- HTML stands for **HyperText Markup Language**.
- Structured text with explicit markup denoted within < and > delimiters.
- Not *what-you-see-is-what-you-get* (WYSIWYG) like MS word.
- Similar to other text markup languages like `latex`.

- HTML was designed as an application of IBM's *Standard Generalized Markup Language* SGML.
- HTML 1.0: used `href` for hyperlinks.
- Evolution added support for tables, client-side image maps.
- Evolution even added support for presentation elements like font, color.
- Modern HTML removes support for presentation elements; moved presentation into CSS.
- HTML documents are often sloppily marked up; standards define explicit behavior for some bad mark up.

HTML 4.0 Never caught on.

```
<!DOCTYPE HTML PUBLIC
    "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
```

XHTML HTML as an XML document. Since XML must be well-formed, difficulties for web authors who were used to sloppy markup.

```
<!DOCTYPE html PUBLIC
    "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/
            xhtml1-strict.dtd">
```

HTML 5 Evolving modern HTML standard; simple DTD
`<!DOCTYPE html>`.

- A HTML document consists of a tree of HTML elements.
- A HTML element delimited between a start tag like <a> and an end-tag like .
- There may be text or other tags between the start tag and end tag.
- The start tag may contain attributes, like <a href="submit.cgi">.
- The set of allowed attributes for any element are predefined with one exception: any element can have attributes with names starting with `data-`. Allows extensible attributes.
- A tag with no content can be denoted as <br />; often simply use opening tag without closing tag; so simply <br>.

href Specifies absolute or relative URL to another resource.

rel Specifies the relationship of the linked to resource from the linking resource.

id Specifies an ID for element. The ID must be unique across the entire document.

class Value consists of multiple space-separated identifiers. Element class can be used for attaching styling and/or behavior to the element.

`class` and `rel` attributes have been used to provide semantics to markup using microformats. Example microformats: *hCalendar* for events, *hCard* for contact information, *geo* for geographical information.

- Absolute URLs are complete URLs containing scheme, hostname and path. Example: `href="https://developer.¬mozilla.org/en-US/docs/Web/JavaScript"`.
- Relative URLs can omit parts of the URL which are filled in from the referring document:

  Use current scheme Use same scheme are current document. Example: `href="//developer.mozilla.org¬/en-US/docs/Web/JavaScript"`.

  Use current scheme and host Same scheme and host as current document. Example: `href="/en-US/docs/Web/JavaScript"`

  Use current scheme, host and path Example: `href="Reference/Global_Objects/Array"` or `href="../HTML"`.

`<html>`   A single `<html>` element must be present enclosing entire content.

`<head>`   Contains meta-content like `<title>` (displays title in browser window bar), `<link>` for loading CSS stylesheets, `<script>` for loading JavaScript files.

`<body>`   Encloses actual document content.

<h1>, ..., <h6>   Headings at different levels.

<section>   Delimits a section of the document. Usually followed by a <h*i*> element.

<nav>   Used for delimiting content used for site navigation.

<div>   Used for delimiting general block content. Usually used to attach style or behavior to a block using `id` or `class` attributes.

<p>   Used for delimiting paragraphs.

List Markup

Unordered Lists Denoted using

```
<ul>
  <li>...</li>
  ...
</ul>
```

Ordered Lists Denoted using

```
<ol>
  <li>...</li>
  ...
</ol>
```

Definition Lists Denoted using

```
<dl>
  <dt>...</dt> <dd>...</dd>
  ...
</dl>
```

- Tables delimited using <table> tags.
- Rows within a table are delimited using <tr> tags.
- Table entries within a row are delimited using <th> tags (for heading entries) or <td> tags (for data entries).
- A table entry can span multiple columns (using attribute colspan) or multiple rows (using attribute rowspan).

<span> Simply used to delimit some content, similar to `<div>`. Example `<span class="keyword">while</span>`.

<em> Emphasized text. Example: `<em>Important</em>`.

<strong> Strongly emphasized text. Example: `<strong class="alert">Warning</strong>`

<img> Used to embed an image Example: `<img src="smile-emoji.gif">`. Can also be used at the block level.

<a> Hyperlinks. Example See `<a href="other-doc.html>other document</a>`.

```
<link rel="stylesheet" href="style.css">
```

- End tag must not be present.
- For best efficiency, include in head section.

```
<script type="text/javascript" src="code.js"></script>
```

- End tag must be present.
- For best efficiency, include after bulk of document body just before </body> tag.

```
<form action="http:www.google.com" method="get">
  Search: <input name="q">
</form>
```

- Forms need to be set up using `<form>` tags.
- `action` gives URI where form should be submitted.
- `method` can be get (default) or post. Other HTTP methods are **not supported**.
- `enctype` used when method is post. Default is `application/x-www-form-urlencoded`. Use `multipart/form-data` if uploading files. HTML5 allows `text/plain`.

Form Controls

- All form controls have a `name` attribute which gives the name by which that control is submitted.
- Usually form controls have to be embedded within a `<form>` element, but HTML5 allows using a `form` attribute specifying the `id` of any `<form>` element on that page.
- Form controls can be `disabled` which makes them inactive.
- Captions for form controls by putting control inside a `<label>` element or by specifying control id in the `for` attribute for `<label>`.
- Can group controls together using `<fieldset>`.

`<input type="`*TYPE*`">` *Live example* from MDN. Less typing using *Local example*

- Main form input field.
- *TYPE* traditionally had values `button`, `checkbox`, `file`, `hidden`, `image`, `password`, `radio`, `reset`, `submit`, `text` (default).
- HTML5 added many more variants: `color` for color-picker, `date`, `datetime-local`, `time`, `month`, `week` for date-time, `email`, `tel` for contact information, `number`, `range` for numeric information, `url` for URLs.
- `autocomplete` attribute allows browser to fill in information previously saved by user.
- `pattern` attribute is a regex the entire value is matched against.

<select>   Provides a menu of options using embedded
<option> elements. Specify `multiple` attribute to
allow multiple options to be selected. Can make an
option selected by setting its `selected` attribute.

<textarea> Multiline text input.

Separation of concerns:

- Presentation relegated entirely to CSS using **external** stylesheets.

- Behavior relegated entirely to *unobstrusive javascript*.

- HTML should specifiy content in as semantically meaningful a way as possible.
  - Do not use tables for layout, only for information which is naturally tabular in nature.
  - Use semantically appropriate HTML tag if possible, minimize use of semantically meaningless `div` and `span`.
  - Link to external semantics using itemscope and friends (a modern way of doing microdata) or *HTML + REFa*.