HTML, CSS
Селекторы, получение объектов страницы

Window: что это? и для чего?

Глобальный объект предоставляет переменные и функции, доступные в любом месте программы. По умолчанию это те, что встроены в язык или среду исполнения.

- Глобальный объект хранит переменные, которые должны быть доступны в любом месте программы.
- Глобальный объект имеет универсальное имя – `globalThis`.
- Следует хранить значения в глобальном объекте, только если они действительно глобальны для нашего проекта. И стараться свести их количество к минимуму.
- В браузерах, если только мы не используем модули, глобальные функции и переменные, объявленные с помощью `var`, становятся свойствами глобального объекта.
- Для того, чтобы код был проще и в будущем его легче было поддерживать, следует обращаться к свойствам глобального объекта напрямую, как `window.x`.

Document

Каждая веб-страница, которая загружается в браузер, имеет свой собственный объект **document**. Интерфейс документа служит точкой входа для получения содержимого веб-страницы (всего DOM - дерева, включая такие элементы как `<body>` и `<table>`), а также обеспечивает функциональность, которая является глобальной для документа, например, для получения URL-адреса страницы или создания новых элементов в документе).

DOM

Основой HTML-документа являются теги.

В соответствии с объектной моделью документа («Document Object Model», коротко DOM), каждый HTML-тег является объектом. Вложенные теги являются «детьми» родительского элемента. Текст, который находится внутри тега, также является объектом.

Все эти объекты доступны при помощи JavaScript, мы можем использовать их для изменения страницы.

Например, `document.body` – объект для тега `<body>`.

Если запустить этот код, то `<body>` станет красным на 3 секунды:

```
document.body.style.background = 'red'; // сделать фон красным

setTimeout(() => document.body.style.background = '', 3000); // вернуть назад
```

Это был лишь небольшой пример того, что может DOM. Скоро мы изучим много способов работать с DOM, но сначала нужно познакомиться с его структурой.

HTML

HTML (от англ. *HyperText Markup Language* — «язык гипертекстовой разметки») — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц имеют содержание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства.

HTML-теги

HTML-теги — основа языка HTML. Теги используются для разграничения начала и конца элементов в разметке.

Основные теги

<html></html>

Указывает программе просмотра страниц, что это HTML документ.

<head></head>

Определяет место, где помещается различная информация не отображаемая в теле документа. Здесь располагается тег названия документа и теги для поисковых машин.

<body></body>

Определяет видимую часть документа

Теги оглавления

<title></title>

Помещает название документа в оглавление программы просмотра страниц

DIV

Элемент `<div>` является блочным элементом и предназначен для выделения фрагмента документа с целью изменения вида содержимого. Как правило, вид блока управляется с помощью стилей. Чтобы не описывать каждый раз стиль внутри тега, можно выделить стиль во внешнюю таблицу стилей, а для тега добавить атрибут class или id с именем селектора.

SPAN

Тег **** предназначен для определения строчных элементов документа. В отличие от блочных элементов, таких как **<table>**, **<p>** или ** **, с помощью тега **** можно выделить часть информации внутри других тегов и установить для нее свой стиль. Например, внутри абзаца (тега **<p>**) можно изменить цвет и размер первой буквы, если добавить начальный и конечный тег **** и определить для него стиль текста. Чтобы не описывать каждый раз стиль внутри тега, можно выделить стиль во внешнюю таблицу стилей, а для тега добавить атрибут class или id с именем селектора.

Input

Тег `<input>` является одним из разносторонних элементов формы и позволяет создавать разные элементы интерфейса и обеспечить взаимодействие с пользователем. Главным образом `<input>` предназначен для создания текстовых полей, различных кнопок, переключателей и флажков. Хотя элемент `<input>` не требуется помещать внутрь контейнера `<form>`, определяющего форму, но если введенные пользователем данные должны быть отправлены на сервер, где их обрабатывает серверная программа, то указывать `<form>` обязательно. То же самое обстоит и в случае обработки данных с помощью клиентских приложений, например, скриптов на языке JavaScript.

Button

Тег **<button>** создает на веб-странице кнопки и по своему действию напоминает результат, получаемый с помощью тега **<input>** (с атрибутом `type="button | reset | submit"`). В отличие от этого тега, **<button>** предлагает расширенные возможности по созданию кнопок. Например, на подобной кнопке можно размещать любые элементы HTML, в том числе изображения. Используя стили можно определить вид кнопки путем изменения шрифта, цвета фона, размеров и других параметров.

Select

Тег `<select>` позволяет создать элемент интерфейса в виде раскрывающегося списка, а также список с одним или множественным выбором, как показано далее. Конечный вид зависит от использования атрибута size тега `<select>`, который устанавливает высоту списка. Ширина списка определяется самым широким текстом, указанным в теге `<option>`, а также может изменяться с помощью стилей. Каждый пункт создается с помощью тега `<option>`, который должен быть вложен в контейнер `<select>`. Если планируется отправлять данные списка на сервер, то требуется поместить элемент `<select>` внутрь формы. Это также необходимо, когда к данным списка идет обращение через скрипты.

CSS

CSS (*каскадные таблицы стилей*) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки.

CSS используется создателями веб-страниц для задания цветов, шрифтов, расположения отдельных блоков и других аспектов представления внешнего вида этих веб-страниц. Основной целью разработки CSS являлось разделение описания логической структуры веб-страницы (которое производится с помощью HTML или других языков разметки) от описания внешнего вида этой веб-страницы (которое теперь производится с помощью формального языка CSS). Такое разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом.

Selectors

Селектор определяет, к какому элементу применять то или иное CSS-правило.

document.querySelector/s(‘selector’) or $/$$(‘selectors’)

Методы HTTP запросов: get, post, put, delete

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*.

- `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.