

Базовые методы обработки данных с использованием Python

Лекция 6. Веб-скрапинг. Парсинг. Основы HTML, CSS. Библиотеки requests, bs4.

Киреев Василий Сергеевич

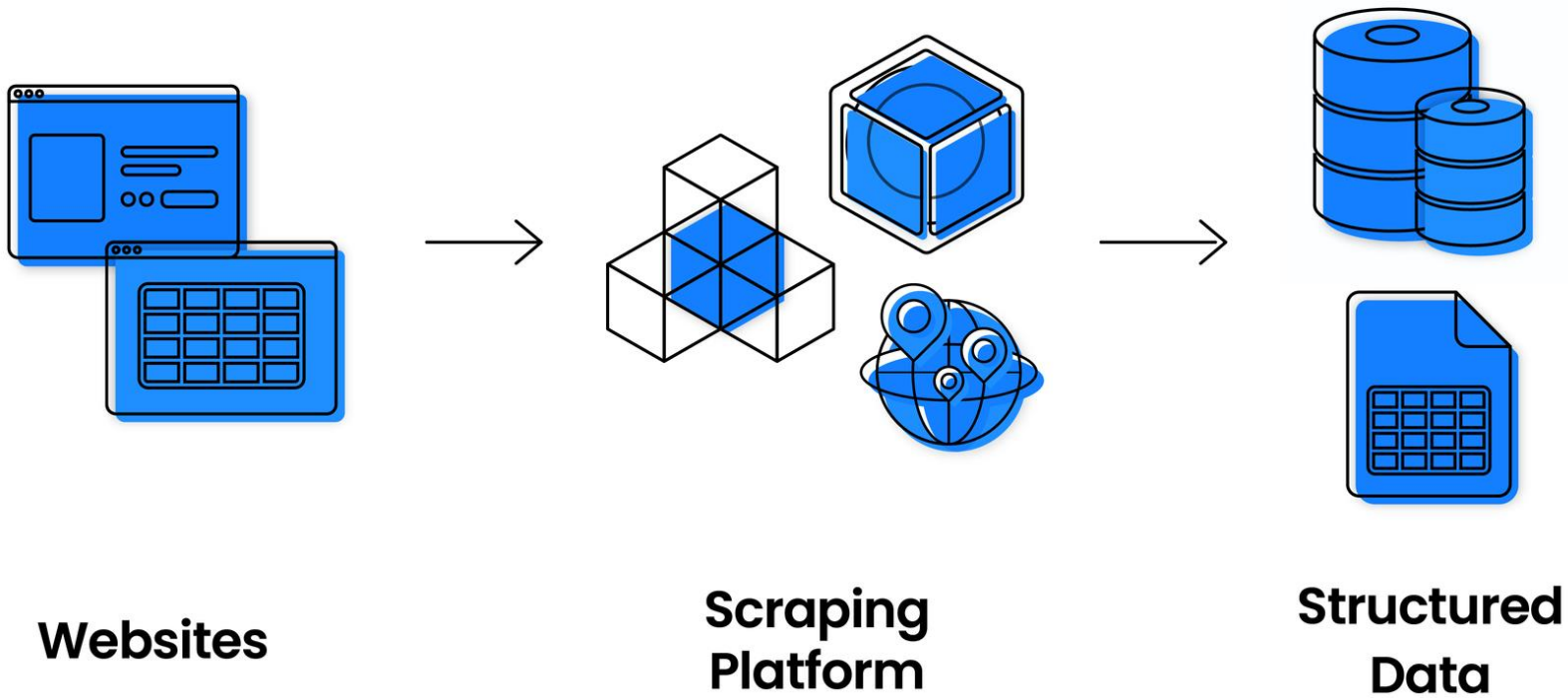
к.т.н., доцент

Москва, 2024

Веб-скрапинг vs парсинг

Веб-скрапинг относится к сбору данных с веб-сайта. Скрапинг данных - это сбор данных, а парсинг данных - их анализ. Результатом скрапинга данных обычно являются необработанные строки HTML. Результатом же парсинга обычно являются структурированные данные в более удобном для чтения формате, например JSON или CSV.

Веб-скрапинг. Процесс



Веб-скрапинг. Применение

Мониторинг брендов и анализ конкурентов

Анализ финансовых данных

Анализ социальных сетей

SEO-мониторинг

Машинное обучение

Скраппинг объявлений о продаже недвижимости

Отраслевая статистика и аналитика

Генерация лидов

Веб-скрапинг. Проблемы

Крупномасштабный веб-скрапинг сопряжен с многочисленными проблемами, обусловленными сложностью обработки больших объемов данных и техническими компонентами:

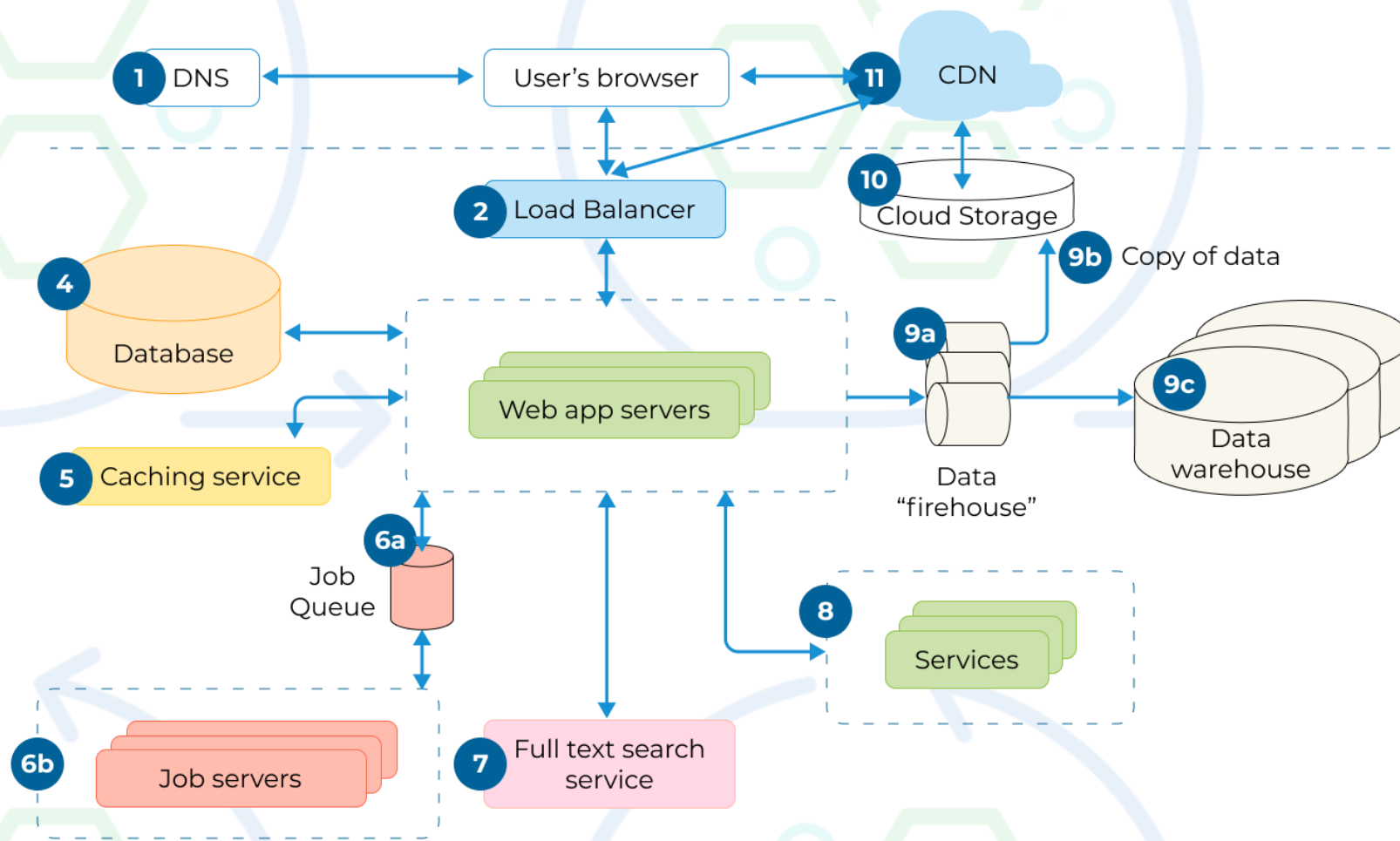
Динамические веб-сайты: Динамические веб-сайты, в отличие от статических, используют JavaScript для загрузки или отображения содержимого, что затрудняет сбор данных традиционными методами. Большинство динамических сайтов требуют взаимодействия с пользователем, например, нажатия кнопок или заполнения форм. Чтобы получить доступ к данным, скрепер должен уметь имитировать эти взаимодействия.

Ограничение скорости: На веб-сайтах используется ограничение скорости, позволяющее контролировать количество запросов, которые клиент может сделать за определенный период. Это позволяет защитить API от вредоносных ботов и предотвратить неправомерное использование данных.

Точность данных: Обеспечить точность данных бывает непросто, особенно при работе с большими массивами данных. Например, большие наборы данных, собранные из нескольких источников, могут привести к несоответствию данных. Проверка новых данных вручную, особенно в больших массивах данных, может быть непрактичной и утомительной. Для проверки и инспекции данных можно использовать автоматизированные метрики, например, алгоритмы машинного обучения или разработку сценариев.

Меры по борьбе с скрапингом: Многие веб-сайты используют механизмы защиты от скрапинга, такие как CAPTCHA, JavaScript-задачи и блокировка IP-адресов, для предотвращения или ограничения деятельности по скрапингу.

Веб-скрапинг. Типовая веб-архитектура



Веб-скрапинг. Минимальная теория. HTTP

URL - это унифицированный указатель ресурса. Он служит в качестве веб-адреса различных веб-страниц. Каждый URL в Интернете работает по принципу "запрос - ответ". Браузер запрашивает у сервера веб-страницу, и ответом сервера является содержимое веб-страницы. Затем это содержимое отображается в браузере.

URL Request - запрос к веб-серверу содержимого для просмотра пользователем. Этот запрос выполняется при щелчке на ссылке или открытии веб-страницы.

URL Response - ответ на запрос независимо от успеха или неудачи. На каждый запрос к веб-серверу веб-сервер предоставляет обязательный ответ, который чаще всего представляет собой соответствующее содержимое, запрошенное в запросе URL.

Веб-скрапинг. Минимальная теория. URL

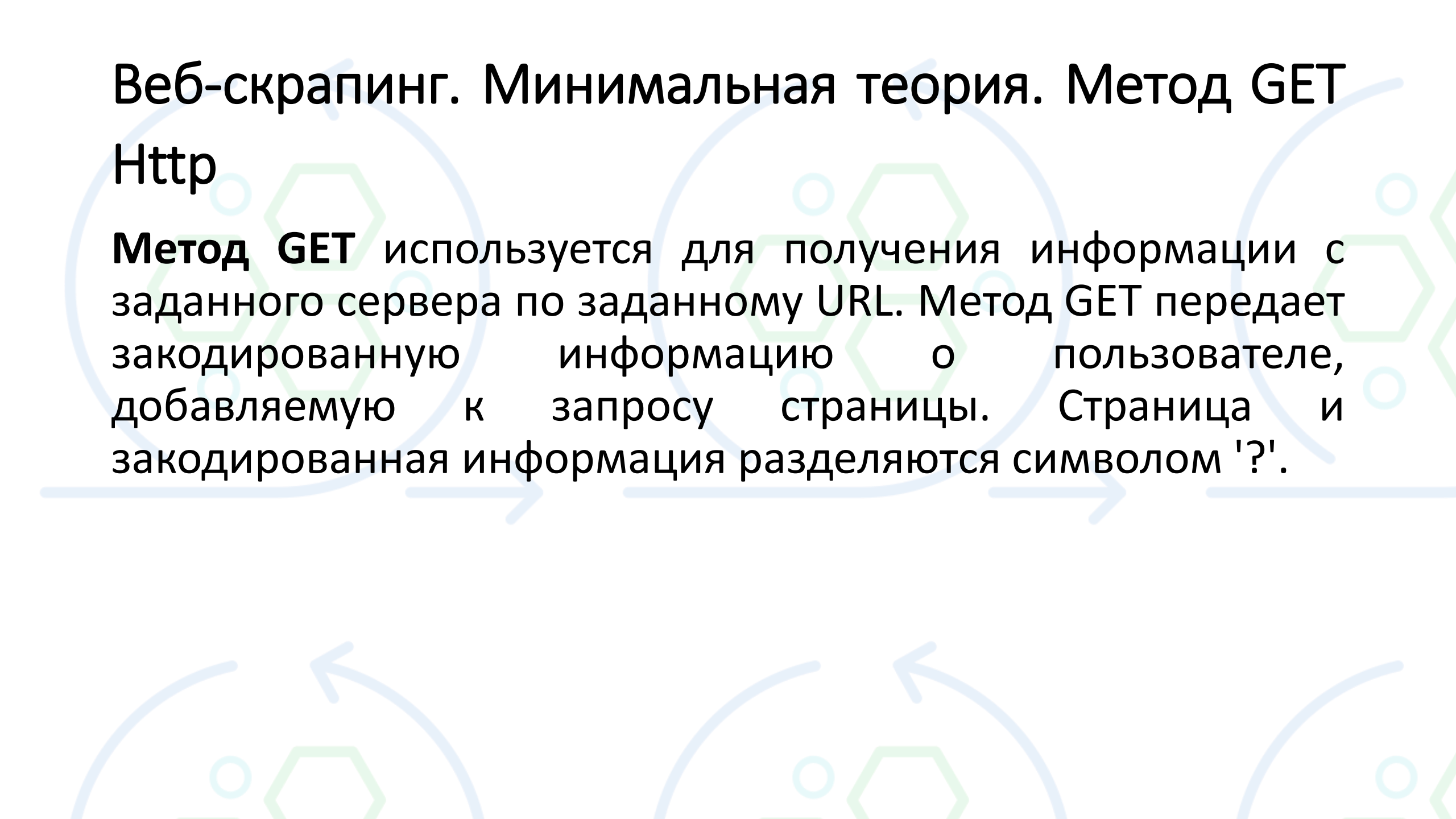
URL - это унифицированный указатель ресурса. Он служит в качестве веб-адреса различных веб-страниц. Каждый URL в Интернете работает по принципу "запрос - ответ". Браузер запрашивает у сервера веб-страницу, и ответом сервера является содержимое веб-страницы. Затем это содержимое отображается в браузере.

URL Request - запрос к веб-серверу содержимого для просмотра пользователем. Этот запрос выполняется при щелчке на ссылке или открытии веб-страницы.

URL Response - ответ на запрос независимо от успеха или неудачи. На каждый запрос к веб-серверу веб-сервер предоставляет обязательный ответ, который чаще всего представляет собой соответствующее содержимое, запрошенное в запросе URL.

Веб-скрапинг. Минимальная теория. Метод GET

Http

Метод GET используется для получения информации с заданного сервера по заданному URL. Метод GET передает закодированную информацию о пользователе, добавляемую к запросу страницы. Страница и закодированная информация разделяются символом '?'.


Парсинг. Метод GET. Коды ответов

Информационные	100- 199
Успешные	200- 299
Перенаправления	300- 399
Клиентские ошибки	400- 499
Серверные ошибки	500-599

Веб-скрапинг. Минимальная теория. Метод POST Http

POST - это метод запроса, поддерживаемый протоколом HTTP и используемый во Всемирной паутине. По своей сути метод запроса POST запрашивает у веб-сервера данные, заключенные в теле сообщения запроса, скорее всего, для их сохранения. Он часто используется при загрузке файла или при отправке заполненной веб-формы.

HTML, CSS и JavaScript

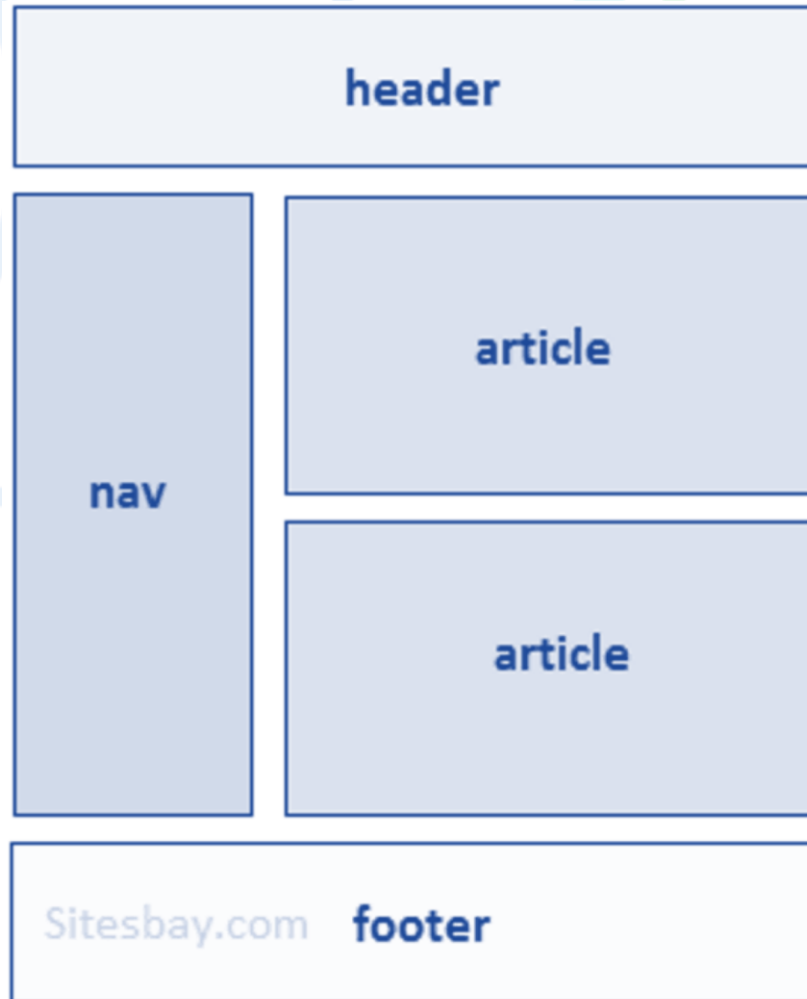
HTML, CSS и JavaScript - три основных языка Интернета. Веб-сайты структурируются с помощью HTML, оформляются с помощью CSS, а интерактивные функции добавляются с помощью JavaScript. Большинство анимаций или действий, которые происходят в результате нажатия, наведения или прокрутки пользователем, создаются с помощью JavaScript.

HyperText Markup Language (HTML)

HTML расшифровывается как HyperText Markup Language. Он используется для разработки веб-страниц с помощью языка разметки. HTML - это комбинация слов Hypertext и Markup language. Гипертекст определяет связь между веб-страницами. Язык разметки используется для определения текстового документа внутри тега, который задает структуру веб-страниц. Этот язык используется для аннотирования (внесения пометок для компьютера) текста, чтобы машина могла понимать его и соответствующим образом манипулировать текстом.

В языке используются теги, определяющие, какие манипуляции необходимо произвести с текстом. TML - это язык разметки, используемый браузером для манипулирования текстом, изображениями и другим содержимым с целью отображения его в требуемом формате. HTML был создан Тимом Бернерсом-Ли в 1991 году. Первой версией HTML был HTML 1.0, но первой стандартной версией стал HTML 2.0, опубликованный в 1995 году.

Структура HTML-5 документа



Теги в HTML

HTML-теги - это ключевые слова, которые используются для создания веб-страниц в различных форматах. В большинстве тегов встречаются открывающие и закрывающие теги. Завершающие теги содержат прямую косую черту (/), а язык начальных тегов остается прежним. Некоторые теги не обязательно должны быть закрытыми.

Теги в HTML

Теги	Описание
<code><!--...--></code>	Используется для добавления комментариев.
<code><!DOCTYPE></code>	Объявляет тип документа и предоставляет основную информацию для браузера — его язык и версия.
<code><a></code>	Создаёт гипертекстовые ссылки.
<code><abbr></code>	Определяет текст как аббревиатуру или акроним. Поясняющий текст задаётся с помощью атрибута title.
<code><address></code>	Задаёт контактные данные автора/владельца документа или статьи. Отображается в браузере курсивом.
<code><area></code>	Представляет собой гиперссылку с текстом, соответствующей определенной области на карте-изображении или активную область внутри карты-изображения. Всегда вложен внутри элемента <code><map></code> .
<code><article></code>	Раздел контента, который образует независимую часть документа или сайта, например, статья в журнале, запись в блоге, комментарий.

Теги в HTML

Теги	Описание
<aside>	Представляет контент страницы, который имеет косвенное отношение к основному контенту страницы/сайта.
<audio>	Загружает звуковой контент на веб-страницу.
	Задаёт полужирное начертание отрывка текста, не придавая акцент или важность выделенному.
<base>	Задаёт базовый адрес (URL), относительно которого вычисляются все относительные адреса. Это поможет избежать проблем при переносе страницы в другое место, так как все ссылки будут работать, как и прежде.
<bdi>	Изолирует отрывок текста, написанный на языке, в котором чтение текста происходит справа налево, от остального текста.
<bdo>	Отображает текст в направлении, указанном в атрибуте dir, переопределяя текущее направление написания текста.
<blockquote>	Выделяет текст как цитату, применяется для описания больших цитат.

Теги в HTML

Теги	Описание
<body>	Представляет тело документа (содержимое, не относящееся к метаданным документа).

	Перенос текста на новую строку.
<button>	Создает интерактивную кнопку. Элемент может содержать текст или изображение.
<canvas>	Холст-контейнер для динамического отображения изображений, таких как простые изображения, диаграммы, графики и т.п. Для рисования используется скриптовый язык JavaScript.
<caption>	Добавляет подпись к таблице. Вставляется сразу после открывающего тега <table>.
<cite>	Используется для указания источника цитирования. Отображается курсивом.
<code>	Представляет фрагмент программного кода, отображается шрифтом семейства monospace.

Атрибуты элементов в HTML

Все HTML-элементы имеют атрибуты, которые предоставляют дополнительную информацию о конкретном элементе. Он принимает 2 параметра, т.е. имя и значение, которые определяют свойства элемента и размещаются внутри тега element:

- Атрибуты всегда идут в паре имя/значение, например, так: имя_атрибута="значение".
- Атрибуты всегда добавляются в начальный тег HTML-элемента.
- Значения атрибутов всегда должны быть заключены в кавычки. Чаще всего используются двойные кавычки (" "), но допускаются и одинарные кавычки (' ').
- В некоторых редких ситуациях, например, когда значение атрибута само содержит кавычки, необходимо использовать одинарные кавычки: name='John "ShotGun" Nelson' и наоборот.

Глобальные атрибуты в HTML

Глобальные атрибуты	Описание
accesskey	Это комбинация клавиш для активизации/фокусировки определенных элементов.
autocapitalize	Используется для автоматического выделения заглавными буквами текста, вводимого/редактируемого пользователем.
autofocus	Атрибут autofocus в HTML используется для указания того, что элемент должен получать фокус при загрузке страницы
class	Указывает одно или несколько имен классов для HTML-элемента.
contenteditable	Используется для указания того, является ли содержимое элемента редактируемым или нет. Когда этот атрибут не задан, элемент наследуется от своего родительского элемента.
contextmenu	Это идентификатор <menu>, обеспечивающего контекстное меню для данного элемента.
data-*	Может использоваться для определения собственных атрибутов данных.

Глобальные атрибуты в HTML

Глобальные атрибуты	Описание
dir	Используется для указания направления текста содержимого элемента.
draggable	Используется для указания того, является ли элемент перетаскиваемым или нет. По умолчанию перетаскиваемыми являются ссылки и изображения.
enterkeyhint	Дает подсказку о том, какой ярлык или пиктограмму следует представить на виртуальной клавиатуре при нажатии клавиш.
hidden	Используется для определения видимости элементов. Он содержит булево значение. Если этот атрибут используется, то браузеры не будут отображать элементы, у которых указан атрибут hidden.
id	Это уникальный идентификатор, который используется для указания документа и применяется CSS и JavaScript для выполнения определенной задачи для уникального элемента.
inputmode	Используется в основном для подсказки браузерам, какую конфигурацию виртуальной клавиатуры следует использовать при редактировании данного элемента или его содержимого.

Cascading Style Sheets (CSS)

Каскадные таблицы стилей (Cascading Style Sheets, CSS), - это язык, предназначенный для упрощения процесса придания веб-страницам презентабельного вида. CSS позволяет применять стили к веб-страницам. Что еще более важно, CSS позволяет делать это независимо от HTML, из которого состоит каждая веб-страница. Он описывает, как должна выглядеть веб-страница: задает цвета, шрифты, интервалы и многое другое.

CSS позволяет разработчикам и дизайнерам определять поведение, в том числе расположение элементов в браузере. Если в HTML используются теги, то в CSS - наборы правил. CSS обеспечивает контроль над представлением HTML-документа.

Синтаксис CSS

CSS состоит из правил стилей, которые интерпретируются браузером и затем применяются к соответствующим элементам документа. Набор стилевых правил состоит из селектора и блока объявления:

- Селектор в CSS используется для указания и выбора конкретных элементов HTML для применения к ним стилей. -- h1
- Объявление: Объявление в CSS представляет собой комбинацию свойства и соответствующего ему значения. -- {color:blue;font size:12px;}

Селектор указывает на HTML-элемент, к которому необходимо применить стиль. Блок объявления содержит одну или несколько деклараций, разделенных точками с запятой. Каждое объявление включает в себя имя CSS-свойства и значение, разделенные двоеточием.

Селекторы в CSS

селекторы CSS	Описание
[attribute*=value]	Выбирает те элементы, значение атрибута которых содержит указанную подстроку str.
[attribute=value]	Выбирает те элементы, значение атрибута которых равно "value".
[attribute\$=value]	Выбрать элементы, значение атрибута которых заканчивается указанным значением "value".
[attribute =value]	Выберите те элементы, значение атрибута которых равно "value". Те элементы, значение атрибута которых начинается со слова "value", за которым сразу следует дефис (-).
[attribute~=value].	Выбрать те элементы, значение атрибута которых содержит указанное слово.
[attribute^=value]	Выбираются те элементы, значение атрибута которых начинается с заданного атрибута.
#id	Устанавливает стиль заданного id. Атрибут id является уникальным идентификатором в HTML-документе.
active	Используется при стилизации активной ссылки на веб-странице. Стиль отображается, когда пользователь щелкает на ссылке.
after	Используется для многократного добавления одного и того же содержимого после содержимого других элементов.

Селекторы в CSS

селекторы CSS	Описание
before	Используется для многократного добавления одного и того же содержимого перед содержимым других элементов
checked	Выделяет все отмеченные элементы в теге input и радиокнопках.
Class	Выбирает все элементы, принадлежащие определенному атрибуту class
default	Устанавливает элемент по умолчанию в группе однотипных элементов в форме.
disabled	Это свойство используется в основном для элементов формы.
element	Выбор HTML-элементов, которые должны быть стилизованы
element element	Выбор элементов внутри элементов
element, element	Используется для стилизации всех элементов, разделенных запятыми, одним и тем же стилем.
element1~element2	Используется для соответствия вхождений элемента2, за которым следует элемент1

CSS. Селектор :root

Селектор `:root` используется для выбора всех элементов HTML-документа. Этот селектор охватывает все элементы или теги HTML.

```
<!DOCTYPE html>
<html>
<head>
  <title>root selector</title>
  <style>
    h1 {
      color: White;
    }
    :root {
      background: green;
    }
    body {
      text-align: center;
    }
  </style>
</head>
<body>
  <h1>Пример</h1>
  <h2>:root селектор</h2>
  <p>
    Корень документа - это тэг body
  </p>
</body>
</html>
```

Пример

:root селектор

Корень документа - это тэг body

jQuery

jQuery - это библиотека JavaScript с открытым исходным кодом, которая упрощает взаимодействие между HTML/CSS-документом, а точнее, объектной моделью документа (DOM), и JavaScript. Если говорить более подробно, то jQuery упрощает обход и манипулирование HTML-документом, обработку событий в браузере, анимацию DOM, взаимодействие с Ajax и кроссбраузерную разработку на JavaScript.

Методы событий в jQuery

Название метода	Описание
click()	Метод click() содержит функцию для обработки событий, вызываемую при щелчке пользователя на конкретном HTML-элементе.
dblclick()	Метод dblclick() содержит функцию обработки событий, вызываемую при двойном щелчке пользователя на конкретном HTML-элементе.
mouseenter()	Метод mouseenter() содержит функцию обработки событий, вызываемую при входе указателя мыши в конкретный HTML-элемент.
mouseleave()	Метод mouseleave() содержит функцию обработки событий, вызываемую при удалении указателя мыши от выбранного ранее HTML-элемента.
mousedown()	Метод mousedown() содержит функцию для обработки событий, которые вызываются при нажатии левой, правой или средней кнопки мыши, когда указатель мыши находится над HTML-элементом.
mouseup()	Метод mouseup() содержит функцию для обработки событий, которые вызываются при отпускании левой, правой или средней кнопки мыши, когда указатель мыши находится над элементом HTML.
hover()	Метод hover() содержит функцию для обработки событий, которые вызываются при входе и выходе указателя мыши из HTML-элемента. Он представляет собой комбинацию методов mouseenter() и mouseleave().

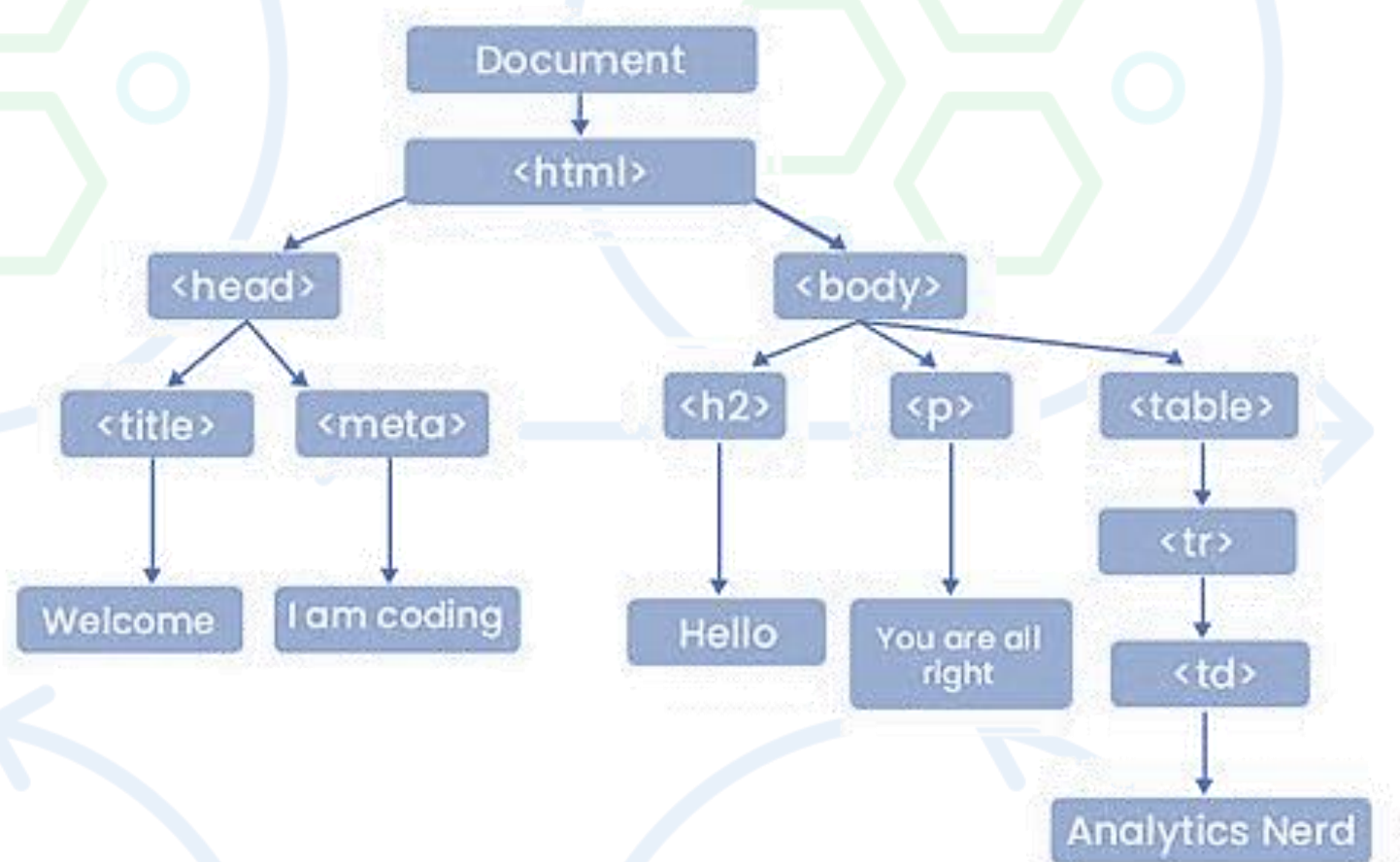
Document Object Model (DOM)

DOM расшифровывается как Document Object Model и представляет собой структурное представление HTML-документа. Реальный DOM - это реальная структура, представленная в пользовательском интерфейсе, а виртуальный DOM - это представление того же самого в памяти. Это древовидная структура, состоящая из всех узлов HTML-документа. DOM представляет UI ваших приложений.

Реальный Document Object Model (DOM)

- Real DOM - это реальная структура веб-страницы.
- DOM представляет собой веб-страницу, часто называемую документом, в виде логического дерева, каждая ветвь которого заканчивается узлом, а каждый узел содержит объекты.
- Разработчик может изменять содержимое документа с помощью языка сценариев, например JavaScript.
- Изменения и обновления в DOM происходят быстро благодаря его древовидной структуре, но перерисовка целых документов делает DOM медленной.
- Все компоненты пользовательского интерфейса должны быть перерисованы при каждом обновлении DOM.

Пример реального DOM



XPath

XPath использует выражения пути для выбора узлов или наборов узлов в XML-документе. Выбор узла осуществляется путем следования по пути или шагам. XPath использует выражения пути для выбора узлов в XML-документе. Выбор узла осуществляется путем следования по пути или шагам. Предикаты используются для поиска определенного узла или узла, содержащего определенное значение. Также как SQL, XPath является декларативным языком запросов.

Выражения XPath

Выражение	Описание
nodename	Выбирает все узлы с именем "nodename".
/	Выбирает из корневого узла
//	Выбирает узлы в документе, начиная с текущего узла, которые соответствуют выбору, независимо от того, где они находятся
.	Выбирает текущий узел
..	Выбирает родителя текущего узла
@	Выбирает атрибуты

Предикаты XPath

Выражение пути	Результат выражения пути
<code>/bookstore/book[1]</code>	Выбирается первый элемент book, являющийся дочерним для элемента bookstore.
<code>/bookstore/book[last()]</code>	Выбирает последний элемент book, являющийся дочерним для элемента bookstore
<code>/bookstore/book[last()-1]</code>	Выбирает предпоследний элемент книги, являющийся дочерним элементом элемента bookstore
<code>/bookstore/book[position()<3]</code>	Выбирает первые два элемента книги, являющиеся дочерними элементами элемента bookstore
<code>//title[@lang]</code>	Выбирает все элементы title, имеющие атрибут lang
<code>//title[@lang='en']</code>	Выбирает все элементы заголовка, имеющие атрибут "lang" со значением "en"
<code>/bookstore/book[price>35.00]</code>	Выбирает все элементы book элемента bookstore, которые имеют элемент price со значением больше 35.00
<code>/bookstore/book[price>35.00]/title</code>	Выбирает все элементы заголовков книг элемента bookstore, которые имеют элемент price со значением больше 35.00

Безголовый (headless) браузер

Безголовый (headless) браузер - это веб-браузер без графического интерфейса (GUI). Безголовые браузеры широко используются для веб-скрапинга, поскольку они могут помочь вам рендерить JavaScript или программно вести себя как пользователь-человек, чтобы избежать блокировки.

Безголовые браузеры также популярны для кроссбраузерного тестирования, поскольку браузер отображает все программно, а не через пользовательский интерфейс. Это означает, что браузеры можно запускать на машинах, к которым не подключены дисплеи или виртуальные дисплеи типа XVFB.

Примеры браузеров, поддерживающих headless - режим

- Chromium - это наиболее популярный браузер, который может работать без головы. Поскольку многие современные браузеры, такие как Edge или Brave, основаны на Chromium, это означает, что их также можно запускать без головы.
- Google Chrome создан на основе Chromium, и также может работать без головы. Он немного более громоздкий, и функции headless появятся в нем позже, но с ним легче имитировать реального пользователя.
- Firefox благодаря встроенным настройкам конфиденциальности подходит для того, чтобы протестировать функции конфиденциальности.
- WebKit (Apple Safari) - браузерный движок с открытым исходным кодом, лежащий в основе Apple Safari
- Splash - безголовый веб-браузер, написанный на языке Python, с HTTP API, поддержкой сценариев Lua и встроенной IDE на основе Python (Jupyter). Он не так популярен, как другие, поскольку не основан на реальном браузере, но он быстр и может выполнять работу в простых сценариях.

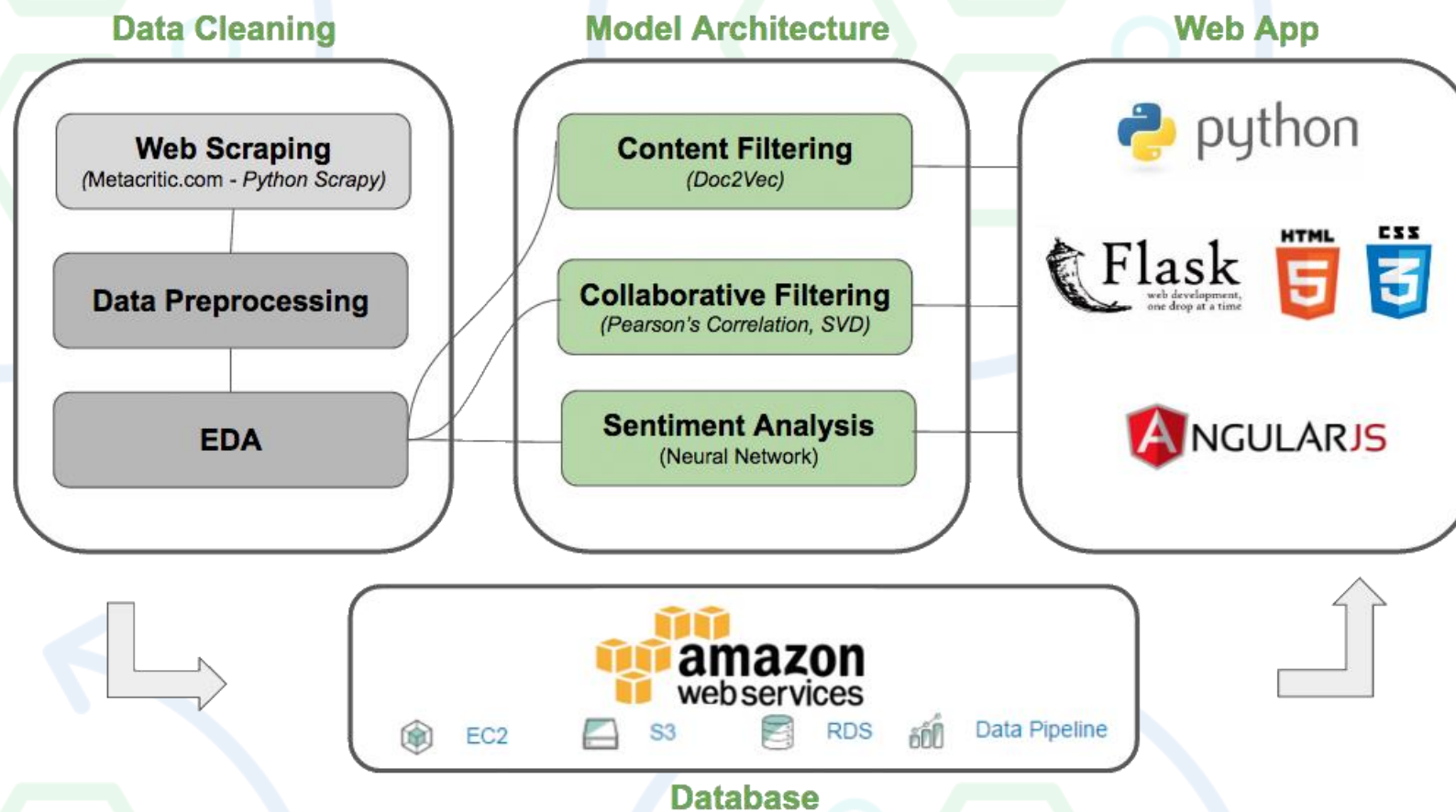
Агент пользователя

Агент пользователя - это информация, передаваемая браузером серверу при каждом HTTP-запросе. Он идентифицирует браузер, его версию и операционную систему, на которой он работает. Безголовый Chrome задает специальную строку агента пользователя для идентификации себя как безголового браузера, и веб-сайт может использовать эту строку для определения того, что к нему обращаются с безголового браузера.

Прокрутка

Прокрутка - это перемещение текста, изображений или любого другого содержимого для обеспечения его правильной видимости в соответствии с разрешением экрана. Это действие известно как прокрутка. Существует два типа прокрутки: горизонтальная и вертикальная.

Веб-скрапинг. Пример архитектуры решения для скрапинга



Веб-скрапинг. Фреймворки Python



Selenium

BeautifulSoup

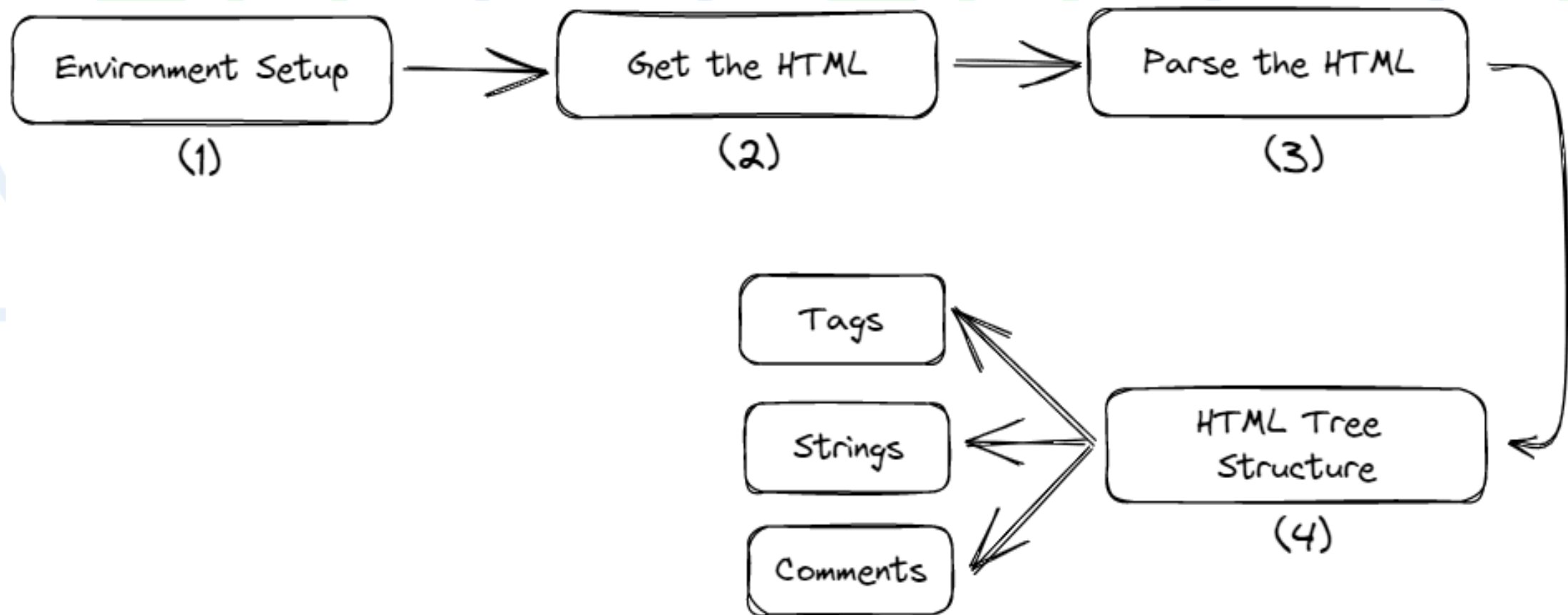


Scrapy



Playwright

Общая схема процесса веб-скрапинга



Библиотеки Python для парсинга HTML страниц

- BeautifulSoup
- html5lib

Библиотеки Python для парсинга HTML страниц. BeautifulSoup

Beautiful Soup - это пакет Python для разбора HTML- и XML-документов (в том числе с некорректной разметкой, т.е. с незакрытыми тегами, названными так в честь tag soup). Он создает дерево разбора разобранных страниц, которое может быть использовано для извлечения данных из HTML, что полезно для веб-скрапинга.

BeautifulSoup. Пример Python

```
1 import requests
2 from bs4 import BeautifulSoup
3
4 url = 'https://quotes.toscrape.com/'
5 response = requests.get(url)
6 soup = BeautifulSoup(response.text, 'lxml')
7
8 print(soup)

</span>
<div class="tags">
  Tags:
    <meta class="keywords" content="aliteracy,books,classic,humor" itemprop="keywords"/>
  <a class="tag" href="/tag/aliteracy/page/1/">aliteracy</a>
  <a class="tag" href="/tag/books/page/1/">books</a>
  <a class="tag" href="/tag/classic/page/1/">classic</a>
  <a class="tag" href="/tag/humor/page/1/">humor</a>
</div>
</div>
<div class="quote" itemscope="" itemtype="http://schema.org/CreativeWork">
  <span class="text" itemprop="text">"Imperfection is beauty, madness is genius and it's better to b
  <span>by <small class="author" itemprop="author">Marilyn Monroe</small>
```

BeautifulSoup. Получение имени тега

Объект Name соответствует имени тега XML или HTML в исходном документе.

tag.name

BeautifulSoup. Получение имени тега.

Пример кода

КОД

```
from bs4 import BeautifulSoup

soup = BeautifulSoup('''
    <root>
        <tag_nm>the first strong tag</tag_nm>
    </root>
''', "lxml")
tag = soup.tag_nm
name = tag.name
print(name)
```

РЕЗУЛЬТАТ

tag_nm

BeautifulSoup. Извлечение значения атрибута

Тег может иметь любое количество атрибутов. Например, тег `<b class="active">` имеет атрибут "class", значение которого равно "active". Можно получить доступ к атрибутам тега, обращаясь к нему как к словарю.

`tag.attrs`

BeautifulSoup. Получение имени тега.

Пример кода

КОД

```
from bs4 import BeautifulSoup

soup = BeautifulSoup('''
    <html>
        <h2 class="hello"> Heading 1 </h2>
        <h1> Heading 2 </h1>
    </html>
''', "lxml")
tag = soup.h2
attribute = tag.attrs
print(attribute)
```

РЕЗУЛЬТАТ

```
{'class': ['hello']}
```


BeautifulSoup. Методы поиска предков и потомков текущего элемента дерева разбора

- `.previous_sibling` используется для поиска предыдущего элемента текущего элемента
- `.next_sibling` используется для поиска следующего элемента текущего элемента
- `.previous_siblings` используется для нахождения всех предыдущих элементов текущего элемента
- `.next_siblings` используется для нахождения всех последующих элементов текущего элемента

BeautifulSoup. Методы поиска предков и потомков текущего элемента дерева разбора. Пример кода

КОД

```
from bs4 import BeautifulSoup

html_code = """<a><b>text1</b><c>text2</c></a>"""
soup = BeautifulSoup(html_code, 'html.parser')
print(soup.b.next_sibling)
```

РЕЗУЛЬТАТ

```
<c>text2</c>
```

HTML-таблица

HTML-таблица задается с помощью тега "table". Каждая строка таблицы определяется тегом "tr". Заголовок таблицы задается тегом "th". По умолчанию заголовки таблиц выделяются жирным шрифтом и выравниваются по центру. Данные/ячейки таблицы определяются тегом "td".

Парсинг результатов запросов в Google поиске. Пример кода

КОД

```
import requests
import bs4

city = "Moscow"
url = "https://google.com/search?q=weather+in+" + city
request_result = requests.get( url )
soup = bs4.BeautifulSoup( request_result.text, "html.parser" )

temp = soup.find( "div" , class_='BNeawe' ).text
print( temp )
```

РЕЗУЛЬТАТ

54°F

BeautifulSoup. Поиск тегов по классу CSS с помощью CSS Selectors

Селектор элементов в CSS используется для выбора HTML-элементов, которые необходимо стилизовать. В объявлении селектора указывается имя HTML-элемента, а внутри скобок {} записываются CSS-свойства, которые должны быть применены к этому элементу.

```
find_all(class_="class_name")
```

BeautifulSoup. Поиск тегов по классу CSS с помощью CSS Selectors. Пример кода

КОД

```
from bs4 import BeautifulSoup

markup = """
<!DOCTYPE>
<html>
  <head><title>Example</title></head>
  <body>
    <div class="first"> Div with Class first
    </div>
    <p class="first"> Para with Class first
    </p>
    <div class="second"> Div with Class second
    </div>
    <span class="first"> Span with Class first
    </span>
  </body>
</html>
"""

soup = BeautifulSoup(markup, 'html.parser')
for i in soup.find_all(class_="first"):
    print(i.name)
```

РЕЗУЛЬТАТ

```
div
p
span
```

The background features a repeating pattern of stylized chemical structures. Each unit consists of a large light blue circle containing three green hexagons and three light blue circles. A curved arrow points clockwise around the circle. Below each circle is a horizontal line with an arrow pointing to the right. The text "Спасибо за внимание!" is centered over the middle unit.

Спасибо за внимание!