Web Programming with XHTML

Christophe Lecoutre
lecoutre@cril.fr

IUT de Lens - CRIL CNRS UMR 8188
Université d'Artois
France

Département SRC - 2010/2011

Outline

Outline

Bibliography

Books

- John Duckett.
  Beginning Web Programming with HTML, XHTML, and CSS,
  2nd Edition, Wrox. 2008.
- ...

Sites

- W3C Specifications at `http://www.w3.org/html/wiki/Specifications`
- W3School-XHTML at `http://www.w3schools.com/xhtml`
- W3C Validator at `http://validator.w3.org`

- SelfHtml-HTML at `http://fr.selfhtml.org/html`
- Giminik-XHTML at `http://giminik.developpez.com/xhtml`

Objective

Writing web pages with:

- CSS: defining the appearance of the pages
- Javascript: adding interactivity/control

Increased emphasis on:

- usability: easiness for users to navigate
- accessibility: making sites available to as many users as possible

Markup Languages

Languages that specifies the logical structure of a document (page). For example:

- XHTML: Extensible Hypertext Markup Language, used for the web
- HTML: Hypertext Markup Language, used for the web
- Latex: used for professional elaboration of scientific documents

Our interest: XHTML, together with CSS

XHTML can be seen as an extension/successor of HTML. XHTML is written in XML (Extensible Markup Language) XHTML is built upon the concepts of:

- tags
- elements
- attributes

Tags and Elements

Tags and elements:

- A tag is formed by an identifier and two enclosing angle brackets.
- An element is formed by a content (possibly empty) and two enclosing similar tags (i.e., two tags with the same identifier).
- The left tag is called the opening tag whereas the right tag is called the closing tag.
- The closing tag always has a forward slash after the first angle bracket.

Example

```
<h1> A propos de XHTML </h1>
```

Warning

A tag `tagName` without any content can be written `<tagName/>`

Attributes

An attribute is formed by:

- a name
- the equal sign (=)
- a value between double quotes, or simple quotes

A space-separated list of attributes (possibly empty) is put after the identifier of the opening tag and before the right angle bracket.

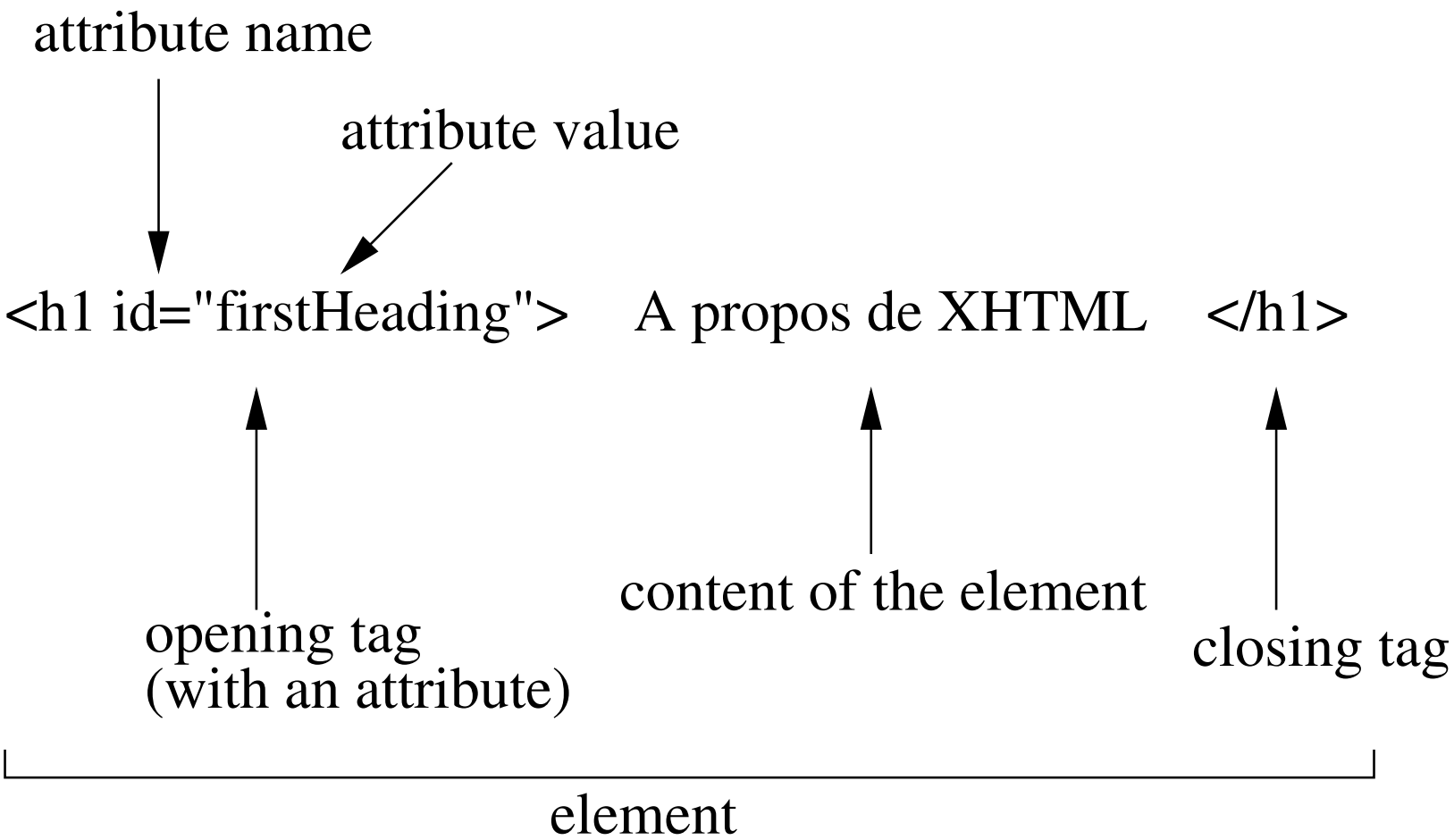
Example

```
<h1 id="firstHeading"> A propos de XHTML </h1>
```

Warning

Tags and attributes must be written in lowercase letters.

Tags, Elements, and Attributes



Tags, Elements, and Attributes

Example

```
<html>
  <head>
    <title>Toto - Home Page</title>
  </head>
  <body>
    <h1> Premier en-tete <\h1>  <!-- ceci est un en-tete -->
    <p> coucou </p>
    <a href="http://www.arte.fr">arte</a>  <!-- un lien -->
  </body>
</html>
```

Remark

Comments are put between any tags using the syntax: `<!-- comment -->`

As an exercice, build the tree structure of the document...

Versions of XHTML

In order to validate an XHTML document, a Document Type Declaration, or DOCTYPE, may be used. A DOCTYPE declares to the browser the Document Type Definition (DTD) to which the document conforms. There are three current versions of XHTML:

- Strict XHTML 1.0

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

- Transitional XHTML 1.0

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

- Frameset XHTML 1.0

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

Warning

Use Strict XHTML 1.0

Skeleton of a Valid Strict XHTML 1.0 Document

```
XML declaration
DOCTYPE declaration
html
  head
    title
  body
```

The first line (XML declaration) is:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

Typically, there are three attributes for the opening tag `html`:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
```

For english language, use:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

Remark

The attributes `xml:lang` and `lang` can be used on other elements

Outline

XHTML offer six levels of headings: <h1>, <h2>, <h3>, <h4>, <h5> and <h6>

By default, browsers display the contents of:

- <h1>, <h2> and <h3> larger than the default size of text
- <h4> as the default size of text
- <h5>, <h6> smaller than the default size of text

Example

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
```

Remark

As for all other elements, CSS can be used to override the size and style of any of these heading elements

Two frequent elements:

- The <p> element is for building paragraphs.
- the <br> element is for creating line breaks

The <br> element is an empty element: just write

Remark

Do not write <br/> or <br>, but

Whatever the number of consecutive white spaces is, only one white space is displayed. This is known as *white space collapsing*.

Example

```
<p> Ceci est un paragraphe sur deux lignes dans le code
    source du document.
</p>
```

The content of a <tt> (teletype) element is written in a monospaced font.

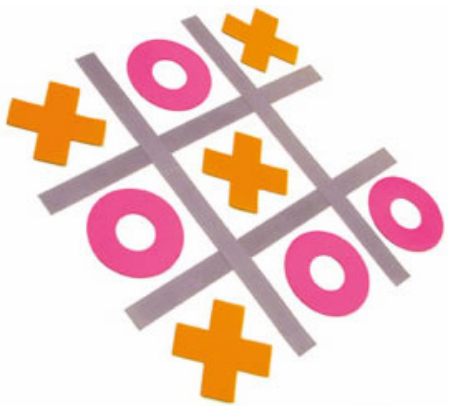
The content in a <pre> element is displayed in a monospaced font, and it preserves both spaces and line breaks.

Remark

Contrary to <pre>, <tt> doesn't affect the formatting of the text.

Example

```
<pre>
     x | o | x
    -----------
     o | o | x
    -----------
     o | x | x
</pre>
```



Phrase Elements: and

phrase.html

These elements add structural information to text fragments.

Important ones:

- adds emphasis (usually displayed in italicized text)
- adds emphasis stronger than (usually displayed in bold text)

Good reasons to use them (instead of presentational elements <i> and <b>):

- applications such as screen readers (to web users with visual impairments) could add suitable intonation
- automated programs could pull them out as keywords or index these words

Example

```
<p> <strong> Fermer </strong> le gaz </p>
<p> En arrivant <em> passer un coup de fil </em> </p>
```

Phrase Elements: <dfn> and <address> *phrase.html*

As phrase elements, we also find:

- <dfn> indicates that you are introducing a special/new term; typically rendered in italic font.
- <address> indicates a snail-adress, ususally at the end of a document.

Example

```
<p> Une fonction est <dfn> recursive </dfn> ssi elle s'appelle
    dans son propre corps. </p>
<address>
    IUT de Lens, <br />
    Rue de l'universite, <br />
    SP 16, 62307 LENS Cedex
 </address>
```

Phrase Elements: <abbr>, <cite>, ... *phrase.html*

- <abbr> indicates an abbreviation; when possible, use a `title` attribute.
- <acronym> indicates an accronym; when possible, use a `title` attribute, and possibly a `xml:lang` attribute.
- <cite> contains a citation or a reference to another source; often used to enclose titles of books, movies or songs.
- <blockquote> adds a long quote (within a block) from another source. Start it after an empty line, and use <p> within the blockquote.
- <q> adds a short quote (within a sentence) from another source.

Warning

The <cite> element is not the `cite` attribute (whose value must be a URL) usually used with <blockquote> and <q>.

- <code>: the content is programming code apprearing on a web page, usually rendered in a monospaced font.
- <var>: the content indicates a variable.
- <kbd>: the content indicates text to be entered by the user.
- <samp>: the content designates sample output from programs, scripts, etc.

Example

```
<pre> <code>
  int sum(int i, int j) {
        return i+j;
  }
</code> </pre>
```

Warning

Some characters are reserved in HTML (see next slide).

Special Characters

Here are the special characters:

Symbol	Description	Entity Name	Number Code
&	Ampersand	&amp;	&#38;
<	Less than	&lt;	&#60;
>	Greater than	&gt;	&#62;
"	Double quote	&quot;	&#34;
	Non-breaking		
	space		

Example

```
<abbr title="Tim">
```

should be written to be displayed as such in a XHTML page as:

```
\&lt;abbr title=\&quot;Tim\&quot;\&gt;
```

Presentational Elements

Warning

Most of these elements should be avoided.

The following elements may be useful:

- `<sup>`: the content is written in superscript
- `<sub>`: the content is written in subscript; helpful when combined with `<a>`
- `<hr>` creates a horizontal rule (line) across the page; it is an empty element, so write `<hr />`

Other elements:

- `<strong>` should be preferred to `<b>` (bold).
- `<em>` should be preferred to `<i>` (italic).
- `<u>` (underline), `<s>` and `<strike>` are deprecated.
- Use CSS rather than employing `<big>` and `<small>`

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Exercice

1. Pour avoir une police à espacement fixe, on utilise :
   a) `<pre>` b) `<b>` c) `<tt>` d) CSS
2. Pour mettre du texte en gras, on utilise :
   a) `<b>` b) `<em>` c) `<strong>` d) CSS
3. Est-ce que ce qui suit est bien formé ?

```
<p> Je suis <em> content </p> </em>
```

4. Est-ce que ce qui suit est bien formé ?

```
<h1> Programmation Web </h1>
<h2> XHTML </h2>
<p> Il faut respecter la syntaxe </p>
```

5. Est-ce que ce qui suit est bien formé ?

```
<adresse>
  Toto, rue des géants de la frite, Trouville
</adresse>
```

6. Est-ce que ce qui suit est bien formé ?

```
<p> <acronym title="Mademoiselle">Mlle </acronym> </p>
```

Editing Text

To show modifications made to a text, use:

- the <ins> element for when you add something; the inserted text is underlined;
- the <del> element for when you delete something; the deleted text will have a strikethough.

Example

```
<p> Le meilleur jeu video de tous les temps
est <del>Starcraft 2</del> <ins
datetime="2010-07-27T17:08:00"
cite="http://www.mamedb.com/game/pengo"
title="c'est trop bien">Pengo</ins> </p>
```

Warning

Only use these elements for a site being developped.

Core Attributes

There are four attributes that can be used on the majority of XHTML elements. They are:

- `id`: used to give a unique identifier to an element; the value must begin with a letter;
- `class`: used to specify that an element belongs to a specific group or class; the value is a space-separated list of class names (ussually, only one);
- `title`: gives a suggested title for an element; often displayed as a tooltip.

Warning

The attribute `style` is deprecated, and the attribute `name` no more valid in XHTML.

Remark

The attributes `id` and `class` will be particularly useful with CSS.

Internationalization Attributes

There are 3 internationalization attributes that help users write pages for different languages and character sets. They are available to most XHTML elements. They are:

- `dir`: indicates to the browser the direction in which the text should flow; its value may be `"ltr"` or `"rtl"`.
- `lang`: indicates the main language; kept for backward compatibility.
- `xml:lang`: indicates the main language.

Values of `lang` and `xml:lang` are ISO-639 standard two-character laguage codes (fr for french, en for english, en-us for U.S. english, ...)

Remark

Typically, you will use these attributes only with the element `html`. However, it is possible to associate them with some particular elements in the body of the document.

UI Event Attributes

The UI event attributes allow you to associate an event (e.g., a key press or a mouse click) with a script.

They are ten attributes known collectively as common event attributes:

`onclick`, `ondoubleclick`

`onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmousepress`

`onkeypress`, `onkeydown`, `onkeyup`

The `<body>` element (and `<frameset>` element) has the following events:

`onload`, `onunload`,

The events associated with forms are:

`onfocus`, `onblur`, `onsubmit`, `onreset`, `onselect`, `onchange`

Block and Inline Elements

Each element belongs to one of two categories:

- the block-level elements start on their own new line, are followed by a new line; for example, `<p>`, `<h1>`, `<h2>`, `<h3>`, `<ul>`, `<ol>`, `<dl>`, `<pre>`, `<blockquote>`, and `<address>` are block-level elements;
- the inline elements can appear within sentences and do not have to appear on a new line; for example, `<em>`, `<strong>`, `<sup>`, `<sub>`, `<big>`, `<small>`, `<ins>`, `<del>`, `<code>`, `<cite>`, and `<dfn>` are inline elements.

Remark

Block-level elements can contain other block-level and inline elements. Inline elements can only appear within block-level elements, and may not contain block-level elements.

Grouping Elements with and

span.html

These elements allow you to group several elements to create sections and subsections of a page. More precisely,

- is used to group block-level elements;
- is used to group inline elements.

Example

```
<div class="gras">
  <p> Lorsque l'alarme <span class="rouge" sonne </span> </p>
  ...
```

Remark

The elements and will not affect the appearance of a page, but are commonly used with CSS to attach a style to a section.

Outline

Unordered lists (<ul>) : are lists of bullet points
Ordered lists (<ol>) : are sequences of numbers, letters or Roman numerals
Definition lists (<dl>) : allow you to specify terms and their definitions

Each item of a list is:

- within a <li> element for both lists <ul> and <ol>;
- composed of a <dt> element (the term to be defined) and a <dd> (the term definition) for lists <dl>.

Example

```
<ul>
  <li>XHTML</li>
  <li>CSS</li>
</ul>
```

Remark

You must use CSS to control numbering (type, starting number) of lists <ol>.

Nested Lists

Be careful of writting properly nested lists. For example,

Example

```
<ul>
  <li>Deux roues</li>
  <li>Quatre roues
    <ol>
      <li>Automobiles</li>
      <li>Camions</li>
    </ol>
  </li>
</ul>
```

```
. Deux roues
. Quatre roues
    1. Automobiles
    2. Camions
```

Tables are made of rows and columns. At each intersection of a row and a column, we find a cell. A table is written out row by row.

The elements used to build tables are:

- `<table>`
- `<tr>` (table row): to build a row
- `<td>` (table data): to build a cell
- `<th>` (table heading): to build a cell as a heading for a row or a column; usually rendered in bold text

Example

```
<table>
  <tr> <th>Quarter 1 (Jan-Mar)</th>
       <td>11200.00</td> ...
```

Warning

Each cell must be a `<td>` or `<th>` element, even if that element is empty.

Attributes for <table> and <tr>

For the element <table>:

- summary provides a summary of the table's purpose and structure for non-visual browsers such as speech browsers and Braille browsers
- other attributes (those that are not universal or event attributes) are deprecated

For the element <tr>:

- char is used to specify that the content of each cell within the row must be aligned around the first instance of a particular character known as an axis separator; currently, not really supported by browsers
- charoff may also be associated with char;
- other attributes (those that are not universal or event attributes) are deprecated

Attributes for <td> and <th>

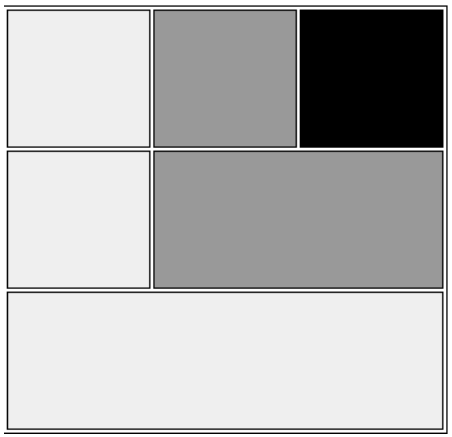
table.html

- `colspan` and `rowspan` specify how many columns and rows of the table a cell will span across
- `abbr` is used to provide an abbreviated version of the cell's content
- `headers` indicate (for voice browsers) which headers correspond to that cell; its value is a space-separated list of the header cell's `id` attribute values
- `scope` defines a way to associate header cells and data cells in a table
- `char` and `charoff`, defined as for `<tr>`
- `axis` allows you to add conceptual categories to cells (to be used programmatically)
- other attributes (not universal or event attributes) are deprecated

Remark

An attribute given for an element `<td>` and `<th>` will override similar attributes or effects of a containing element such as `<tr>` or `<table>`.

Note that if a table has 3 columns and a cell spans 2 columns in a row, you only need 2 <td> elements in that row. A similar reasoning has to be done for cells that span across rows.



Example

```
<table>
  <tr>
    <td class="col1" colspan="3"> </td>
    ...
```

Remark

If a cell has no real content, include   to avoid problems with some browsers.

Head, Body and Foot

splitting.html

A table can be divided into three portions:

- a header, using the element `<thead>`
- a body using the elment `<tbody>`
- a foot using the elment `<tfoot>`

The separation of the parts of the table allows for richer formatting of tables by browsers (e.g., for aural browsers, or when printing a table).

Remark

A table may contain several `<tbody>` elements to indicate different "pages".

Warning

The `<tfoot>` element must appear before the `<tbody>` element in the code.

Captions and Groups of Columns

To add a caption, just insert an element `<caption>` before the first row or header.

To format (with CSS) groups of adjacent columns, define such groups with the element `<colgroup>` and its attribute `span`.

Example

```
<table>
   <caption>This is the caption</caption>
   <colgroup span="4" />
   <colgroup span="2" />
   <tr>
      ...
```

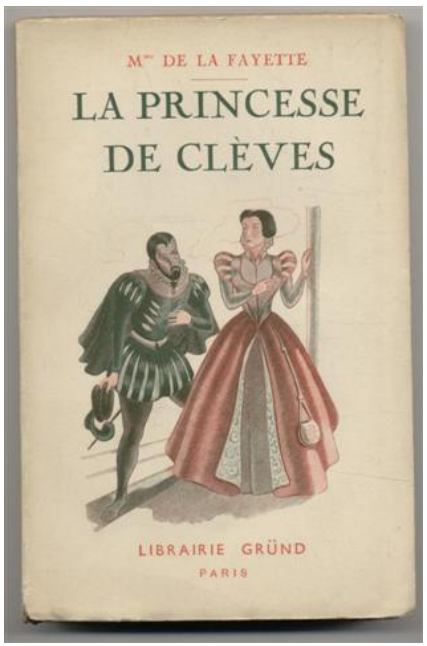
Remark

It is possible to refine column groups by using the element `<col>`.

Exercice

Ecrire le code HTML pour reproduire le tableau suivant :

Ecrivain	Oeuvre
Stendhal	Le rouge et le noir
Madame de la Fayette	La princesse de Clèves



Outline

Hyperlinks

Hyperlinks allow visitors to navigate between web sites by clicking on words, phrases and images.
To link to another document, we use the element <a>. The minimal syntax for using it is: `<a href="URL"> </a>`

It is a good practice to:

- make the content of your links concise but precise;
- use the `title` attribute, displayed as a tooltip when the user hovers over the link.

Example

```
<a href="http://www.google.com/" title="Search the Web with
Google">Google</a> is a very popular search engine.
```

URL

URL stands for Uniform Resource Locator. The general syntax is:
`protocol://user:pass@host:port/path?query#anchor`

Elements of a URL

- The protocole, or scheme name, identifies the type of URL you are linking; typically, this is http, but it may be https, ftp, file, ...
- The username and password are optional (required for a password-protected part of a site).
- The host (name) or IP address gives the destination location for the URL.
- The port number is optional; if omitted, the default for the scheme (protocol) is used.
- The path is used to find the resource specified; it is case-sensitive.
- The query string contains data to be passed to web applications.
- The anchor part when used with HTTP specifies a precise location on the page.

URL

Most of the time, an URL is a simple *absolute* URL:

```
http://host/path
```

It may also be a *relative* URL (i.e. the host is absent):

```
path
```

Example

```
http://www.lemonde.fr/international/
page2.html
http://www.arte.tv/fr/70.html
```

Remark

When the path does not end with a filename, the web server returns a default file (e.g. index.html), or an error message.

Anchors

There are two kinds of anchors that can be created with <a>:

- source anchors, which are built using <a> with the attribute `href`
- destination anchors, which are built using <a> with the attribute `id`

Remark

Source anchors are what most people think of when talking abour links on the web. It is something you can click and then expect to be taken somewhere else.

Destination Anchors

A destination anchor allows the page author to mark specific points in a page that a source link can point to.

Common uses are:

- "Back to top" links at the bottom of (sections) of long pages;
- A list of contents for a page that takes the user to the relevant section;
- Links to footnotes or definitions.

The value of the `href` attribute of a source anchor that establishes a link to a destination anchor must be the value of the `id` attribute of the destination anchor preceded by the character #.

Remark

It is possible for a source anchor to link to any element with an `id` attribute.

Source and Destination Anchors

Example

```
<a id="pagetop">Navigation</a>
...
...
<a href="#pagetop">Page top</a>
```

Warning

The content of a destination anchor must not be empty.

Attributes for <a>

In addition to the universal attributes (i.e. core and internationalization attributes) and the UI event attributes, we find:

- `accesskey`: provides a keyboard shortcut on a source anchor to activate a link; ALT + SHIFT + accesskey for Firefox (≥ 2)
- `target`: specifies in which window/frame the linked document will be opened; use `"_blank"` to open it in a new window and `"_self"` for the same window
- `tabindex`: specifies the order in which, when the TAB key is pressed, the links (or form controls) obtain focus
- `charset`: indicates the character encoding of the linked document (e.g., UTF-8 or ISO-8859-1)
- `hreflang`: specifies the language of the linked document (e.g., fr)
- `type`: specifies the MIME type of the linked document
- `coords` and `shape`: to deal with image maps
- `rel` and `rel`: see documentation

Linking to E-mail Addresses

To open a new e-mail in the user's default e-mail program, with a given e-mail address, the syntax is:

```
<a href="mailto:toto@usine.fr">E-mail me</a>
```

Remark

It is possible to specify the subject, the body, the cc and the bcc of the message. For example,

```
<a href="mailto:toto@usine.fr?subject=XHTML&cc=titi@usine.fr">...</a>
```

Warning

The address can be retrieved to be spammed. Use e-mail forms (with a server-side scripting language such as JSP or PHP) or Javascript to protect it.

Images

In XHTML, we can:

- add images to the documents
- use images as links
- divide an image into sections with different links; this is called *image map*

Warning

Be careful of the size of the image files.

Warning

Images are subject to copyright.

Images

Browers tend to support three common bitmap graphics format:

- GIF (Graphics Interchange Format) using 256 colors (8-bit GIF) or 16 colors (4-bit GIF)
- JPEG (Joint Photographic Experts Group Format)
- PNG (Portable Network Graphics) using 256 colors (8-bit PNG) or many more (24-bit PNG)

The most popular vector graphics format on the web is Flash.

The element <img>

To include an image, use <img> with at least two attributes;

- src: required to specify the URL of the image to load; the URL can be absolute or relative
- alt: required to specify a text alternative for the image in case the user cannot see it

Example

```
<img src="logo-IUT.gif" alt="logo IUT de Lens" />
```

Remark

It is important that the value of the attribute alt really describes the image. An image cannot be seen for two common reasons:

- the file has not been found by the browser;
- the user has visual impairment.

However, if the image is only used to enhance the layout of the page (and provides no information), just put alt="".

In addition to the universal and UI event attributes, <img> can also carry the following attributes:

- `width` and `height`: specify the width and height of the image (in pixels); it allows the browser to lay out the image quicker
- `longdesc`: used to indicate the URL of a document containing a detailed description of the image
- `ismap` and `usesmap`: used with image maps
- `align`, `border`, `hspace`, `vspace` and `name`, which are deprecated

Warning

If you want to display the image a lot smaller, you should create a new reduced-size version of the image (thumbnail), instead of using `width` and/or `height` attributes.

The element <object>

This element is used to embed all media types (e.g., MP3s, Flash movies, Java applets, ...) into documents.
To embed an object into a page, you need to specify:

- the location of the software used to run the object
- the actual data to be run
- any additional values the object needs at runtime within <param> elements (that are childs of <object>)

Remark

The content of <object> (except for <param> childs) is displayed if the browser cannot run the object. For example,

```
<object> <p> your browser does not appear to support
         the format used in this film clip. </p>
</object>
```

Inserting a Flash Movie into XHTML

media.html

```
<!--[if !IE]> -->
<object type="application/x-shockwave-flash"
      data="movie.swf" width="300" height="135">
<!-- <![endif]-->

<!--[if IE]>
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
      codebase="http://download.macromedia.com/pub/shockwave/
                cabs/flash/swflash.cab#version=6,0,0,0"
      width="300" height="135">
  <param name="movie" value="movie.swf" />
<!-->
  <param name="loop" value="true" />
  <param name="menu" value="false" />
  <p>The movie cannot be played.</p>
</object>
<!-- <![endif]-->
```

Inserting a Java Applet

media.html

```
<!--[if !IE]> Firefox and others will use outer object -->
<object type="application/x-java-applet"
      classid="java:DrawingLines.class" archive="DrawingLines.jar"
      height="300" width="550">
<!--<![endif]-->

<!--[if IE]>
<object classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
      codebase="http://java.sun.com/update/1.5.0/
                jinstall-1_5_0-windows-i586.cab"
      height="300" width="550" >
      <param name="code" value="DrawingLines" />
<!-->
  <param name="codebase" value="java/" />
  <param name="archive" value="DrawingLines.jar" />
  <p>This browser does not have a Java Plug-in. </p>
</object>
<!--<![endif]-->
```

Images as Links and Image Maps

To turn an image into a link, rather than putting text inside an element <a>, just place an image.

Example

```
<a href="index.html" title="click to return home">
  <img src="images/banana.gif" alt="Banana" />  </a>
```

On the other hand, image maps allow you to specify several links that correspond to different clickable areas, called hotspots, of one single image. There are two types of image maps:

- Server-side image maps
- Client-side image maps

Warning

The hotspots shouldn't be too small. Besides, you should put text links at the bottom of the page, and indicate this with the `alt` attribute of the image.

Server-side Image Maps

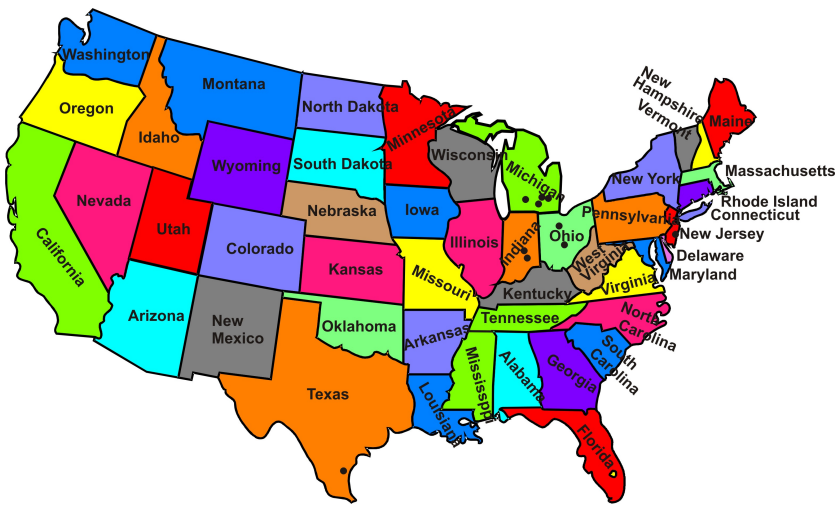
The image is put inside a <a> element. A special attribute `ismap` is used, which tells the browser to send the server the coordinates x, y where the user's mouse was when he/she clicked.

Example

```
<a href="map.php">
  <img src="images/states.gif"
       alt="Map of United States"
       ismap="ismap" />
</a>
```

If the user clicks the image at position (50,75) then the browser will send the following query to the server:

`http://www.example.org/map.php?50,75`



Client-side Image Maps

map.html

To build one, we use the elements <map> and <area>. The image that forms the map is an element <img> with an attribute usemap. The value of this attribute is the value of the name attribute of the element <map> preceded by #. The element <map> carries the attribute name and contains several <area> elements.

Example

```
<img src="images/gallery_map.gif" alt="Gallery Map"
     width="500" height="300" usemap="#gallery" />
<map id="idGallery" name="gallery">
   <area shape="circle" coords="154,150,59"
         href="foyer.html" alt="Gallery foyer" />
   ...
```

Remark

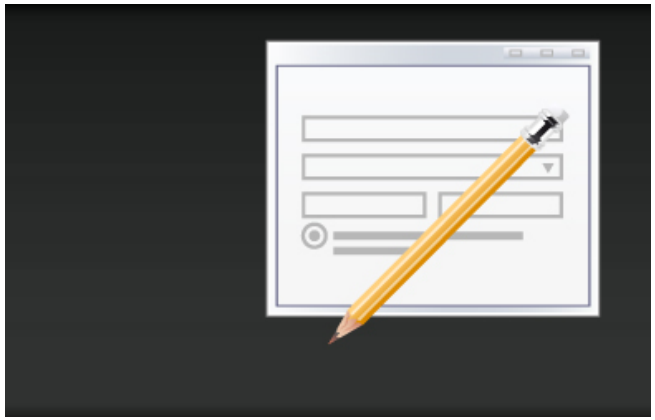
There is another way of building an image map with an element <map> inside an <object> element.

Outline

Forms

Everytime you want to collect information from a visitor to your site, a form is required. You can create a form by combining what are known as form controls:

- Text fields
- Buttons
- Checkboxes
- Select lists
- ...



Warning

XHTML is used only to present the form to the user; it does not allow you to say what happens with that data once it has been collected.

Forms

Once users have entered information into a form, they usually have to click on a submit button to send the form data to a web server. The data are sent to the server in name/value pairs, one pair per form control:

- the name corresponds to the name of the form control
- the value is what the user has entered or the value of the selected option

Example

```
<form action="http://www.totosa.fr/search.php" method="get"
      id="idSearch">
  ...
</form>
```

Warning

Every `<form>` element should carry at least the two attributes `action` and `method`.

Attributes for the Element <form>

- `action` specifies where to send the form data when the form is submitted; the value is a URL corresponding to a page/program on a web server
- `method` specifies the method used to send the data; it may be `"get"` or `"post"`
- `enctype` specifies how form data must be encoded before being sent to the server; use `"multipart/form-data"` as value when you are using forms that have a file upload control
- `accept-charset` specifies the character-sets the server can handle for form data
- `onsubmit` and `onreset` may be used to make controls (e.g. with Javascript) when the user clicks on a submit button or a reset button

Warning

A form can contain elements such as paragraphs, headings, ... but must not contain another form.

Text Inputs and Buttons

There are three types of text input controls used in forms:

- Single-line text input controls by using the `<input>` element with a `type` attribute whose value is `"text"`.
- Password input controls by using the `<input>` element with a `type` attribute whose value is `"password"`.
- Multi-line text input controls by using the `<textarea>` element

There are three ways of creating a button in forms:

- by using the `<input>` element with a `type` attribute whose value is `"submit"`, `"reset"` or `"button"`;
- by using the `<input>` element with a `type` attribute whose value is `"image"`;
- by using the `<button>` element with a `type` attribute whose value is `"submit"`, `"reset"` or `"button"`.

Single-line Text Input

textInputs.html

Such a control is created by an element `<input>` with an attribute `type` whose value is `"text"`. In addition to the universal attributes, the other attributes of element `<input>` are:

- `name`: gives the name part of the name/value pair that is sent to the server
- `value`: provides an initial value
- `size`: specifies the width of the control in terms of characters that can be displayed
- `maxlength`: specifies the maximum number of characters a user can enter
- `disabled`, `readonly`, `tabindex` and `accesskey`

Example

```
<p> Search the site: <input type="text" name="txtSearch" /> </p>
```

Password Input

textInputs.html

Such a control is created by an element `<input>` with an attribute `type` whose value is `"password"`. The password input masks the characters the user types by replacing them with either a dot or an asterisk.

Example

```
<input type="password" name="pwdPassword"
       size="20" maxlength="20" />
```

Warning

Passwords are hidden on the screen but are sent across the Web as clear text. In order to make them secure, you should use an SSL connection between the client and the server.

Multiple-line Text Input

textInputs.html

Such a control is created by an element `<textarea>`. It allows the user to enter more than one line of text.

Example

```
Please, tell us what you think: <br />
<textarea name="txtFeedback" rows="10" cols="50" >
   Enter your feedback here.
</textarea>
```

Remark

The attributes `rows` and `cols` specify the number of rows and columns of the `textarea` control.

Warning

You always need an opening and closing tags for the `<textarea>` element, even if the content is empty, because some browsers require these two tags.

Buttons using the <input> Element

buttons.html

With the <input> element, the type of button you create is specified using the type attribute. Its value may be:

- "submit": creates a button that automatically submits a form
- "reset": creates a button that automatically resets form controls to their intial values
- "button": creates button used to trigger a client-side script

Example

```
<input type="submit" name="btnBlue" value="Vote for blues" />
<input type="reset" value="Clear" />
<input type="button" value="Calculate" onclick="calculate()" />
```

Remarks

The attribute value specifies the text displayed on the button. The attributes onclick, onfocus and onblur may be used to make some controls.

Using Images for Buttons

You can use an image for a button rather than using the standard button. Such a control is created by an element `<input>` with an attribute `type` whose value is `"image"`. You also need two additional attributes:

- `"src"`: specifies the source of the image file
- `"alt"`: provides alternative text for the image

Example

```
<input type="image" name="btnPanic"
       src="imagePanic.gif" alt="submit" />
```

Remark

An image button is a submit button. Besides, the x and y coordinates of the place where the user has clicked on the button are sent to the server.

The Element <button>

The element <button> is a more recent introduction (than <input>) that allows us to specify what appears on a button between the opening and closing tags.

Example

```
<button type="submit"> Submit </button>
<button type="reset">
  <strong> Clear this form </strong>
</button>
<button type="button"> Compute something </button>
<button type="button">
  <img src="images/submit.gif" alt="Another submit" />
</button>
```

Warning

Be careful with this element as browsers may treat it differently.

The user can toggle between on and off positions by clicking on a checkbox. Such a control is created by an element `<input>` with an attribute `type` whose value is `"checkbox"`. This control is ideal for letting the user to:

- provide a simple yes or no response with one checkbox
- select several items from a list of possible options

Example

```
<input type="checkbox" name="chkAccept" checked="checked" />
   I accept the terms ...  <br />
<input type="checkbox" name="chkSkills" value="css"/> CSS <br />
<input type="checkbox" name="chkSkills" value="php"/> PHP <br />
```

Remarks

In the absence of a `value` attribute, the value is automatically `"on"`.
Several checkboxes may share the same name.

The user can only select one option at a given time from several options declared in a group. Such a control is created by an element `<input>` with an attribute `type` whose value is `"radio"`.

Example

```
<input type="radio" name="radClass"
  value="First" /> First class
<input type="radio" name="radClass"
  value="Business" /> Business class
```

Remark

A group of radio buttons corresponds to all radio buttons sharing the same value of the `name` attribute.

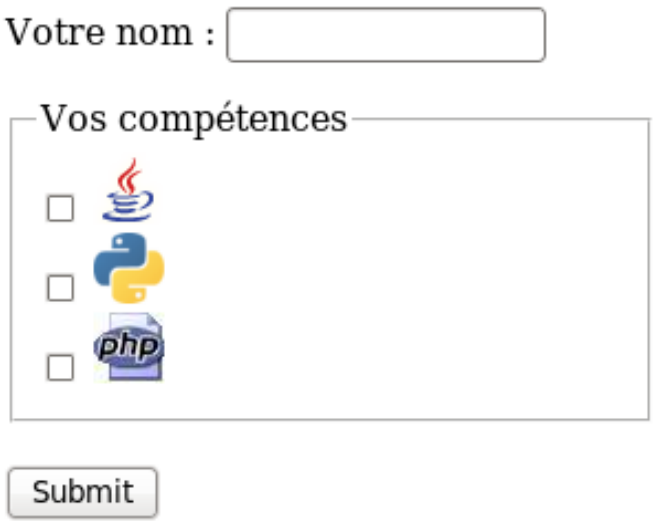
Warning

You must have at least two radio buttons in each group.

Exercice

imageForm.html

On souhaite écrire le formulaire suivant qui comporte un champ de saisie, trois cases à cocher et un bouton submit. Pour chaque case à cocher, la description est donnée ici par une image. Les images sont des icones de taille 32x32 pixels et ont pour nom `java.png`, `python.png` et `php.png`.



Warning

Dans la "vraie vie", on évitera d'utiliser une image plutôt que du texte pour une utilisation aussi sensible.

Select Lists

select.html

A select list allows the user to select one item from a drop-down menu, and can take up far less space than a group of radio buttons. Such a control is created by an element `<select>` containing several `<option>` elements.

Example

```
<select name="selColor">
  <option value="red"> Red </option>
  <option value="green"> Green </option>
  ...
</select>
```

Among attributes for `<select>`, we have:

- `size`: specifies the number of rows in the list that can be visible at the same time; this is used to present a scrolling select list
- `multiple`: allows a user to select multiple items in the list; you must write `multiple="multiple"`

It is possible to group options using the `<optgroup>` element which acts as a container for several options. The `<optgroup>` element can carry a `label` attribute.

Example

```
<select name="sellInformation">
  <optgroup label="Hardware">
    <option value="Desktop"> Desktop computers </option>
    <option value="Laptop"> Laptop computers </option>
  </optgroup>
  ...
```

Remark

To initially have one option selected, you must write

```
<option selected="selected" ...
```

File Upload Controls

fileHidden.html

You can propose a form control in order to allow the user to upload a file to the server. Such a control is created by an element `<input>` with an attribute `type` whose value is `"file"`.

Example

```
<form enctype="multipart/form-data" ...>
  <input type="file" name="fileUpload" accept="image/*" />
  ...
```

Remark

The `enctype` attribute of the `<form>` alement indicates here that only files with an image format can be uploaded; but this is partially supported by browsers.

Warning

The `method` and `enctype` attributes of the `<form>` element must be given the values `"post"` and `"multipart/form-data"`, repectively.

Hidden Controls

fileHidden.html

To pass information between pages without the user seeing it, you can use an hidden control. Such a control is created by an element `<input>` with an attribute `type` whose value is `"hidden"`.

Example

```
<input type="hidden" name="hidName" value="toto" />
```

Warning

The information contained in hidden controls must not be confidential because it is visible in the source code.

The Element <label>

labelLegend.html

For any form control that does not have a `label` attribute, it is good practice to use the `<label>` element. The value of the `for` attribute of the `<label>` element must be the value of the `id` attribute of the form control.

Example

```
<tr>
  <td> <label for="idName"> User name </label> </td>
  <td> <input type="text" id="idName" name="txtName" /> </td>
```

Remark

For a text input, it is usually better to put the label on the left, whereas for checkboxes and radio buttons, it is usually better to have it on the right.

Remark

As an alternative, the `<label>` element can be used as a container, but this does not allow you to completely control where the label appears.

Elts <fieldset> and <legend>

For large forms, the Elements <fieldset> and <legend> help you to group controls:

- <fieldset> creates a border
- <legend> adds a caption

Example

```
<fieldset>
  <legend> <em> Contact Information </em> </legend>
  <label> First name: <input type="text" name="txtFName" />  ...
```

Warning

When used, the <legend> element should always be the first child of the <fieldset> element.

Elements that a user can interact with can receive focus. To control the order in which elements can gain focus, use the `tabindex` attribute whose value is a number between 0 and 32767 and which forms part of the tabbing order.

Example

```
<input type="checkbox" name="chkN" value="1" tabindex="3" /> One
<input type="checkbox" name="chkN" value="2" tabindex="7" /> Two
```

Remark

Give your `tabindex` attributes, values of 1 or higher, because 0 is for elements without the `tabindex` attribute.

Warning

A disabled element cannot gain focus.

Access keys are just like keyboard shortcuts.

Example

```
<legend accesskey="c">
  Contact Information (SHIFT + ALT + C)
</legend>
...
<input accesskey="e" type="submit" value="Enter" />
```

The following elements can carry an `accesskey` attribute:

`<a>`, `<area>`, `<button>`, `<input>`, `<label>`, `<legend>`, `<textarea>`

The following elements can carry a `tabindex` attribute:

`<a>`, `<area>`, `<button>`, `<input>`, `<object>`, `<select>`, `<textarea>`

Disabled and Read-only Controls

Some form controls can carry attributes called `disabled` and `readonly`.

- `readonly`: specifies that the user cannot change the value of the form control (but a script can). A name/value pair for the form control is sent to the server when the form is submitted.
- `disabled`: specifies that the user cannot change the value of the form control (but a script can re-enable it). A name/value pair for the form control is not sent to the server when the form is submitted.

These attributes are set as follows:

```
readonly="readonly"
disabled="disabled"
```