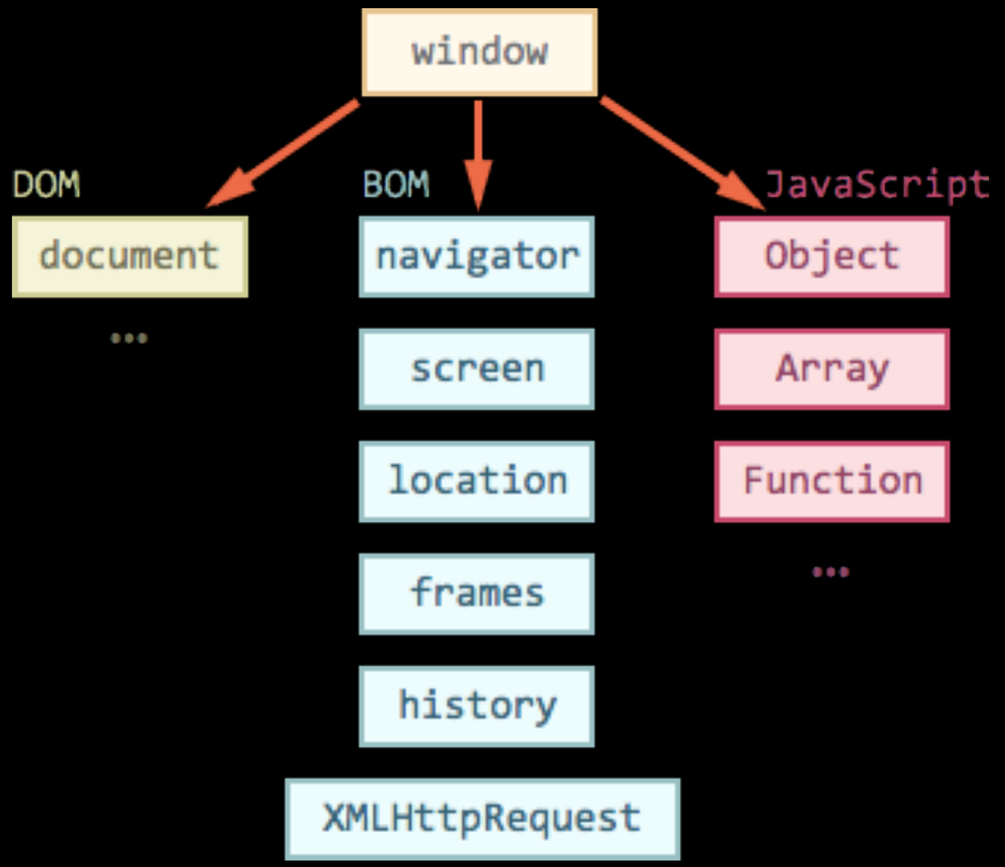
Документ и объекты страницы

JavaScript

Окружение


window
DOM
document
...
BOM
navigator
screen
location
frames
history
XMLHttpRequest
JavaScript
Object
Array
Function
...

Объектная модель документа

- Document Object Model (DOM)
- Глобальный объект document даёт возможность взаимодействовать с содержимым страницы
- Объект и его свойств и методы описаны в стандарте W3C DOM
- document.body.style.background = 'red';
  alert('BODY временно стал красным');
  document.body.style.background = '';

Объектная модель браузера

- Browser Object Model (BOM)
- BOM — это объекты для работы с чем угодно, кроме документа, например:
  - navigator
  - location
  - alert/confirm/prompt
- Большинство возможностей BOM стандартизированы в HTML 5
- alert(location.href); // текущий адрес

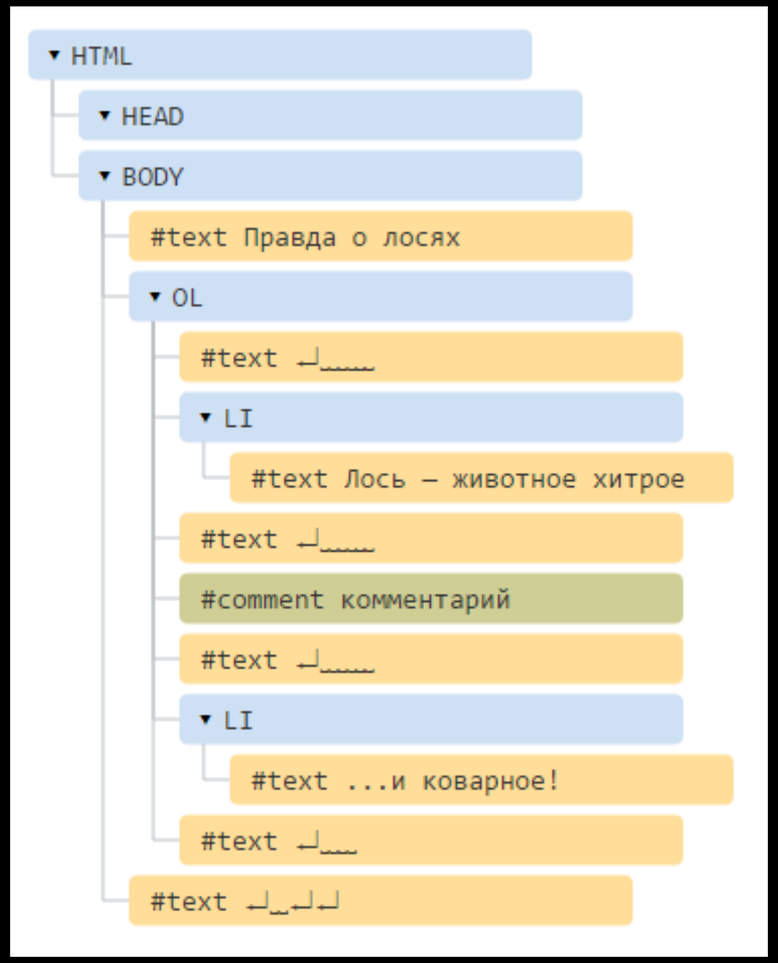
Дерево DOM

- Согласно DOM-модели, документ является иерархией, деревом
- DOM — это представление документа в виде дерева объектов, доступное для изменения через JavaScript
- При чтении неверного HTML браузер автоматически корректирует его для показа и при построении DOM

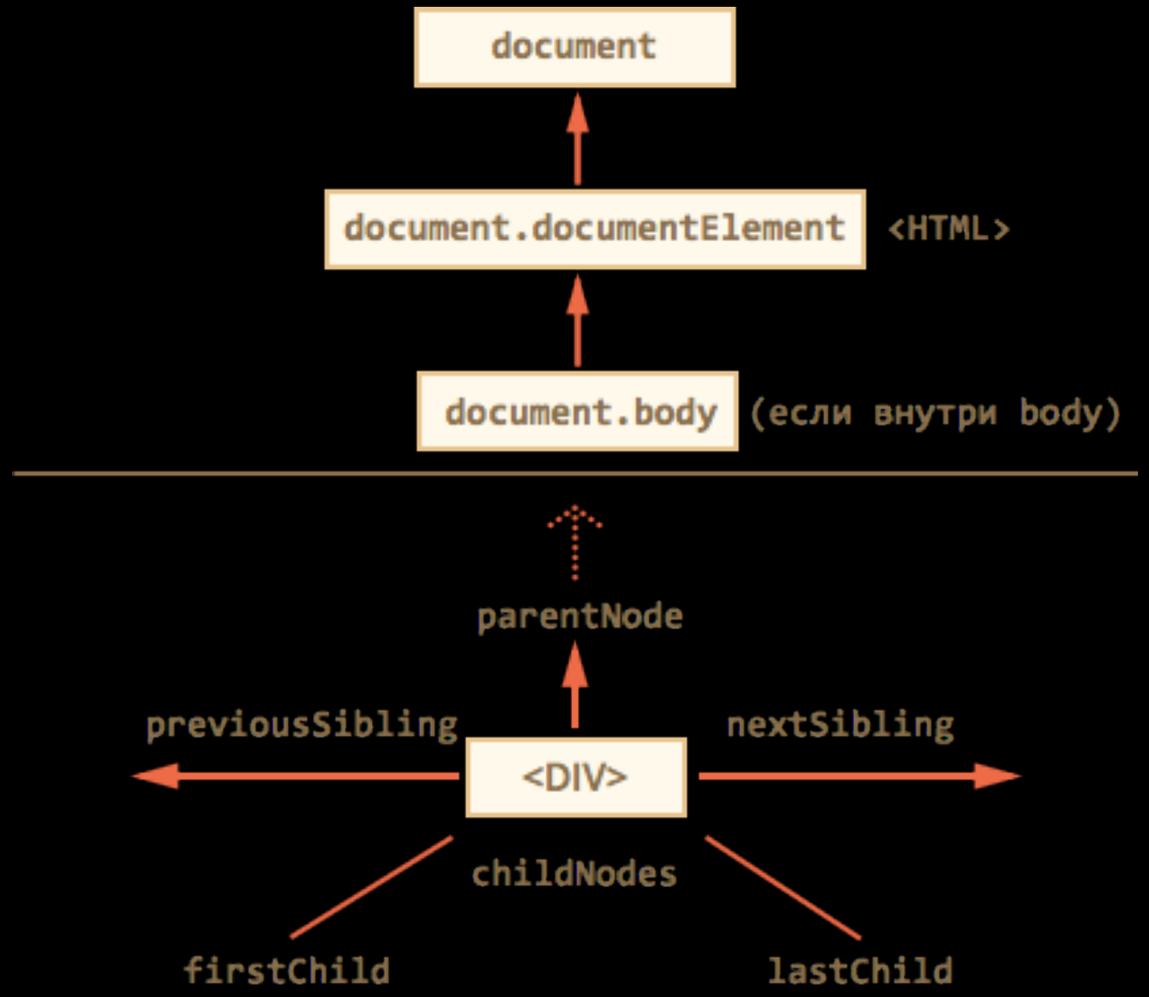
Типы узлов

- Всего различают 12 типов узлов, но на практике работают только с 4 из них:
  - Документ — точка входа в DOM
  - Элементы — основные строительные блоки
  - Текстовые узлы — содержат, собственно, текст
  - Комментарии — информация, которая не будет показана, но доступна из JS

Пример дерева



Навигация по DOM-узлам



Верхние элементы дерева

- Самые верхние элементы дерева доступны напрямую из document
- <HTML> = document.documentElement
  Это свойство ссылается на DOM-объект для тега <html>
- <BODY> = document.body
  Это свойство ссылается на DOM-объект для тега <body>

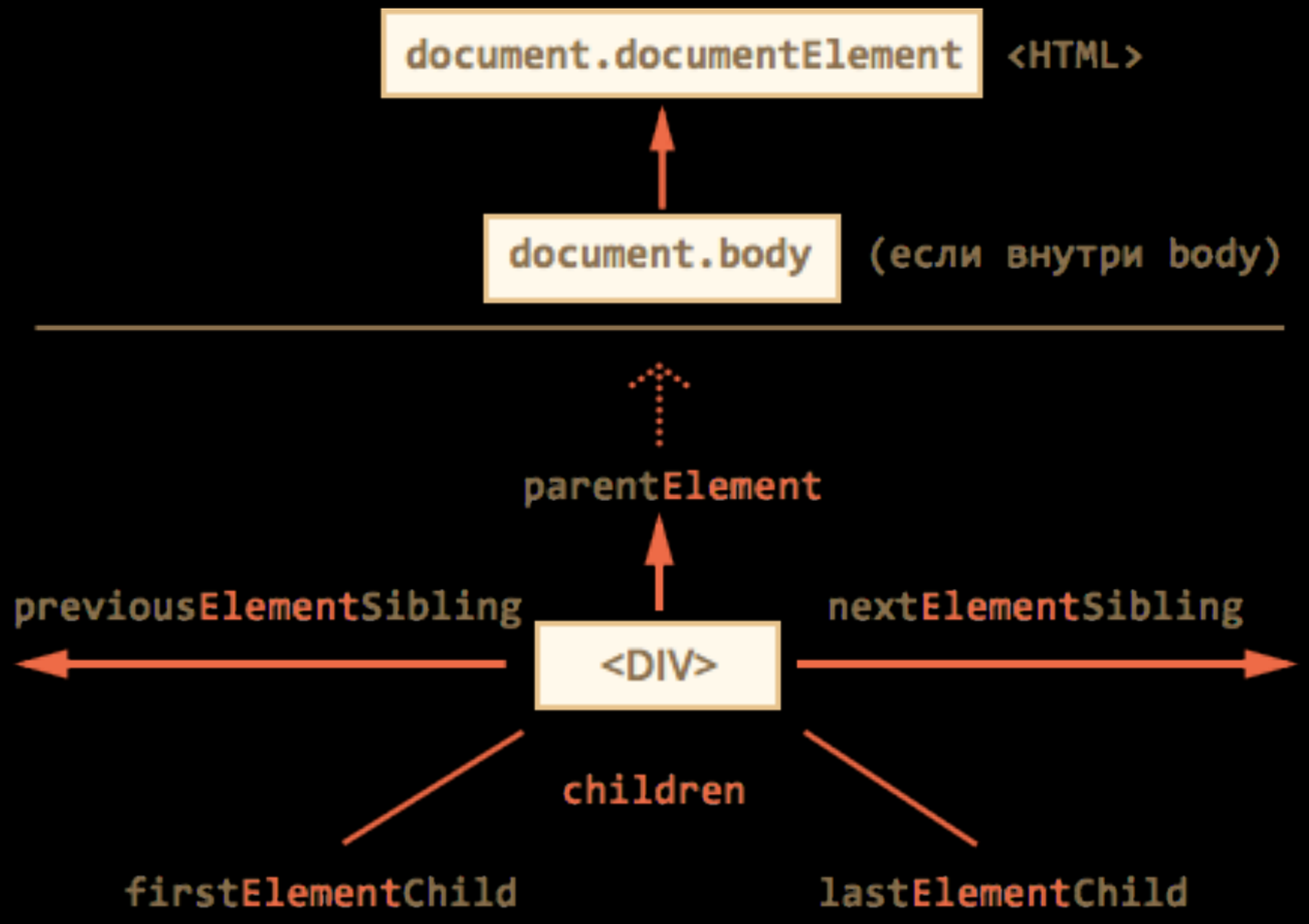
Дети

- **Дочерние элементы (или дети)** — элементы, которые лежат *непосредственно* внутри данного элемента
- **Потомки** — все элементы, которые лежат внутри данного, вместе с их детьми, детьми их детей и так далее
- Псевдо-массив childNodes хранит все дочерние элементы, включая текстовые

Пример

- <html>
- <body>
- Начало
-
- Информация
-
- Конец
- <script>
- for (var i=0; i<document.body.childNodes.length; i++) {
  alert(document.body.childNodes[i]);
  }
- </script>
- …
- </body>
- </html>

Навигация по элементам



Соседи, родители, дети

- children — только дочерние узлы-элементы
- firstElementChild, lastElementChild —первый и последний дети-элементы
- previousElementSibling, nextElementSibling — соседи-элементы
- parentElement — родитель-элемент

Особые ссылки

- У конкретных элементов DOM могут быть свои дополнительные ссылки для большего удобства навигации

Table

- **table.rows** — коллекция строк таблицы
- table.caption/tHead/tFoot — ссылки на элементы таблицы caption, thead, tfoot
- table.tBodies — коллекция элементов таблицы tbody

Thead/tfoot/tbody и td/th

- tbody.rows — коллекция строк секции

- **td.cellIndex** — номер ячейки в строке

Tr

- **tr.cells** — коллекция ячеек td/th
- **tr.sectionRowIndex** — номер строки в текущей секции thead/tbody
- tr.rowIndex — номер строки в таблице

Основные методы поиска элементов DOM

Метод	Ищет по…	Ищет внутри	Поддержка
getElementById	id	-	везде
getElementsByName	name	-	везде
getElementsByTagName	тег или '*'	✔	везде
getElementsByClassName	классу	✔	кроме IE8-
querySelector	CSS-селектор	✔	везде
querySelectorAll	CSS-селектор	✔	везде

Доступ к элементу по id

- Существует 2 способа обратиться к элементу по его ID:
  - document.getElementById('ID');
  - window['ID'];

- document.getElementById('test');
- window['test'];

Доступ к элементу по имени

- document.getElementsByName(name) позволяет получить все элементы с данным атрибутом name

- var age;
- age = document.getElementsByName('age');

Доступ к элементу по тегу

- element.getElementsByTagName(tag) позволяет получить все элементы с заданным тегом tag

- var el;
- el=document.getElementsByTagName('div');
- el[0].getElementsByTagName('li');

Доступ к элементу по классу

- element.getElementsByClassName(class) позволяет получить все элементы с заданным классом

- var el;
- el=document.getElementsByClassName('art');
- el[0].document.getElementsByClassName('li');

Доступ к элементу через CSS-селекторы

- elem.querySelectorAll(css) возвращает все элементы внутри elem, удовлетворяющие CSS-селектору
- elem.querySelector(css) возвращает первый элемент, соответствующий CSS-селектору

- var el;
- el=document.querySelector('ul > li:last-child');

Проверка элемента

- elem.matches(css) проверяет, удовлетворяет ли elem селектору css
- Возвращает true либо false
- Метод бывает полезным, когда в массиве элементов необходимо выбрать только некоторые из них

Пример

- ...
- ...
- <script>
- var elems = document.body.children;
- for (var i = 0; i < elems.length; i++) {
  if (elems[i].matches('a[href$="zip"]')) { alert("Архив: " + elems[i].href);
  }
  }
- </script>

Родитель по селектору

- elem.closest(css) ищет ближайший элемент выше по иерархии DOM, подходящий под CSS-селектор
- Сам элемент тоже включается в поиск

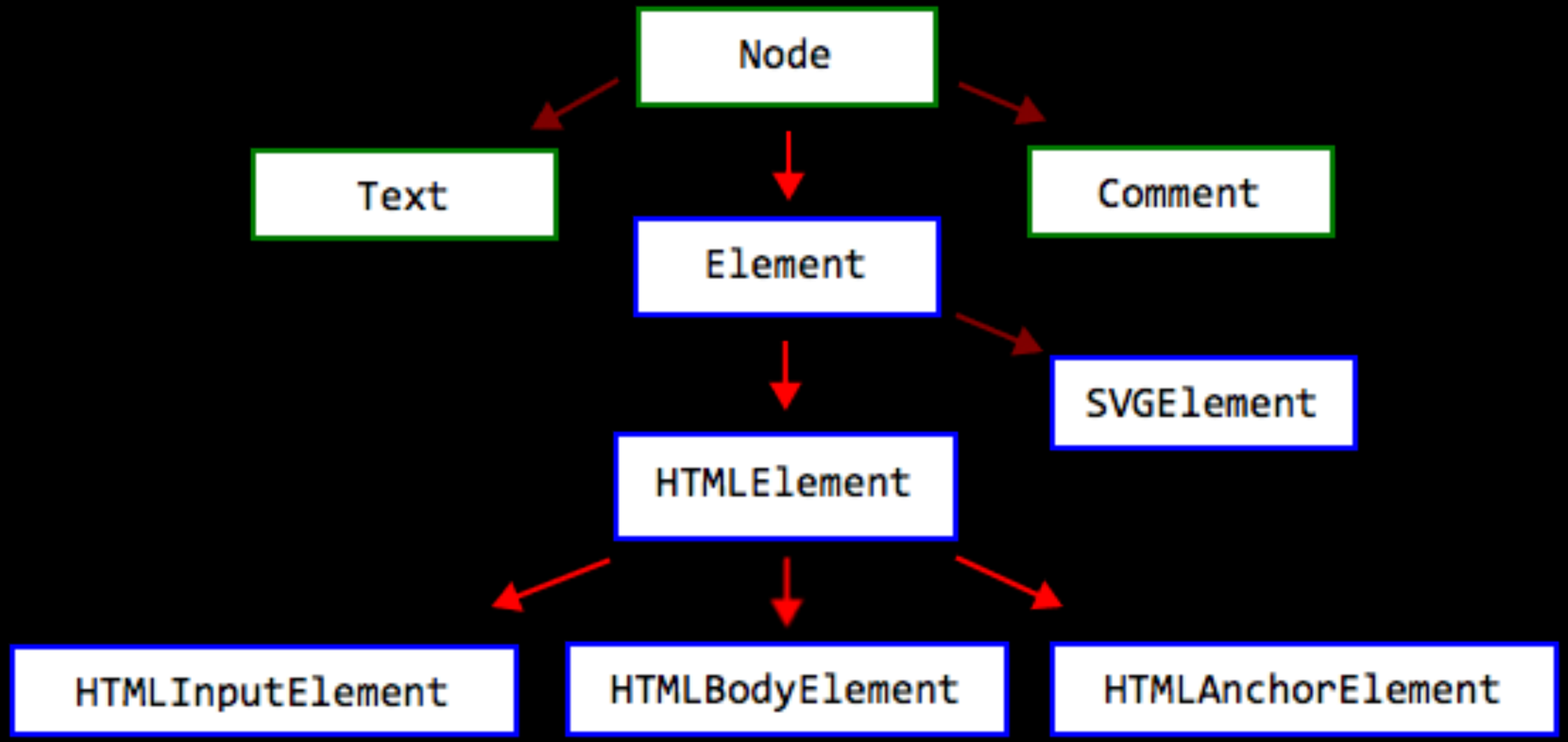
Пример

-
  - <li class="chapter">Глава I
    -
      - <li class="subchapter">Глава 1.1
      - <li class="subchapter">Глава 1.2
    -
  -
-
- <script>
- var numberSpan = document.querySelector('.num');
- alert(numberSpan.closest('li').className)
- </script>

Свойства узлов

- Разные узлы являются объектами различных классов и имеют различные свойства
- Классы DOM образуют иерархию
- Основной объект в ней: Node, от которого наследуют остальные

Иерархия DOM



Иерархия DOM

- От Node наследуют text, comment и element
- От элементов наследуются SVG и HTML
- От HTMLElement наследуются разнообразные узлы HTML:
  - Для <input> — HTMLInputElement
  - Для <body> — HTMLBodyElement
  - Для <a> — HTMLAnchorElement
  - …

Определение класса

- Определить класс элемента можно 2-мя способами:
  - Привести его к строке
    ```
    alert( document.body );
    // [object HTMLBodyElement]
    ```
  - Проверить при помощи instanceof
    ```
    alert( document.body instanceof HTMLElement );
    alert( document.body instanceof Element );
    alert( document.body instanceof Node );
    ```

Содержимое элемента

- Свойство innerHTML позволяет получить HTML-содержимое элемента в виде строки
- В innerHTML можно и читать и писать
- document.getElementById("t1").innerHTML;
- document.getElementById("t1").innerHTML = "1";

Элемент целиком

- Свойство outerHTML содержит HTML элемента целиком
- В outerHTML можно и читать и писать, однако при записи формируется новый элемент
- После перезаписи, переменная, в которой содержался старый элемент, продолжает храниться информация о нем

Пример

- Привет, Мир!
- <script>
  - var div = document.body.children[0];
  - div.outerHTML = ' Новый элемент! ';
  - alert(div.outerHTML);
- </script>

Содержимое текстового узла

- Свойство innerHTML есть только у узлов-элементов
- Содержимое других узлов доступно на чтение и запись через свойство data
- Свойство data доступно на чтение и запись

Пример

- <body>
  - Привет
  - <!-- Комментарий -->
  - <script>
  - for (var i = 0; i < document.body.childNodes.length; i++) {
    - alert(document.body.childNodes[i].data);
  - }
  - </script>
  - Пока
- </body>

Получение текста

- Свойство textContent содержит только текст внутри элемента, за вычетом всех <тегов>
- возвращается в точности весь текст, включая переводы строк и пробелы, но без тегов
- Работает на чтение и на запись

Пример

-
  - <h1>Срочно в номер!</h1>
  - Марсиане атакуют людей!
-
- <script>
  - var news = document.body.children[0];
  - alert(news.textContent);
- </script>

Пример

-
-
- <script>
  - var name = prompt("Имя?", "noname");
  - document.body.children[0].innerHTML = name;
  - document.body.children[1].textContent = name;
- </script>

Невидимость узла

- Свойство hidden делает элемент видимым/невидимым

- Текст
- <div hidden>С атрибутом hidden
- <script>
  - var lastDiv = document.body.children[2];
  - lastDiv.hidden = true;
- </script>

Исследование элементов

- У DOM-узлов есть и другие свойства, зависящие от типа, например:
  - value — значение для INPUT, SELECT и тд
  - id — идентификатор
  - href — адрес ссылки
  - …
- Метод console.dir выводит развертку элемента
- console.dir(document);