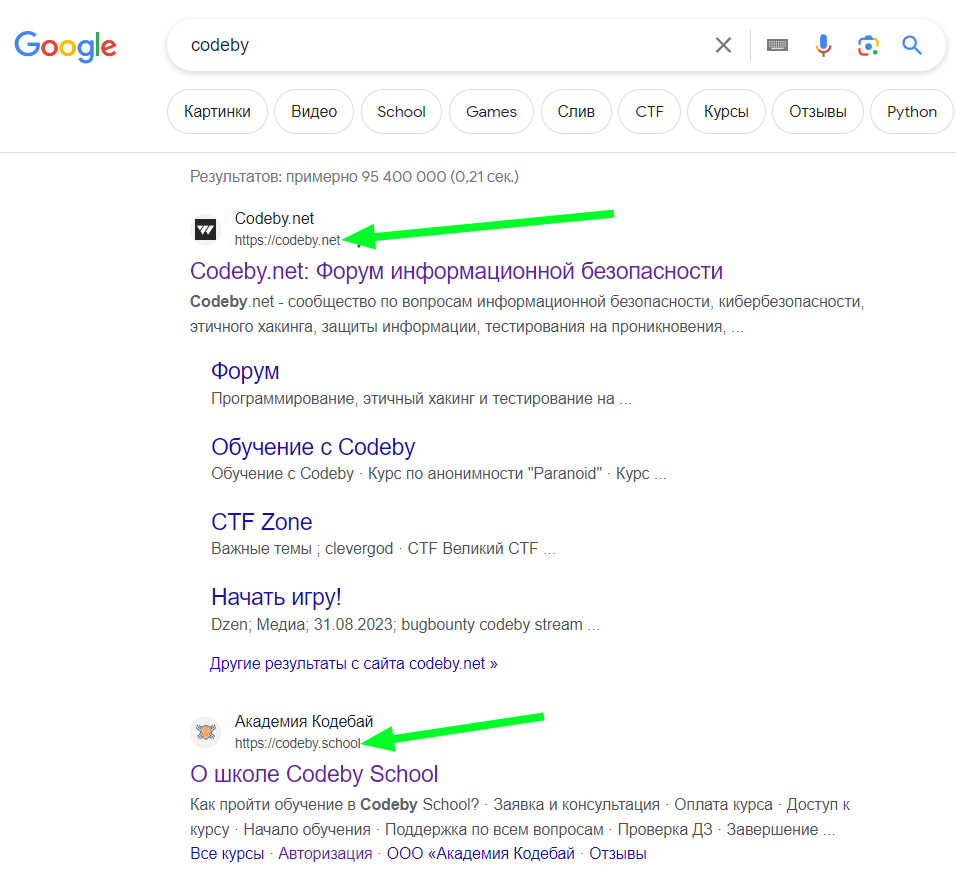
# **Общие понятия**

Как вы уже знаете, Интернет - это глобальная сеть, которая соединяет миллионы компьютеров по всему миру. Он позволяет людям обмениваться информацией, общаться, искать и делиться данными. Интернет работает через провайдеров интернет-услуг, которые обеспечивают подключение к сети. Давайте начнём с основ веб-технологий.

### **Сайт**

Сайт - это набор связанных веб-страниц и другого контента, доступный через интернет по определенному URL-адресу. Он может содержать информацию о компании, продуктах, услугах, новостях, блоге, фотографиях, видео, формы обратной связи и многое другое. Сайты разрабатываются с целью предоставления информации, привлечения посетителей и взаимодействия с ними. Сайты могут быть статичными (с постоянным контентом) или динамическими (с возможностью взаимодействия с пользователем и обновления контента).



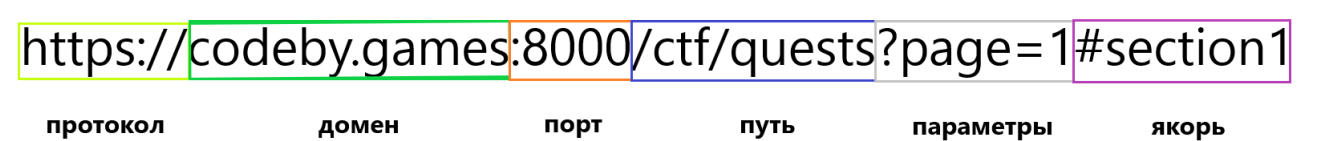
Стрелками показаны адреса веб-сайтов.

### **URL**

Как и в любой локальной сети, в интернете у каждого ресурса есть свой IP-адрес. Введя IP-адрес 64.233.163.113 мы попадём на главную страницу поисковика Google. Подобная адресация удобна для компьютеров, но не для человека, поэтому со временем IP-адресам стали присваивать имена - URL (Uniform Resource Locators). Сопоставление IP-адресов URL происходит средствами системы DNS (Domain Name System). Вводя URL в адресную строку браузере, он отправляет запрос на DNS-сервер, который содержит базу данных доменных имен и их соответствующих IP-адресов. DNS-сервер выполняет поиск в этой базе данных и возвращает браузеру IP-адрес, связанный с введенным URL.

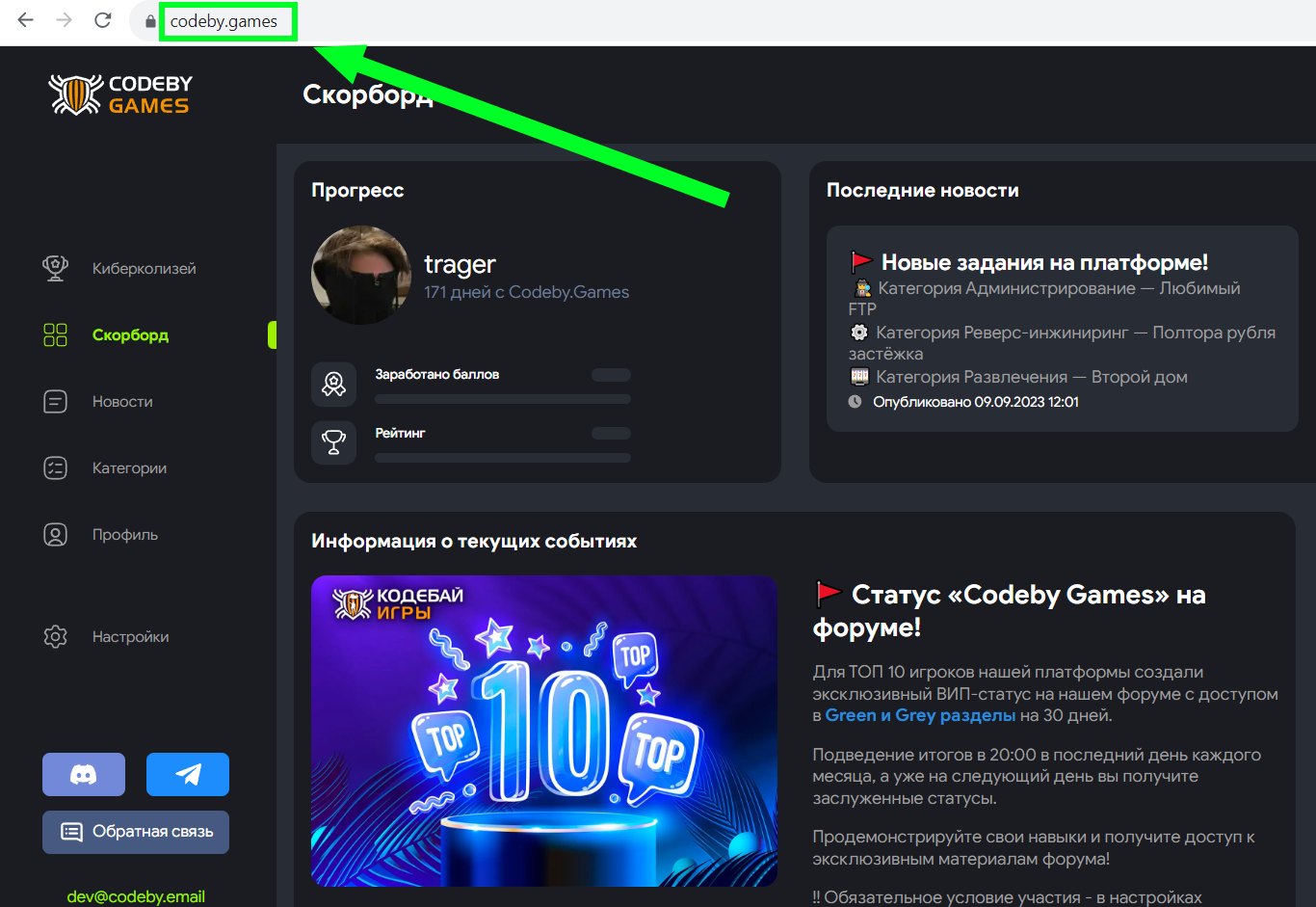
URL обычно состоит из нескольких компонентов:

1. Протокол (Обязателен): обозначает способ взаимодействия с ресурсом. Например, http:// или https:// для обычного веб-сайта или ftp:// для FTP-сервера.
2. Домен или IP-адрес (Обязателен): указывает на конкретный веб-сервер или ресурс, к которому нужно обратиться. Например, example.com.
3. Порт: число, которые указывает на конкретную службу/сервис.
4. Путь: указывает на конкретное местоположение ресурса на сервере. Например, "/categories/quests".
5. Параметры запроса: опциональные данные, передаваемые вместе с запросом для настройки поведения ресурса. Обычно они представлены в виде “ключ-значение” и разделяются символом ? и амперсандом &. Например, ?page=2&sort=asc.
6. Якорь: опциональная ссылка на специфическую часть страницы. Отображается после символа #. Например, #section1.



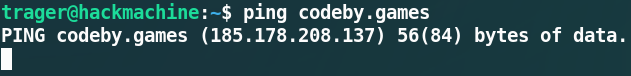
### **Домен**

Домен - это часть интернет-адреса URL, которая идентифицирует определенный веб-сайт. Домен обычно состоит из двух основных компонентов: имени хоста (например, example) и доменного имени верхнего уровня (например, .com). Например, в URL www.codeby.games доменным именем является codeby.games.



Изначально, сервер имеет только IP-адрес. Администратор может купить доменное имя, чтобы людям было проще получить доступ к веб-ресурсу. Например, мы хотим открыть свою CTF-площадку. Покупая сервер, нам выдают его IP-адрес, например, 185.178.208.137 с помощью которого мы можем с ним коммуницировать: подключаться удалённо, пользоваться его сервисами и т.д. Но нашим клиентам будет неудобно постоянно вводить https://185.178.208.137/, чтобы попасть на нашу площадку. В таком случае мы можем купить доменное имя, например, codeby.games. Домен намного проще вводится и быстрее запоминается.

IP-адрес любого домена можно получить всегда с помощью терминальной утилиты ping или whois:



Важно понимать, что не всегда показывается IP-адрес реального сервера. Сервер может обладать защитой, например, CloudFlare, которая подменяет IP-адрес защищаемого ресурса своим.

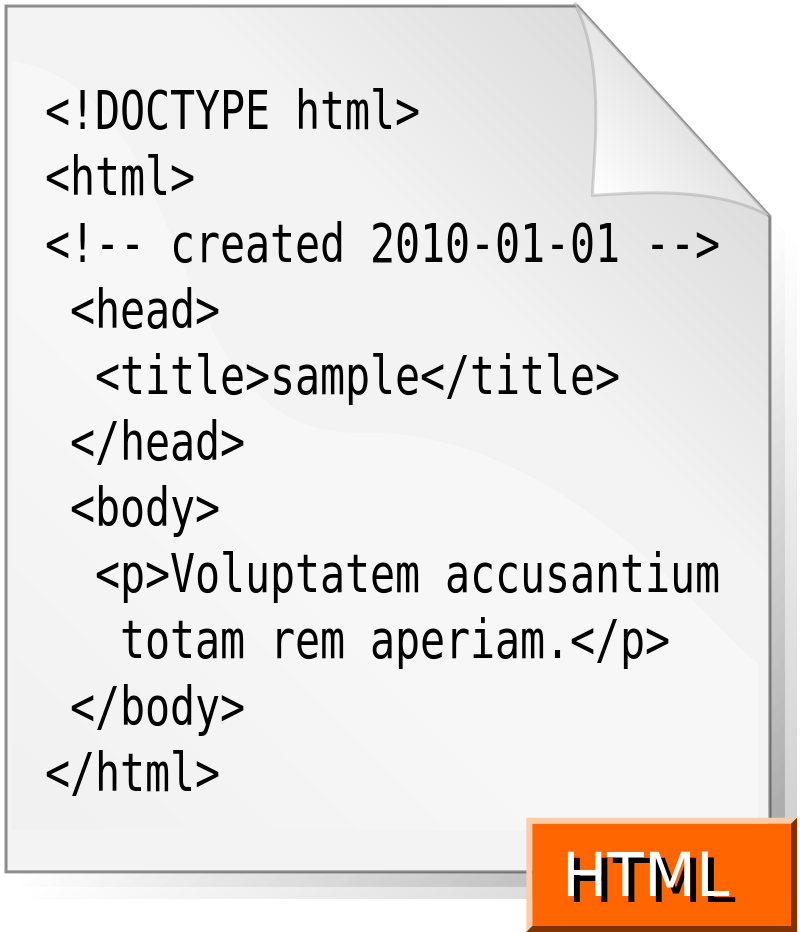
### **IP-адрес**

IP-адрес (Internet Protocol address) - это уникальный идентификатор, присвоенный каждому устройству, подключенному к компьютерной сети, использующую протокол TCP/IP. Наш с вами Интернет использует как раз такую архитектуру. Он состоит из четырех чисел, разделенных точками, например, 192.168.0.1. IP-адрес используется для определения местоположения и связи между устройствами в сети. Он позволяет отправлять и получать данные через интернет. IP-адресы бывают двух типов: IPv4 (четыре числа) и IPv6 (шестнадцать чисел).

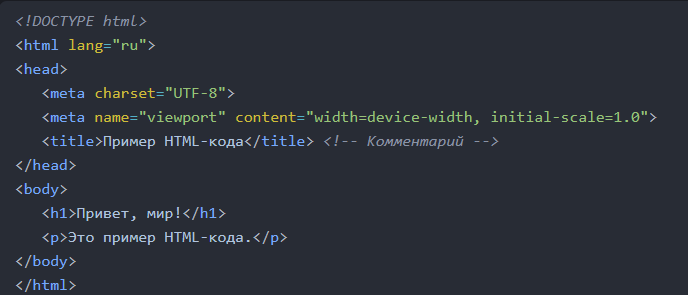
### **Порт**

Порт - это числовой идентификатор, который используется на сервере или на хосте для определения конкретного приложения или службы, с которой устройство хочет установить связь через сеть. Протоколы TCP и UDP поддерживают до 65535 портов, пронумерованных от 0 до 65535. Когда устройство отправляет данные в сеть, оно указывает какой порт использует для отправки, а также к какому порту они должны быть доставлены на удаленном устройстве. Это позволяет точно направить данные определенной программе или службе, работающей на удаленном устройстве. Более углублённо с портами Вы познакомитесь в следующей главе.

### **HTML (HyperText Markup Language)**



HTML (англ. аббр. HyperText Markup Language “язык гипертекстовой разметки”) - стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц. Важно отметить, что данная технология не является языком программирования. Ниже Вы можете посмотреть пример HTML-кода:

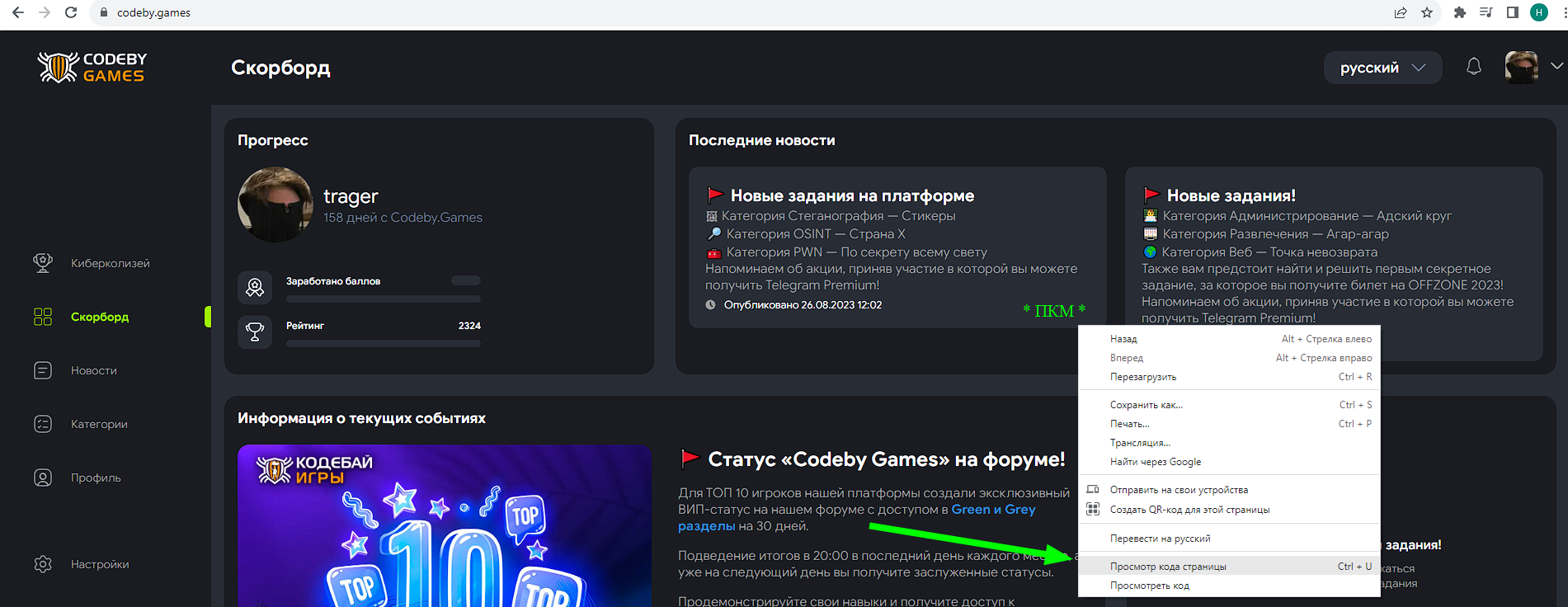


HTML документ состоит из различных тегов, например: <html>, <head>, <body>, <h1>, <p> и других. Каждый тег выполняет свою определённую функцию:

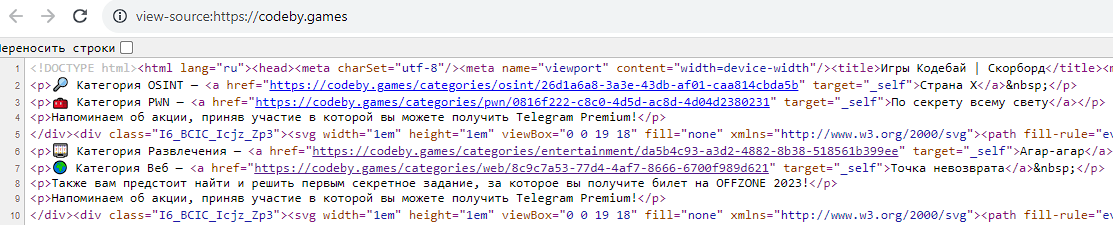
* <!DOCTYPE html> - объявление типа документа для браузера. Браузер понимает, что перед ним HTML-страница.
* <html lang=”ru”> - тег, который заключает в себе всё содержимое веб-страницы. Открывающий и закрывающий теги <html> в документе необязательны, но при этом это является хорошим стилем написания страниц. Атрибут lang используется для определения языка контента на странице.
* Атрибут - это дополнительная информация, которая добавляется к HTML-элементу для определения его свойств и характеристик. Атрибуты используются для указания различных параметров элементов, таких как цвет, размер, шрифт, ссылки, изображения и т.д.
* <head> - предназначен для хранения служебной информации о странице.
* <meta> - используется для определения метаданных документа, таких как описание страницы, ключевые слова, автор и другие данные, которые не отображаются на странице, но могут быть использованы поисковыми системами и другими инструментами для анализа и индексации содержимого страницы.
* <body> - определяет основное содержимое документа, которое отображается на странице. Внутри тега body могут находиться другие элементы HTML, такие как заголовки, параграфы, списки, изображения и другие элементы.
* <h1> - используется для определения заголовка первого уровня на странице. Обычно он используется для указания основного заголовка страницы.
* <p> - используется для определения абзаца на странице.

Важно отметить, что в HTML есть и закрывающиеся теги (например, <html></html>) и не закрывающиеся (например, <br>). В HTML-страницу можно внедрить код языка программирования JavaScript и каскадные таблицы стилей CSS. Исходный код HTML любого сайта всегда виден посетителям. Его можно посмотреть через “Просмотр исходного кода страницы”:

* Откроем [codeby.games](http://codeby.games/) и нажмём правой кнопкой мыши по любой области сайта (либо можно использовать сочетание клавиш CTRL + U), а затем “Просмотр кода страницы”:



* Открылся HTML-код главной страницы [https://codeby.games](https://codeby.games/):

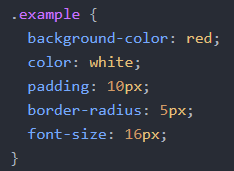


Знание HTML может помочь при анализе веб-страницы на наличие уязвимостей.

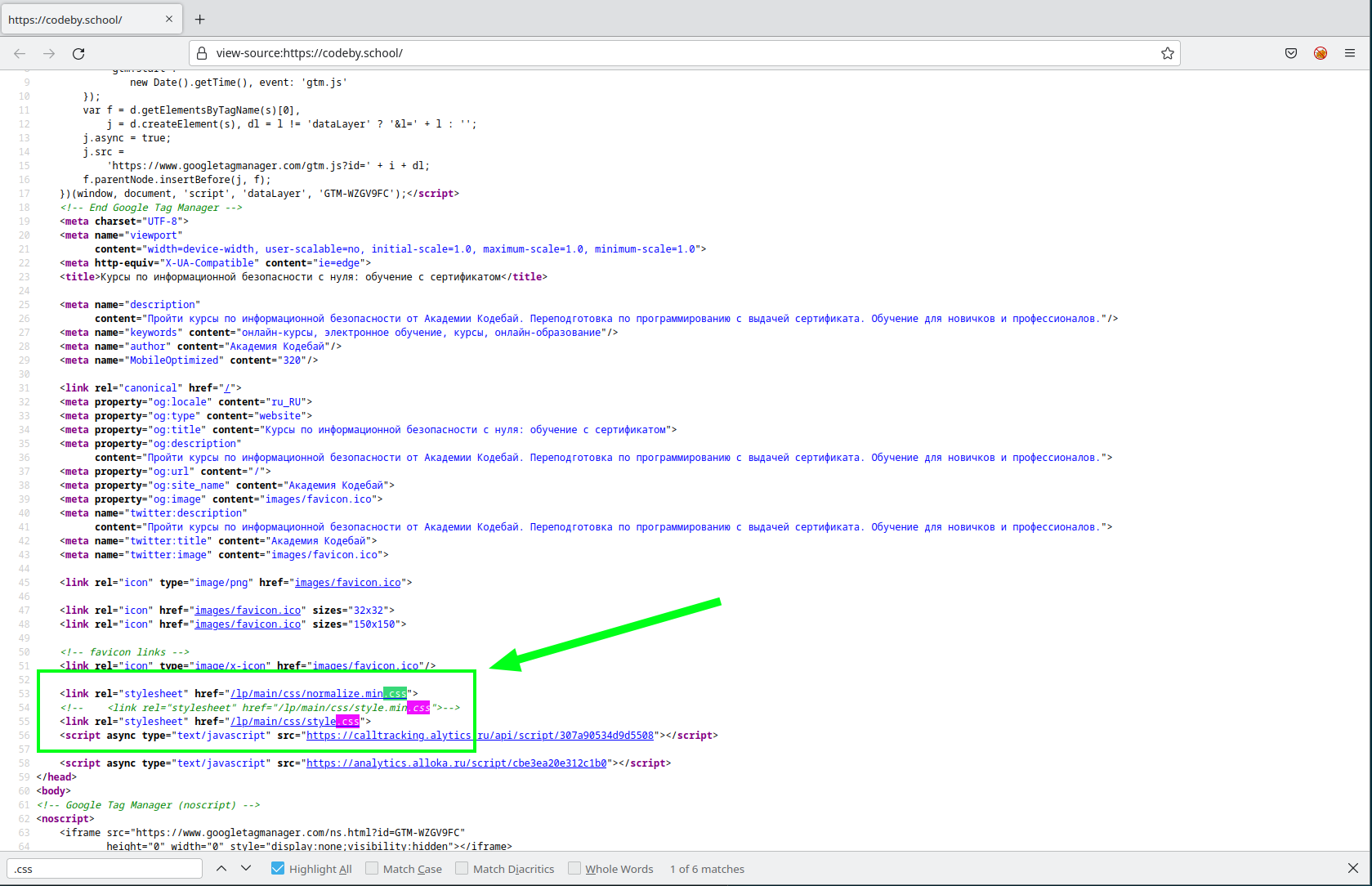
### **CSS (Cascading Style Sheets)**



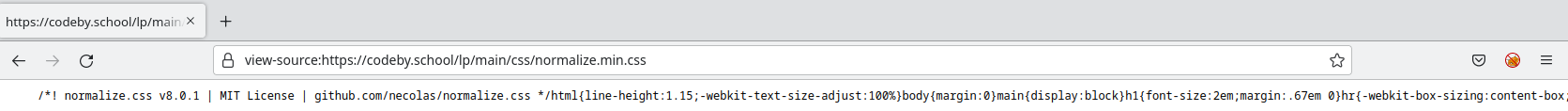
CSS (англ. аббр. Cascading Style Sheets “каскадные таблицы стилей”) - это язык, используемый для описания внешнего вида веб-страницы. Он располагается либо в самом HTML-коде, либо в отдельном файле, который подключается к HTML-странице и имеет расширение .css. Пример CSS кода:



Ровно как и HTML, CSS не является языком программирования и он также доступен всем посетителям веб-страницы. Перейдём на [codeby.school](http://codeby.school/), нажмём сочетание клавиш CTRL + U и CTRL + F, а затем введём .css:



Выделились пути к .css файлам, которые мы можем посмотреть:

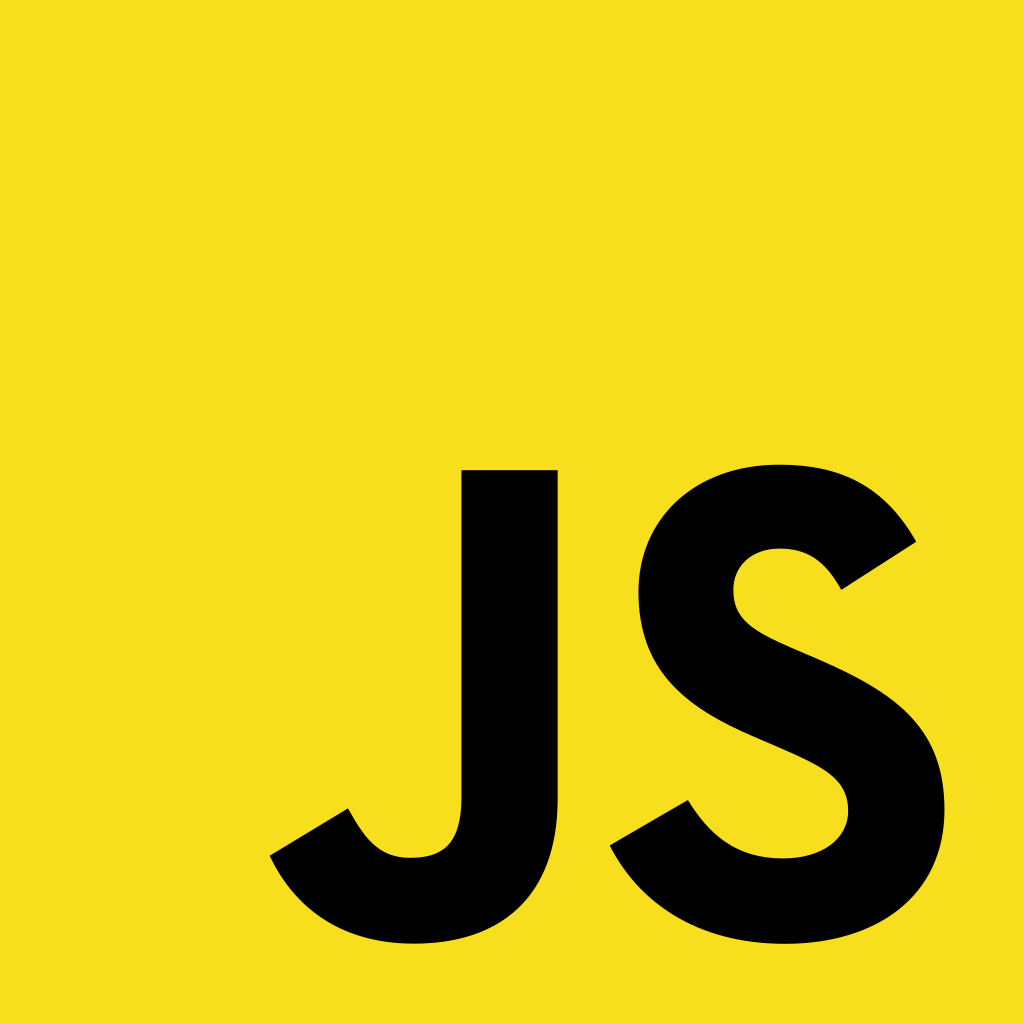


CSS-код можно внедрить в HTML тремя способами:

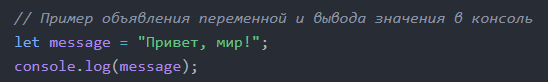
* Через внешние таблицы стилей с помощью тега <link>, подключив файл, содержащий код CSS.
* Через встроенные таблицы стилей с помощью тега <style>.
* Через внутренние таблицы стилей с помощью атрибута style.

Знание CSS полезно при веб-эксплуатации, так как это позволяет понять, как веб-страницы оформлены и как они могут быть изменены или скомпрометированы.

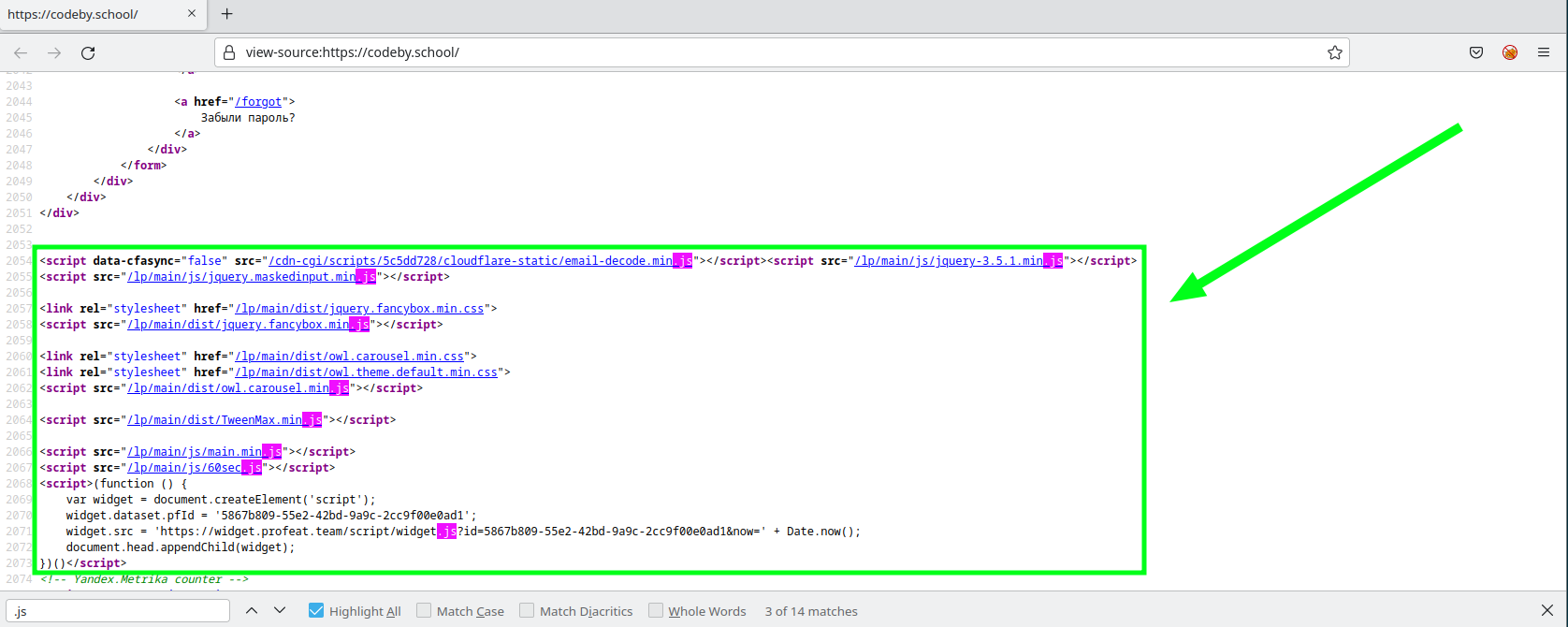
### **JavaScript**



JavaScript, в отличие от упомянутых уже выше HTML и CSS, является языком программирования, который используется для создания интерактивных веб-страниц и приложений. Ниже Вы можете увидеть пример JS-кода:



JavaScript-код можно внедрить в HTML-страницу напрямую с помощью тега <script>, либо подключив JS-файл с помощью атрибута src:



JavaScript-код виден всем посетителям веб-сайта. Знание JavaScript позволяет более полноценно оценивать безопасность веб-приложений и проводить анализ клиентской стороны, где могут находиться уязвимости.

# **Протокол HTTP**



HTTP (англ. HyperText Transfer Protocol - “протокол передачи гипертекста”) - протокол прикладного уровня передачи данных, который используется для обмена данных во Всемирной Паутине. HTTPS (HyperText Transfer Protocol Secure) - это защищенная версия HTTP, которая использует шифрование данных с помощью SSL (Secure Sockets Layer) или его более современной версии TLS (Transport Layer Security). Это обеспечивает безопасность передачи данных между веб-сервером и веб-браузером, что делает HTTPS более безопасным, чем HTTP.

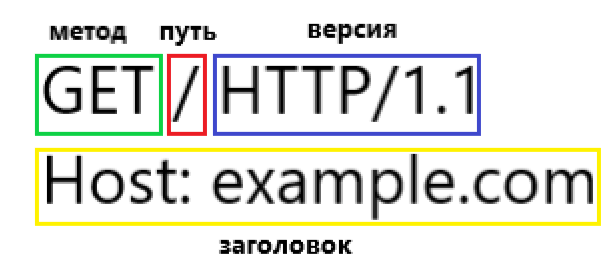
Протокол HTTP применяется во множестве приложениях и веб-сервисах. HTTP широко используется в следующих областях:

* Веб-сайты: Когда Вы вводите URL-адрес в браузер, браузер отправляет HTTP запросы на сервер, чтобы получить веб-страницы, изображения, стили, скрипты и другие ресурсы.
* Веб-сервисы API: HTTP используется для обмена данными между клиентскими приложениями и серверами с использованием RESTful API или других стандартов веб-сервисов, таких как SOAP или GraphQL.
* Аутентификация и авторизация: HTTP используется для передачи учетных данных и токенов авторизации, таких как токены доступа OAuth или JWT, для обеспечения безопасности и контроля доступа.
* Формы и отправка данных: Когда вы заполняете веб-форму и нажимаете на кнопку отправки, данные отправляются на сервер с помощью POST запросов HTTP.
* Интеграция приложений: HTTP используется для обмена данными между различными приложениями и системами, и они могут взаимодействовать, отправляя и получая HTTP запросы и ответы.

Это только несколько примеров применения HTTP. Протокол HTTP является основой передачи данных веб-приложений и играет ключевую роль в связи между клиентами и серверами в Интернете.

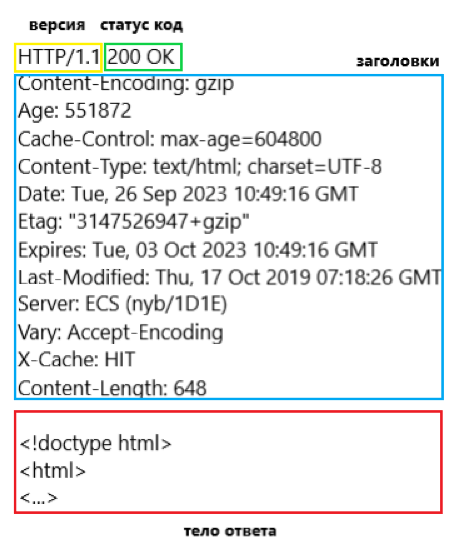
### **Структура HTTP-запроса и HTTP-ответа**

Рассмотрим простейший HTTP-запрос с методом GET:



В самом начале указывается метод HTTP-запроса, затем путь к ресурсу на сервере, используемая версия HTTP и заголовок Host, где указан домен example.com. Количество заголовков может быть неограниченным.

Рассмотрим ответ веб-сервера к этому GET-запросу:



В ответе возвращается используемая версия HTTP, статус код, HTTP-заголовки и тело ответа.

### **Методы HTTP**

Каждый HTTP-запрос использует метод, который предназначен для определённой цели. Ниже перечисленны все методы, которые могут использоваться в HTTP-запросе.

|  |  |
| --- | --- |
| GET | Запрашивает представление ресурса. Запросы с использованием этого метода могут только извлекать данные. |
| HEAD | Запрашивает ресурс так же, как и метод GET, но без тела ответа. |
| POST | Используется для отправки сущностей к определённому ресурсу. Часто вызывает изменение состояния или какие-то побочные эффекты на сервере. |
| PUT | Заменяет все текущие представления ресурса данными запроса. |
| DELETE | Удаляет указанный ресурс. |
| CONNECT | Устанавливает “туннель” к серверу, определённому по ресурсу. |
| OPTIONS | Используется для описания параметров соединения с ресурсом. |
| TRACE | Выполняет вызов возвращаемого тестового сообщения с ресурса. |
| PATCH | Используется для частичного изменения ресурса. |

### **Заголовки HTTP**

Заголовки HTTP позволяют клиенту и серверу передавать дополнительную информацию с помощью HTTP-запроса или ответа. Заголовок HTTP состоит из имени без учета регистра, за которым следует двоеточие :, а затем его значение. Пробелы перед значением игнорируются. Пример заголовка - Cookie: user=admin.

Все заголовки разделяются на четыре основных группы:

* General Headers (рус. Основные заголовки) - должны включаться в любое сообщение клиента и сервера.
* Request Headers (рус. Заголовки запроса) - используются только в запросах клиента.
* Response Headers (рус. Заголовки ответа) - только для ответов от сервера.
* Entity Headers (рус. Заголовки сущности) - сопровождают каждую сущность сообщения.
* **GET**

Ниже приведены некоторые виды заголовков для GET-запросов:

|  |  |
| --- | --- |
| **Название заголовка** | **Функция** |
| Host | Указывает на имя целевого сервера или хоста, на который должен быть отправлен запрос |
| User-Agent | Содержит информацию о программе или браузере, используемом для отправки запроса |
| Accept | Определяет типы контента, которые клиент ожидает получить в ответ от сервера. |
| Accept-Language | Определяет предпочтительный язык ответа. |
| Referer | Указывает на URL-адрес страницы, с которой был осуществлен переход на текущую страницу. |
| Cache-Control | Задает инструкции по кешированию страницы. |

* **POST**

Ниже приведены некоторые виды заголовков для POST-запросов:

|  |  |
| --- | --- |
| **Название заголовка** | **Функция** |
| Host | Указывает на имя целевого сервера или хоста, на который должен быть отправлен запрос |
| User-Agent | Содержит информацию о программе или браузере, используемом для отправки запроса |
| Accept | Определяет типы контента, которые клиент ожидает получить в ответ от сервера. |
| Content-Type | Определяет тип данных, отправляемых в теле запроса. |
| Content-Length | Указывает размер тела запроса в байтах. |
| Referer | Указывает на URL-адрес страницы, с которой был осуществлен переход на текущую страницу. |

### **Статус коды**

Веб-серверы используют статус-коды HTTP для информирования клиентов о результате выполнения запроса. Ниже приведен список таких кодов:

|  |  |
| --- | --- |
| 1xx | Информационные |
| 2xx | Успешные |
| 3xx | Перенаправления |
| 4xx | Клиентские ошибки |
| 5xx | Серверные ошибки |