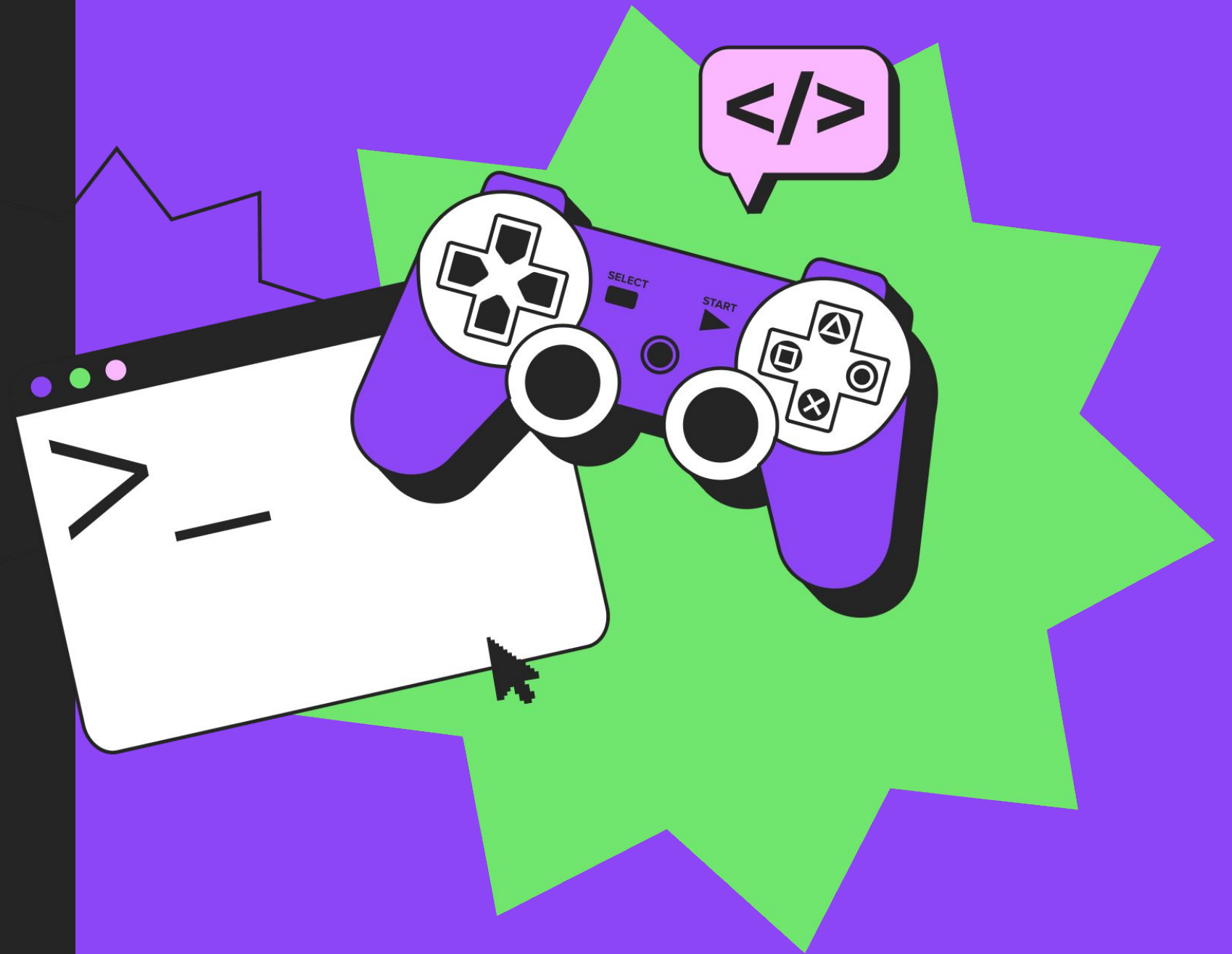


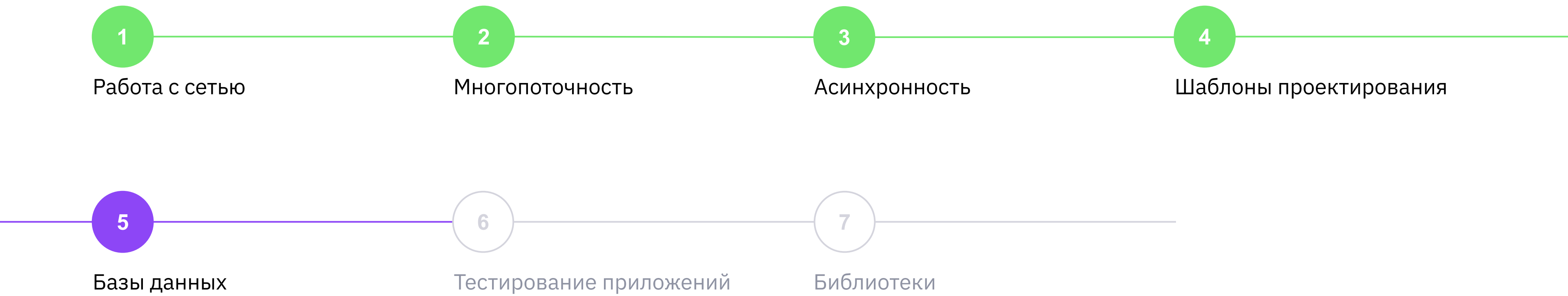
Разработка сетевого приложения на C#

Урок 5
Базы данных





План курса





Содержание урока

- Как работать с базами данных
- ADO.Net, Linq2Sql и Entity Framework
- Научимся работать с Entity Framework и выполнять все необходимые операции для работы с базами



Базы данных

Базы данных играют ключевую роль в современной информационной технологии и используются для хранения, управления и организации больших объемов данных. Они имеют множество применений в различных областях, включая бизнес, науку, здравоохранение, образование и другие.

- Хранение данных
- Управление данными
- Поиск и извлечение данных
- Анализ данных
- Совместное использование данных
- Сохранение структуры данных
- Системы управления базами данных (СУБД)
- Безопасность данных





SQL

Структурированный язык запросов, который позволяет совершать все необходимые операции, такие как: создание, удаление базы; создание, удаление и модификация таблиц; получение, модификация и вставка данных.

Примеры SQL-запросов:

- выбор всех всех строк из таблицы `some_table1`:

```
SELECT * FROM some_table1;
```

- выбор столбца `name` всех строк из таблицы `some_table`:

```
SELECT name FROM some_table
```

- выбор столбца `name` всех строк из таблицы `some_table` , где возраст равен 18:

```
SELECT name FROM some_table WHERE age = 18
```

- удаление всех столбцов и всех строк из таблицы `some_table1`:

```
DELETE * FROM some_table
```



SQL

Пример сложного SQL-запроса:

- Выбор данных из двух таблиц, связанных по ключу:

```
SELECT * from
```

```
some_table JOIN another_table
```

```
ON some_table.anotherId = another_table.id
```

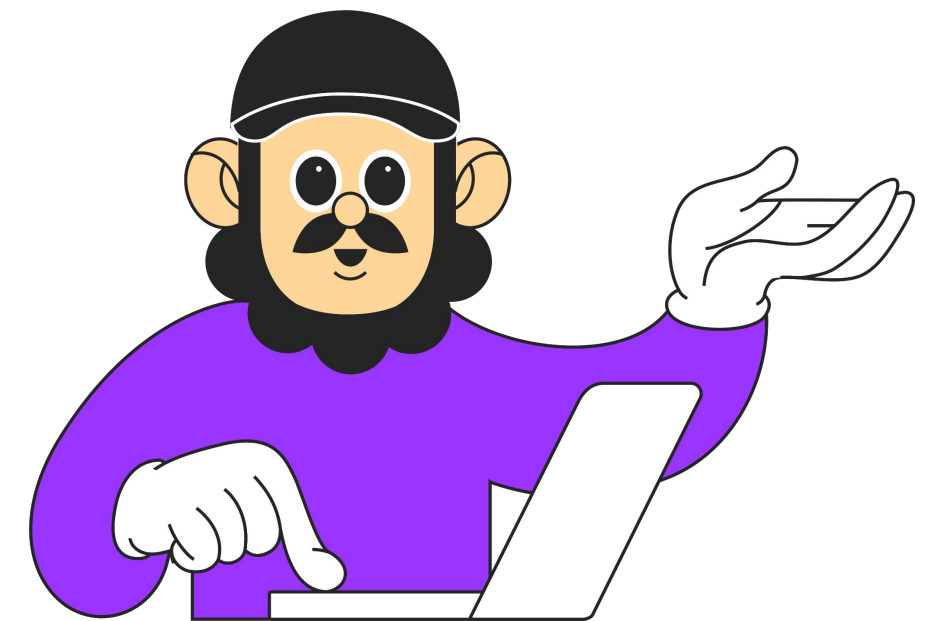


Docker

Docker – это платформа для разработки, доставки и запуска приложений в контейнерах. Контейнеры – это изолированные среды, которые включают в себя все необходимое для запуска приложения: код, библиотеки, настройки и зависимости. Docker обеспечивает стандартизацию окружения между разработкой и продакшн средами, что упрощает развертывание и масштабирование приложений.

Основные концепции Docker:

- Образы (Images)
- Контейнеры (Containers)
- Dockerfile
- Регистры (Registries)
- Docker Compose
- Сети и тома
- Оркестрация

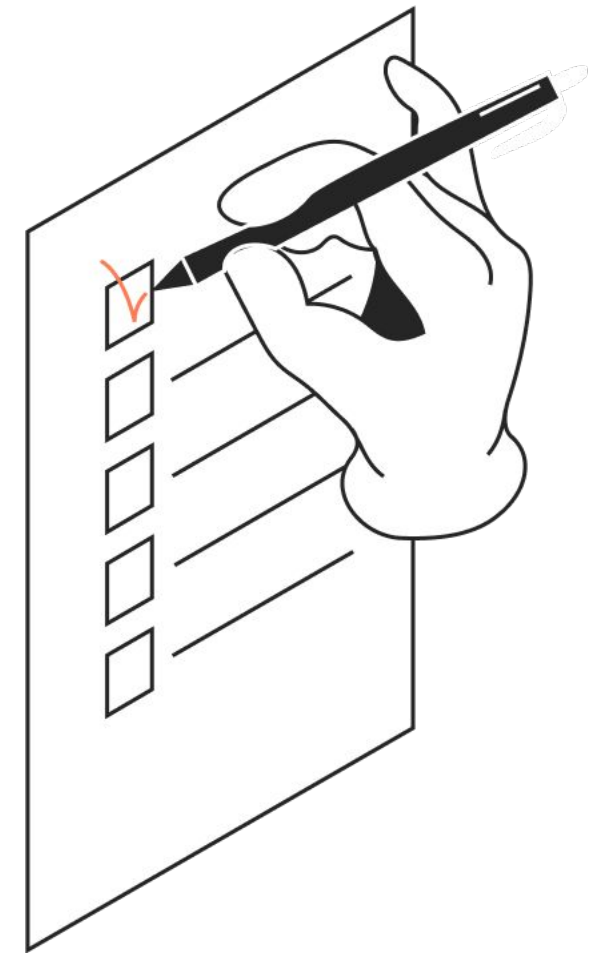




ADO.Net

Основные компоненты ADO.NET:

- Connection
- Command
- DataReader
- DataAdapter
- DataSet
- DataTable
- DataView
- Transaction





ORM

ORM (Object-Relational Mapping) – это технология, которая предоставляет абстракцию над реляционными базами данных и позволяет разработчикам работать с данными в виде объектов и классов вместо табличных структур и SQL-запросов. ORM устанавливает связь между объектно-ориентированным программированием и реляционными базами данных, обеспечивая более удобный и естественный способ взаимодействия с данными.

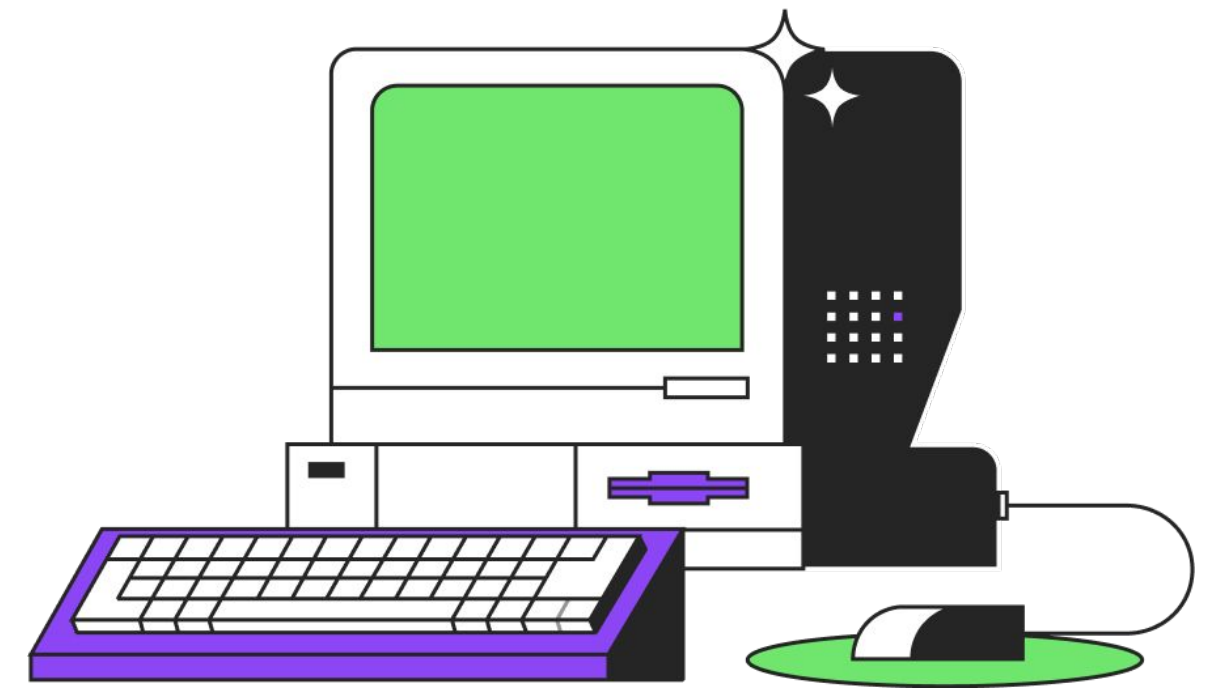
Основные принципы и возможности ORM:

- Отображение объектов на таблицы
- Автоматическое создание SQL-запросов
- Управление отношениями
- Автоматическое создание схемы базы данных
- Управление состоянием объектов
- Кэширование
- Поддержка транзакций



Entity Framework

Entity Framework (EF) – это ORM (Object-Relational Mapping) для языка программирования C#. Он предоставляет удобные и эффективные средства для работы с данными и реляционными базами данных в приложениях на платформе .NET



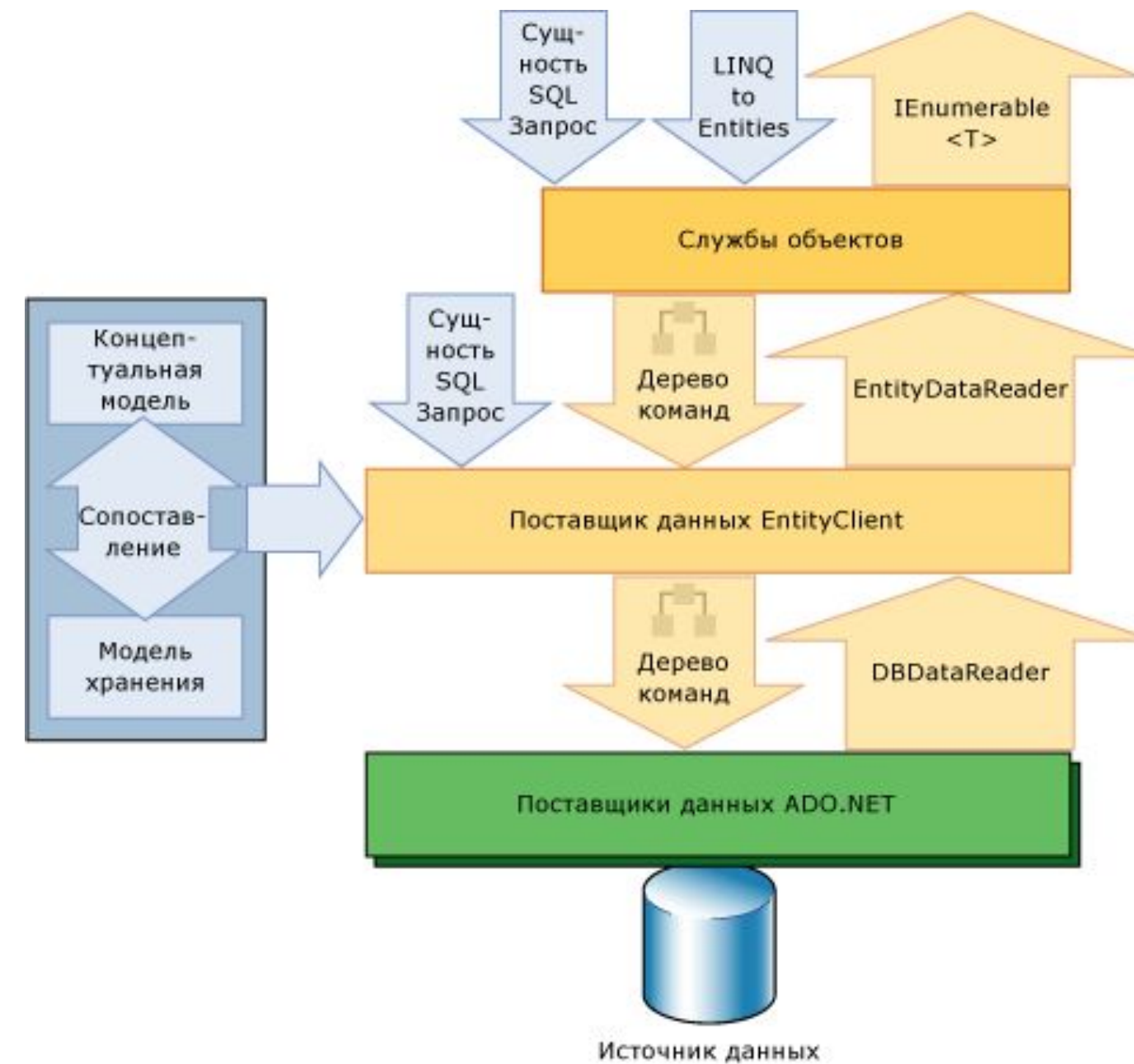


Entity Framework

- Увеличение производительности разработки
- Автоматическое создание SQL-запросов
- Маппинг данных
- Поддержка LINQ
- Поддержка отношений
- Управление состоянием объектов
- Кэширование
- Поддержка транзакций
- Поддержка множественных источников данных
- Интеграция с Visual Studio
- Открытый исходный код
- Сообщество и экосистема



Компоненты EF





Концептуальная модель данных

Концептуальная модель (Conceptual Model) – это абстрактное представление данных и их отношений в информационной системе или приложении. Эта модель описывает структуру и семантику данных, независимо от способа их хранения и физической реализации. Концептуальная модель служит основой для проектирования базы данных и обеспечивает понимание информационных потребностей и бизнес-процессов.

Важные аспекты концептуальной модели:

- Сущности (Entities)
- Атрибуты (Attributes)
- Отношения (Relationships)
- Ограничения (Constraints)
- Абстракция от физической реализации



Маппинг-модель (сопоставление)

Mapping Model в контексте баз данных и ORM (Object-Relational Mapping) представляет собой способ сопоставления концептуальной модели данных (определенной на уровне объектов в коде) с физической моделью данных (структура таблиц и отношений в реляционной базе данных). Она определяет то, как сущности и связи между ними в коде соотносятся с таблицами, столбцами и ключами в базе данных.

Ключевые аспекты маппинг-модели:

- Сущности (Entities)
- Атрибуты (Attributes)
- Отношения (Relationships)
- Ограничения (Constraints)
- Хранимые процедуры и функции
- Оптимизации запросов



Модель хранилища

Storage Model в контексте базы данных и ORM (Object-Relational Mapping) представляет собой физическую модель данных, которая описывает структуру базы данных, включая таблицы, столбцы, ключи, ограничения и другие объекты базы данных. Эта модель представляет данные в том формате, в котором они хранятся и обрабатываются в реальной реляционной базе данных.

Важные аспекты хранилища модели:

- Таблицы и столбцы
- Ограничения (Constraints)
- Индексы
- Представления (Views)
- Хранимые процедуры и функции
- Безопасность (Security)



LINQ to Entities

Предоставляет интегрированный язык запросов для выполнения запросов к данным, хранящимся в базе данных, с использованием концептуальной модели объектов. Он позволяет разработчикам использовать выразительный и типобезопасный синтаксис C# (или других .NET языков) для формулирования запросов к данным, а EF обеспечивает преобразование этих запросов в SQL-запросы, которые выполняются на сервере базы данных.

Ключевые особенности LINQ to Entities:

- Интеграция с C#
- Типобезопасность
- Использование концептуальной модели
- Генерация SQL
- Ленивая и жадная загрузка
- Выражение запроса



Entity SQL

Язык запросов, предназначенный для выполнения запросов к данным в Entity Framework (EF). Этот язык создан специально для работы с концептуальной моделью данных и позволяет выражать запросы к данным в более низком уровне, чем LINQ to Entities. Entity SQL предоставляет возможность формулировать запросы к данным, используя строковый синтаксис, а не интегрированный язык запросов, как LINQ.

Важные особенности Entity SQL:

- Строковый синтаксис
- Низкоуровневый доступ
- Прямой SQL-запрос
- Операторы LINQ



Object Services Layer

Object Services работают с объектами, представляющими сущности базы данных, и предоставляют API для выполнения запросов и операций с этими объектами. Она выполняет следующие функции:

- Отслеживание объектов
- Выполнение запросов
- Создание, обновление и удаление данных
- Управление транзакциями
- Оптимизация запросов
- Загрузка связанных данных



Entity Client Data provider

Обеспечивает доступ к данным в базах данных с использованием Entity Framework. Этот провайдер данных предоставляет абстракцию между приложением и базой данных, что позволяет Entity Framework взаимодействовать с различными системами управления базами данных (СУБД).

Ключевые характеристики Entity Client Data Provider:

- Поддержка разных СУБД
- Генерация SQL-запросов
- Преобразование данных
- Управление подключением
- Конфигурация и настройка



ADO.Net Data Provider

Обеспечивает доступ к данным в базе данных из приложения, используя технологию ADO.NET. ADO.NET Data Providers предоставляют интерфейс для взаимодействия с разными системами управления базами данных (СУБД) из .NET-приложений. В контексте Entity Framework (EF) ADO.NET Data Provider используется как нижележащий компонент для взаимодействия с базой данных.

Основные аспекты ADO.NET Data Provider в разрезе Entity Framework:

- Совместимость с разными СУБД
- Установка и конфигурация
- Генерация SQL-запросов
- Преобразование данных
- Управление подключением
- Кэширование и оптимизация



Database first

В этом подходе вы начинаете с существующей базы данных и создаете модель данных на основе этой базы данных.

Основные шаги и характеристики подхода Database First в Entity Framework:

- Существующая база данных
- Создание модели данных
- Создание классов сущностей
- Генерация SQL-запросов
- Обновление модели данных
- Использование готовой модели



Model first

В этом подходе вы начинаете с создания модели данных в среде разработки, а затем Entity Framework автоматически генерирует схему базы данных

Основные шаги и характеристики подхода "Model First" в Entity Framework:

- Создание модели данных
- Генерация схемы базы данных
- Создание классов сущностей
- Работа с данными
- Изменение модели



Code first

В этом подходе вы начинаете с создания классов сущностей (Entity Classes) и определения модели данных в коде вашего приложения, а затем Entity Framework автоматически создает базу данных и схему на основе этой модели.

Основные шаги и характеристики подхода Code First в Entity Framework:

- Создание классов сущностей
- Контекст данных (DbContext)
- Конфигурация модели
- Инициализация базы данных
- Миграции
- Работа с данными



Транзакция

Это инструмент, позволяющий выполнить ряд операций с базой данных атомарно или же отменить все сделанные изменения.






Подведение итогов

На этой лекции вы:

- Научились работать с базами данных с применением различных подходов
- Узнали про ADO.Net и Entity Framework
- Узнали про транзакции
- Научились выполнять отладки запросов

Спасибо 
за внимание

