***Ответ от чата, поскольку в лк практически нету инфы***

**ASP.NET Web API**

ASP.NET Web API — это платформа для построения RESTful сервисов, которая позволяет создавать HTTP API, поддерживающие взаимодействие различных приложений через интернет. Веб-API позволяет обмениваться данными в формате JSON или XML, а также взаимодействовать через HTTP-запросы (GET, POST, PUT, DELETE и др.). Это один из популярных способов создания серверной логики для мобильных, веб- и настольных приложений.

**Основные понятия Web API:**

1. **Контроллеры (Controllers):** Контроллер — это класс, который обрабатывает входящие HTTP-запросы. В Web API контроллеры наследуются от класса ApiController. Каждый метод контроллера обычно соответствует определенному HTTP-методу:
   * GET — для получения данных,
   * POST — для отправки данных,
   * PUT — для обновления данных,
   * DELETE — для удаления данных.
2. **Маршрутизация (Routing):** В Web API маршрутизация определяет, как запросы URL сопоставляются с контроллерами и их методами. В Web API по умолчанию используется атрибутивная маршрутизация (attribute routing), где маршруты могут задаваться атрибутами прямо над методами контроллера, например, [Route("api/values/{id}")].
3. **HTTP-глаголы (HTTP Verbs):** Web API использует стандартные HTTP-глаголы для определения типов запросов:
   * **GET**: используется для получения данных,
   * **POST**: используется для отправки новых данных на сервер,
   * **PUT**: обновляет существующие данные,
   * **DELETE**: удаляет данные.
4. **Данные (JSON, XML):** Web API может работать с различными форматами данных, но чаще всего данные передаются в формате **JSON** (JavaScript Object Notation). Этот формат удобен для передачи структурированных данных и широко используется в веб-приложениях.
5. **CORS (Cross-Origin Resource Sharing):** Чтобы API могло быть доступно с других доменов, используется механизм CORS. Этот механизм позволяет контролировать, с каких доменов могут быть отправлены запросы к вашему API. В ASP.NET Web API CORS можно включить с помощью атрибута [EnableCors].

**Ключевые преимущества Web API:**

* Поддержка стандартных HTTP-глаголов и статусов.
* Легкость интеграции с разными платформами (веб, мобильные приложения, десктоп).
* Простота разработки и масштабируемость.
* Поддержка форматов JSON и XML.
* Возможность работы с различными механизмами аутентификации и авторизации.

**ASP.NET MVC**

**ASP.NET MVC (Model-View-Controller)** — это архитектурный шаблон, который разделяет приложение на три основных компонента:

* **Модель (Model)**,
* **Представление (View)**,
* **Контроллер (Controller)**.

Эта архитектура помогает в разработке приложений, обеспечивая лучшее разделение логики, представления данных и взаимодействия с пользователем.

**Основные понятия MVC:**

1. **Модель (Model):** Модель представляет собой данные приложения, а также логику, связанную с этими данными. Это может быть как объект данных (например, пользователь), так и логика взаимодействия с базой данных или другими источниками данных.
2. **Представление (View):** Представление отвечает за визуальное отображение данных. Оно получает информацию от контроллера и формирует страницу для отображения пользователю. В ASP.NET MVC представления обычно создаются с использованием Razor-движка (шаблоны с расширением .cshtml).
3. **Контроллер (Controller):** Контроллер управляет взаимодействием между моделью и представлением. Когда пользователь отправляет запрос (например, открывает страницу или отправляет форму), контроллер получает этот запрос, обрабатывает его, взаимодействует с моделью и передает данные представлению для отображения.
4. **Маршрутизация (Routing):** В ASP.NET MVC также используется маршрутизация для сопоставления запросов URL с действиями контроллера. Маршруты определяются в файле RouteConfig или с помощью атрибутивной маршрутизации.

**Пример работы MVC:**

* Пользователь делает запрос на получение данных (например, через URL /Products/Details/1).
* Контроллер получает этот запрос, извлекает данные из модели (например, информацию о продукте с ID 1).
* Эти данные передаются представлению, которое генерирует HTML-код для страницы.
* Пользователь видит сформированную страницу с информацией о продукте.

**Преимущества MVC:**

* Разделение приложения на компоненты делает код более организованным и поддерживаемым.
* Легкость в тестировании отдельных частей приложения (например, контроллеров).
* Полный контроль над HTML-разметкой и результатами рендеринга.
* Возможность создания легко расширяемых и масштабируемых приложений.

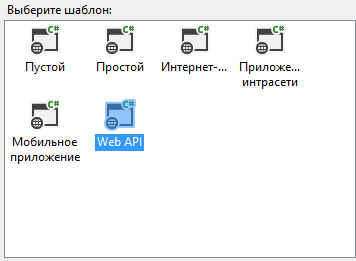
**Различия между Web API и MVC:**

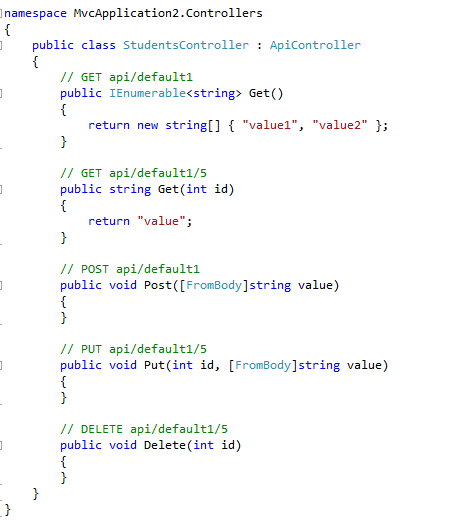
1. **Цель:**
   * **Web API** предназначен для создания RESTful сервисов, которые возвращают данные (JSON, XML) и взаимодействуют с другими системами через HTTP.
   * **MVC** фокусируется на рендеринге представлений (HTML), управлении страницами и пользовательским интерфейсом.
2. **Тип ответов:**
   * **Web API** возвращает данные в формате JSON, XML или других форматов.
   * **MVC** возвращает представления (HTML), которые отображаются в браузере.
3. **Использование HTTP-глаголов:**
   * В **Web API** HTTP-глаголы (GET, POST, PUT, DELETE) соответствуют методам контроллера.
   * В **MVC** контроллеры могут использовать HTTP-глаголы для разных действий, но основное взаимодействие происходит через формы и стандартные действия.

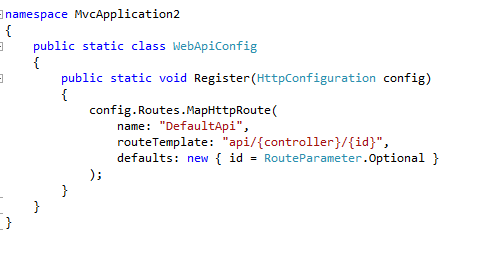
Обе технологии могут использоваться совместно в одном приложении: **MVC** для рендеринга представлений, а **Web API** — для предоставления данных другим системам через API.

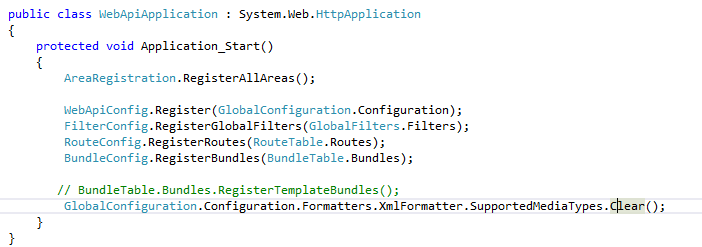
Лекция Смелова:

1. **REST: создание REST-службы на базе ASP.NET MVC WEB API**

****

****

****

****

От себя скажу, что отличие Web Api (2nd laba) от Asp.net http-handler (1st laba) в том, что у нас уже нету файлика web.config, маршрутизация определяется другим файлом (пр., WebApiConfig.cs):

config.Routes.MapHttpRoute(

name: "DefaultApi",

routeTemplate: "api/{controller}/{id}",

defaults: new { id = RouteParameter.Optional }

);

Как видим, наименование контроллера определяется динамически. Т.е. если у нас будет запрос на api/Values, то нас переадресует на ValuesController (согласование имен)