP методами POST и PUT?
- Опишите жизненный цикл запроса в ASP.NET MVC.
- Для чего предназначена спецификация OWIN?
- Как сохранить состояние на стороне сервера для не аутентифицированного пользователя?
- Для чего применяется версионирование API?
- Какой транспортный протокол использует библиотека SignalR?
- Что такое JWT токены?
- Что такое E-Tag?

## Фронтенд разработка: JavaScript / TypeScript / SPA фреймворки



Уровень	Темы	Материалы
★	HTML: Структура HTML-документа, DOM Основные теги: html, body, title, head, p, br, img и другие. Атрибуты Id и Class.	<a href="#">HTML Tags</a> ● Структура HTML-кода
★	CSS: классы, идентификаторы, селекторы, применение стилей, позиционирование элементов	<a href="#">Основы CSS</a> ● CSS для JavaScript-разработчика
★	Понимание основных принципов работы браузеров	<a href="#">Как работают браузеры: принципы работы современных веб-браузеров</a> ● <a href="#">Важные аспекты работы браузера для разработчиков. Часть 1</a> ● <a href="#">Современный учебник JavaScript, Часть 2</a>
★	Введение в JavaScript	<a href="#">Введение в JavaScript</a>
★	Основы JavaScript: Переменные, Строки, Функции, Типы данных, Операторы, Преобразования типов, Циклы, Массивы, Объекты	<a href="#">Современный учебник JavaScript</a>
★	Классы в JavaScript: Синтаксис, Наследование, Приватные члены, Статические члены, Оператор instanceof	<a href="#">Современный учебник JavaScript</a>

Уровень	Темы	Материалы
★	Коллекции Map и Set	Современный учебник JavaScript
★	Использование setTimeout и setInterval	Планирование: setTimeout и setInterval
★	Разница между JavaScript и TypeScript	Разница между Javascript и TypeScript
★	Понятие Single Page Application (SPA)	Одностраничное приложение ● Single Page Applications — что это?
★	Работа с jQuery	jQuery
★	Промисы (Promises), Ключевые слова async/await	Промисы ● Цепочка промисов ● Async/await
★★	Работа с SPA фреймворками: React.js, Angular.js, Vue.js	React Tutorial ● Vue.js Tutorial ● Angular7 Tutorial
★★	Работа с Node Package Manager (NPM)	NPM для простых смертных ● NPM для простых смертных
★★	Работа с Cookies, Local Storage, Session Storage	Cookie ● LocalStorage, sessionStorage ● Local Storage vs Cookies
★★	Работа с Developer Tools: Отладка, Анализ исходящих запросов, Использование профайлеров	Chrome DevTools ● Chrome DevTools - 20+ Tips and Tricks
★★	Работа с объектами window, document, screen	The Window Object ● Document ● The Screen Object
★★★	Модули: Экспорт и импорт, Динамические импорты	Модули, введение ● Экспорт и импорт ● Динамические импорты
★★★★	Работа с замыканиями (closures)	Замыкание
★★★★	Прототипное наследование, Свойство prototype	Прототипное наследование ● Прототипное наследование
★★★★	Работа с IndexedDB API	HTML5 - IndexedDB

Уровень	Темы	Материалы
★★★	Работа с WebWorkers	<a href="#">HTML5 Web Workers</a>
★★★	Понимание Event Loop в JavaScript	<a href="#">The JavaScript Event Loop</a>

### Контрольные вопросы:

- JavaScript компилируемый или интерпретируемый? Однопоточный или многопоточный?
- В чем разница между EcmaScript и JavaScript?
- В чем разница между ключевыми словами var и let?
- Какие типы данных существуют в JavaScript
- В чем разница между значениями null и undefined?
- Для чего применяется оператор instanceof?
- В чем разница между JavaScript и TypeScript?
- Для чего необходимы многоуровневые промисы?
- В чем преимущества SPA над веб-сайтами?
- В чем разница между Cookies и Local Storage?
- Как реализовать замыкание в JavaScript?
- Для чего необходимо свойство prototype?
- Как работает Event Loop в JavaScript?

## Веб-безопасность, Шифрование данных, OWASP

Уровень	Темы	Материалы
★	Концепции Аутентификации и Авторизации	<a href="#">В чем состоит разница между аутентификацией и авторизацией</a>
★	Шифрование vs Хеширование	<a href="#">Что такое хеширование, шифрование и кодировка</a>
★	Симметричное и Асимметричное шифрование	<a href="#">Описание симметричного и асимметричного шифрования ● Асимметричное шифрование. Как это работает?</a>

Уровень	Темы	Материалы
★	Лучшие практики для веб-разработчиков по работе с паролями	6 ways developers can build in better password security
★	Простые веб-атаки: DoS/DDoS, XSS, CSRF, SQL-injection и механизмы защиты	DoS и DDoS-атаки: значение и различия ● XSS уязвимость ● Атака CSRF ● SQL injection для начинающих. Часть 1
★	OAuth 2.0 протокол	Введение в OAuth 2 ● OAuth 2.0 простым и понятным языком
★	Аутентификация на основе токенов, JWT токены	Про токены, JSON Web Tokens (JWT), аутентификацию и авторизацию. Token-Based Authentication
★	Алгоритмы симметричного шифрования в .NET: AES, DES 3DES	Cryptography in .NET
★	Алгоритмы асимметричного шифрования в .NET: RSA, DSA, ECDiffieHellman	Cryptography in .NET
★	Алгоритмы хеширования в .NET: MD5, SHA256	Hashing algorithms and their practical usage in .NET Part 1
★★	Механизм работы HTTPS протокола, HSTS	Чем отличается HTTP от HTTPS ● HSTS ● Принудительное применение HTTPS в ASP.NET Core
★★	Cross-Origin Resource Sharing (CORS)	Cross-Origin Resource Sharing (CORS)
★★	OWASP TOP 10	OWASP Top Ten Project ● OWASP TOP-10: практический взгляд на безопасность веб-приложений
★★	Microsoft Security Development Lifecycle	Microsoft Security Development Lifecycle

## Контрольные вопросы:

- В чем отличия Аутентификации от Авторизации?
- Что такое хеширование?
- В чем разница между Симметричным и Асимметричным шифрованием?
- В чем смысл протокола OAuth?
- Как происходит XSS атака?
- Как происходит CSRF атака?
- Откуда берутся сертификаты, которые использует HTTPS протокол?
- В чем разница между HTTPS и HSTS?
- В чем смысл CORS?
- Какие уязвимости из OWASP TOP 10 Вам знакомы?
- Какие алгоритмы симметричного шифрования Вам знакомы?

## Распределенные системы и Микросервисная архитектура



Уровень	Темы	Материалы
★	Что такое распределенная система?	Введение в распределенные системы ● Lecture 9 Scalability Harvard Web Development David Malan
★★	Вертикальное и горизонтальное масштабирование	Горизонтальное и вертикальное масштабирование веб приложений ● Lecture 9 Scalability Harvard Web Development David Malan
★★	Теорема CAP	CAP Theorem: Revisited ● The System Design Primer
★★	Концепция балансировки нагрузки	Lecture 9 Scalability Harvard Web Development David Malan ● Load balancer

Уровень	Темы	Материалы
★★★★	Кеширование в распределенных системах	The System Design Primer, Cache ● Scalability for Dummies - Part 3: Cache
★★★★	Репликации баз данных	The System Design Primer, Replications
★★★★	Выбор между Performance vs scalability, Latency vs throughput, Availability vs consistency в распределенных системах	The System Design Primer, Trade offs
★★★★	Consistency patterns: Weak consistency, Eventual consistency, Strong consistency	The System Design Primer, Consistency Patterns
★★★★	Availability patterns: Active-passive Failover, Active-active Failover, Master-slave replication, Master-master replication	The System Design Primer, Availability Patterns
★★★★	SQL or NoSQL, NoSQL: Key-value store, Document store, Wide column store, Graph Database	The System Design Primer, NoSQL
★★★★	Построение микросервисной архитектуры	Статьи Мартина Фаулера по Микросервисам ● Книга Building Microservices: Designing Fine-Grained Systems 1st Edition by Sam Newman
★★★★	Распределенные транзакции, 2PC, Saga Pattern	Patterns for distributed transactions within a microservices architecture ● Pattern: Saga
★★★★	Миграция с монолита на микросервисы, Шаблон Strangler	Шаблон Strangler ● Monolith to Microservices Using the Strangler Pattern

### Контрольные вопросы:

- В чем разница между Вертикальным и горизонтальным масштабированием?
- В чем смысл теоремы CAP?
- Какие существуют алгоритмы балансировки нагрузки?

- Опишите шаблон кеширования Cache-aside.
- В чем разница между SQL и No-SQL
- Что такое Eventual consistency?
- Как добиться транзакционности в микросервисной архитектуры?
- Опишите идею шаблона Strangler.

## Базы данных: T-SQL, MS SQL, NoSQL

Уровень	Темы	Материалы
★	Концепция нормализации данных, Нормальные формы	Нормализация отношений. Шесть нормальных форм ● Описание основных приемов нормализации базы данных
★	Создание таблиц, Выбор типов данных	Создание таблиц ● Типы данных
★	Создание ограничений: NOT NULL, UNIQUE, CHECK, DEFAULT, PRIMARY KEY, FOREIGN KEY	Ограничения уникальности и проверочные ограничения
★	Моделирование отношений: один к одному, один ко многим, многие ко многим, Primary Key & Foreign Key	Типы связей в реляционных базах данных
★	Работа с оператором SELECT, Использование операторов WHERE, INNER/LEFT/RIGHT/CROSS JOIN, GROUP BY, HAVING, DISTINCT	Примеры использования инструкции SELECT ● Предложение HAVING ● Предложение GROUP BY
★	Работа агрегатными функциями: SUM, MIN, MAX, AVG, COUNT	Обобщение данных с помощью агрегатных функций
★	Модификация данных с использованием INSERT, UPDATE, DELETE операторов	Учебник по языку SQL (DDL, DML) на примере диалекта MS SQL Server. Часть вторая ● SQL команды DELETE и TRUNCATE
★★	Работа с операторами UNION, UNION ALL, INTERSECT, EXCEPT	UNION / EXCEPT / INTERSECT



Уровень	Темы	Материалы
★★	Работа с хранимыми процедурами, функциями и триггерами	Основы T-SQL и примеры — функции (UDF), триггеры, процедуры, курсоры, циклы
★★	Использование SQL профайлера	РАБОТА С SQL SERVER PROFILER. ПРИМЕРЫ НАСТРОЙКИ ТРАССИРОВОК
★★	Кластерные и некластерные индексы	Описания кластеризованных и некластеризованных индексов ● 14 вопросов об индексах в SQL Server, которые вы стеснялись задать
★★	SQL vs NoSQL	SQL или NoSQL — вот в чём вопрос
★★★★	Необходимость денормализации, Техники денормализации данных	Денормализация БД. Зачем? Когда? Как?
★★★★	Работа с оконными функциями: OVER, ROW_NUMBER, RANK, NTILE...	Оконные функции ● Оконные функции – то, что должен знать каждый T-SQL программист
★★★★	Работа с CTE, Рекурсивные CTE	Общие табличные выражения
★★★★	Анализ плана выполнения запроса: Nested Loops, Index Seek, Index Scan, Key, Lookup...	Отображение действительного плана выполнения ● Изучение плана запроса в SQL

Уровень	Темы	Материалы
☆☆☆	Работа с транзакциями, ACID, Уровни изоляций транзакций, Snapshot, Мертвые блокировки	<a href="#">T-SQL - Transactions</a> ● <a href="#">A Primer on ACID</a> <a href="#">Transactions: The Basics</a> <a href="#">Every Cloud App Developer Must Know</a> ● <a href="#">Snapshot Isolation in SQL Server</a> ● <a href="#">SQL Server Transactions and Isolation Levels</a>

### Контрольные вопросы:

- Какие нормальные формы Вам знакомы?
- Сравните нормализацию и денормализацию баз данных.
- Как моделируется отношение "многие ко многим" в MS SQL?
- В чем разница между Primary Key и Foreign Key?
- В чем разница между операторами LEFT JOIN vs INNER JOIN?
- В чем разница между операторами CROSS JOIN и CROSS APPLY?
- В чем отличия Кластерных индексов от Некластерных?
- Для чего применяется оператор INCLUDE при создании некластерного индекса?
- Какие уровни изоляций транзакций существуют?
- Опишите свойства ACID.
- Как избегать мертвых блокировок при работе с транзакциями?
- В чем разница между временными таблицами и CTE?

### Системы контроля версий: Git, Mercurial, TFS

Уровень	Темы	Материалы
☆	Понимание необходимости систем контроля версий	<a href="#">Система управления версиями</a> ● <a href="#">Введение в системы контроля версий</a>
☆	Понимание разницы между централизованными и распределенными системами контроля версий	<a href="#">Сравнение централизованных и распределенных систем управления версиями</a>
☆	Жизненный цикл в Git	<a href="#">Руководство по Git.</a> <a href="#">Жизненный цикл Git.</a>

Уровень	Темы	Материалы
★	Работа с Git, Понимание ключевых терминов: repository, branch, commit, push, tag, pull request, merge, revert, stage, blame...	<a href="#">Git tutorial. Become a git guru.</a>
★★	Понимание Git Flow	<a href="#">GitHub Flow ●</a> <a href="#">Introducing GitFlow</a>
★★	Работа с ветками, стратегии "бранчевания"	<a href="#">Branching стратегии в Git</a>
★★★	Понимание merge-стратегий: merge commit, merge squash, fast-forward merge, rebase	<a href="#">Pull request merge strategies</a>
★★★★	Миграция на Git с других систем контроля версий	<a href="#">SVN to Git - prepping for the migration</a>

### Контрольные вопросы:

- В чем разница между централизованными и распределенными системами контроля версий?
- В чем разница между командами git fetch и git pull в Git?
- В чем разница между merge commit и merge squash в Git?
- Что такое cherry-pick в Git?
- Для чего применяются теги в Git?
- Для чего применяются пул-реквесты в Git?

## Методологии разработки: Scrum, Kanban и Scrumban

Уровень	Темы	Материалы
★	Понимание Scrum процесса	<a href="#">Гибкая методология разработки "Scrum" ●</a> <a href="#">Что такое Scrum</a>
★	Роли в Scrum: Product Owner, Scrum Master, Team	<a href="#">Роли в Scrum ●</a> <a href="#">Scrum: 12 терминов, которые нужно запомнить</a>

Уровень	Темы	Материалы
★	Эстимации в Scrum: Story points, Planning poker, Story Points vs Man-hours	12 советов по эстимации проектов по разработке от практикующих пресейл-специалистов ● Как научиться оценивать задачи, если не умеешь: 4 фактора сложности ● Стори поинты: как это работает
★	Церемонии в Scrum: Sprint/release planning, Daily scrum, sprint demo, sprint retrospective	Agile-церемонии
★★	Понимание Kanban методологии, Scrum vs Kanban	Что такое Канбан ● Разбираемся в Scrum и Kanban ● Scrum vs Kanban: в чем разница и что выбрать?
★★	Понимание Scrumban методологии, Необходимость применения	Scrum-ban
★★	Методология экстремального программирования	Книга Экстремальное программирование, Кент Бек ● Методики и принципы экстремального программирования

### Контрольные вопросы:

- Опишите Scrum процесс.
- Какие роли существуют в Scrum?
- Какие церемонии существуют в Scrum?
- В чем сходства и отличия методологий Scrum и Kanban?
- В чем разница между оценкой в Story points и человеко-часах?

Комментарии (0)

Авторизируйтесь для участия в дискуссии