# Оглавление

[**Оглавление**](#_8y22pz1dd6rt) **1**

[**1. Основы программирования**](#_wbubud5qwel8) **2**

[1.1. Алгоритмы](#_i8yjasn5l3ve) 2

[1.2. Эволюция языков](#_24ncbi35ep7c) 2

[1.3. Обзор платформы .NET](#_up6x2o1rk8l0) 2

[1.4. Язык C#, общие концепции синтаксиса](#_sqp0kqqimayw) 2

[**1. Основы ООП**](#_nfcr6endum8b) **5**

[1.1. Принципы ООП](#_peuc8zmhc1kw) 5

[1.2. Инкапсуляция](#_alf1pej2e07a) 5

[1.3. Пространство имен](#_pkx8w9bw6kq3) 5

[1.4. Перегрузка операций](#_1bt1hsb8z5a6) 5

[**2. Наследование классов**](#_y548m2sujvtt) **5**

[9.5. Обзор WPF](#_nkdnao46a3mf) 6

[**1. Инструментарий разработчика**](#_7uttn32yzbg0) **8**

[1.1 Visual Studio 2017](#_4kct5uc1csp2) 8

[**2. Введение в Asp.NET MVC 5**](#_il6otsp4p36y) **8**

[1.1 Коротко о браузере](#_ukxohnc9qgl) 8

[1.2 Знакомство с работой HTTP протокола](#_ygujfa5lgyye) 9

[1.2.1 Заголовок (HTTP заголовки)](#_vhi6014dqf2n) 9

[1.2.2 Типы запроса (Методы запросов)](#_b6095rw7tazc) 9

[1.2.3 Актуальные проблемы работы с HTTP](#_oipbsxuqcgqp) 10

[1.2.4 Простейшие инструменты для анализа HTTP запроса](#_eeu3n8bkfmcy) 10

[1.3. Нововведение в ASP .NET MVC 5](#_ydgksjrum7xl) 10

[1.4. Предназначение ASP .NET MVC](#_uc0gsoz3ica7) 10

[**25 Bootstrap 3**](#_vrkipgmplyhs) **13**

[**28 ORM. ENTITY FRAMEWORK 6**](#_yas3avq9xocy) **14**

[**29 Asp.Net Identity Framework**](#_jo6franjtm1x) **15**

[**30 WEB API 2**](#_ijxmrxmswcpw) **15**

[**31 OWIN И KATANA**](#_lvt2rdn46urb) **15**

[**31 Библиотека SignalR. Push-уведомления**](#_k5oebju28k0d) **16**

# 1. Основы программирования

## 1.1. Алгоритмы

## 1.2. Эволюция языков

## 1.3. Обзор платформы .NET

## 1.4. Язык C#, общие концепции синтаксиса

2. Система типов языка С#

2.1. Ключевые слова, идентификаторы, литералы

2.2. Выражения и операции

2.3. Переменные и константы

2.4. Преобразование и приведение типов

3. Операторы языка C#

3.1. Арифметические

3.2. Отношения

3.3. Логические

3.4. Присваивания

3.5. Поразрядные

3.6. Оператор

3.7. Использование скобок

3.8. Деление

4. Операторы языка C#

4.1. Управляющие операторы

4.2. Операторы переходов

4.3. Операторы проверки условий

4.4. Операторы циклов

5. Массивы и строки

5.1. Начальные сведения о массивах

5.2. Одномерные массивы

5.3. Многомерные массивы

5.4. Ступенчатые массивы

5.5. Строки

6. Коллекции

6.1. Основы работы со стандартными коллекциями

6.2. Коллекции-списки

6.3. Коллекции-словари

6.4. Пользовательские коллекции

7. Введение в классы

7.1. Синтаксис объявления класса

7.2. Методы, объявление и вызов

7.3. Использование параметров

7.4. Модификаторы доступа

7.5. Разделяемые классы и методы

8. Введение в классы

8.1. Свойства и индексаторы

8.2. Конструкторы

8.3. Статические классы

8.4. Методы расширения

8.5. Конструкторы класса

8.6. Рекурсия

8.7. Стратегия поиска ошибок и отладка кода

8.8. Области видимости переменных

8.9. Класс System.Object

8.10. Жизненный цикл объекта

8.11. Структуры

8.12. Перечисления

# 1. Основы ООП

## 1.1. Принципы ООП

## 1.2. Инкапсуляция

## 1.3. Пространство имен

## 1.4. Перегрузка операций

# 2. Наследование классов

2.1. Основы наследования

2.2. Конструкторы и наследование

2.3. Виртуальные методы

2.4. Полиморфизм

2.5. Абстрактные классы

3. Классы, интерфейсы

3.1. Статические классы

3.2. Статик конструкторы

3.3. Абстрактные классы

3.4. Абстрактные компоненты

3.5. Интерфейсы

3.6. Наследование интерфейсов

4. Сборка мусора

4.1. Оператор new

4.2. Работа сборщика мусора

4.3. Поколения

4.4. Применение деструкторов

5. Делегаты и события

5.1. Делегаты

5.2. Анонимные функции

5.3. Лямбда-выражения

5.4. События

6. Генерация и обработка исключительных ситуаций

6.1. Основы обработки исключительных ситуаций

6.2. Последствия не перехвата исключений

6.3. Перехват исключений

7. Работа с объектами файловой системы

7.1. Использование потоков данных

7.2. Классы для работы с потоками

7.3. Основы XML

7.4. Сериализация

8. Основы многопоточного программирования

8.1. Многопоточность

8.2. Синхронизация потоков (lock, mutex, семафор)

8.3. Процессы и домены

8.4. TPL

9. Перечислители и итераторы, визуальная разработка

9.1. Универсальные шаблоны

9.2. Класс System.Object

9.3. Операции упаковки, распаковки

9.4. Основы Unit Testing

## 9.5. Обзор WPF

# 1. Инструментарий разработчика

## 1.1 Visual Studio 2017

• Инструменты веб разработчика  
• Средства отладки приложений.  
• Использование NuGet пакетов.

* + Системы контроля версий и работа с ними.

• Основы Git.   
• Обзор приложений для работы с git.  
• GitFlow.

* + Тестирование кода и средства для тестирования.

• Модульное тестирование.   
• Интеграционное тестирование.   
• Нагрузочное тестирование.  
• Библиотека Moq.

# 2. Введение в Asp.NET MVC 5

## 1.1 Коротко о браузере

Браузер — это движок рендеринга. Его работа заключается в том, чтобы загрузить веб-страницу и представить её в понятном для человека виде.

**Что делает браузер?**

Разрешение DNS - Этот процесс помогает браузеру узнать, к какому серверу он должен подключиться, когда пользователь вводит URL. Браузер связывается с DNS-сервером и обнаруживает, что google.com соответствует набору цифр 216.58.207.110 — IP-адресу, к которому может подключиться браузер;

HTTP-обмен - Как только браузер определит, какой сервер будет обслуживать наш запрос, он установит с ним TCP-соединение и начнет HTTP-обмен. Это не что иное, как способ общения браузера с нужным ему сервером, а для сервера — способ отвечать на запросы браузера. HTTP — это просто название самого популярного протокола для общения в сети, и браузеры в основном выбирают HTTP при общении с серверами. HTTP-обмен подразумевает, что клиент (наш браузер) отправляет запрос, а сервер присылает ответ.

Рендеринг - Последним по счёту, но не последним по значению идет процесс рендеринга. В теле ответа сервер включает представление запрашиваемого документа в соответствии с заголовком Content-Type. В нашем случае тип содержимого был установлен на text/html, поэтому мы ожидаем HTML-разметку в ответе — и именно ее мы и находим в теле документа.

Это как раз тот момент, где браузер действительно проявляет свои способности. Он считывает и анализирует HTML-код, загружает дополнительные ресурсы, включенные в разметку (например, там могут быть указаны для подгрузки JavaScript-файлы или CSS-документы) и представляет их пользователю как можно скорее.

Еще раз, конечным результатом должно стать то, что доступно для восприятия среднестатистического Васи;

Сброс и повтор.

## 1.2 Знакомство с работой HTTP протокола

HTTP (HyperText Transfer Protocol) – это протокол седьмого уровня [модели OSI](https://zametkinapolyah.ru/veb-programmirovanie/servernoe-programmirovanie-server-apache/osnovy-interneta-chast-1-sem-urovnej-modeli-osi-princip-raboty-etalonnoj-modeli.html) для передачи данных, в основе которого лежит архитектура взаимодействие [клиент-сервер](https://zametkinapolyah.ru/servera-i-protokoly/o-modeli-vzaimodejstviya-klient-server-prostymi-slovami-arxitektura-klient-server-s-primerami.html). Изначально протокол HTTP разрабатывался для передачи [HTML документов](https://zametkinapolyah.ru/verstka-sajtov/struktura-html-dokumenta-tip-html-dokumenta-zagolovok-html-dokumenta-telo-html-dokumenta-granicy-html-dokumenta.html) между [сервером](https://zametkinapolyah.ru/servera-i-protokoly/chto-takoe-server-servernyj-kompyuter-i-servernoe-prilozhenie.html) и [клиентом](https://zametkinapolyah.ru/servera-i-protokoly/chto-takoe-klient-klientskij-kompyuter-i-klientskoe-prilozhenie.html) при помощи [HTTP сообщений](https://zametkinapolyah.ru/servera-i-protokoly/tema-3-http-soobshhenie-tipy-http-soobshhenij-zagolovki-http-soobshhenij-telo-soobshheniya-i-primery-http-soobshhenij.html). Поскольку в основе протокола лежит взаимодействие клиент-сервер, то предполагается, что есть клиент, который делает [HTTP запросы](https://zametkinapolyah.ru/servera-i-protokoly/tema-4-http-zapros-zagolovki-http-zaprosa-metody-http-zaprosa-stroka-http-zaprosa-resursy-http-zaprosa-primery-zaprosov.html) и есть [HTTP сервер](https://zametkinapolyah.ru/servera-i-protokoly/http-server-ili-veb-server-naznachenie-funkcii-i-rol-servera-v-http.html), который обрабатывает эти запросы и дает клиенту [HTTP ответы](https://zametkinapolyah.ru/servera-i-protokoly/tema-5-http-otvety-servera-stroka-sostoyaniya-http-otveta-kody-sostoyaniya-zagolovki-http-otveta-primery.html). Все ответы сервера содержат [коды состояния](https://zametkinapolyah.ru/servera-i-protokoly/spisok-kodov-sostoyaniya-http-servera-spravochnik-kodov-sostoyaniya-http-servera.html), а все запросы клиента имеют [HTTP методы](https://zametkinapolyah.ru/servera-i-protokoly/tema-7-opredelenie-metodov-http-http-method-definitions-metody-http-zaprosov.html).

## 1.2.1 Заголовок (HTTP заголовки)

HTTP заголовки сопровождают обмен данными по протоколу HTTP. Они могут содержать описание данных и информацию необходимую для взаимодействия между клиентом и сервером. Заголовки и их статусы перечислены в [реестре IANA](http://www.iana.org/assignments/message-headers/perm-headers.html), который постоянно обновляется.

## 1.2.2 Типы запроса (Методы запросов)

HTTP определяет множество методов запроса, которые указывают, какое желаемое действие выполнится для данного ресурса. Несмотря на то, что их названия могут быть существительными, эти методы запроса иногда называются HTTP глаголами. Каждый реализует свою семантику, но каждая группа команд разделяет общие свойства: так, методы могут быть [безопасными](https://developer.mozilla.org/ru/docs/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8C/safe), [идемпотентными](https://developer.mozilla.org/ru/docs/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8C/idempotent) или [кэшируемыми](https://developer.mozilla.org/ru/docs/%D0%A1%D0%BB%D0%BE%D0%B2%D0%B0%D1%80%D1%8C/cacheable).

[GET](https://developer.mozilla.org/ru/docs/Web/HTTP/Methods/GET)- Метод GET запрашивает представление ресурса. Запросы с использованием этого метода могут только извлекать данные;

[HEAD](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/HEAD) - Метод HEAD запрашивает ресурс так же, как и метод GET, но без тела ответа;

[POST](https://developer.mozilla.org/ru/docs/Web/HTTP/Methods/POST) - Метод POST используется для отправки сущностей к определённому ресурсу. Часто вызывает изменение состояния или какие-то побочные эффекты на сервере;

[PUT](https://developer.mozilla.org/ru/docs/Web/HTTP/Methods/PUT) - Метод PUT заменяет все текущие представления ресурса данными запроса;

[DELETE](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/DELETE) - Метод DELETE удаляет указанный ресурс;

[CONNECT](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/CONNECT) - Метод CONNECT устанавливает "туннель" к серверу, определённому по ресурсу;

[OPTIONS](https://developer.mozilla.org/ru/docs/Web/HTTP/Methods/OPTIONS) - OPTIONS используется для описания параметров соединения с ресурсом;

[TRACE](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/TRACE) - Метод TRACE выполняет вызов возвращаемого тестового сообщения с ресурса;

[PATCH](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/PATCH) - Метод PATCH используется для частичного изменения ресурса.

## 1.2.3 Актуальные проблемы работы с HTTP

## 1.2.4 Простейшие инструменты для анализа HTTP запроса

## 1.3. Нововведение в ASP .NET MVC 5

Аутентификация и авторизация;

Создание адаптивного и расширяемого интерфейса с использованием css-фреймворка Bootstrap;

Добавлены фильтры аутентификации, а также появилась функциональность переопределения фильтров;

Добавлены атрибуты маршрутизации;

Использование по умолчанию Entity Framework 6.

## 1.4. Предназначение ASP .NET MVC

Платформа ASP.NET MVC представляет собой фреймворк для создания сайтов и веб-приложений с помощью реализации паттерна MVC.

* + Обзор Asp.Net

• Web Pages Applications.  
• Web Forms Applications.  
• MVC Applications.

* + Общие возможности для Asp.Net приложений (конфигурация, аутентификация, роли и провайдеры, управление состоянием, кэширование).
  + Введение в MVC приложение

• Модель, представление, контроллер. Понимание паттерна MVC.  
• Демонстрация работы MVC приложение и взаимосвязи его частей.  
• Обзор новых возможностей в MVC разработке.

* + Хостинг Web приложений. Знакомство с IIS и IIS Express.
  + Жизненный цикл запроса к серверу.

[**Разработка моделей Asp.NET MVC**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Понятие модели
  + Создание модели

• Проектирование модели  
• Использование аннотаций данных на свойствах модели.  
• Валидация пользовательского ввода с помощью аннотаций данных.  
• Расширение моделей. Понятие сервисов.  
• Демонстрация практической реализации модели и ее использование

[**Разработка контроллеров Asp.NET MVC**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Взаимодействие контроллера с пользовательским запросом.

• Маршрутизация запроса

* + Определение действий контроллера.
  + Виды возвращаемого результата.
  + Использование параметров в действиях.
  + Передача информации в представление

• Модель  
• ViewBag  
• TempData

* + Практическая демонстрация

[**Разработка представлений Asp.NET MVC**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Создание представлений с помощью движка Razor

• Создание нового представления.  
• Знакомство с Razor синтаксисом.  
• Присоединение представлений к моделям и отображение данных.  
• Понятие модели представления  
• Отличия серверного кода представления от HTML

* + Использование HTML Helpers

• Action Helpers  
• Display Helpers  
• Form Helpers  
• Editor Helpers  
• Validation Helpers  
• Проектирование и реализация своего HTML Helper

* + Повторное использование кода в представлениях

• Создание и использование частичных представлений  
• Использование RenderAction()  
• Шаблоны элементов представления  
• Слои и темы

[**Маршрутизация**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Введение в маршрутизацию MVC.
  + Введение в шаблоны URL.
  + Создание простого маршрута
  + Создание сложного маршрута с использованием специальных переменных сегментов.
  + Маршрутизация с помощью атрибутов.
  + Основы зон.
  + Маршрутизация с использованием зон.

[**Фильтры**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Понятие фильтра.
  + Использование фильтра аутентификации.
  + Использование фильтра авторизации.
  + Использование фильтра действий.
  + Использование фильтра результата.
  + Использование фильтра исключений.
  + Создание своего фильтра.

[**Привязка моделей**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Понятие привязки моделей.
  + Использование стандартной привязки.
  + Настройка системы привязки моделей.
  + Создание собственной привязки.

[**Безопасность в разработке приложений MVC**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Способы взлома Web приложений
  + Конфигурирование приложения
  + Защита форм и данных.

• Использование протокола HTTPS.

* + Основы аутентификации и авторизации.
  + Аутентификация с использованием форм.

[**Интеграция JavaScript и MVC**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Рендеринг и исполнение JavaScript кода

• Добавление JavaScript файлов  
• Использование Content Delivery Network для JavaScript библиотек  
• Введение в jQuery  
• Доступ к HTML элементам с помощью jQuery (селекторы)  
• Введение в jQueryUI

* + Использование AJAX и частичного обновления страницы

• Причины использования AJAX  
• Использование AJAX в MVC

# 25 Bootstrap 3

* + Введение в Bootstrap
  + Скрипты и стили
  + Сетка
  + Responsive Design
  + Компоненты
  + Типографика
  + Формы
  + Таблицы
  + Кнопки
  + JavaScript
  + Кастомизация

[**Паттерны проектирования**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Понятие паттерна.
  + Знакомство с основными актуальными паттернами.
  + Анти-паттерны.
  + Принципы SOLID.

[**IoC. Dependency Injection (внедрение зависимостей)**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Паттерн IoC.
  + Основы DI.
  + Знакомство с наиболее используемыми DI контейнерами. NInject, Autofac. Castle Windsor.
  + Внедрение зависимостей в MVC.

[**Производительность и оптимизация веб-приложений**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Общее понимание узких мест приложения. Способы диагностики.
  + Многопоточность в веб
  + TPL
  + PLINQ
  + Асинхронное программирование.
  + Профилировка веб приложений. Нагрузочное тестирование.
  + Методы оптимизации и рефакторинга кода.

# 28 ORM. ENTITY FRAMEWORK 6

* + Понятие ORM.
  + Основные подходы к проектированию БД: CodeFirst, DatabaseFirst, ModelFirst.
  + Создание и конфигурирование моделей БД. Fluent vs DataAnnotation.
  + DbContext.
  + Механизм миграций.
  + Ленивая загрузка.
  + DbSet и DbEntry. Механизм отслеживания изменений.
  + Тонкая настройка.
  + Оптимизация запросов к БД и тонкости использования EF 6.
  + Стратегии хранения данных: TPT, TPH.
  + Паттерн репозиторий и UnitOfWork.
  + Entity Framework Core. Ключевые отличия.

# 29 **Asp.Net Identity Framework**

* + Введение в Asp.Net Identity.
  + Ключевые отличия в новой Identity модели.
  + Identity Context.
  + Пользователи и роли.
  + Использование стандартных средств для конфигурирования пользователей и ролей.
  + Понятие Claim.
  + Аутентификация и авторизация.
  + OAuth.
  + Понятие JWT токена и его использование.
  + Использование сторонних сервисов для аутентификации (Google, Facebook, SSO Claims)

# 30 **WEB** API **2**

* + Введение в архитектуру REST.
  + Стратегия проектирования RESTfull сервиса.
  + ApiController. Ключевые особенности.
  + Средства для тестирования и документирования сервиса. Swagger.

# 31 **OWIN И KATANA**

* + Введение
  + KATANA
  + OWIN
  + WebServer Console
  + AppFunc
  + Middleware
  + Создание API
  + Миграция на IIS, OWIN и MVC 5

# 31 Библиотека SignalR. Push-уведомления

* + Введение
  + Web Sockets
  + SignalR
  + Установка и настройка
  + Создание клиентской части
  + Создание серверной части

[**SINGLE PAGE APPLICATION**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Введение
  + Основные JavaScript Frameworks: Angular, React.
  + SSE
  + Серверная архитектура

[**Основы Asp.NET Core**](https://www.it-academy.by/course/asp-net-developer/nd2-razrabotka-prilozheniy-na-asp-net/#)

* + Введение в Asp.NET Core
  + Основные отличия от классического MVC приложения.
  + JavaScript и TypeScript в Core приложениях.
  + Использование Nuget, npm и bower.
  + Конфигурирование приложения. Прощай web.config.
  + Тестирование.
  + Расширение фреймворка. Инструменты dotnet.
  + Развертывание приложения Core.
  + Понятие контейнера. Docker.
  + Основы MS Azure.