HTML Introduction

Fajiang Yu, fjyu@whu.edu.cn

School of Computer, Wuhan University

2017.3

Agenda

Agenda

Agenda

A brief history of HTML

HTML was originally developed by Tim Berners-Lee while at CERN.

HTML 2.0 (November 1995) was developed under the aegis of the Internet Engineering Task Force (IETF) to codify common practice in late 1994.

HTML+ (1993) and HTML 3.0 (1995) proposed much richer versions of HTML. Drafts.

HTML 3.2 (January 1997), W3C HTML Working Group.

HTML 4.0, 24 April 1998.

HTML 4.01 is a revision of HTML 4.0, 24 December 1999.

HTML 5, since 2008

Agenda

SGML and HTML

SGML is a system for defining markup languages.
HTML is one example of a markup language.
An SGML application (a markup language) is characterized by:

- An SGML declaration.
- A **document type definition (DTD)**, defines the **syntax** of markup constructs.
- A specification that describes the **semantics**.
- Document instances.

HTML document type definition mainly declares

- element types
- and character references.

Agenda

HTML Elements

Each element type declaration generally describes three parts:

- **a start tag**
  the element's name appears in the start tag
  (written `<element-name>`)
- **content**
- **an end tag**
  written `</element-name>`

Some HTML element types allow authors to omit end tags.
A few element types also allow the start tags to be omitted.
Some HTML element types have no content.

Elements must be **properly nested**.
HTML element names are always **case-insensitive**.
Note: *elements are not tags*.

Agenda

HTML Element Attributes

Elements **may** have associated properties, called attributes, which **may** have values.

Attribute/value pairs appear before the final ”>” of an element’s start tag.
Any number of attribute value pairs, separated by **spaces**, may appear **in any order**.

All attribute values are delimited using **either** double quotation marks **or** single quotation marks.
Single quote marks can be included within the attribute value when the value is delimited by double quote marks, and *vice versa*.
In certain cases, authors may specify the value of an attribute **without** any quotation marks. The attribute value can not contain some special characters.

Attribute names are **always** case-insensitive.
Attribute values are **generally** case-insensitive.

Agenda

HTML Character References

Hardware or software configurations do not allow users to input some document characters directly.
They begin with a "&" sign and end with a semi-colon (;).
Character references in HTML may appear in two forms:

- Numeric character references (either decimal or hexadecimal).
- Character entity references.

< >	< >
é è ñ	é è ñ
™ ©	™ ©
π δ Δ	π δ Δ
И	И
" &	" &

List of character references `http://www.w3schools.com/tags/ref_entities.asp`

It is possible to eliminate the final ";" in some cases.
In other cases it may not be eliminated (e.g., in the middle of a word).

Agenda

HTML Comments

HTML comments have the following syntax:

```
<!-- this is a comment -->
<!-- and so is this one,
     which occupies more than one line -->
```

Agenda

Agenda

Introduction to the structure of an HTML document

An HTML document is composed of three parts:

1) a line containing **HTML version information**,
2) a declarative header section (delimited by the **head** element),
3) a body, which contains the document's actual content (implemented by the **body** element or the **frameset** element).

White space (spaces, newlines, tabs, and comments) may appear before or after each section.

Introduction to the structure of an HTML document

Example

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
   <head>
      <title>My first HTML document</title>
   </head>
   <body>
      <p>Hello world!</p>
   </body>
</html>
```

Agenda

elements

1. General Information
2. The global structure of an HTML document
   - Introduction to the structure of an HTML document
   - HTML version information
   - The `title` element
   - Meta data
   - The `body` element
   - Headings
   - Rules: the `hr` element
   - Block-level and inline
3. Text
4. List
5. Table
6. Link
7. Objects, Images, and Applets
8. HTML 5 Introduction

HTML version information

HTML 4.01 specifies three DTDs

- HTML 4.01 Strict DTD

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

- HTML 4.01 Transitional DTD

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

- HTML 4.01 Frameset DTD

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

Agenda

The `title` element

Every HTML document must have a `title` element in the head section.

Displayed in the browser's title bar and when bookmarking the page.

Agenda

Meta data

Authors specify machine-readable metadata about HTML documents and other network-accessible resources, which involves two steps

1) Declaring a property and responding value, being done in two ways:
   - From within a document, via the `meta` element.
   - From outside a document, linking to meta data via the `link` element.
2) Referring to a profile where the property and its legal values are defined by using the `profile` attribute of the `head` element.

The meta element

meta and HTTP headers

HTTP servers use the property value specified by the `http-equiv` attribute to create an header in the HTTP response.

```
<meta http-equiv="Expires"
   content="Tue, 20 Aug 1996 14:25:27 GMT">
```

The meta element

meta and search engines

A common use for meta is to specify keywords that a search engine may use to improve the quality of search results.

```
<meta name="keywords"
    content="vacation, Greece, sunshine">
```

The meta element

meta and default information

The following example specifies the character encoding for a document

```
<meta http-equiv="Content-Type"
   content="text/html; charset=ISO-8859-5">

<meta http-equiv="Content-Type"
   content="text/html; charset=gb2312" />
```

Agenda

The body element

Some Attributes

- `background`, an image resource
- `bgcolor`, background color
- `text`, the foreground color for text
- `link`, the color of text marking unvisited hypertext links
- `vlink`, the color of text marking visited hypertext links
- `alink`, the color of text marking hypertext links when selected

Basic HTML data type - **Uniform Resource Identifier (URI)** for an image resource, etc.

Note that URIs include URLs.
URIs in general are **case-sensitive**.

Relative URIs are resolved to **full URIs** using a **base URI**.
The base URI

- is set by the `base` element,

```
<base href="http://www.aviary.com/products/intro.html">
```

- or is given by meta data discovered during a protocol interaction,
- by default is that of the current document. Not all HTML documents have a base URI (e.g., a valid HTML document may appear in an email).

Example

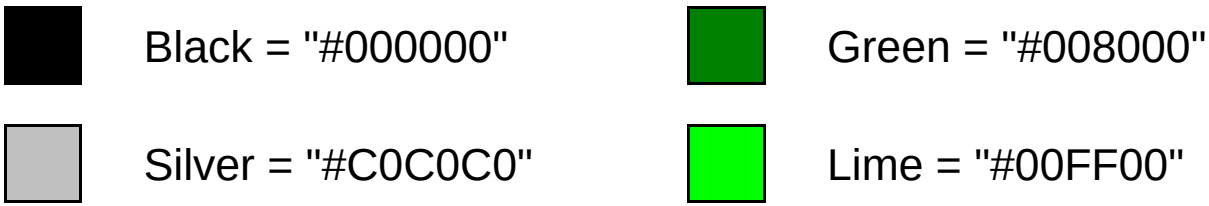
```
<body background="../Images/bg.gif">
```

Basic HTML data type - **Color**

A color value may either be

- a hexadecimal number (prefixed by a hash mark)
- or one of sixteen (constant) color names.

Some color names and sRGB values



Agenda

Headings

The `h1`, `h2`, `h3`, `h4`, `h5`, `h6` elements

Example

```
<h1>Forest elephants</h1>

<h2>Habitat</h2>
```

Agenda

Rules: the `hr` element

Causes a horizontal rule
Start tag: required, End tag: forbidden

Some attributes

- `align` = `left` | `center` | `right`
- `noshade`
  When set, this **boolean** attribute requests that the user agent render the rule in a solid color rather than as the traditional two-color "groove".
- `size`
  This attribute specifies the height of the rule.
- `width`
  Specifies the width of the rule, default width is 100%.

Basic HTML data type - **Length**

HTML specifies three types of length values for attributes:

- Pixels: The value is an integer that represents the number of pixels of the canvas (screen, paper).
- Length: The value is a percentage of the available horizontal or vertical space.
- MultiLength: The value is a relative length, which has the form "`i*`", where "`i`" is an integer.

Agenda

Block-level and inline elements

The distinction is founded on several notions:

- **Content model**
  Generally, block-level elements may contain inline elements and other block-level elements. Generally, inline elements may contain only data and other inline elements.
- **Formatting**
  By default, block-level elements are formatted differently than inline elements. Generally, block-level elements begin on new lines, inline elements do not.
- **Directionality**
  Block-level and inline elements differ in how they inherit directionality information.

Agenda

Agenda

Paragraphs: the p element

Attribute `align`

- `left`: text lines are rendered flush left.
- `center`: text lines are centered.
- `right`: text lines are rendered flush right.
- `justify`: text lines are justified to both margins.

Agenda

Controlling line breaks and space

Forcing a line break: the `br` element
Start tag: required, End tag: forbidden

Prohibiting a line break
The ` ` entity acts as a space where user agents should not cause a line break.

Agenda

Phrase elements

Phrase elements add structural information to text fragments.

- `em`: Indicates emphasis.
- `strong`: Indicates stronger emphasis.
- `cite`: Contains a citation or a reference to other sources.
- `dfn`: Indicates a defining instance of the enclosed term.
- `code`: Designates a fragment of computer code.
- `samp`: Designates sample output from programs, scripts, etc.
- `kbd`: Indicates text to be entered by the user.
- `var`: Indicates an instance of a variable or program argument.
- `abbr`: Indicates an abbreviated form (e.g., ARG, CN, etc.).
- `acronym`: Indicates an acronym (e.g., WWW, HTML, etc.).

Agenda

Quotations

`blockquote` element is for long quotations (block-level content).

`q` is intended for short quotations (inline content) that don't require paragraph breaks.

Agenda

Preformatted text

The `pre` element
tells visual user agents that the enclosed text is ”preformatted”.

Attribute `width`
provides a hint to visual user agents about the desired width of the formatted block.

Agenda

Subscripts and superscripts

The `sub` and `sup` elements

Agenda

Font style elements

Rendering of font style elements depends on the user agent.

- `tt`: Renders as teletype or monospaced text.
- `i`: Renders as italic text style.
- `b`: Renders as bold text style.
- `big`: Renders text in a ”large” font.
- `small`: Renders text in a ”small” font.
- `strike` and `s`: Render strike-through style text.
- `u`: Renders underlined text.

Font modifier elements

The `font` element sets the font size, color and face for its text contents. Some attributes

- `size`, possible values:
  An integer between 1 and 7.
  A relative increase in font size. The value "+1" means one size larger. The value "-3" means three sizes smaller. All sizes belong to the scale of 1 to 7.

  The `basefont` element sets the base font size (using the size attribute). If `basefont` is not used, the default base font size is 3.
- `color`
- `face`
  This attribute defines a comma-separated list of font names th user agent should search for in order of preference.

Font face

Generic font families

Family	Example (browser dependent)
serif	The quick brown fox jumps over the lazy dog. 0123456789
sans-serif	The quick brown fox jumps over the lazy dog. 0123456789
cursive	The quick brown fox jumps over the lazy dog. 0123456789
fantasy	**The quick brown fox jumps over the lazy dog. 0123456789**
monospace	The quick brown fox jumps over the lazy dog. 0123456789

Generic fonts (e.g. Latin fonts)

- 'serif' (e.g., Times)
- 'sans-serif' (e.g., Helvetica)
- 'cursive' (e.g., Zapf-Chancery)
- 'fantasy' (e.g., Western)
- 'monospace' (e.g., Courier)

Agenda

List

Unordered lists (`ul`), ordered lists (`ol`) and list items (`li`)

some attributes

- `type`
  This attribute sets the style of a list item.
- `start`
  For `ol` only, Specifies the starting number of the first item in an ordered list.
- `value`
  For `li` only, sets the number of the current list item.
- `compact`
  When set, this **boolean** attribute gives a hint to visual user agents to render the list in a more compact way.

List

`type` attribute value

For the `ul` element, possible values are `disc`, `square`, and `circle`.

For the `ol` element, possible values are summarized in the table below:

Type	Numbering style	
1	arabic numbers	1, 2, 3, ...
a	lower alpha	a, b, c, ...
A	upper alpha	A, B, C, ...
i	lower roman	i, ii, iii, ...
I	upper roman	I, II, III, ...

Agenda

Agenda

The `table` element

The `table` element contains all other elements that specify caption, rows, content, and formatting.

some attributes

- `align` = `left` | `center` | `right`
- `width`
- `bgcolor`
- `frame`, `rules`, `border`
- `cellspacing`, `cellpadding` (cell margins)

The table element

The `frame` attribute specifies which sides of the frame surrounding a table will be visible. Possible values:

- `void`: No sides. This is the default value.
- `above`: The top side only.
- `below`: The bottom side only.
- `hsides`: The top and bottom sides only.
- `vsides`: The right and left sides only.
- `lhs`: The left-hand side only.
- `rhs`: The right-hand side only.
- `box`: All four sides.
- `border`: All four sides.

The table element

The `rules` attribute specifies which rules will appear between cells within a table. Possible values:

- `none`: No rules. This is the default value.
- `groups`: Rules will appear between row groups and column groups.
- `rows`: Rules will appear between rows only.
- `cols`:Rules will appear between columns only.
- `all`: Rules will appear between all rows and columns.

The `table` element

The `border` attributes specifies the width (in pixels only) of the frame around a table.

The `cellspacing` attribute specifies how much space the user agent should leave between the left side of the table and the left-hand side of the leftmost column, the top of the table and the top side of the topmost row, and so on for the right and bottom of the table. The attribute also specifies the amount of space to leave between cells.

The `cellpadding` attribute specifies the amount of space between the border of the cell and its contents.

Agenda

Table Caption

The `caption` element is only permitted immediately after the `table` start tag. A `table` element may only contain one `caption` element.

`align` attribute
`align` = `top` | `bottom` | `left` | `right`

Agenda

Table rows

The `tr` element acts as a container for a row of table cells.

some attributes

- `bgcolor`
- `align`, `char`, `charoff`, `valign` (cell alignment)

Agenda

Table cells

The `th` element defines a cell that contains header information.
The `td` element defines a cell that contains data.
Cells may be empty.

some attributes

- `rowspan`
  Specifies the number of rows spanned by the current cell.
- `colspan`
  Specifies the number of columns spanned by the current cell.
- `width`
- `height`
- `bgcolor`
- `align`, `char`, `charoff`, `valign` (cell alignment)

Agenda

Horizontal and vertical alignment

`align` = `left` | `center` | `right` | `justify` | `char`

`char`: Align text around a specific character.

`valign` = `top` | `middle` | `bottom` | `baseline`

`baseline`: All cells in the same row as a cell whose valign attribute has this value should have their textual data positioned so that the first text line occurs on a baseline common to all cells in the row.

The `char` attribute specifies a single character within a text fragment to act as an axis for alignment.

The `charoff` attribute specifies the offset to the first occurrence of the alignment character on each line.

Agenda

Row Group and Column Group

Row group: the `thead`, `tfoot`, and `tbody` elements

Column group: the `colgroup` and `col` elements

Agenda

Agenda

Introduction to Link and Anchor

A link is a connection from one Web resource to another.
Although a simple concept, the link has been one of the primary forces driving the success of the Web.

A link has two ends – called **anchors** – and a **direction**.
The link starts at the "source" anchor and points to the "destination" anchor, which may be any Web resource (e.g., an image, a video clip, a sound bite, a program, an HTML document, an **element** within an HTML document, etc.).

Agenda

The a Element

The `a` element may only appear in the body.
Each `a` element defines an source (a link) or destination anchor.

- The `name` attribute defines one destination anchor, which is an **element** within an HTML document.
- The `href` attribute makes this anchor the source anchor of exactly one link and its responding destination anchor.

Authors may set the `name` and `href` attributes simultaneously in the same `a` instance.
The `a` element has content (text, images, etc.).

some attributes

- `name`
- `href`
- `target` = `_blank` | `_self` | `_parent` | `_top`

The destination anchor being an element

If the destination anchor is a HTML element, it must be given an **anchor name** and any URI addressing this anchor must include the name, the URIs that designate anchors contain a "#" character followed by the anchor name.

Here are some examples of such URIs:

- An absolute URI
  `http://www.mycompany.com/one.html#anchor-one`
- A relative URI
  `./one.html#anchor-one` or `one.html#anchor-one`
- When the link is defined in the same document: `#anchor-one`

Special URI: `mailto:joe@someplace.com`

Anchors with the `id` attribute

Destination anchors in HTML documents may be specified by the `id` attribute at the start tag of any element (including the `a` element).

Example

```
<h2 id="section2">Section Two</h2>
<a id="anchor-two">photo of my family at the lake</a>
```

The `id` and `name` attributes share the same name space.
This means that they cannot both define an anchor with the same name in the same document.
It is permissible to use both attributes to specify an element's unique identifier for some elements. When both attributes are used on a single element, their values must be **identical**.

Anchors with the `id` attribute

Use `id` or `name`?

- The `id` attribute can act as more than just an anchor name.
- Some older browsers don't support anchors created with the `id` attribute.
- The `name` attribute allows richer anchor names.

Agenda

The `link` element

The `link` element defines a relationship between the current document and another resource.
The `link` element may only appear in the head of a document.
`link` has no content.

some attributes

- `type`
- `rel`
- `rev`

The `link` element

Forward and reverse links

The `rel` attribute specifies a forward link.
The `rev` attribute specifies a reverse link.

```
<link href="docB" rel="foo">
<link href="docA" rev="foo">
```

Links and external style sheets

Links and search engines

Favorites icon ?

Agenda

Agenda

Introduction to objects, images, and applets

HTML's multimedia features allow authors to include images, applets (programs that are automatically downloaded and run on the user's machine), video clips, and other HTML documents in their pages.

HTML 4 introduces the `object` element, which offers an all-purpose solution to generic object inclusion.
The term "object" is used to describe the things that people want to place in HTML documents.

Introduction to objects, images, and applets

The `object` element subsumes some of the tasks carried out by other elements.

Type of inclusion	**Specific element**	**Generic element**
Image	`img`	`object`
Applet	`applet` (*Deprecated*)	`object`
Another HTML doc	`iframe` (*Deprecated*)	`object`

Images and other included objects may have hyperlinks associated with them.

Agenda

The `img` element

The `img` element has no content.
Start tag: required, End tag: forbidden

some attributes

- `src`
- `alt`, alternate text
- `align`
- `width`, `height`
- `border`,
- `hspace`, `vspace`

The `img` element

`align` attribute

The following values for align concern the **object**'s position with respect to surrounding text

- `bottom`
- `middle`
- `top`

Two other values, `left` and `right`, cause the image to float to the current left or right margin.

Image maps

Client-side image maps: the `map` and `area` elements.

Agenda

The `object` element

Most user agents have built-in mechanisms for rendering common data types such as text, GIF images, etc..
The `object` element allows authors to control whether data should be rendered externally or by some program, specified by the author, that renders the data within the user agent.
In the most general case, an author may need to specify three types of information, but authors may not have to specify all three at once.

- The implementation of the included object, via the `object` element.
- The data to be rendered, via the `object` element.
- Additional values required by the object at run-time, via the `param` element.

The `object` element

some attributes

- `classid`, specify the location of an object's implementation via a URI.
- `codebase`, specify the base path used to resolve relative URIs specified by the `classid`, `data`, and `archive` attributes.
- `codetype`, specify the content type of data expected when downloading the object specified by `classid`.
- `data`, specify the location of the object's data. (Inline vs. external data.)
- `type`, specify the content type for the data specified by `data`.
- `archive`, specify a space-separated list of URIs for archives containing resources relevant to the object, which may include the resources specified by the `classid` and `data` attributes.
- `declare`, when present, this `boolean` attribute makes the current `object` definition a declaration only.

The object element

Example

```
<object data="TheEarth.gif" type="image/gif"> </object>
<object data="TheEarth.mpeg" type="application/mpeg"> </object>
```

Examples of content types include:
”text/html”, ”image/png”, ”image/gif”,
”video/mpeg”, ”text/css”, and ”audio/basic”.

Object initialization: the `param` element

Including an applet

```
<object classid="http://www.miamachina.it/analogclock.py">
   <param name="height" value="40" valuetype="data">
   This user agent cannot render Python applications.
</object>
```

Global naming schemes for objects

The location of an object's implementation is given by a URI.
Some objects might employ other naming schemes.

```
<object classid="java:program.start"> </object>

<object classid="clsid:663C8FEF-1EF9-11CF-A3DB-080036F12502"
   data="http://www.acme.com/ole/clock.stm">  </object>
```

Agenda

HTML 5 Introduction

HTML 5 will be the new standard for HTML, HTML 5 is not yet an official standard. HTML 5 is still a work in progress.

However, the major browsers (Safari, Chrome, Firefox, Opera, Internet Explorer) support many of the new HTML 5 elements and APIs.

Some rules for HTML5 were established:

- New features should be based on HTML, CSS, DOM, and JavaScript
- The need for external plugins (like Flash) should be reduced
- Error handling should be easier than in previous versions
- Scripting has to be replaced by more markup
- HTML5 should be device-independent
- The development process should be visible to the public

HTML 5 Introduction

New Elements:

- New <canvas> Element
- New Media Elements
  - <audio>, <video>, <source>, <track>, <embed>
- New Form Elements
  - <datalist>, <keygen>, <output>
- New Semantic/Structural Elements
  - <header>, <nav>, <section>, <main>, <article>, <aside>, <footer>,
  - <details>, <summary>, <figure>, <figcaption>, <mark>, <time>, <bdi>,
  - <wbr>, <dialog>, <command>, <meter>, <progress>, <ruby>, <rt>, <rp>

HTML 5 Introduction

Another new features:

- Geolocation
- Drag and Drop
- Web Storage
- Applicaiton Cache
- Web Workers
- Sever-Sent Events

HTML 5 Introduction

The following HTML 4.01 elements has been removed from HTML5:

- <acronym>, <applet>
- <basefont>, <big>, , <dir>, <font>
- <frame>, <frameset>, <noframes>
- <strike>, <tt>

HTML 5 Introduction

In HTML 5 there is only one declaration, and it is very simple:

```
<!DOCTYPE html>
```

HTML 5 Audio & Video

```
<!DOCTYPE html>
<html>
<body>

<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>

<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>

</body>
</html>
```

HTML 5 Canvas

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100"
        style="border:1px solid #c3c3c3;">
 Your browser does not support the HTML5 canvas tag.
</canvas>

<script>
 var c=document.getElementById("myCanvas");
 var ctx=c.getContext("2d");
 ctx.fillStyle="#FF0000";
 ctx.fillRect(0,0,150,75);
</script>

</body>
</html>
```

HTML 5 Inline SVG

```
<!DOCTYPE html>
<html>
<body>

<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="190">
   <polygon points="100,10 40,180 190,60 10,60 160,180"
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;">
</svg>

</body>
</html>
```

Thank You!
Any Questions?