



HTML Basics

AGENDA

- 1 HyperText Markup Language
- 2 Tags
- 3 Validation
- 4 Elements with no semantic meaning
- 5 Semantic Tags
- 6 Html5. API

HyperText Markup Language - Features

- Web page is a web document that is suitable for the World Wide Web and the web browser
- HTML or HyperText Markup Language is the standard markup language used to create web pages
- HTML is written in the form of HTML elements consisting of tags enclosed in angle brackets (like `<html>`).
- HTML describes the structure of a website semantically along with cues for presentation, making it a markup language rather than a programming language

Main points

- Markup language not equal programming language
- Hypertext is text displayed on a computer display or other electronic devices with references

Web-page structure

<html> -

This is a container that includes the entire page content. There is always two sections on the page : head and body.

<head> -

contains information about the HTML file: the name of the page, style, meta tags and additional information.

<meta> -

meta tags that contain information for browsing and retrieval systems (charset, description, keywords).

<body> -

contains all the text and tags that are displayed on the page.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-
width, initial-scale=1.0">
  <title>Document</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  ...
  <script src="scripts/script.js"></script>
</body>
</html>
```

Tags

- Opening and closing tags:

```
<tag>Some text</tag>
```

- Single (empty) tag:

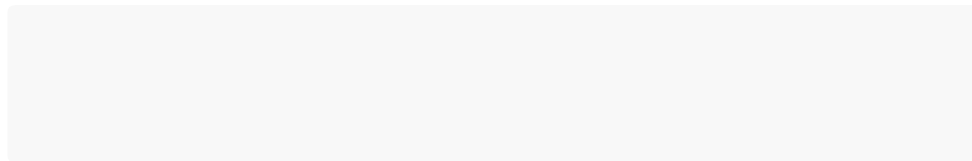
```
<tag>
```

- Parent and child tags:

```
<first-tag>  
  <second-tag>Some text</second-tag>  
</first-tag>
```

- Comment:

```
<!-- Commentcontent -->
```



Attributes

Example:

```
<tag attribute1="value" attribute2="value"> ... </tag>
```

- Extend the capabilities of individual tags. The
- order of the attributes doesn't matter.
- Examples: id, class, name, title, style, src, type

<!DOCTYPE>

Element **<!DOCTYPE>** is intended to indicate the type of the document - DTD (document type definition), for the browser to know how to parse the page.

There are several versions HTML i XHTML (eXtensible HyperText Markup Language), as well as HTML5, which differ in syntax.:

The **<!DOCTYPE>** declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

Validation

Validator checks the markup validity of Web documents in HTML validator.w3.org

The screenshot shows the W3C Markup Validation Service interface. At the top, there's a blue header with the W3C logo and the text "Markup Validation Service" and "Check the markup (HTML, XHTML, ...) of Web documents". Below the header, there are three tabs: "Validate by URI", "Validate by File Upload", and "Validate by Direct Input". The "Validate by Direct Input" tab is selected. Under this tab, there's a section titled "Validate by direct input" with a sub-label "Enter the Markup to validate:" and a large text area for input. Below the text area, there's a "More Options" section with a dropdown arrow. This section contains several options: "Validate Full Document" (selected), "Use Doctype:" (HTML5 (experimental)), "Only if Doctype is missing" (checkbox), "Validate HTML fragment" (radio), "Use Doctype:" (HTML 4.01, XHTML 1.0), "List Messages Sequentially" (selected), "Group Error Messages by Type" (radio), "Show Source" (checkbox), "Clean up Markup with HTML-Tidy" (checkbox), "Show Outline" (checkbox), "Validate error pages" (checkbox), and "Verbose Output" (checkbox). At the bottom right of the form is a "Check" button.

Block and inline element

block :

- occupy all available width of container, height of these component is determined by its content, and it always starts on a new line.

- contain an entire **large** region of **content**

- examples: <h1>, <h2>, <p>, <nav>, <article>, <form>

- the **browser** **places** a **margin** of **whitespace** between **block** elements for separation

inline :

- are directly part of another element, such as a text paragraph. They are mainly used to change the appearance of text or a logical selection.

- elements** **affect** a **small** amount of **content**
- examples: <a>, <label>, , <code>, <q>, <cite>, <big>, <small>, , , <i>, ,
, <hr>

- the browser allows **many** **inline** elements to **appear** on the **same** **line**

- must be **nested** **inside** a **block** element

<https://developer.mozilla.org/en-US/docs/Web/HTML>

Lists

Ordered list:

```
<ol>
  <li>Item first</li>
  <li>Item second</li>
</ol>
```

Demo:

1. Item first
2. Item second

Unordered list:

```
<ul>
  <li>Item 1</li>
  <li>Item 2
    <ul>
      <li>Item 2.1</li>
      <li>Item 2.2</li>
    </ul>
  </li>
</ul>
```

Demo:

- Item 1
- Item 2
 - Item 2.1
 - Item 2.2

More about ordered list. Attributes

reversed This Boolean attribute specifies that the items of the list are specified in reversed order.

start This integer attribute specifies the start value for numbering the individual list items.

type Indicates the numbering type.

```
<ol type="I" start="20" reversed>  
  <li>first item</li>  
  <li>second item</li>  
  <li>third item</li>  
</ol>
```

XX. first item

XIX. second item

XVIII. third item

```
<div id="links">  
  <a href="#">Cat</a>  
  <a href="#">Dog</a>  
  <a href="#">Unicorn</a>  
</div>
```



```
<ul class="menu">  
  <li><a href="#">Cat</a></li>  
  <li><a href="#">Dog</a></li>  
  <li><a href="#">Unicorn</a></li>  
</ul>
```

Lists

Definitions list:

```
<dl>
  <dt>Title 1</dt>
  <dd>Definition</dd>
  <dt>Title 2</dt>
  <dd>Definition</dd>
  <dd>Definition</dd>
</dl>
```

Title 1
Definition
Title 2
Definition
Definition

Elements of the page

Images

```

```

Links:

```
<a href="http://www.epam.com/">Go to Epam</a>
```

Demo:

[Go to Epam](http://www.epam.com/)

Image inside a link:

```
<a href="http://www.epam.com/">  
    
</a>
```

Demo:



More about images

- inserts a graphical image into the page (inline)
- the **src** attribute specifies the image URL
 - **Relative** URL is a partial URL specified in relation to some existing URL:
src="images/gollum.jpg"
 - **Absolute** URL is a complete URL to a web source:
src="http://i.imdb.com/images/nb15/logo2.gif"
- HTML5 also **requires an alt attribute** describing the image
 - If the browser is unable to fetch the image, it will show the **alt** text
- if placed in an **anchor**, the image **becomes a link**
- Other attributes are **width** and **height**: their value can be given as pixels or percentage of window.
 - If not used, the image will be shown in its actual size

More about tag `<a>`. Attributes

`href` elements must have a value that is a valid URL .

`target` if present, must be a valid browsing context name or keyword. It gives the name of the browsing context that will be used..

`type` The type attribute, if present, gives the MIME type of the linked resource. The value must be a valid mime type.

`download` if present, indicates that the author intends the hyperlink to be used for downloading a resource

More about tag <a>. Attribute target

The target attribute specifies where to open the linked document

<code>_blank</code>	Opens the linked document in a new window or tab
<code>_self</code>	Opens the linked document in the same frame as it was clicked (this is default)
<code>_parent</code>	Opens the linked document in the parent frame
<code>_top</code>	Opens the linked document in the full body of the window
<code>framename</code>	Opens the linked document in a named frame

```
<a href="https://www.example.com" target="_blank">example</a>
```

More about tag <a>. Attribute rel


Specifies the relationship between the current document and the linked document

Value	Description
alternate	Provides a link to an alternate representation of the document (i.e. print page, translated or mirror)
author	Provides a link to the author of the document
bookmark	Permanent URL used for bookmarking
next	Provides a link to the next document in the series
nofollow	Links to an unendorsed document, like a paid link. ("nofollow" is used by Google, to specify that the Google search spider should not follow that link)
noreferrer	Requires that the browser should not send an HTTP referer header if the user follows the hyperlink
noopener	Requires that any browsing context created by following the hyperlink must not have an opener browsing context
prev	The previous document in a selection
search	Links to a search tool for the document
tag	A tag (keyword) for the current document

<https://quasi-art.ru/library/it/use-rel-noopener>

Anchor

hash mark (#), specifies an internal target location (an ID of an HTML element) within the current document. URLs are not restricted to Web (HTTP)-based documents, but can use any protocol supported by the browser.



```
<a id="top">logo</a>
```



```
<a href="#top">go to top</a>
```



Uniform Resource Locator (URL)

Relative Paths

index.html

/graphics/image.png

/help/articles/how-do-i-set-up-a-webpage.html

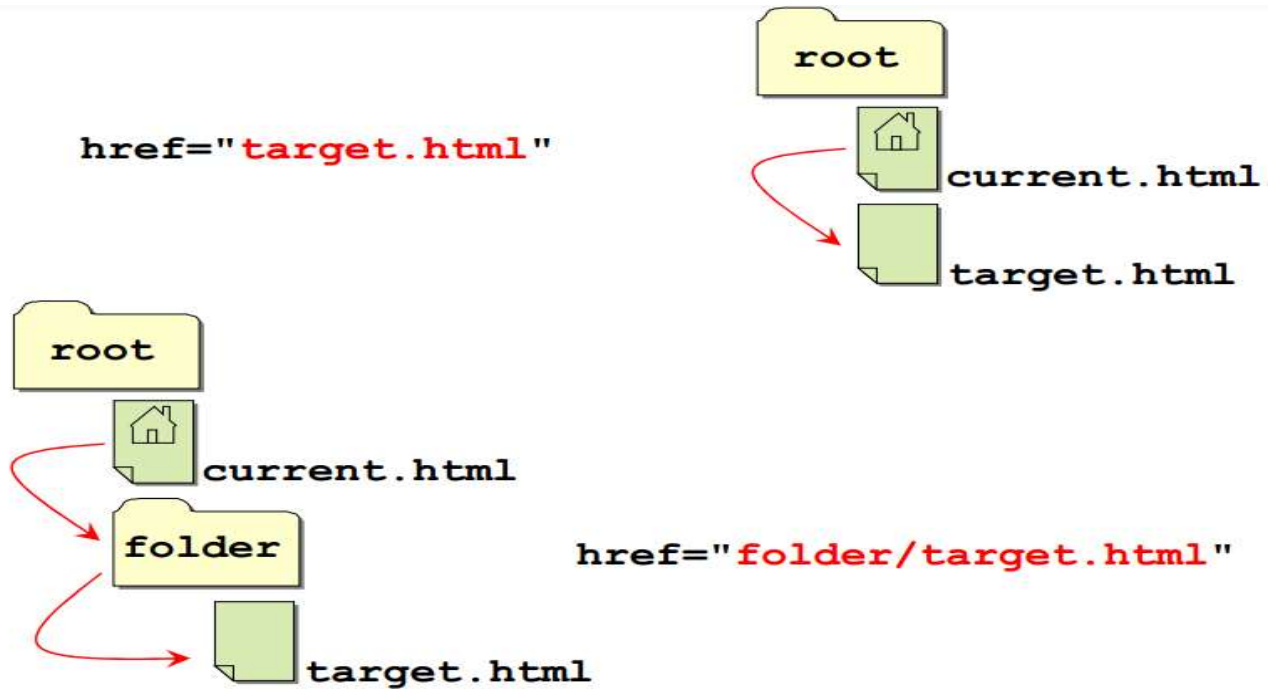
Absolute Paths

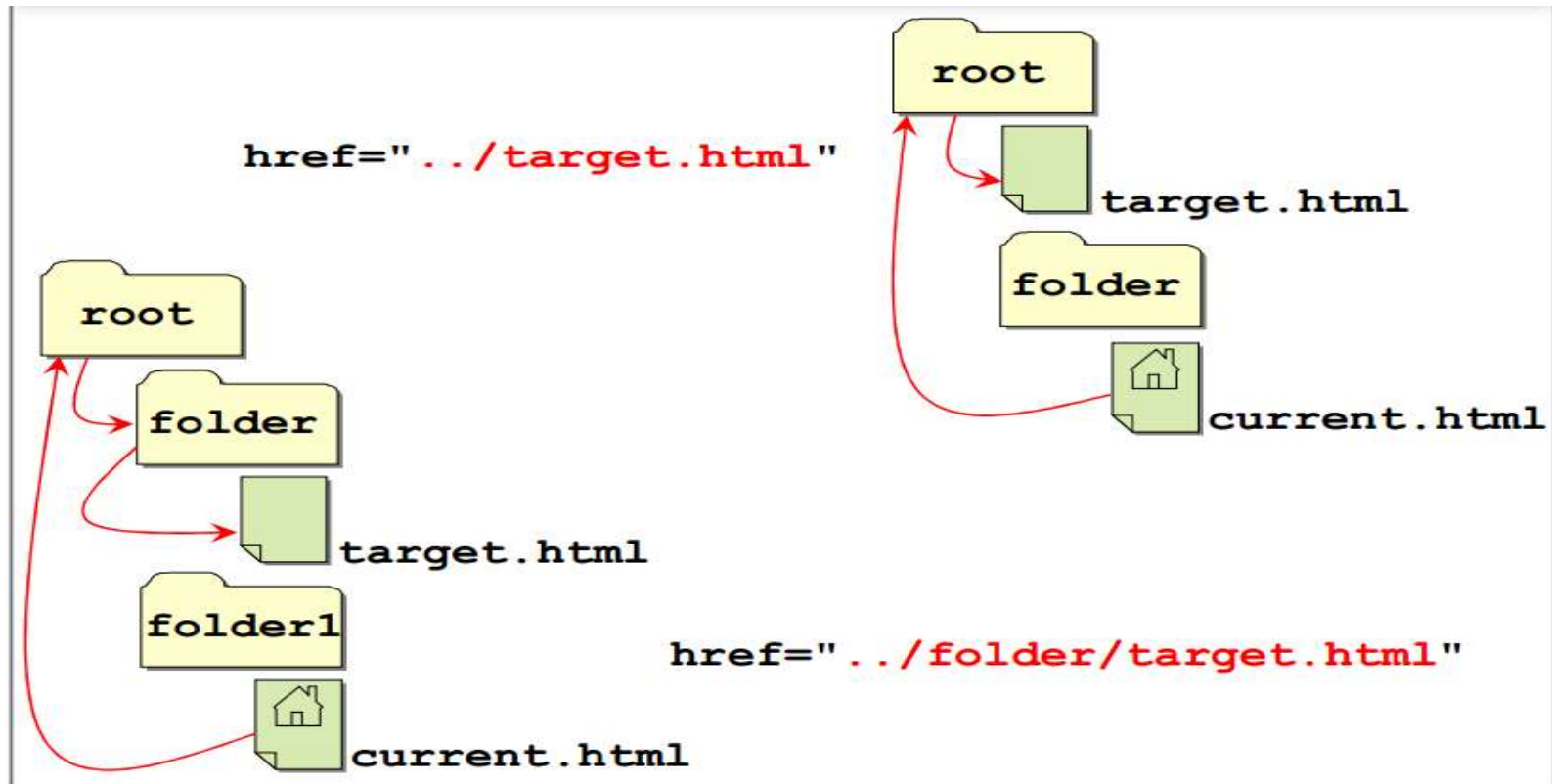
http://www.mysite.com

http://www.mysite.com/graphics/image.png

http://www.mysite.com/help/articles/how-do-i-set-up-a-webpage.html

Relative links





email link, skype link, phone link

```
<a href="tel: +04951234567">+0 (495) 123-45-67</a>
```

```
<a href="mailto: example@mail.com">example@mail.ru</a>
```

```
<a href="skype: someskye?call">someskye</a>
```



Details about the tag ``

`` represents an image in the document.

JPG Images



GIF Images



PNG Images

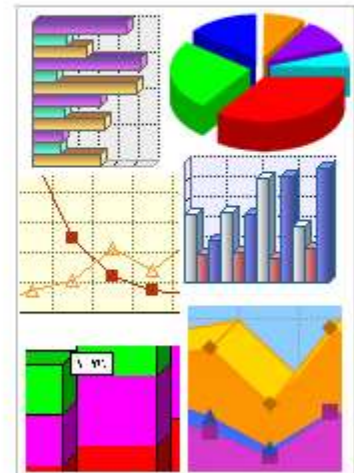


Image file

+

-

Jpeg

Full-color information

Great for photographs, banner
etc.

Cannot be animation

Doesn't support transparency

Gif

support transparency

can be animated

only supports 256 colors

file size can be quite large

Png

Great for logos

Great for text images

Cannot be animation

Not great for large images

alt

Provide alternative contents for multimedia.

```

```



```

```



Tables

`<table>`

It is used as a container for elements that define the contents of the table.

Each table row is defined with the `<tr>` tag. A table header is defined with the `<th>` tag. By default, table headings are bold and centered. A table data/cell is defined with the `<td>` tag.

colspan attribute is used to combine neighboring columns in the table

rowspan - to combine rows

Inside `<table>` you can use only the following elements:

`<caption>`, `<col>`, `<colgroup>`, `<tbody>`, `<td>`, `<tfoot>`, `<th>`, `<thead>` and `<tr>`

For common column styles we can specify the information once, on a `<col>` element.

`<col>` elements are specified inside a `<colgroup>` container just below the opening `<table>` tag.

Tables

```
<table>
  <caption>Prices for buses</caption>
  <thead>
    <tr>
      <th>Bus number</th>
      <th>Price</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>3A</td>
      <td rowspan="2">3 hrn</td>
    </tr>
    <tr>
      <td>25</td>
    </tr>
    <tr>
      <td>133</td>
      <td>4 hrn</td>
    </tr>
  </tbody>
</table>
```

Demo:

Prices for buses

Bus number	Price
3A	3 hrn
25	
133	4 hrn

tbody

<p>Table with thead, tfoot, and tbody</p>

```
<table>
  <thead>
    <tr>
      <th>Header content 1</th>
      <th>Header content 2</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Body content 1</td>
      <td>Body content 2</td>
    </tr>
  </tbody>
  <tfoot>
    <tr>
      <td>Footer content 1</td>
      <td>Footer content 2</td>
    </tr>
  </tfoot>
</table>
```

The <thead> tag is used to group header content in an HTML table.

The <tbody> tag is used to group the body content in an HTML table.

The <tbody> tag must be used in the following context: As a child of a <table> element, after any <caption>, <colgroup>, and <thead> elements.

The <tfoot> tag is used to group footer content in an HTML table.

caption

The <caption> tag defines a table caption.

The <caption> tag must be inserted immediately after the <table> tag.

```
<table>
  <caption>Monthly savings</caption>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
</table>
```

Monthly savings

Month Savings

January \$100

columns

The `<colgroup>` tag specifies a group of one or more columns in a table for formatting. The `<colgroup>` tag must be a child of a `<table>` element, after any `<caption>` elements and before any `<thead>`, `<tbody>`, `<tfoot>`, and `<tr>` elements. The `<col>` tag specifies column properties for each column within a `<colgroup>` element. The `<col>` tag is useful for applying styles to entire columns, instead of repeating the styles for each cell, for each row.

ISBN	Title	Price
3476896	My first HTML	\$53

```
<table>
  <colgroup>
    <col span="2">
    <col>
  </colgroup>
  <tr>
    <th>ISBN</th>
    <th>Title</th>
    <th>Price</th>
  </tr>
  <tr>
    <td>3476896</td>
    <td>My first HTML</td>
    <td>$53</td>
  </tr>
</table>
```


colspan, rowspan

The colspan attribute defines the number of columns a cell should span

```
<table>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>February</td>
    <td>$80</td>
  </tr>
  <tr>
    <td colspan="2">Sum: $180</td>
  </tr>
</table>
```

Month	Savings
February	\$80
Sum: \$180	

The rowspan attribute specifies the number of rows a cell should span.

```
<table>
  <tr>
    <th>Month</th>
    <th>Savings</th>
    <th rowspan="2">Savings for holiday!
  </th>
</tr>
  <tr>
    <td>February</td>
    <td>$80</td>
  </tr>
</table>
```

Month	Savings	Savings for holiday!
February	\$80	

Elements with no semantic meaning

These tags are designed to highlight the portion of the document to change the appearance of the content but without creating additional semantic value fragment.

`<div>` - block element

`` - inline element

Special Characters and Signs

Many mathematical, technical, and currency symbols, are not presented on a normal keyboard. To add these symbols to an HTML page, you can use an HTML entity name.

If no entity name exists, you can use an entity number; a decimal (or hexadecimal) reference.

```
<p>Euro symbol: &euro; &#8364; &#x20AC;</p>
```

```
<p>&copy; &trade; &laquo; &raquo; &amp; &nbsp; &lt; &gt; &ndash; &mdash;</p>
```

Demo:

Euro symbol: € € €

© ™ « » & < > — —

Semantic Tags

<article>

<section>

<aside>

<nav>

<figure>

<figcaption>

<header>

<footer>

<mark>

<audio>

<video>

<source>

<canvas>

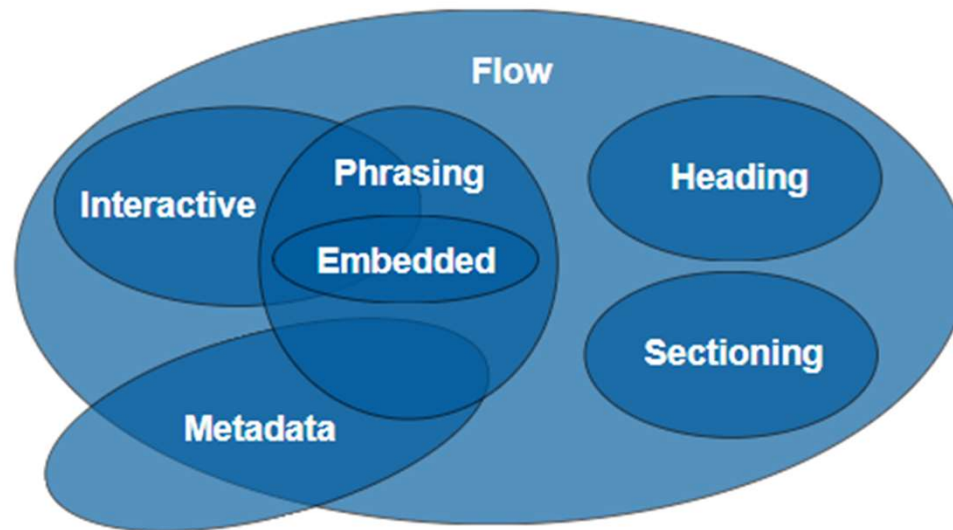
<svg>

<datalist>

<progress>

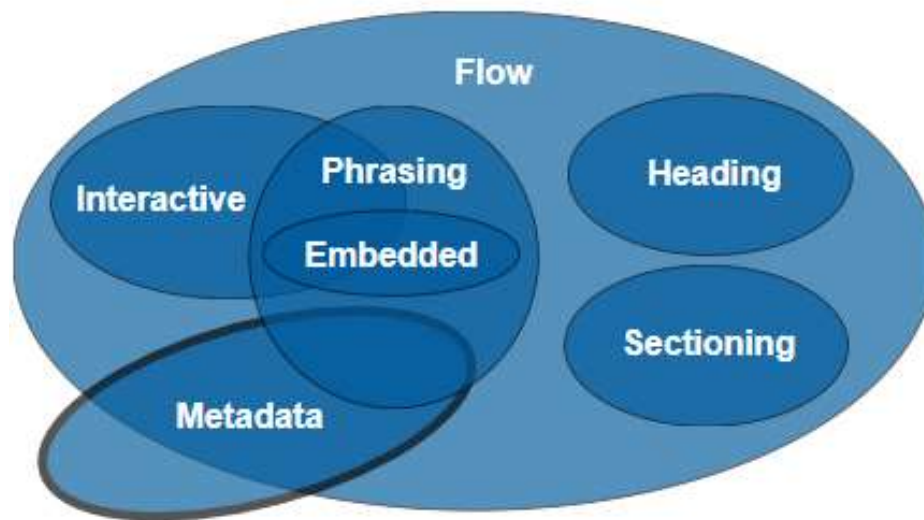
kinds of content

- ☐ metadata content
- ☐ flow content
- ☐ sectioning content
- ☐ heading content
- ☐ phrasing content
- ☐ embedded content
- ☐ interactive content



<https://www.w3.org/TR/2011/WD-html5-20110525/content-models.html>

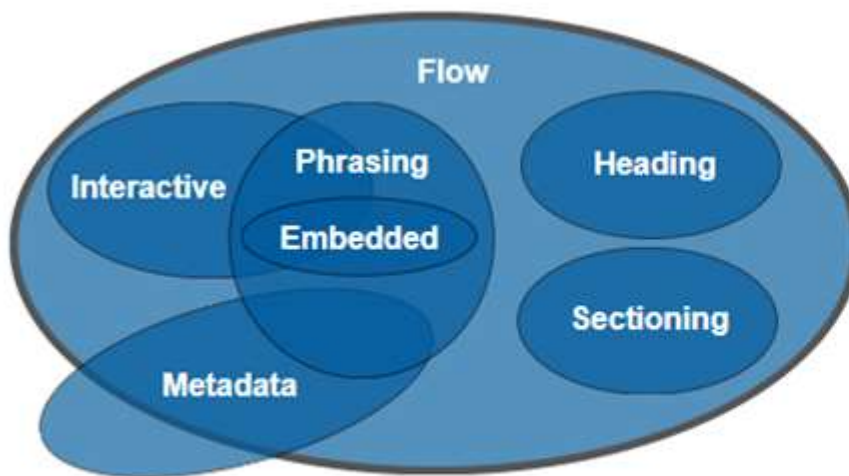
metadata content



Metadata content

`<base>`, `<meta>`, `<noscript>`,
`<script>`, `<style>`, `<title>`.

flow content



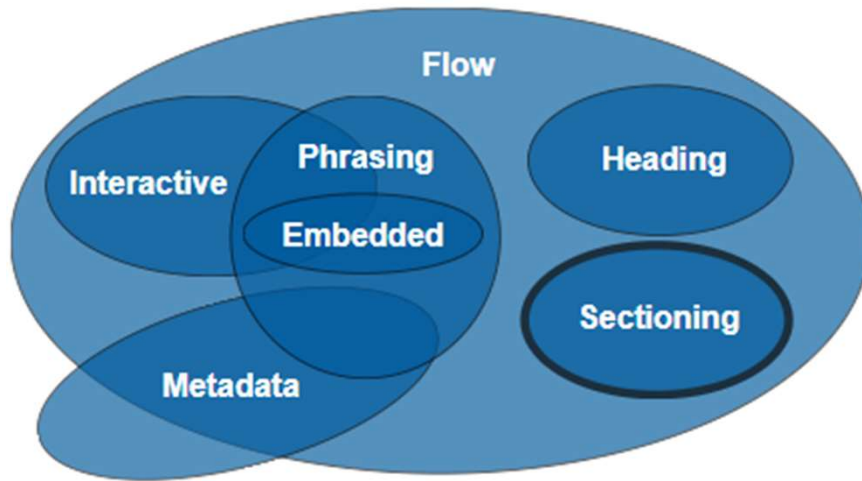
Flow content

`<a>`, `<abbr>`, `<address>`, `<article>`, `<aside>`, `<audio>`, ``, `<bdo>`, `<bdi>`, `<blockquote>`, `
`, `<button>`, `<canvas>`, `<cite>`, `<code>`, `<command>`, `<data>`, `<datalist>`, ``, `<details>`, `<dfn>`, `<div>`, `<dl>`, ``, `<embed>`, `<fieldset>`, `<figure>`, `<footer>`, `<form>`, `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>`, `<h6>`, `<header>`, `<hgroup>`, `<hr>`, `<i>`, `<iframe>`, ``, `<input>`, `<ins>`, `<kbd>`, `<keygen>`, `<label>`, `<main>`, `<map>`, `<mark>`, `<math>`, `<menu>`, `<meter>`, `<nav>`, `<noscript>`, `<object>`, ``, `<output>`, `<p>`, `<picture>`, `<pre>`, `<progress>`, `<q>`, `<ruby>`, `<s>`, `<samp>`, `<script>`, `<section>`, `<select>`, `<small>`, ``, ``, `<sub>`, `<sup>`, `<svg>`, `<table>`, `<template>`, `<textarea>`, `<time>`, ``, `<var>`, `<video>`, `<wbr>` and Text.

A few other elements belong to this category, but only if a specific condition is fulfilled:

- `<area>`, if it is a descendant of a `<map>` element
- `<link>`, if the `itemprop` attribute is present
- `<meta>`, if the `itemprop` attribute is present
- `<style>`, if the `scoped` attribute is present

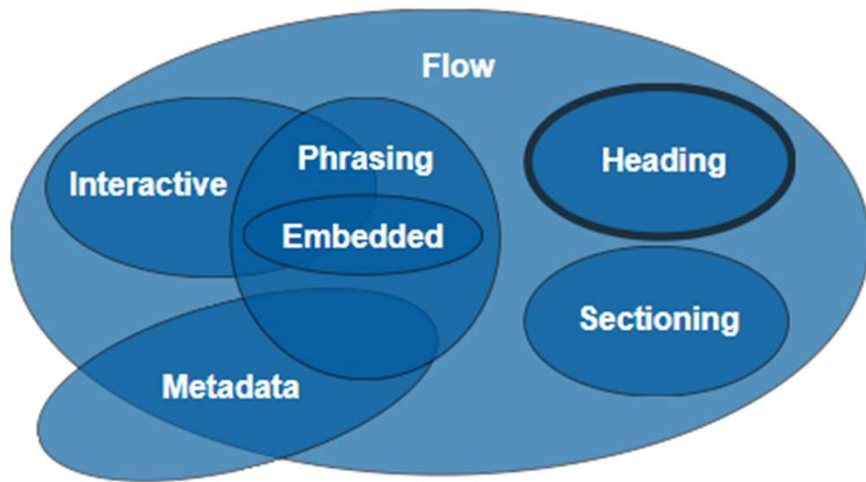
sectioning content



Sectioning content

`article, aside, nav, section`

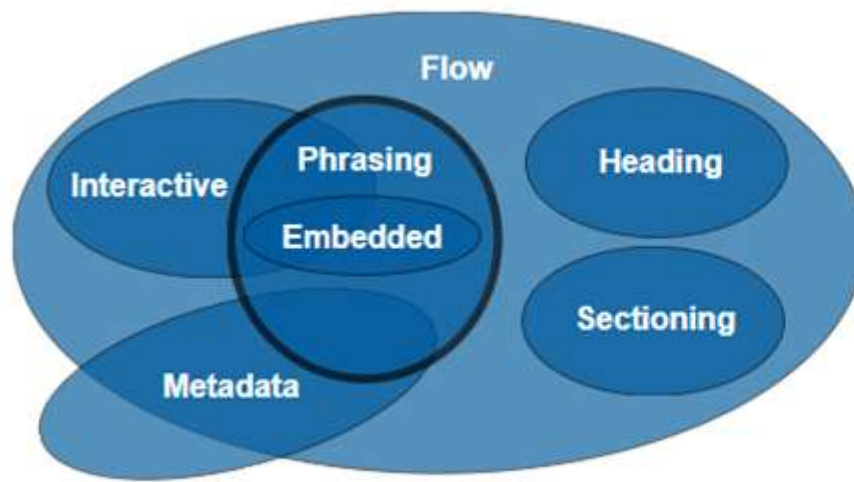
heading content



Heading content

h1, h2, h3, h4, h5, h6

phrasing content



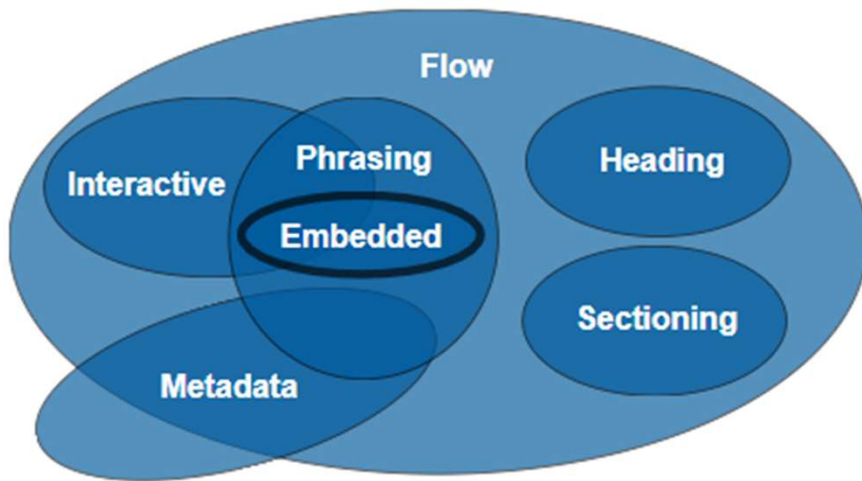
Phrasing content

`<abbr>`, `<audio>`, ``, `<bdo>`, `
`, `<button>`, `<canvas>`, `<cite>`, `<code>`,
`<command>`, `<data>`, `<datalist>`, `<dfn>`, ``, `<embed>`, `<i>`, `<iframe>`,
``, `<input>`, `<kbd>`, `<keygen>`, `<label>`, `<mark>`, `<math>`,
`<meter>`, `<noscript>`, `<object>`, `<output>`, `<picture>`, `<progress>`, `<q>`, `<ruby>`,
`<samp>`, `<script>`, `<select>`, `<small>`, ``, ``, `<sub>`, `<sup>`, `<svg>`,
`<textarea>`, `<time>`, `<var>`, `<video>`, `<wbr>` and plain text (not only consisting of white spaces characters).

A few other elements belong to this category, but only if a specific condition is fulfilled:

- `<a>`, if it contains only phrasing content
- `<area>`, if it is a descendant of a `<map>` element
- ``, if it contains only phrasing content
- `<ins>`, if it contains only phrasing content
- `<link>`, if the `itemprop` attribute is present
- `<map>`, if it contains only phrasing content
- `<meta>`, if the `itemprop` attribute is present

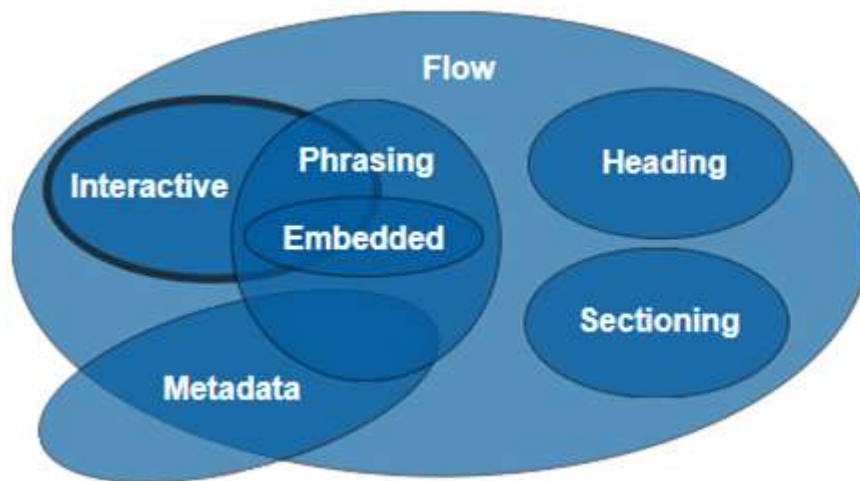
embedded content



Embedded content

audio, canvas, embed, iframe, img, math, object, svg, video

interactive content



Interactive content

`<a>`, `<button>`, `<details>`, `<embed>`, `<iframe>`,
🔊 `<keygen>`, `<label>`, `<select>`, and `<textarea>`.

Some elements belong to this category only under specific conditions:

- `<audio>`, if the `controls` attribute is present
- ``, if the `usemap` attribute is present
- `<input>`, if the `type` attribute is not in the hidden state
- `<menu>`, if the `type` attribute is in the toolbar state
- `<object>`, if the `usemap` attribute is present
- `<video>`, if the `controls` attribute is present

Section

Represents a generic document or application section. In this context, a section is a thematic grouping of content, typically with a header, possibly with a footer. Examples include chapters in a book, the various tabbed pages in a tabbed dialog box, or the numbered sections of a thesis. A web site's home page could be split into sections for an introduction, news items, contact information.

```
<section>
  <h1>Level 1</h1>
  <section>
    <h1>Level 2</h1>
  </section>
</section>
```

Article

Represents a section of a page that consists of a composition that forms an independent part of a document, page, or site. This could be a forum post, a magazine or newspaper article, a Web log entry, a user-submitted comment, or any other independent item of content

Each `<article>` should be identified, typically by including a heading (`<h1>` - `<h6>` element) as a child of the `<article>` element.

```
<article>
  <header>
    <h4><a href="#comment-2" rel="bookmark">Comment #2</a> by <a
      href="http://example.com/">Jack Osborne</a></h4>
    </header>
    <p>Pellentesque habitant morbi tristique senectus.</p>
</article>
```

Nav

- Represents navigation for a document. The nav element is a section containing links to other documents or to parts within the current document.
- Not all groups of links on a page need to be in a nav element — only groups of primary navigation links. In particular, it is common for footers to have a list of links to various key parts of a site, but the footer element is more appropriate in such cases.

```
<nav>
  <ul>
    <li>Nav Link</li>
    <li>Nav Link</li>
    <li>Nav Link</li>
  </ul>
</nav>
```

Aside

Represents a section of a page consisting of content that is tangentially related to the content around the aside element, and which could be considered separate from that content.

Such sections are often represented as sidebars in printed typography.

```
<aside>
  <h2>Blogroll</h2>
  <ul>
    <li><a href="#">My Friend</a></li>
    <li><a href="#">My Other Friend</a></li>
    <li><a href="#">My Best Friend</a></li>
  </ul>
</aside>
```


Main

- Represents a the dominant content of the **<body>** of a document, portion of a document or application.
- The main content area consists of content that is directly related to or expands upon the central topic of a document, or the central functionality of an application.
- The contain of a **<main>** element should be unique to the document

```
<main>
  <h1>Level 1</h1>
  <section>
    <h2>Level 2</h2>
  </section>
</main>
```

Header

Represents the "header" of a document or section of a document. The header element is typically used to group a set of h1–h6 elements to mark up a page's title with its subtitle or tagline.

header elements may, however, contain more than just the section's headings and subheadings — e.g., version history information or publication date.

```
<header>  
  <h1>Header string</h1>  
  <p>Other content</p>  
</header>
```

Footer

- Represents the "footer" of a document or section of a document. The footer element typically contains metadata about its enclosing section, such as who wrote it, links to related documents, copyright data, etc.
- When the footer element contains entire sections, they represent appendices, indexes, long colophons, verbose license agreements, and other such content.

```
< footer >  
  <h3>Contacts</h3>  
  <p>Other content</p>  
</footer>
```

Figure, Figcaption

The figure element represents some flow content, optionally with a caption, that is self-contained and is typically referenced as a single unit from the main flow of the document.

The figure element can be used to annotate illustrations, diagrams, photos, code listings, etc., that are referenced in the main content of the document, but that could, without affecting the flow of the document, be moved away from that primary content — e.g., to the side of the page, to dedicated pages, or to an appendix.

```
<figure>
  <img ... >(or video, table etc)
  <figcaption>A rabid unicorn goring afairy.</figcaption>
</figure>
```

Mark

Represents a run of text in one document marked or highlighted because of its relevance in another context.

When used in a quotation or other block of text referenced in a document, it indicates a highlight that was not present in the original document — e.g., a portion of text in an academic publication that has recently come under additional scrutiny.

```
<p>In this sentence we will be using the mark element. <mark>HTML5</mark> Can  
you see where it has been used?</p>
```

Demo:

In this sentence we will be using the mark element. **HTML5** you see where it has been used?

Audio

Represents a sound or audio stream.

Content may be nested inside the audio element. User agents should not show this content to the user. Authors should use this content to force older browsers to use a legacy audio plugin or to inform the user of how to access the audio content.

```
<audio src="music.oga" controls>  
  <a href="music.oga">Download song</a>  
</audio>
```



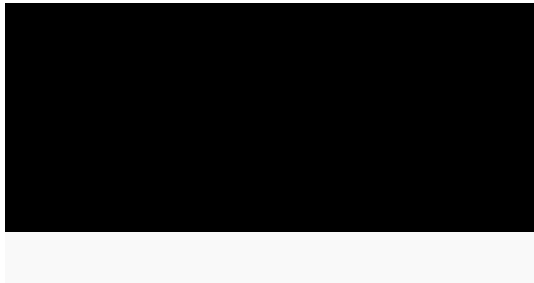
Video

Represents a video or movie.

Content may be nested inside the video element. User agents should not show this content to the user. Authors should use this content to force older browsers to use a legacy video plugin or to inform the user of how to access the video content.

```
<video src="video.ogv" controls width="240" height="200" poster="poster.jpg">  
  <a href="video.ogv">Download Video</a>  
</video>
```

Demo:



CONFIDENTIAL

Forms Introduction

- HTML forms are used to create (rather primitive) GUIs on Web pages
- Usually the purpose is to ask the user for information
- The information is then sent back to the server
- A form is an area that can contain form elements
- The syntax is: `<form> ... form elements ... </form>`
- Form elements include: buttons, checkboxes, text fields, radio buttons, drop-down menus, etc
- Other kinds of HTML tags can be mixed in with the form elements
- A form usually contains a Submit button to send the information in the form elements to the server

The <form> tag

The `<form arguments> ... </form>` tag encloses form elements

The arguments to form tell what to do with the user input:

- **action="url"** specifies where to send the data when the Submit button is clicked
- **method="get"**
 - Form data is sent as a URL with ?form_data info appended to the end;
 - Can be used only if data is all ASCII and not more than 100 characters;
- **method="post"**
 - Form data is sent in the body of the URL request;
 - Cannot be bookmarked by most browsers;
- **target="target"** - Tells where to open the page sent as a result of the request
 - target="_blank" means open in a new window;
 - target="_top" means use the same window.

Methods: GET and POST

GET - Requests data from a specified resource

`/test/demo_form.php?name1=value1&name2=value2`

POST - Submits data to be processed to a specified resource

`POST /test/demo_form.php HTTP/1.1`

`Host: example.com`

`name1=value1&name2=value2`

Compare GET vs. POST

GET	POST
<ul style="list-style-type: none">GET requests can be cachedGET requests remain in the browser historyGET requests can be bookmarkedGET requests should never be used when dealing with sensitive dataGET requests have length restrictionsGET requests should be used only to retrieve data	<ul style="list-style-type: none">POST requests are never cachedPOST requests do not remain in the browser historyPOST requests cannot be bookmarkedPOST requests have no restrictions on data length

More attributes of the `<form>` Element

accept-charset

action

autocomplete

enctype

method

name

novalidate

target

Form Elements

Tag	Description
<form>	Defines an HTML form for user input
<input>	Defines an input control
<textarea>	Defines a multiline input control (text area)
<label>	Defines a label for an <input> element
<fieldset>	Groups related elements in a form
<legend>	Defines a caption for a <fieldset> element
<select>	Defines a drop-down list
<optgroup>	Defines a group of related options in a drop-down list
<option>	Defines an option in a drop-down list
<button>	Defines a clickable button
<datalist>	Specifies a list of pre-defined options for input controls
<output>	Defines the result of a calculation

Input Types

The most important form element is the `<input>` element. This element can be displayed in several ways, depending on the type attribute:

- ❑ `text;`
- ❑ `password;`
- ❑ `submit;`
- ❑ `radio;`
- ❑ `checkbox;`
- ❑ `button;`
- ❑ `color;`
- ❑ `date;`
- ❑ `datetime-local;`
- ❑ `email;`
- ❑ `month;`
- ❑ `number;`
- ❑ `range;`
- ❑ `search;`
- ❑ `tel;`
- ❑ `time;`
- ❑ `url;`
- ❑ `week;`

The <fieldset>, <legend> tag

The fieldset is a useful tool for organizing and grouping related items within a form, and has been used for a long time in desktop applications.

```
<form action="script.php">
  <fieldset>
    <legend>User data</legend>
    <label for="userName">Enter your name:</label>
    <input type="text" id="userName" />
  </fieldset>

  <fieldset>
    <legend>User Skills</legend>
    <label for="userSkills">Enter your skills:</label>
    <input type="text" id="userSkills" />
  </fieldset>
  <input type="submit" name="Submit" value="Send">
</form>
```

Demo:

User data

Enter your name:

User Skills

Enter your skills:

Send

The `<textarea>` tag

The `textarea` element represents a multiline plain text edit control for the element's raw value. The contents of the control represent the control's default value.

```
<textarea cols="20" rows="10"></textarea>
```

Demo:

The <input> tag

The input element represents a typed data field, usually with a form control to allow the user to edit the data.

Most, but not all, form elements use the input tag, with a type="..." argument to tell which kind of element it is. The most used types text, checkbox, radio, password, hidden, submit, reset, button, file, or image.

Input tag arguments:

name: the name of the element

value: the “value” of the element; used in different ways for different values of type

readonly: the value cannot be changed

disabled: the user can't do anything with this element

Text input

A text field:

```
<input type="text" name="textfield" value="Hello">
```

Hello

A password field:

```
<input type="password" name="text2" value="secret">
```

.....

A multi-line text field:

```
<textarea name="textarea" cols="50" rows="3">
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Eveniet, ipsum, officia, doloremque sapiente nesciunt dolore.

```
</textarea>
```

Lorem ipsum dolor sit amet, consectetur
adipisicing elit. Eveniet, ipsum, officia,
doloremque sapiente nesciunt dolore.

Buttons

A submit button - send data:

```
<input type="submit" name="Click me!" value="Submit">
```

A reset button - restore all form elements to their initial state:

```
<input type="reset" name="Submit2" value="Reset">
```

A plain button - take some action as specified by JavaScript:

```
<input type="button" name="Submit3" value="Push Me">
```

Checkboxes

A checkbox:

```
<input type="checkbox" name="checkbox" value="checkbox" checked>
```

Attributes:

- **type**: "checkbox";
- **name**: used to reference this form element from JavaScript;
- **value**: value to be returned when element is checked.

There is no text associated with the checkbox - you have to supply text in the surrounding HTML

Checkboxes

Example 1:

```
<input type="checkbox" name="vehicle1" value="Bike"> I have a bike<br>
<input type="checkbox" name="vehicle2" value="Car"> I have a car<br>
<input type="checkbox" name="vehicle3" value="Boat"> I have a boat<br>
```

Demo:

- ☐ I have a bike
- ☐ I have a car
- ☐ I have a boat

Example 2:

```
<input type="checkbox" name="male" checked="" /> male <br />
<input type="checkbox" name="female" checked="" /> female
```

☒ male
☒ female

Radio buttons

Radio buttons:

```
<input type="radio" name="sex" value="myValue1"> male<br>
<input type="radio" name="sex" value="myValue2" checked> female
```

If two or more radio buttons have the same `name`, the user can only select one of them at a time.

This is how you make a radio button "group".

As with checkboxes, radio buttons do not contain any text.

Demo:

☐ male

☒ female

Drop-down menu

A menu or list:

```
<select name="select">
  <option value="red">red</option>
  <option value="green">green</option>
  <option value="BLUE">blue</option>
</select>
```

Demo:

red ▼

red
green
blue

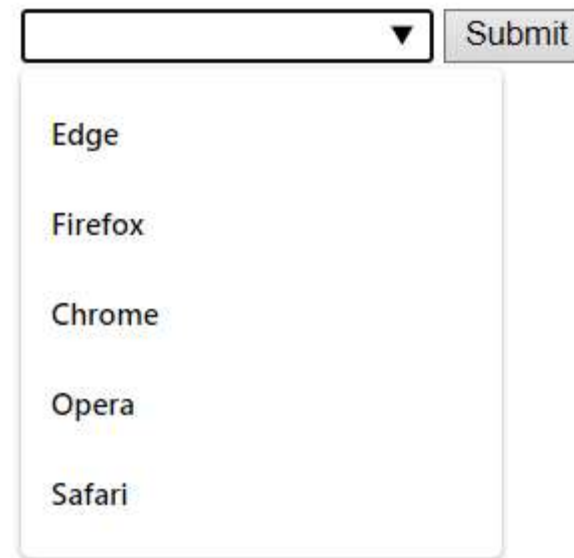
Additional arguments:

- **size**: the number of items visible in the list (default is "1")
- **multiple**: if set to "true", any number of items may be selected (default is "false")

Datalist

The **<datalist>** element specifies a list of pre-defined options for an **<input>** element.

```
<input list="browsers">
  <datalist id="browsers">
    <option value="Edge">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
<input type="submit">
```



The image shows a web form with a dropdown menu and a submit button. The dropdown menu is open, displaying a list of browser names: Edge, Firefox, Chrome, Opera, and Safari. The submit button is labeled "Submit".

The <label> tag (part 1)

Label is used to link the descriptive text for a given form control directly to that control.

It provides a usability improvement for mouse users, because if the user clicks on the text within the element, it toggles the control.

The for attribute of the tag should be equal to the id attribute of the related element to bind them together.

The <label> tag (part 2)

Example 1:

```
<label for="userName">Enter your name:</label>  
<input type="text" name="username" id="userName" value="Author Name" />
```

Demo:

Enter your name:

Example 2:

```
<label>  
  Enter your name:  
  <input type="text" name="username" value="Author Name" />  
</label>
```

Demo:

Enter your name:

Hidden fields

- All input fields are sent back to the server, including hidden fields. This is a way to include information that the user doesn't need to see.
- The value of a hidden field can be set programmatically (by JavaScript) before the form is submitted.

```
<input type="hidden" name="secretMessage" value="hello">
```

HTML5 input types (part 1)

- Tells the browser not to submit the form, unless the user has entered a valid email address:

```
<input type=email>
```

- Causes the browser to ensure that the entered field is a correct URL:

```
<input type=url>
```

- Browsers will display a (popup) calendar widget:

```
<input type=date>
<input type="month" />
<input type="week" />
<input type="time" />
<input type="datetime" />
<input type="datetime-local" />
```

Demo:

Email:

URL:

Date: дд.мм.yyyy

Month: ----- p.

Time: Datetime

local:
дд.мм.yyyy --:--

Ok

HTML5 input types (part 2)

- Throws an error if the user doesn't enter a numeric value. Works with min, max and step attributes:

```
<input type=number>
```

- Renders as a slider:

```
<input type=range>
```

- Expects a search term:

```
<input type=search>
```

- Expects a telephone number. The smartphone brings up a telephone input keyboard:

```
<input type=tel>
```

- Allows the user to input a color value via a picker:

```
<input type=color>
```

Demo:

Number:

Range:

Search

Tel:

Color:

More about Number and Range

The `<input type="number">` defines a **numeric** input field.

```
<form>  
  Quantity (between 1 and 5):  
  <input type="number" name="quantity" min="1" max="5">  
</form>
```

The `<input type="range">` defines a control for entering a number whose exact value is not important (like a slider control). Default range is 0 to 100. However, you can set restrictions on what numbers are accepted with the min, max, and step attributes

```
<form>  
  <input type="range" name="points" min="0" max="10">  
</form>
```



Attributes

- disabled
- max
- maxlength
- **placeholder**
- min
- pattern
- readonly
- tabindex
- required
- size
- step
- value

```
<input type="number" min="10" max="100"  
value="50">
```

HTML5 attributes

Placeholder attribute:

```
<input type="search" placeholder="Place a search term her
```

Required attribute:

```
<input type="email" required />
```

Multiple attribute for multiple uploads or email addresses:

```
<input type="file" multiple />
```

Pattern attribute match input to custom regular expression:

```
<input pattern="[0-9][A-Z]{3}" title="A digit follow by t
```

Demo:

Placeholder attribute:

Required attribute:

Multiple attribute for multiple uploads or email addresses:

Вибрати файли Файл не вибрано

Pattern attribute match input to custom regular expression:

More about pattern

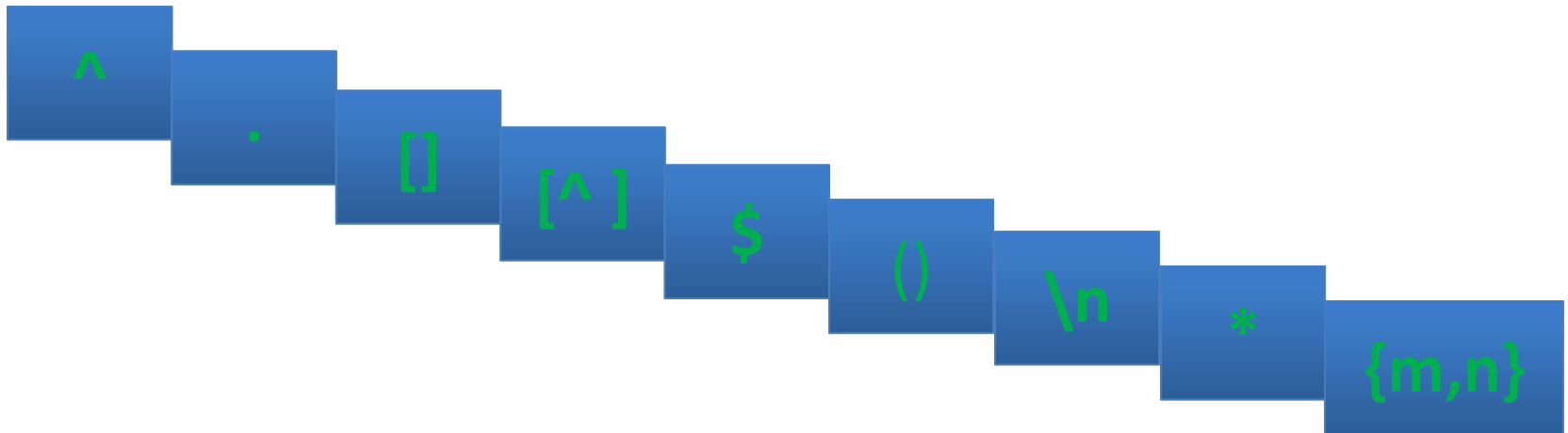
```
<input pattern="regexp">
```

The pattern attribute specifies a regular expression that the <input> element's value is checked against. The pattern attribute works with the following input types: text, date, search, url, tel, email, and password.

Use the global title attribute to describe the pattern to help the user.

patterns

Metacharacter



patterns

Meta-character(s)	Description
.	Normally matches any character except a newline. Within square brackets the dot is literal.
()	Groups a series of pattern elements to a single element.
+	Matches the preceding pattern element one or more times.
?	Matches the preceding pattern element zero or one time.
?	Modifies the *, +, ? or {M,N}'d regex that comes before to match as few times as possible.
*	Matches the preceding pattern element zero or more times.

patterns

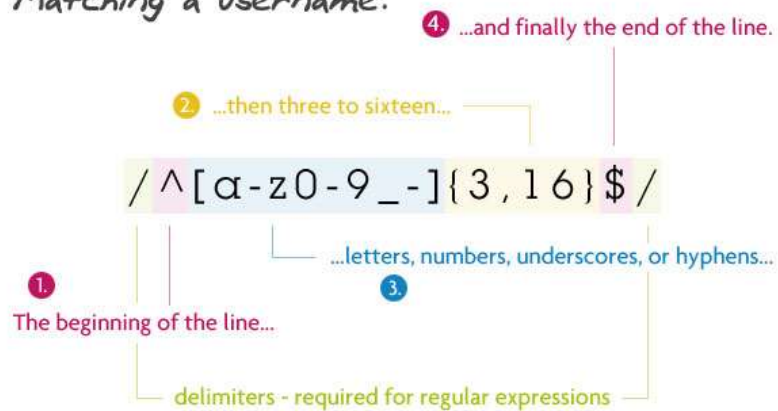
Meta-character(s)	Description
{M,N}	Denotes the minimum M and the maximum N match count.
[...]	Denotes a set of possible character matches.
	Separates alternate possibilities.
\w	Matches an alphanumeric character, including "_";
\W	Matches a non-alphanumeric character, excluding "_";
\s	Matches a whitespace character
\S	Matches anything BUT a whitespace.
\d	Matches a digit; same as [0-9] in ASCII;
\D	Matches a non-digit; same as [^0-9] in ASCII or \P{Digit} in Unicode.

patterns

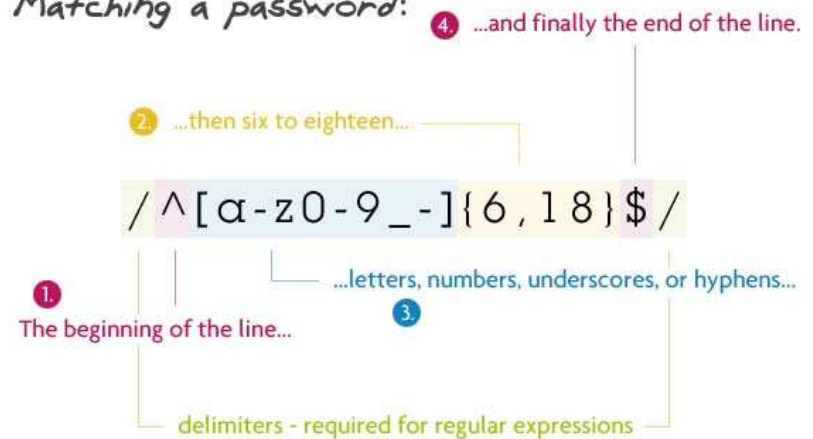
Meta-character(s)	Description
<code>^</code>	Matches the beginning of a line or string.
<code>\$</code>	Matches the end of a line or string.
<code>\A</code>	Matches the beginning of a string (but not an internal line).
<code>\z</code>	Matches the end of a string (but not an internal line).
<code>[^...]</code>	Matches every character except the ones inside brackets.

example

Matching a username:

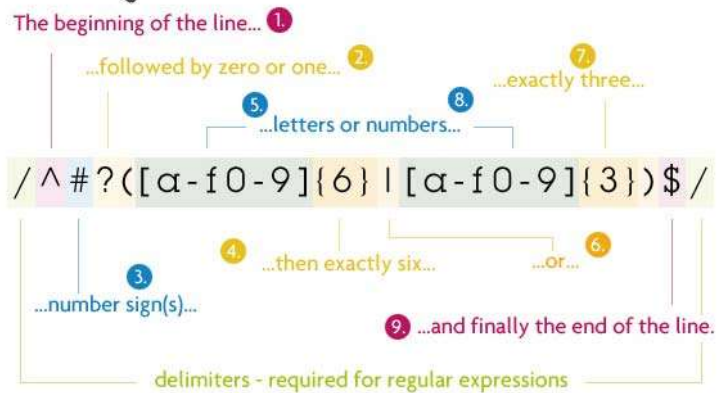


Matching a password:

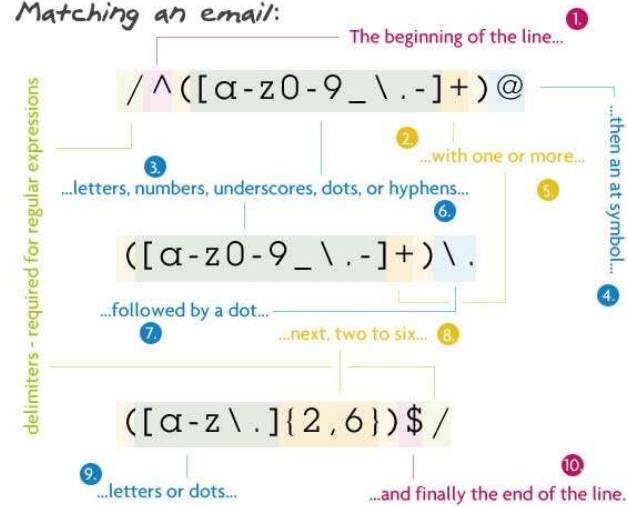


example

Matching a hex value:



Matching an email:



HTML5 Shiv

- **HTML5Shiv** is a JavaScript workaround, invented to enable styling of HTML5 elements in versions of Internet Explorer prior to version 9, which do not allow unknown elements to be styled without JavaScript.
- HTML5Shiv is calling **document.createElement("tagname")** for each of the new HTML5 elements, which causes IE to parse them correctly.
- It also includes basic default styling for those HTML5 elements.

```
<head>  
  <!--[if lt IE 9]>  
    <script src="//cdnjs.cloudflare.com/ajax/libs/html5shiv/r29/html5.min.js">  
    </script>  
  <![endif]-->  
</head>
```




Graphics Elements

Tag <picture>

- serves as a container for zero or more <source> elements and one element to provide versions of an image for different display device scenarios.
- The browser will consider each of the child <source> elements and select one corresponding to the best match found; if no matches are found among the <source> elements, the file specified by the element's src attribute is selected.

Demo:



```
<picture>
  <source srcset="/media/examples/picture-
    small.png"
    media="(max-width: 1000px)">
  <source srcset="/media/examples/picture-
    wide.png"
    media="(min-width: 1000px)">
  
</picture>
```

Srcset image attribute

- allows you to list multiple alternative image sources which vary in pixel density
- This allows the browser to pick an image of the appropriate quality for the user's device

```

```

Canvas and SVG

- Provide native drawing functionality
- Completely integrated into HTML5 documents (part of DOM)
- Can be styled with CSS
- Can be controlled with JavaScript
- Use for animation, charts, images, pixel manipulation, and so on
- Canvas supports 2D and 3D (WebGL)

HTML5 Canvas



The HTML <canvas> element is used to draw graphics, on the fly, via JavaScript.

The <canvas> element is only a container for graphics.

Canvas has several methods for drawing paths, boxes, circles, text, and adding images.

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

Element					
<canvas>	4.0	9.0	2.0	3.1	9.0

Scalable Vector Graphics

SVG stands for Scalable Vector Graphics and it is a language for describing 2D-graphics and graphical applications in XML and the XML is then rendered by an SVG viewer.

- Provide native drawing functionality

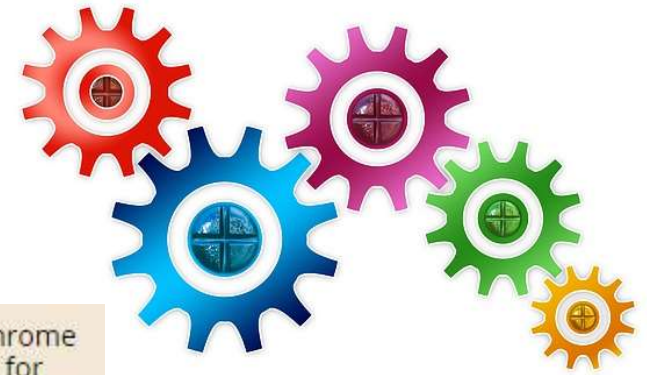
- Completely integrated into HTML5 documents

- (part of DOM) Can be styled with CSS

- Can be controlled with JavaScript

HTML5 SVG

SVG stands for Scalable Vector Graphics
SVG is used to define graphics for the Web
SVG is a W3C recommendation



IE	Edge [*]	Firefox	Chrome	Safari	iOS Safari [*]	Opera Mini [*]	Chrome for Android
			86		12.4		
	87	84	87	13.1	13.7		
^{2 3} 11	88	85	88	14	14.4	all	88
		86	89	TP			
		87	90				
			91				

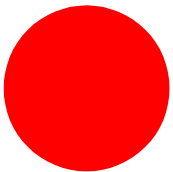
Scalable Vector Graphics

```
<h4>HTML5 SVGCircle</h4>
<svg id="svgelem" height="150" xmlns="http://www.w3.org/2000/svg">
  <circle id="redcircle" cx="50" cy="50" r="50" fill="red" />
</svg>
```

```
<h4>HTML5 SVGRectangle</h4>
<svg id="svgelem" height="150" xmlns="http://www.w3.org/2000/svg">
  <rect id="redrect" width="300" height="100" fill="red" />
</svg>
```

Demo:

HTML5 SVG Circle



Demo:

HTML5 SVG Rectangle



SVG Icons

Icons are a great place to start using SVG on the web. SVGs are flexible, resolution independent, and light weight, so icons naturally lend themselves to the vector format.

With SVG, the freedom exists to style individual elements within an icon, which opens up entirely new possibilities.



[SVG icons](#)

Canvas and SVG

Canvas	SVG
<ul style="list-style-type: none">•Resolution dependent•No support for event handlers•Poor text rendering capabilities•You can save the resulting image as .png or .jpg•Well suited for graphic-intensive games	<ul style="list-style-type: none">•Resolution independent•Support for event handlers•Best suited for applications with large rendering areas (Google Maps)•Slow rendering if complex (anything that uses the DOM a lot will be slow)•Not suited for game applications

Html5. API

Geolocation API

Drag & Drop

Local storage

Geolocation API



Geolocation (geolocation) - determining the current location of the user

Issuance of content in accordance with the location of the user (geotargeting)

Update information on the page as the user moves

Point-to-Point Routing

Geometrics in social networks

Geolocation API

Does not depend on the device

Does not depend on the source of information

The location is determined based on the IP address, MAC address, WiFi access point coordinates, GPS, GSM / CDMA identifiers, etc.

The location information is represented by latitude and longitude.

API					
Geolocation	5.0 - 49.0 (http) 50.0 (https)	9.0	3.5	5.0	16.0

Drag & drop API



Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location.

In HTML5, drag and drop is part of the standard: Any element can be draggable.

API					
Drag and Drop	4.0	9.0	3.5	6.0	12.0

Web Storage



Web applications can store data locally within the user's browser.

Web storage is more secure, and large amounts of data can be stored locally, without affecting website performance.

Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.

Web storage is per origin (per domain and protocol). All pages, from one origin, can store and access the same data.

API					
Web Storage	4.0	8.0	3.5	4.0	11.5

HTML5 Features

- Simplified Markup
- New semantic tags and attributes
- Audio, video support
- New graphic elements
- New form control attributes
- New HTML5 API's

Html Linter

- Is used to avoid potential errors.
- Ready to use, you don't need additional configurations.
- You **CAN'T** change any rules.
- You won't be able to push your code if there any errors occurred.
- You can find all rules in .htmlintrc file in the root of your project.
- Detailed rules description:
<https://github.com/htmlhint/htmlhint/wiki/Options>

Additional links

- <https://validator.w3.org/>
- <https://developer.mozilla.org/>
- <https://webref.ru>
- <http://htmlbook.ru>
- <https://caniuse.com/>
- <https://github.com/htmlhint/htmlhint/wiki/Options>
- [w3schools. HTML5 Introduction](#)
- [Справочник по HTML5](#)
- [SVG Tutorial](#)
- [Canvas Tutorial](#)
- [Manipulating SVG Icons With Simple CSS](#)

FE Online UA Training Course Feedback

I hope that you will find this material useful.

If you find errors or inaccuracies in this material or know how to improve it, please report on to the electronic address:

serhii_shcherbak@epam.com

With the note [FE Online UA Training Course Feedback]

Thank you.

Q&A



DRIVEN



CANDID



CREATIVE



ORIGINAL



INTELLIGENT



EXPERT

UA Frontend Online LAB