

- Создание документов HTML5
- Создание форм HTML5 для запроса и обработки информации
- Валидация форм HTML5
- Обработка событий HTML5 функциями JavaScript
- Управление элементами HTML5 через DOM
- Практика: Написание кода JavaScript для изменения элементов документа

`<form>`

`<input type="...">`

`<select>`

`<textarea>`

`<button>`

`<datalist>`

```
<form>
  Имя:<br>
  <input type="text" name="username"><br>
  Роль:<br>
  <input type="text" name="role" list="roles">
  <datalist id="roles">
    <option value="Администратор">
    <option value="Пользователь">
    <option value="Гость">
  </datalist>
  <br>
  Пароль:<br>
  <input type="password" name="password">
</form>
```

```
<form>
  Имя:<br>
  <input type="text" name="name"><br>
  <input type="radio" name="gender" value="M" checked>
  М<br>
  <input type="radio" name="gender" value="F">
  Ж<br>
  <br>
  <input type="checkbox" name="agree" value="yes">
  Я согласен<br>
  <input type="button" onclick="alert('Нажали!')"
  value="Нажмите">
  <input type="reset" value="Сброс">
  <input type="submit" value="OK">
</form>
```

Типы <input> в HTML5



color

date

datetime-local

email

month

number

range

search

tel

time

url

week

```
<form>
  Имя:<br>
  <input type="text" name="name"><br>
  <input type="radio" name="gender" value="M" checked>
  М<br>
  <input type="radio" name="gender" value="F">
  Ж<br>
  <br>
  <input type="checkbox" name="agree" value="yes">
  Я согласен<br>
  <input type="button" onclick="alert('Нажали!')"
  value="Нажмите">
  <input type="reset" value="Сброс">
  <input type="submit" value="OK">
</form>
```

```
<form>
```

Вы узнали о нас

```
<select name="source">
```

```
  <option value="ad">Из рекламы</option>
```

```
  <option value="ad" selected>От друзей</option>
```

```
  <option value="ad">Другое</option>
```

```
</select>
```

Комментарии

```
<textarea name="comments"></textarea>
```

```
<input type="submit" value="OK">
```

```
</form>
```

Атрибуты форм

target
method
enctype

autocomplete
novalidate

```
<form
  method="post"
  target="http://localhost/index.html"
  autocomplete="on"
  novalidate
>
  <input type="text" name="name"><br>
  <input type="radio" name="gender" value="M" checked>
  <input type="submit" value="OK">
</form>
```


Атрибуты <input>

value	autofocus	autocomplete
readonly	height, width	form
disabled	list	formaction
size	min, max	formenctype
maxlength	multiple	formmethod
	pattern (regexp)	formnovalidate
	placeholder	formtarget
	required	
	step	

```
<input type="number"  
  step="5" min="10" max="100" value="30"  
>
```

<meta>

<title>

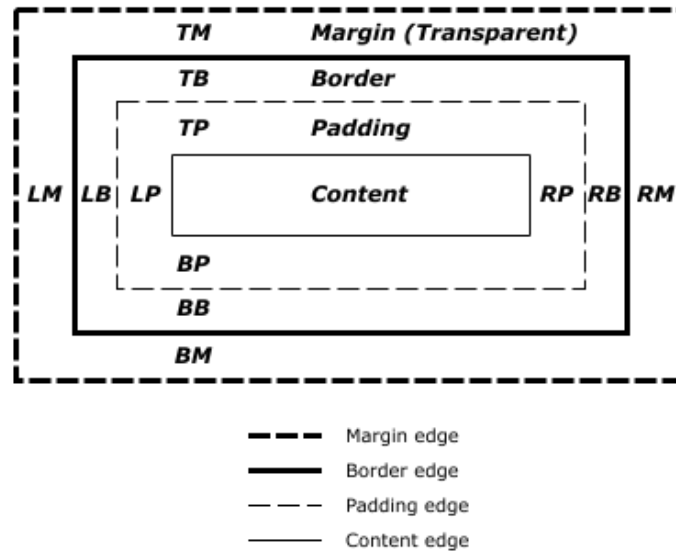
<style>

<script>

<link>

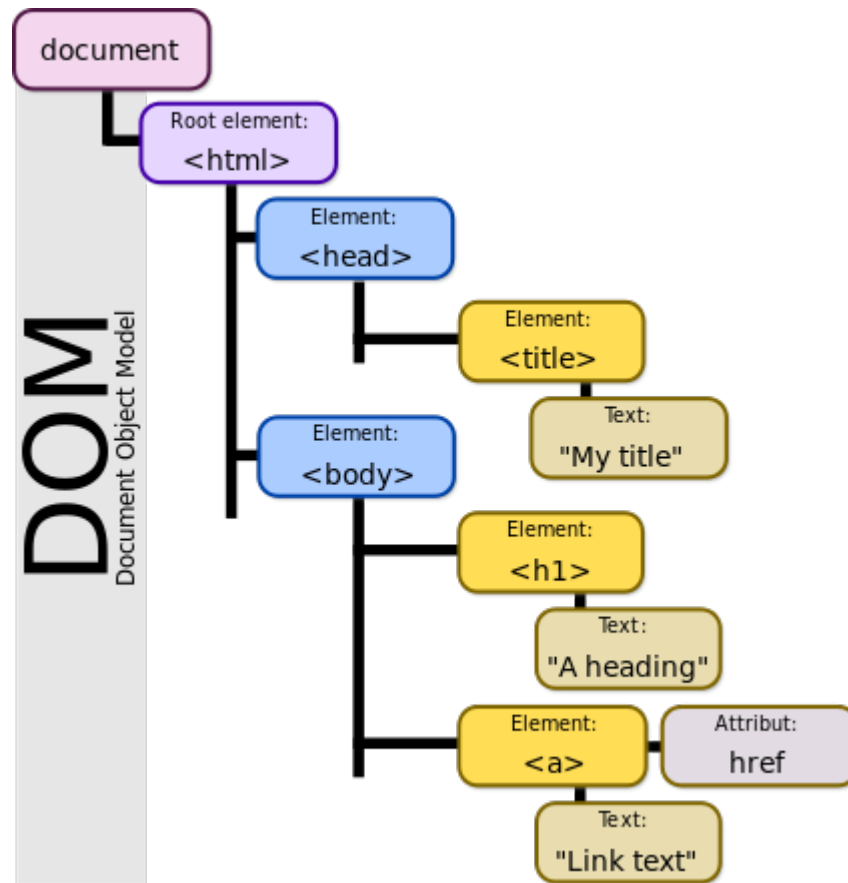
Элементы body

- Блочные
 - div
 - p, h1 ... h6
 - dl (dt, dd)
 - ul, ol (li)
 - hr
 - form
- Линейные
 - span
 - a
 - br
 - img
 - object

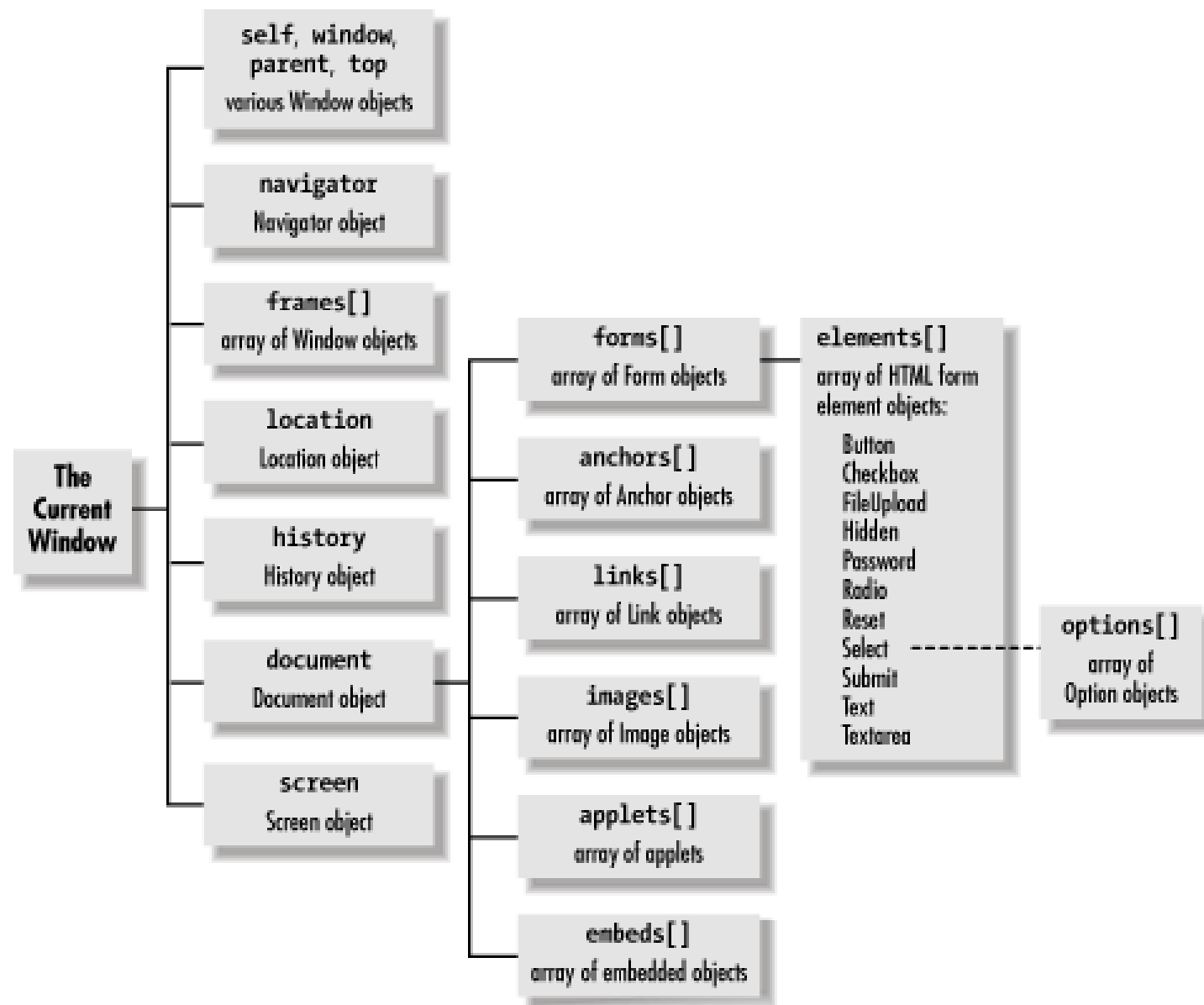


- Табличные
 - table
 - caption
 - thead, tbody, tfoot
 - col, colgroup
 - tr, th, td

window.document



Level 0 DOM



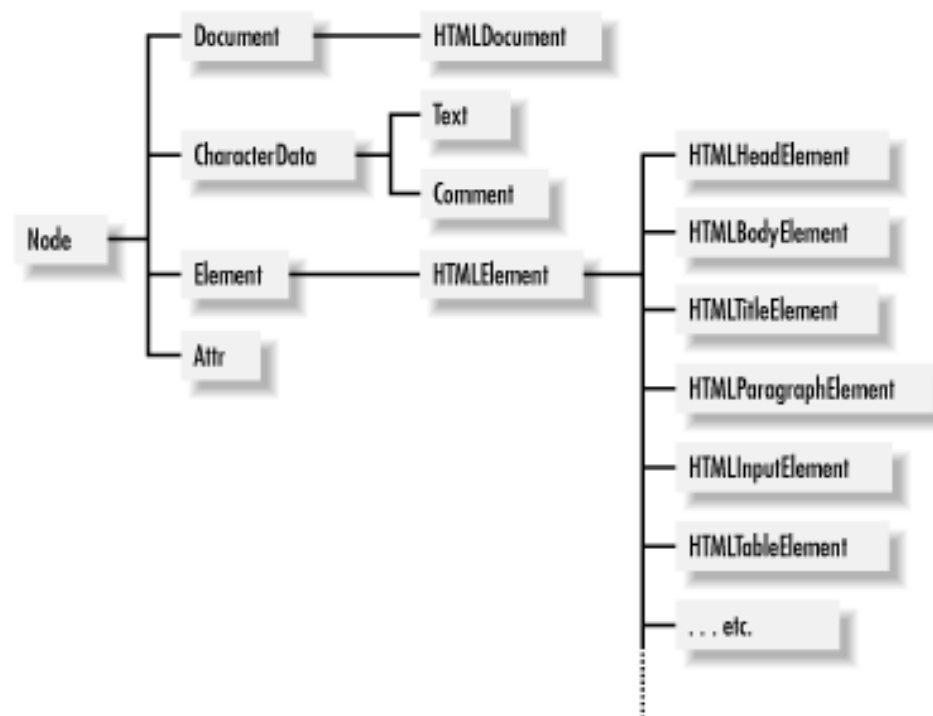
- Level 0 – до стандартизации
- Level 1 – 1988, 2000
- Level 2 – 2000, 2003 (HTML)
 - getElementById
 - модель событий
 - XML namespace
 - CSS
- Level 3 – 2003-2004
 - XPath
 - события клавиатуры
 - сериализация как XML
- DOM 4 – проект

Node

- nodeType
- nodeValue
- appendChild
- replaceChild
- removeChild
- insertBefore
- childNodes
- firstChild
- lastChild
- nextSibling
- previousSibling
- parentNode

Типы узлов

- Документ
- Фрагмент документа
- Элемент
- Атрибут
- Текстовый узел
- Комментарий



Node.DOCUMENT_NODE
Node.DOCUMENT_FRAGMENT_NODE
Node.ELEMENT_NODE

Node.ATTRIBUTE_NODE
Node.TEXT_NODE
Node.COMMENT_NODE

Полифилл (polyfill)

```
if (document.documentElement.firstChild ===  
undefined) {  
  
    Element.prototype.firstChild = function() {  
        var el = this.firstChild;  
        do {  
            if (el.nodeType === Node.ELEMENT_NODE) {  
                return el;  
            }  
            el = el.nextSibling;  
        } while (el);  
        return null;  
    }  
};
```

```
<script  
    src="//cdn.polyfill.io/v1/polyfill.js?features=es6"  
></script>
```

Обход дерева DOM



```
function displayIds (node) {  
    if (node.id != '') {  
        console.log (node.id) ;  
    }  
    for (  
        let i=0;  
        i<node.childNodes.length;  
        i++  
    ) {  
        displayIds (node.childNodes[i]) ;  
    }  
}  
  
displayIds (document) ;
```

getElementById или id

```
<div id="content-holder">
  <div id="content">Элемент</div>
</div>

<script>
  console.log(content);
  console.log(window['content-holder']);
</script>
```

```
<div id="content">Выделим этот элемент</div>

<script>
  var elem = document.getElementById('content');
  elem.style.background = 'red';
  console.log(elem == content); // true
  content.style.background = ""; // тот же элемент
</script>
```

element.getElementsByTagName

```
var tableElem = document.getElementById('age-table');  
var elems = tableElem.getElementsByTagName('input');  
var allElems = tableElem.getElementsByTagName('*');
```

element.getElementsByClassName (кроме IE8-)

```
<div class="article">Статья</div>  
<div class="long article">Длинная статья</div>  
  
<script>  
  var articles =  
document.getElementsByClassName('article');  
  alert( articles.length ); // 2  
</script>
```

element.querySelectorAll

```
var elements = document.querySelectorAll(':hover');
```

element.querySelector

```
document.querySelector('#list > li:last-child').  
    style.display = 'none';
```

element.matches

```
if(elem.matches('a[href$="zip"]'))  
    elem.title = 'Скачать архив';
```

element.closest

```
document.querySelector('td').closest('tr').  
    style.backgroundColor = '#55f';
```

element.querySelectorAll

```
var elements = document.querySelectorAll(':hover');
```

element.querySelector

```
document.querySelector('#list > li:last-child').  
    style.display = 'none';
```

element.matches

```
if(elem.matches('a[href$="zip"]'))  
    elem.title = 'Скачать архив';
```

element.closest

```
document.querySelector('td').closest('tr').  
    style.backgroundColor = '#55f';
```

Модификация документа

```
var div = document.createElement('div');
div.className = "alert alert-success";
div.innerHTML = "<strong>Внимание!</strong>";
document.body.insertBefore(
    div, document.body.firstChild
);
let text = "Важное сообщение.";
div.appendChild(document.createTextNode(text));
```

- createElement(tag)
- createTextNode(value)
- cloneNode(deep)

- innerHTML

- appendChild(elem)
- parent.insertBefore(elem, nextSibling)
- removeChild(elem)
- replaceChild(newElem, elem)

```
var int = document.getElementById('first-name');  
int.className = "form-input";  
console.log(int.tagName);  
int.setAttribute ('max', '1000');
```

| | | |
|-----------------------|-----------|---------------|
| getAttribute (...) | tagName | type |
| setAttribute (...) | id | name |
| removeAttribute (...) | className | value |
| hasAttribute (...) | style | checked |
| | title | selectedIndex |

Событийно-ориентированное программирование



- Слабые связи в коде
- Гибкость, простота адаптации
- Бесшовная связь с сервером через события



- Трудно отлаживать
- Трудно сопровождать

Событийно-ориентированное программирование

```
var Foo = function() {  
    this.lorem = function()  
    {  
        alert("HELLO FOO");  
    }  
};
```

```
var Bar = function() {  
    var foo = new Foo();  
    foo.lorem();  
};
```

Событийно-ориентированное программирование

```
var Foo = function() {  
  
    myEventSystem.on("lorem.event", lorem);  
  
    function lorem()  
    {  
        alert("HELLO FOO");  
    }  
  
}
```

```
var Bar = function() {  
    var foo = new Foo();  
    myEventSystem.send("lorem.event");  
};
```

Добавление обработчиков



```
<input
  id="b1"
  type="button"
  value="Click me"
  onclick="alert('Thanks!');"
/>
```

```
<input id="myElement" type="button" value="Жми"/>
<script>
  document.getElementById('myElement').onclick =
  function() {
    alert('Thanks');
  }
</script>
```

Добавление обработчиков



```
var element = document.getElementById("myElement");  
var handler = function() { alert('Thanks'); }
```

```
element.addEventListener("click", handler);
```

```
element.removeEventListener("click", handler);
```

- target
- currentTarget
- type
- timestamp
- preventDefault()
- stopPropagation()
- stopImmediatePropagation()

```
addEventListener(  
  "click",  
  e=>  
    alert(e.target.innerHTML);  
);
```

- online
- offline
- error
- open
- popstate
- load
- unload
- beforeunload

```
addEventListener(  
    "beforeunload",  
    ()=>confirm('Вы уверены?')  
);
```

- DomContentLoaded

```
document.addEventListener(  
    "DomContentLoaded",  
    function() {  
        document.body.addEventListener('click', ...);  
    }  
);
```


- submit
- reset
- focus
- blur
- change
- input
- select

```
function areYouSure(e) {  
    if (!confirm('Are you sure?')) {  
        e.preventDefault();           //return false;  
    }  
}  
  
myForm.addEventListener("submit", areYouSure);
```

События клавиатуры



- keypress
- keydown
- keyup

- altKey
- ctrlKey
- shiftKey
- charCode
- key
- keyCode
- which

```
element.addEventListener('keydown', function(e) {  
    if(  
        e.ctrlKey &&  
        String.fromCharCode(e.keyCode) == 'W'  
    ) {  
        alert('You pressed Ctrl-W');  
    }  
})
```

- click
- dblclick
- mousedown
- mouseup
- mouseenter
- mouseleave
- mousemove
- mouseover
- mouseout
- wheel

```
element.addEventListener(  
  'mouseover',  
  function(e) {  
    e.target.style.borderColor  
      = 'blue';  
    e.stopPropagation();  
  }  
)
```

- | | |
|------------|-------------------------------|
| • altKey | • clientX, clientY |
| • ctrlKey | • pageX, pageY |
| • shiftKey | • screenX, screenY |
| • button | • detail (сколько раз нажали) |

- drag
- dragstart
- dragend
- dragenter
- dragleave
- dragover
- drop
- scroll

```
<div id="dr" draggable="true">  
  ...  
</div>
```

```
document  
  .getElementById('dr')  
  .addEventListener(  
    'dragstart',  
    e => {this.style.opacity = '0.4';}  
  )  
;
```

- События прикосновения
 - touchstart
 - touchend
 - touchmove
 - touchcancel
- События анимации
- События мультимедиа

- Создание документов HTML5
- Создание форм HTML5 для запроса и обработки информации
- Валидация форм HTML5
- Обработка событий HTML5 функциями JavaScript
- Управление элементами HTML5 через DOM
- Практика: Написание кода JavaScript для изменения элементов документа

<form>

<input type="...">

<select>

<textarea>

<button>

<datalist>

```
<form>
  Имя:<br>
  <input type="text" name="username"><br>
  Роль:<br>
  <input type="text" name="role" list="roles">
  <datalist id="roles">
    <option value="Администратор">
    <option value="Пользователь">
    <option value="Гость">
  </datalist>
  <br>
  Пароль:<br>
  <input type="password" name="password">
</form>
```



```
<form>
  Имя:<br>
  <input type="text" name="name"><br>
  <input type="radio" name="gender" value="M" checked>
  М<br>
  <input type="radio" name="gender" value="F">
  Ж<br>
  <br>
  <input type="checkbox" name="agree" value="yes">
  Я согласен<br>
  <input type="button" onclick="alert('Нажали!')"
  value="Нажмите">
  <input type="reset" value="Сброс">
  <input type="submit" value="OK">
</form>
```

Типы <input> в HTML5



color

date

datetime-local

email

month

number

range

search

tel

time

url

week

```
<form>
  Имя:<br>
  <input type="text" name="name"><br>
  <input type="radio" name="gender" value="M" checked>
  М<br>
  <input type="radio" name="gender" value="F">
  Ж<br>
  <br>
  <input type="checkbox" name="agree" value="yes">
  Я согласен<br>
  <input type="button" onclick="alert('Нажали!')"
  value="Нажмите">
  <input type="reset" value="Сброс">
  <input type="submit" value="OK">
</form>
```

```
<form>
```

Вы узнали о нас

```
<select name="source">
```

```
  <option value="ad">Из рекламы</option>
```

```
  <option value="ad" selected>От друзей</option>
```

```
  <option value="ad">Другое</option>
```

```
</select>
```

Комментарии

```
<textarea name="comments"></textarea>
```

```
<input type="submit" value="OK">
```

```
</form>
```

target
method
enctype

autocomplete
novalidate

```
<form
  method="post"
  target="http://localhost/index.html"
  autocomplete="on"
  novalidate
>
  <input type="text" name="name"><br>
  <input type="radio" name="gender" value="M" checked>
  <input type="submit" value="OK">
</form>
```

Атрибуты <input>

JW

value	autofocus	autocomplete
readonly	height, width	form
disabled	list	formaction
size	min, max	formenctype
maxlength	multiple	formmethod
	pattern (regexp)	formnovalidate
	placeholder	formtarget
	required	
	step	

```
<input type="number"  
  step="5" min="10" max="100" value="30"  
>
```

<meta>

<title>

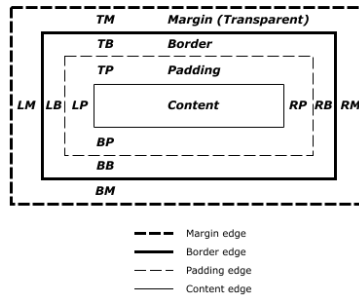
<style>

<script>

<link>

- Блочные

- div
- p, h1 ... h6
- dl (dt, dd)
- ul, ol (li)
- hr
- form



- Линейные

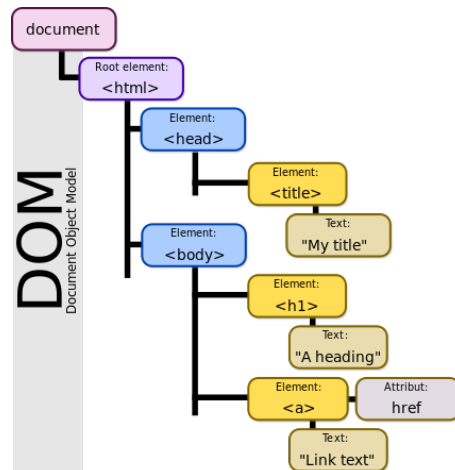
- span
- a
- br
- img
- object

- Табличные

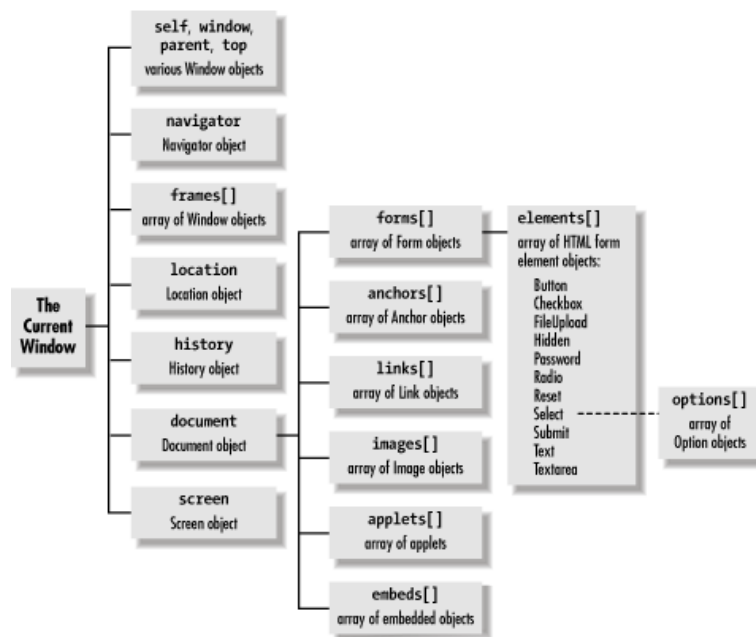
- table
- caption
- thead, tbody, tfoot
- col, colgroup
- tr, th, td

<https://www.w3schools.com/tags/default.asp>

window.document



Level 0 DOM



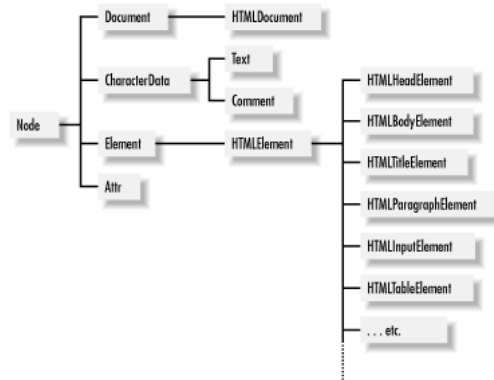
- Level 0 – до стандартизации
- Level 1 – 1988, 2000
- Level 2 – 2000, 2003 (HTML)
 - getElementById
 - модель событий
 - XML namespace
 - CSS
- Level 3 – 2003-2004
 - XPath
 - события клавиатуры
 - сериализация как XML
- DOM 4 – проект

Node

- nodeType
- nodeValue
- appendChild
- replaceChild
- removeChild
- insertBefore
- childNodes
- firstChild
- lastChild
- nextSibling
- previousSibling
- parentNode

Типы узлов

- Документ
- Фрагмент документа
- Элемент
- Атрибут
- Текстовый узел
- Комментарий



Node.DOCUMENT_NODE
Node.DOCUMENT_FRAGMENT_NODE
Node.ELEMENT_NODE

Node.ATTRIBUTE_NODE
Node.TEXT_NODE
Node.COMMENT_NODE

Полифилл (polyfill)

```
if (document.documentElement.firstChild ===
undefined) {

  Element.prototype.firstChild = function() {
    var el = this.firstChild;
    do {
      if (el.nodeType === Node.ELEMENT_NODE) {
        return el;
      }
      el = el.nextSibling;
    } while (el);
    return null;
  }
};
```

```
<script
  src="//cdn.polyfill.io/v1/polyfill.js?features=es6"
></script>
```

```
function displayIds (node) {  
  if (node.id !== '') {  
    console.log (node.id);  
  }  
  for (  
    let i=0;  
    i<node.childNodes.length;  
    i++  
  ) {  
    displayIds (node.childNodes[i]);  
  }  
}  
  
displayIds (document);
```

getElementById или id

```
<div id="content-holder">
  <div id="content">Элемент</div>
</div>

<script>
  console.log(content);
  console.log(window['content-holder']);
</script>
```

```
<div id="content">Выделим этот элемент</div>

<script>
  var elem = document.getElementById('content');
  elem.style.background = 'red';
  console.log(elem == content); // true
  content.style.background = ""; // тот же элемент
</script>
```


element.getElementsByTagName

```
var tableElem = document.getElementById('age-table');  
var elems = tableElem.getElementsByTagName('input');  
var allElems = tableElem.getElementsByTagName('*');
```

element.getElementsByClassName (кроме IE8-)

```
<div class="article">Статья</div>  
<div class="long article">Длинная статья</div>  
  
<script>  
  var articles =  
document.getElementsByClassName('article');  
  alert( articles.length ); // 2  
</script>
```

element.querySelectorAll

```
var elements = document.querySelectorAll(':hover');
```

element.querySelector

```
document.querySelector('#list > li:last-child').  
  style.display = 'none';
```

element.matches

```
if(elem.matches('a[href$=".zip"]'))  
  elem.title = 'Скачать архив';
```

element.closest

```
document.querySelector('td').closest('tr').  
  style.backgroundColor = '#55f';
```

element.querySelectorAll

```
var elements = document.querySelectorAll(':hover');
```

element.querySelector

```
document.querySelector('#list > li:last-child').  
style.display = 'none';
```

element.matches

```
if(elem.matches('a[href$=".zip"]'))  
    elem.title = 'Скачать архив';
```

element.closest

```
document.querySelector('td').closest('tr').  
style.backgroundColor = '#55f';
```

Модификация документа

JW

```
var div = document.createElement('div');
div.className = "alert alert-success";
div.innerHTML = "<strong>Внимание!</strong>";
document.body.insertBefore(
    div, document.body.firstChild
);
let text = "Важное сообщение.";
div.appendChild(document.createTextNode(text));
```

- createElement(tag)
- createTextNode(value)
- cloneNode(deep)

- innerHTML

- appendChild(elem)
- parent.insertBefore(elem, nextSibling)
- removeChild(elem)
- replaceChild(newElem, elem)

Атрибуты элемента



```
var int = document.getElementById('first-name');  
int.className = "form-input";  
console.log(int.tagName);  
int.setAttribute ('max', '1000');
```

| | | |
|-----------------------|-----------|---------------|
| getAttribute (...) | tagName | type |
| setAttribute (...) | id | name |
| removeAttribute (...) | className | value |
| hasAttribute (...) | style | checked |
| | title | selectedIndex |



- Слабые связи в коде
- Гибкость, простота адаптации
- Бесшовная связь с сервером через события



- Трудно отлаживать
- Трудно сопровождать

Событийно-ориентированное программирование



```
var Foo = function() {  
    this.lorem = function()  
    {  
        alert("HELLO FOO");  
    }  
};
```

```
var Bar = function() {  
    var foo = new Foo();  
    foo.lorem();  
};
```

```
var Foo = function(){  
    myEventSystem.on("lorem.event", lorem);  
    function lorem()  
    {  
        alert("HELLO FOO");  
    }  
}
```

```
var Bar = function(){  
    var foo = new Foo();  
    myEventSystem.send("lorem.event");  
};
```



```
<input
  id="b1"
  type="button"
  value="Click me"
  onclick="alert('Thanks!');"
/>
```

```
<input id="myElement" type="button" value="Жми"/>
<script>
  document.getElementById('myElement').onclick =
  function() {
    alert('Thanks');
  }
</script>
```

```
var element = document.getElementById("myElement");  
var handler = function(){ alert('Thanks'); }
```

```
element.addEventListener("click", handler);
```

```
element.removeEventListener("click", handler);
```

- target
- currentTarget
- type
- timestamp
- preventDefault()
- stopPropagation()
- stopImmediatePropagation()

```
addEventListener(  
  "click",  
  e=>  
    alert(e.target.innerHTML);  
);
```

- online
- offline
- error
- open
- popstate
- load
- unload
- beforeunload

```
addEventListener(  
  "beforeunload",  
  ()=>confirm('Вы уверены?')  
);
```

- DomContentLoaded

```
document.addEventListener(  
  "DomContentLoaded",  
  function() {  
    document.body.addEventListener('click', ...);  
  }  
);
```

- submit
- reset
- focus
- blur
- change
- input
- select

```
function areYouSure(e) {  
    if (!confirm('Are you sure?')) {  
        e.preventDefault();           //return false;  
    }  
}  
  
myForm.addEventListener("submit", areYouSure);
```

- keypress
- keydown
- keyup
- altKey
- ctrlKey
- shiftKey
- charCode
- key
- keyCode
- which

```
element.addEventListener('keydown', function(e) {  
    if(  
        e.ctrlKey &&  
        String.fromCharCode(e.keyCode) == 'W'  
    ) {  
        alert('You pressed Ctrl-W');  
    }  
})
```

- click
- dblclick
- mousedown
- mouseup
- mouseenter
- mouseleave
- mousemove
- mouseover
- mouseout
- wheel

```
element.addEventListener(  
  'mouseover',  
  function(e) {  
    e.target.style.borderColor  
      = 'blue';  
    e.stopPropagation();  
  }  
)
```

- | | |
|------------|-------------------------------|
| • altKey | • clientX, clientY |
| • ctrlKey | • pageX, pageY |
| • shiftKey | • screenX, screenY |
| • button | • detail (сколько раз нажали) |

- drag
- dragstart
- dragend
- dragenter
- dragleave
- dragover
- drop
- scroll

```
<div id="dr" draggable="true">
...
</div>
```

```
document
  .getElementById('dr')
  .addEventListener(
    'dragstart',
    e => {this.style.opacity = '0.4';}
  )
;
```

- События прикосновения
 - touchstart
 - touchend
 - touchmove
 - touchcancel
- События анимации
- События мультимедиа