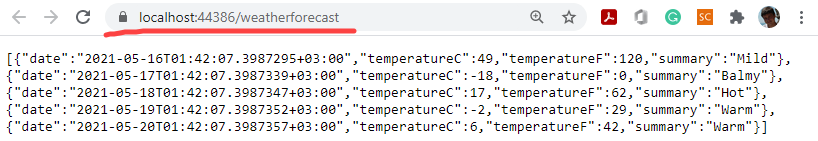
**Создание серверной части приложения**

**Задача 1. Получение конкретного товара**

**Создайте с помощью мастера в Visual Studio новый проект Web API (ASP.Net Core Web API).**

**Изучите проект, в нем уже есть контроллер и при запуске можно обратиться к нему по относительному пути /WeatherForecast/, получив JSON:**

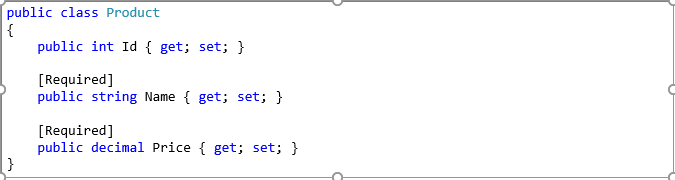


**Вам необходимо внести изменения в проект, добавив возможность работы с товарами через Web API.**

**Для этого создайте в папке Controllers новый контроллер с именем** ProductsController**, осуществляющий работу со списком товаров, добавив перед классом атрибут:**

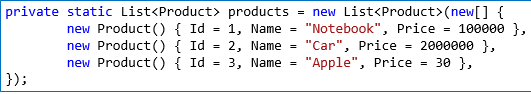
[Route("/api/[controller]")]

**Для Web API необходимо определить модели. Для этого создайте папку Models (если ее нет) и разместите в ней код, описывающий товар:**



**Подсказка:** атрибут [Required] из System.ComponentModel.DataAnnotations указывает на обязательность заполнения данных.

**В соответствии с моделью данных определите в классе контроллера** ProductsController **список товаров (для простоты – в виде статического поля):**



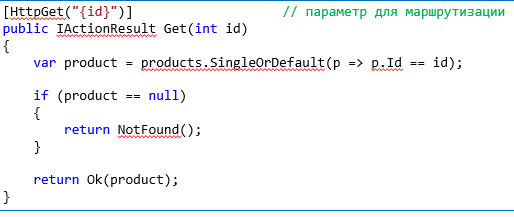
**Поскольку список товаров необходимо получать через API, добавьте метод для обработки HTTP GET-запроса к приложению:**



**Запустите приложение и убедитесь, что при обращении из браузера к относительному адресу /api/products**

**получаете список товаров в формате JSON.**

**Для получения сведений по конкретному товару вам необходимо добавить в класс контролера код, отвечающий за поиск и предоставление данных по товару с выбранным идентификатором (id):**



Проверьте, что по относительным адресам вы получаете правильный ответ:

**/api/products/1/ - есть товар**

**/api/products/1234/ - нет товара (HTTP 404)**

**CRUD –** аббревиатура, обозначающая четыре базовые функции, используемые при работе с базами данных: создание (create), чтение (read), модификация (update), удаление (delete). Термин введен James Martin в 1983 г. как стандартная классификация функций по манипуляции данными.

В системах, реализующих доступ к базе данных через API в стиле REST, эти функции реализуются зачастую (но не обязательно) через HTTP-методы POST, GET, PUT и DELETE соответственно:

Действие -> HTTP-метод

**Create -> POST**

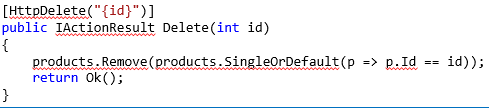
**Read -> GET**

**Update -> PUT**

**Delete -> DELETE**

**Задача 2. Удаление**

**Для удаления товара с использованием метода DELETE разместите в контролере следующий код:**



Проверьте, что вы можете удалить товар с идентификатором 1, отправив DELETE-запрос по относительному адресу **/api/products/1/**

Получите список товаров, чтобы убедиться, что товар с идентификатором 1 был удален.

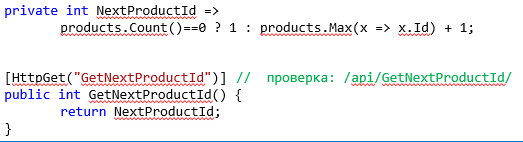
***Подсказка:***

Для отправки DELETE-запроса можно воспользоваться утилитой CURL (встроена в Windows):

**curl -X DELETE** [**https://localhost:44386/api/products/1**](https://localhost:44386/api/products/1)

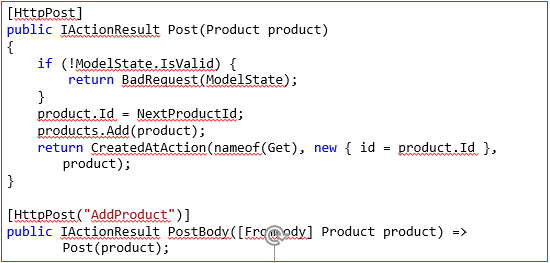
**Задача 3. Свободный идентификатор**

Как вы знаете, идентификаторы должны быть уникальны, поэтому при добавлении товара, необходимо использовать уникальный id. Добавьте в класс контроллера соответствующий код:



Проверьте с помощью браузера правильность нахождения следующего свободного идентификатора.

**Задача 4. Добавление товара**



Обратите внимание, что мы можем добавлять товар двумя способами:

* передавая характеристики товара в виде полей формы:

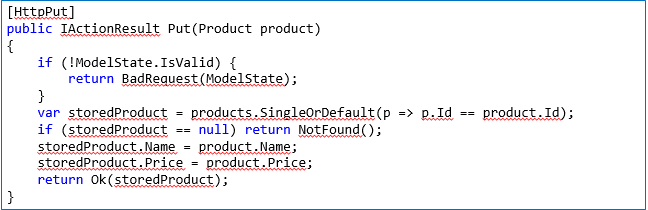
**curl -X POST -F Name=someName -F Price=123 https://localhost:44386/api/products/**

* передавая характеристики товара в виде JSON-объекта:

**curl -X POST -H "Content-Type: application/json" -d "{\"name\":\"JsonName\",\"price\":1234}" https://localhost:44386/api/products/AddProduct/**

**Задача 5. Изменение товара**

Добавьте в код контроллера поддержку обработки метода HTTP PUT для обновления данных по товару:



Обратите внимание, что изменение реализовано только через поля формы:

curl -X PUT -F Id=3 -F Name=UpdatedName -F Price=1234 https://localhost:44386/api/products/

**TODO:** самостоятельно реализуйте и протестируйте изменение товара через JSON (подсказка: параметр метода необходимо пометить атрибутом [FromBody]).