

CGS 3175: Internet Applications Fall 2007

XHTML – Part 1

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
<http://www.cs.ucf.edu/courses/cgs3175/fall2007>

School of Electrical Engineering and Computer Science
University of Central Florida



Overview of Markup Languages

- A **markup language** is simply a set of rules that defines the layout, format, or structure of text within a document.
- After markup instructions are added to a document, the document must be read, or processed, by a program that know how to interpret the markup elements.
- Markup languages existed long before the World Wide Web. They were used primarily in the publishing industry prior to their adaptation to computer programming.
- Work began in the 1960s to develop a standardized document markup language that would be platform independent.



Overview of Markup Languages

- The **Standard Generalized Mark Language (SGML)** was the result of this initiative and was the first standardized markup language to gain acceptance.
- It wasn't until the Web exploded in popularity in the mid-1990s that the benefits of an open standard for markup languages became overwhelmingly apparent.



Overview of Markup Languages

- SGML is the ancestor of, and provides the framework for, current Web markup languages, including XHTML, XML, and HTML.
- SGML was developed as a markup language for large documents, such as technical documentation.
- It was adopted as an international standard by the [International Organization for Standards \(ISO\)](#) in 1986, and has been widely used by many industries, including the automotive industry, the health care industry, the IRS, and the US Department of Defense, for large scale documentation projects.
- SGML is extremely complex, and thus very expensive. SGML has proved useful mainly to organizations that have the expertise and budget to implement the expansive SGML specification.



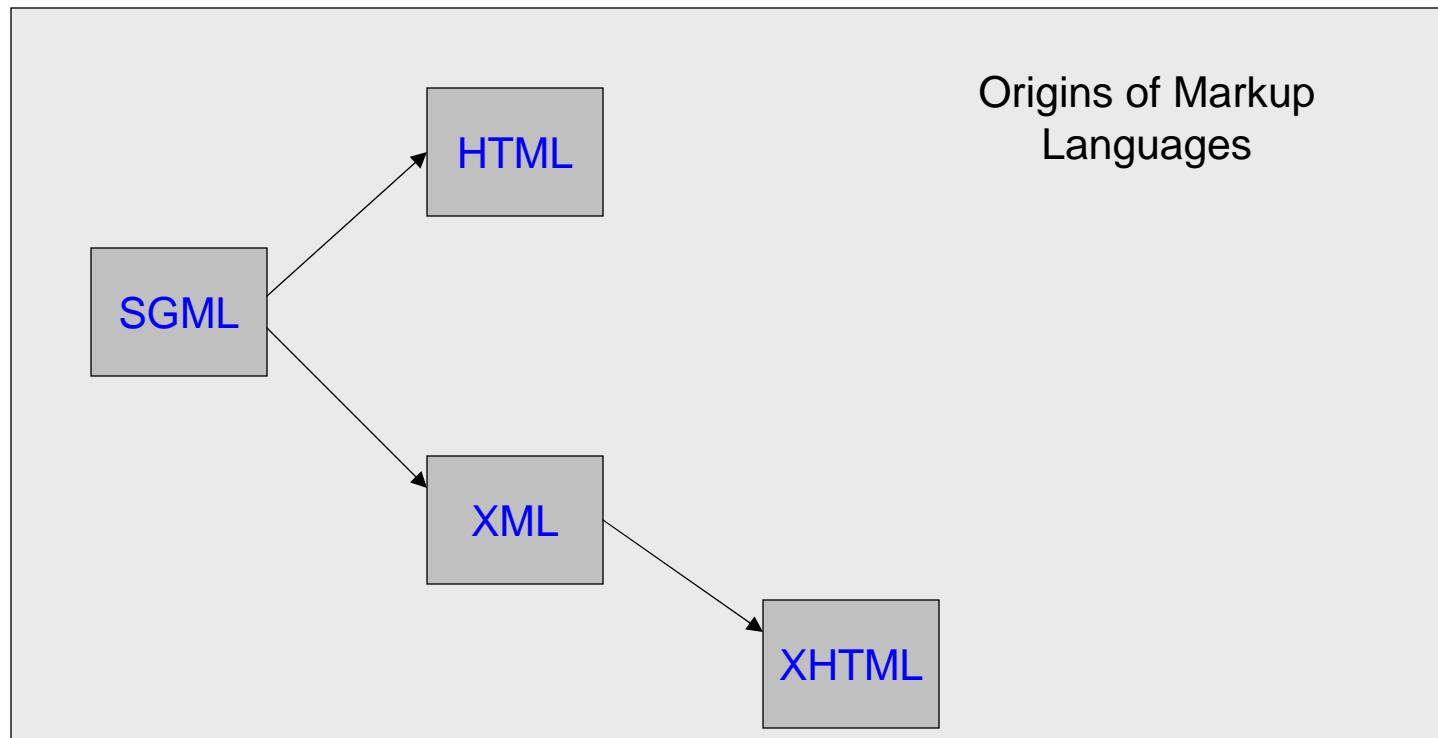
Overview of Markup Languages

- When the World Wide Web was in its infancy in the late 1980s and early 1990s, SGML was the perfect tool for building the markup language that would be used to create documents for this new medium.
- **Hypertext Markup Language (HTML)** was developed as a lightweight SGML by researchers at CERN (European Organization for Nuclear Research) in the early 1990s.
- CERN had been involved with working on the SGML specification for many years.
- HTML was much smaller than SGML and gained widespread acceptance very quickly. It provided content developers with a portable document format that was not tied to any particular program or platform, and being an open standard, it was completely free to use.



Overview of Markup Languages

- HTML was adopted shortly after its development by the W3C (www.w3c.org), which continues to maintain the HTML specification, currently at version 4.01.



Overview of Markup Languages

- Because HTML documents are simple text documents embedded with markup elements, they are completely portable across platforms and programs.
- HTML documents can be displayed using any program running on any operating system that knows how to interpret the HTML language.
- This gives developers an incredible amount of flexibility and allows them to move files freely among platforms and programs.
 - For example, an HTML file created with a Mac text editor would look the same when open in a Windows text editor, and would be displayed the same when viewed using Netscape Navigator or Internet Explorer, either on a Mac or a PC.



Limitations of HTML

- As web technologies continue to advance at a very rapid pace, HTML has been pushed to its limits by developers and vendors.
- Somewhat ironically, the traits of HTML that helped build its popularity – its small size, limited number of elements, and ease of use – have become its downfall.
- HTML is a fixed specification with a finite set of elements. It is not extendable, and as a result of this limitation, Web developers and software vendors have stretched the usefulness of HTML almost to a breaking point.
- Browser vendors such as Microsoft and Netscape, have added proprietary features and additional HTML elements to their browsers based on demands for more functionality, but in so doing, they have compromised one of the most important benefits that HTML has to offer – portability. Given these proprietary additions to particular browsers, HTML pages developed for use in one Web browser may not display the same way when displayed in another browser or on another platform.



Limitations of HTML

- Even though HTML has syntax rules, Web browsers have always been fairly forgiving in that they allow poorly written HTML code to be displayed.
- If a browser encounters code that is written incorrectly, it can usually compensate and will display the correct output.
- This is unfortunate in that it has allowed millions of poorly written pages to be published on the Web. It is also unfortunate because the newest breed of Web clients, including cell phones, PDAs, and other devices are not as forgiving as Web browsers and will either display the pages incorrectly or not at all.



Limitations of HTML

- Despite all of its benefits and the revolution it helped to spark, HTML has some serious limitations that inhibit its future usefulness as Internet technologies continue to advance. These include:
 - HTML elements are primarily used for defining presentation and formatting styles, but they do not provide any information about the data itself (data without context).
 - HTML has a finite number of elements, which cannot be extended or customized.
 - HTML does not force documents to adhere to strict syntax rules, making it difficult for parsers to interpret poorly written code.
 - Until recently, Web browsers were the primary client program used to view content on the Web. However, HTML's limitations are further being felt with the introduction of new technologies, such as wireless devices like cell phones, voice, and speech programs, and PDAs.
 - Even though HTML has syntax rules, Web browsers have always been fairly forgiving in that they allow poorly written HTML code to be displayed.



Limitations of HTML

- As a result of these limitations, the most recent version of HTML, HTML 4.01, will be the last.
- The first version of XHTML, 1.0, was released by the W3C in 2000 as the successor to HTML. This is the reason that the terms HTML and XHTML can be used interchangeably.
- **Extensible Hypertext Markup Language (XHTML)** is the modern “version” of HTML.
- XHTML 1.0 is the current standard adopted by W3C in 2002.



XML – The Future of Web Markup Languages

- The limitations of HTML made it very clear that a new and better language was necessary for formulating Web documents.
- In addition, many companies were adding transactional functionality to their Web sites, such as allowing visitors to purchase items and services online.
- This marked a radical departure from the first generation of websites, which mainly provided static information that was easily stored as text. These new websites relied heavily on data gathered from different sources, such as databases, news feeds, and other Web sites.
- The resulting language was the **Extensible Markup Language (XML)**. XML is an extensible language because, unlike HTML, it allows users to define their own tags.



XML – The Future of Web Markup Languages

- The first recommendation, XML 1.0, was released in 1998.
- The XML family of technologies was developed to separate document data from presentation and to give developers the ability to extend the element sets of XML languages as needed.
- XML itself is **not** a language – it is a **meta language**. A meta language is a set of rules used for building markup languages.
- Structured languages can be developed that describe certain types of data rather than just the presentation of the data. Such structured languages include elements that describe documents containing information about an account, an item, a service, or a transaction.
- XHTML is an application of XML that is used for formatting Web documents. There are many other XML languages, some still under development, such as RSS (Really Simple Syndication), MathML, GraphML, Scalable Vector Graphics, and MusicXML.



Some Of The Benefits Of XML

- It allows data to be self-describing, as opposed to being limited by a pre-defined set of elements.
- You can provide rules for XML elements that limit the type of data an element can contain, such as only letters, only numbers, or a certain number of characters.
- Lets you create custom data structures for industry-specific or company-specific needs.
- Because XML describes data, you can present the data any number of ways by applying different presentation styles.
- It provides a rich set of tools for linking.
- You can use it to interchange data between proprietary formats and between databases or data structures.
- You can use the tools to defines a standard syntax for many markup languages.
- It has much more robust and reliable data searching capabilities than HTML.



Some Of The Benefits Of XML

- As its name implies, XHTML is extensible, meaning that its element set is not finite like the element set of HTML.
- Like any XML language, XHTML can be extended to add elements if needed or it can incorporate elements from various XML languages into its element set.
- Keep in mind, however, that if your goal is to keep XHTML compatible with current browsers (certainly one of our goals in this class), you may not be able to use some of its more advanced features. But if your documents are written in XHTML now, you will be able to integrate these features with ease as soon as they become mainstream.



XHTML Document Building Blocks

- Like any language, XHTML has a number of building blocks that are used to create complete documents.
 - The English language, for example, is made up of nouns, verbs, adjectives, adverbs, prepositions, and so on, which are used in conjunction with each other to form sentences and paragraphs.
- Both HTML and XHTML provide language building blocks that can be added to any text document. Web browsers know how to interpret these elements in order to present the document based on formatting rules.

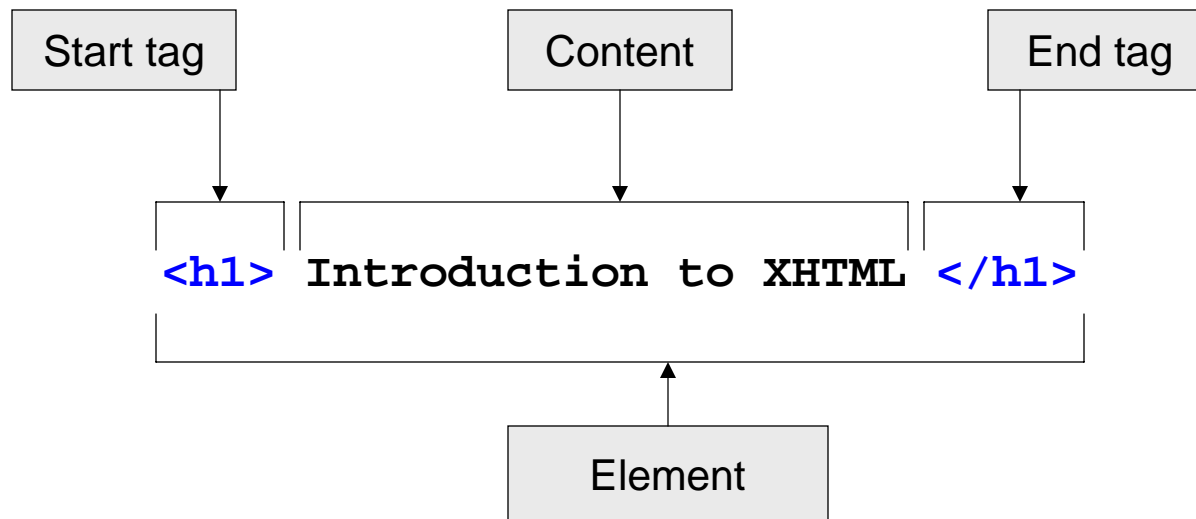


XHTML Elements

- XHTML **elements** are the core components of XHTML documents and are used to describe the data in a document.
 - Elements are like nouns in the English language.
- **Elements** are the **markup**, or formatting instructions, of the XHTML document.
- Elements define the text styles, formatting links, and other pieces of the document.
- Elements and **tags** are sometimes used interchangeably, but strictly speaking, a tag is a piece of an element.



XHTML Elements



- All elements, except for empty elements, consist of three pieces: a **start tag**, **content**, and an **end tag**.
- XHTML element names must be written in lowercase letters.



Empty Elements

- Empty elements are used primarily to describe pieces of data that don't contain any content.
- For example, some common empty elements in HTML are:
 - `
` for line break
 - `` for image
 - `<p>` for paragraph
- In XML and XHTML, all elements must have a start tag and end tag.
- The XML specification provides a shortcut for writing an empty element using a single tag. This is shown below:
 - `
` for line break
 - `` for image
 - `<p />` for paragraph

NOTE: The practice of adding the extra space is **not** part of the syntax of either XML or XHTML. The reason for the space is to make this code compatible with older versions of Web browsers. Because HTML does not require end tags for these elements, most older browsers do not know how to handle the new empty element syntax. The extra space allows these browsers to interpret the syntax correctly.



Attributes

- XHTML **attributes** are pieces of information that help to describe elements. Some elements have required attributes, others are optional and depend on the content that is being marked up.
- Attributes are referred to as **name-value pairs** and have the following syntax: The name of the attribute is on the left, followed by an equal sign, then the value.

`name = "value"`

- Here are a few examples:

```
<a href="http://mark.com">Click here</a>
```

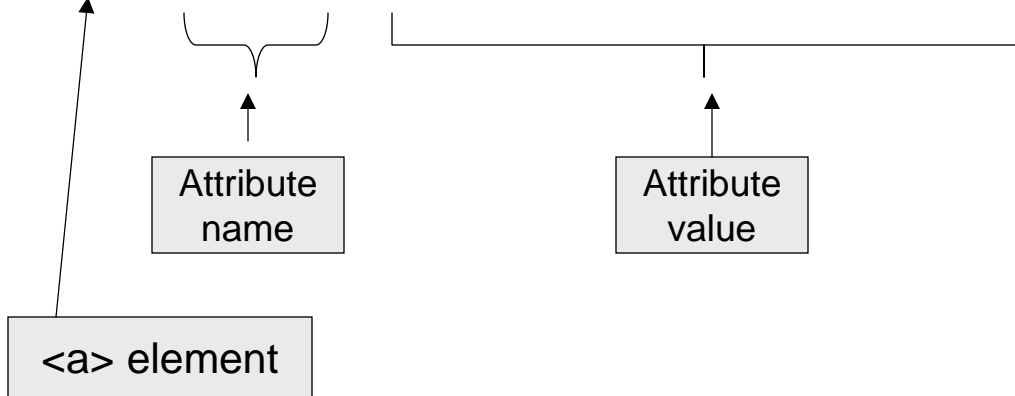
```

```



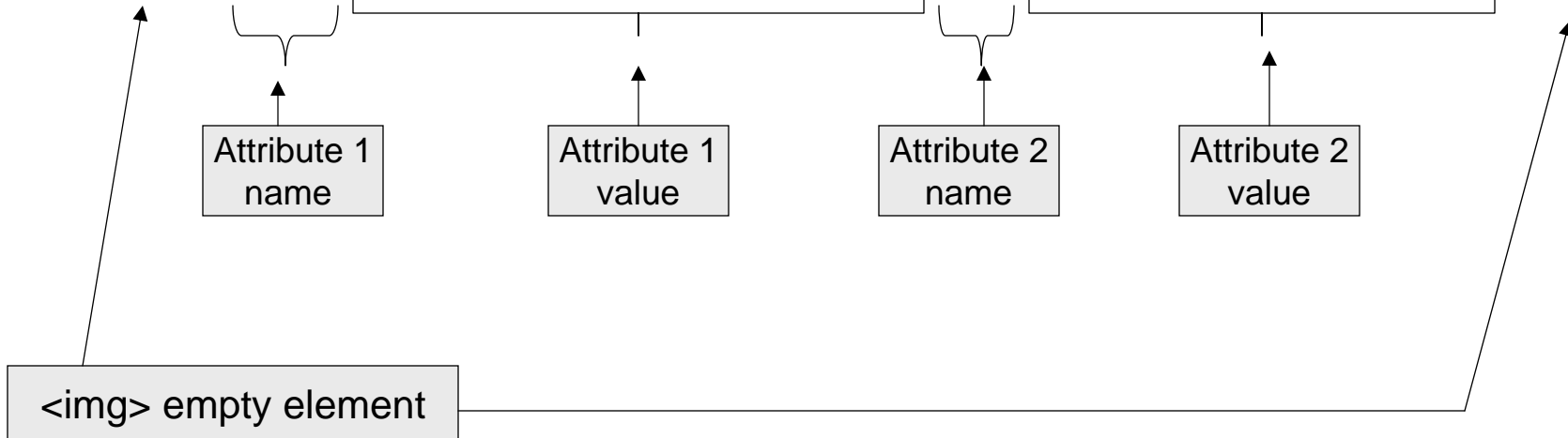
Attributes

```
<a href="http://mark.com">Click here</a>
```



```

```



Attribute Rules

- We'll see more rules later, but here are a few rules about XHTML attributes:
 1. Attributes are always contained within the start tag of an element.
 2. Names must be in lowercase letters.
 3. Attributes must have a value that is surrounded by quotes.



XHTML Core Attributes

- In the XHTML 1.0 specification, there are a set of **core attributes** that can be used with most XHTML elements.
- They cannot be used in `<base>`, `<head>`, `<html>`, `<meta>`, `<param>`, `<script>`, `<style>`, and `<title>` elements.
- The core attributes are:
 1. **id** – document wide unique id.
 2. **class** – list of classes of the element.
 3. **style** – associated style information.
 4. **title** – advisory title amplification.



XHTML Comments

- **Comments** in XHTML are notations that are ignored by programs and parsers.
- The syntax of an XHTML comment is identical to HTML and XML comments.
- You can use comments to document your code, add additional information about a piece of data, add visual breaks, or add information that other people working on or using your document would find useful.
- The following is an XHTML comment:

```
<!-- This is a comment --->
```



The XML Declaration

- The XML declaration defines which version of XML the document is using.
- Stating which version of XML is being used is important in order not to confuse parsers and programs as more versions of XML are developed.
- **The XML declaration always appears as the first line in an XML document. It cannot be preceded by any blank lines or white space.**
- The XML declaration tag begins with `<?xml` and ends with `?>`.
- The XML declaration can contain the following three attributes: `version`, `encoding`, and `standalone`.
- Although the XML declaration is optional in XML and XHTML, it is good practice to at least declare the `version` of XML being used:

```
<?xml version="1.0"?>
```



The XML Declaration

- The second piece of information that the declaration can contain is the **encoding** attribute, which defines the character set that the document uses.
- A character set is a grouping of characters, such as the Latin character set, a character set of symbols, or a character set of Greek letters.
- The encoding attribute is optional in the XML declaration, and if no character set is defined, the default is UTF-8, which is the 8-bit Unicode character encoding scheme.
 - All XML processors are required to handle UTF-8, which is why it is the default character set.
- Other character sets include UTF-16, UTF-32, and ISO-10646-UCS-2. There are literally thousands of character sets available for use.



The XML Declaration

- The third piece of information that the declaration can contain is the **standalone** attribute. The value of this attribute must be either **yes** or **no**.
- This attribute is also optional, and the default value is **no**.
- This attribute tells the processor whether this document contains all of the pertinent information within itself or relies on external **Document Type Definitions (DTDs)** for its declarations.
- Setting the value to **yes** tells the processor that everything needed to process the document is within the document, and to ignore any references to external files. Setting the value to **no** tells the processor that the document can reference external files.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```



The Three Flavors of XHTML 1.0

- The XHTML 1.0 specification comes in three “flavors” or versions. XHTML authors can choose the version that best fits the needs of their documents.
- This is done by inserting a **Document Type Definition (DOCTYPE) declaration**. This is part of the document prolog that we’ll look at more closely later.
- The DTD is the specification from the W3C that defines the language, including the elements and attributes. The declaration was optional for HTML documents, but is required for XHTML documents, and it needs to match one of the DTDs for the three flavors of XHTML.
- The following three pages define each of the three flavors and when they should be used, and shows the DOCTYPE declaration for each and where you can view the DTD on the Web.



XHTML Transitional Version

- XHTML Transitional is currently the most used version of XHTML 1.0. This version most closely resembles HTML 4.0.1 and is the best choice when documents need to use HTML's presentational elements or when pages need to be developed without using style sheets.
- Use this version if you want to convert existing HTML pages to XHTML.
- The caveat to the Transitional version is that it contains support for certain elements and attributes that are being deprecated, or phased out. It also does not contain support for frames.
- For more details on the Transitional flavor of XHTML see: <http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd>.
- The declaration is:

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



XHTML Frameset Version

- XHTML Frameset should be used when your documents need to use the frame elements that partition the browser into multiple independent windows.
- The element set for this version contains all of the elements from XHTML Transitional plus the elements needed to support frames, such as `<frame>` and `<frameset>`.
- For more details on the Frameset flavor of XHTML see: <http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd>.
- The declaration is:

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



XHTML Strict Version

- XHTML Strict most closely represents the future of XHTML. The element set for XHTML Strict contains a subset of the elements from XHTML Transitional, but does not include support for strictly presentational elements or elements that will not likely be included in future versions of XHTML.
- In the future, XHTML documents will separate presentation from content and use style sheets to define presentation formatting such as font types, colors, and styles.
- Use XHTML Strict with Cascading Style Sheets (CSS). We will learn all about CSS later in the course.
- For more details on the Strict flavor of XHTML see: <http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>.
- The declaration is:

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



Creating Your First XHTML Document

- We'll begin creating our first XHTML document by examining a plain text document that we would like to markup with formatting elements.
- Using NotePad or some similar text editor, create a file with the name “`unformatted.html`”, that includes the text shown below:

Course Name: Internet Applications

Course Number: CGS 3175

Instructor: Dr. Mark Llewellyn

Class Meets: Tuesday and Thursday, 1:30pm-2:45pm, HEC 104

Course Description: This course covers internet applications including how to write XHTML Web documents.

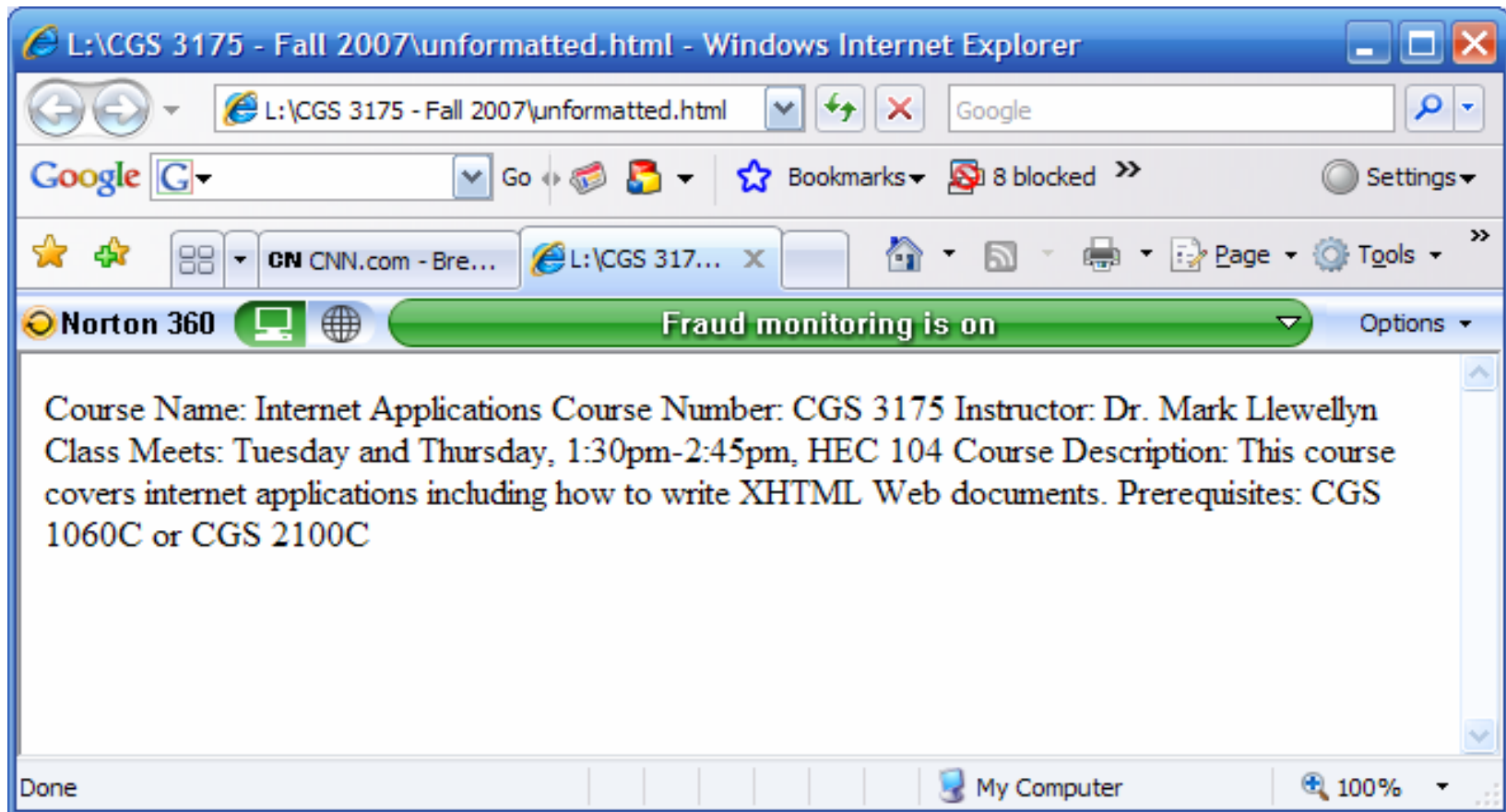
Prerequisites: CGS 1060C or CGS 2100C

Unformatted Text Document – name it “`unformatted.html`”



Creating Your First XHTML Document

- Next, open your Web browser and load the file “unformatted.html” that you just created. The screen shot below, shows what the file looks like in Internet Explorer on my computer at home.



Creating Your First XHTML Document

- The screen shot on the previous page illustrates how the Web browser can open the document, but without formatting instructions, it does not know how to correctly format the document.
- Web browsers ignore all whitespace characters, including line breaks, so without the proper markup, our document displays as just a block of text.
- Next, we'll add some formatting elements (markup) to our document. For comparison purposes, we'll markup the document in both HTML and XHTML, and then look at the differences between the two.
- **NOTE:** Do not type the line numbers in your document, they are for only used in these notes to make references to specific lines.



The HTML Version

```
1. <HTML>
2.   <HEAD>
3.     <TITLE>Internet Applications Fall 2007</TITLE>
4.   </HEAD>
5.   <BODY>
6.     <STRONG>Course Name: </STRONG> Internet Applications <BR>
7.     <STRONG>Course Number: </STRONG> CGS 3175 <BR>
8.     <STRONG>Instructor: </STRONG> Dr. Mark Llewellyn <BR>
9.     <STRONG>Class Meets: </STRONG> Tuesday and Thursday, 1:30pm-2:45pm, HEC 104
10.    <BR>
11.    <P>
12.    <STRONG>Course Description: </STRONG> This course covers Internet applications
13.    including how to write XHTML Web documents.
14.    <P>
15.    <STRONG>Prerequisites: </STRONG>
16.    <UL>
17.      <LI> CGS 1060C or,
18.      <LI> CGS 2100C
19.    </UL>
20.  </BODY>
21. </HTML>
```

The document starts with an opening tag for the HTML element

This is followed by the header section and the title of the document.

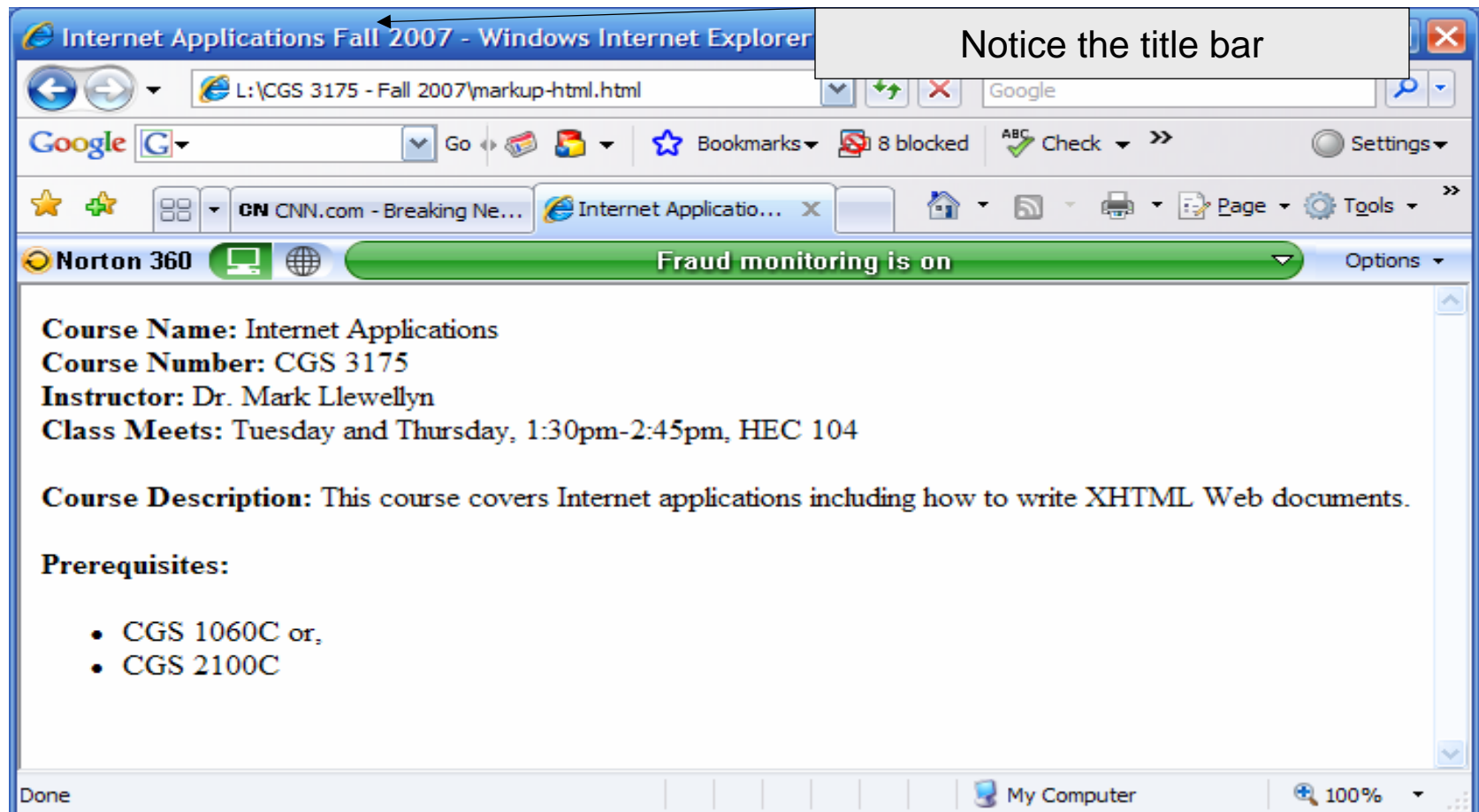
The main body of the document contains a mix of markup elements and content.

The document always ends with the closing tag for the HTML element



Viewing The HTML Version

- Create a new file that contains the text from the previous page (remember – do not type in the line numbers). I called my version of this file: “markup-html.html”, but you can call it whatever you would like. Next, open your Web browser and load the file “markup-html.html” that you just created. The screen shot below, shows what the file looks like in Internet Explorer on my computer at home.



The XHTML Version

Begin with XHTML document heading

```
1. <?xml version="1.0"?>
2. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3.   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4. <html xmlns="http://w3.org/199/xhtml">
5.   <head>
6.     <title>Internet Applications Fall 2007</title>
7.   </head>
8.   <body>
9.     <strong>Course Name: </strong> Internet Applications <br />
10.    <strong>Course Number: </strong> CGS 3175 <br />
11.    <strong>Instructor: </strong> Dr. Mark Llewellyn <br />
12.    <strong>Class Meets: </strong> Tuesday and Thursday, 1:30pm-2:45pm, HEC 104
13.    <br />
14.    <strong>Course Description: </strong> This course covers Internet applications
15.    including how to write XHTML Web documents.
16.    <br />
17.    <strong>Prerequisites: </strong>
18.    <ul>
19.      <li> CGS 1060C or,</li>
20.      <li> CGS 2100C</li>
21.    </ul>
22.  </body>
  </html>
```

The document starts with an opening tag for the <html> element.

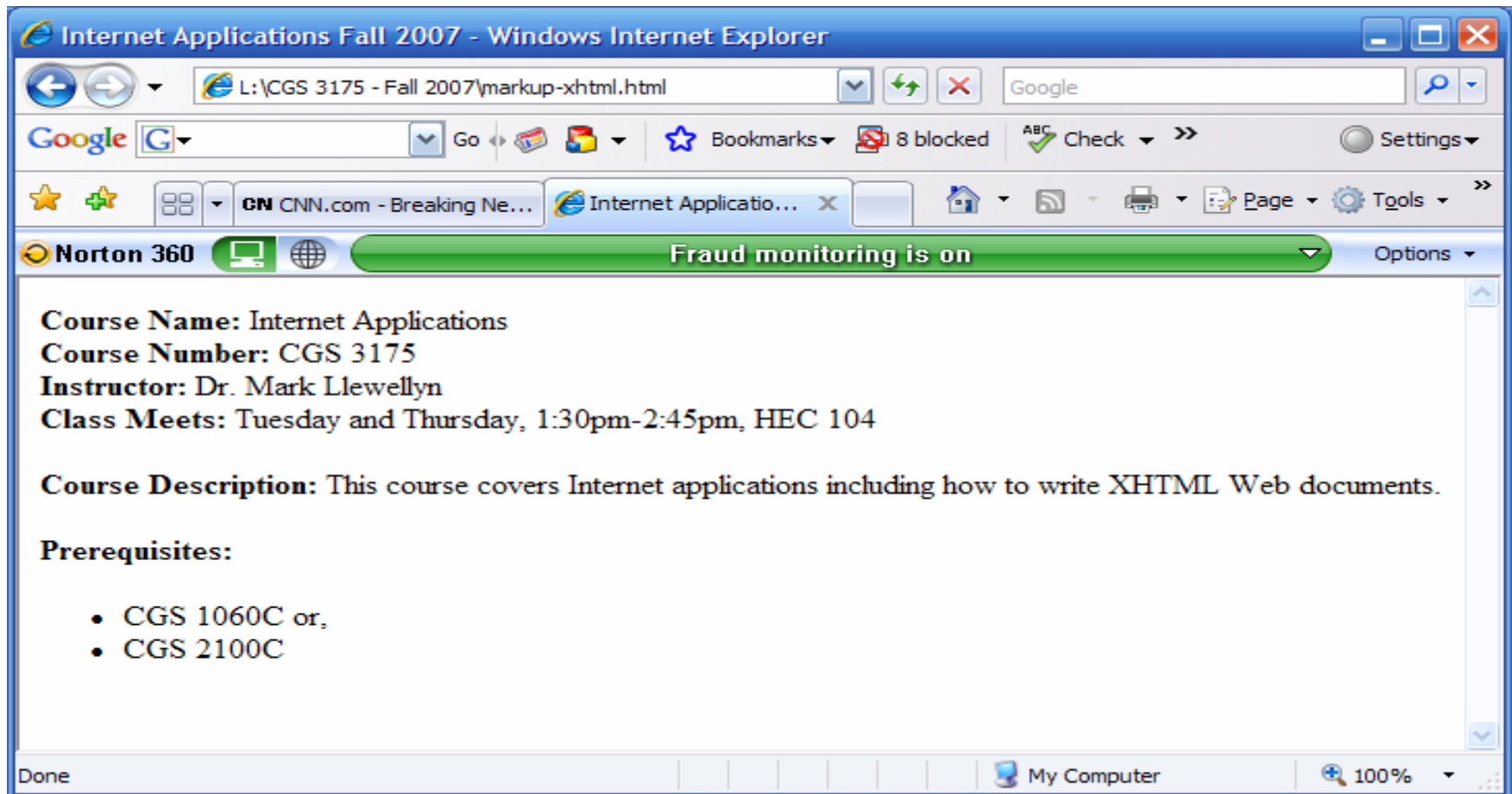
The main body of the document contains a mix of markup elements and content.

The document ends with a closing tag for the <html> element



Viewing The XHTML Version

- Create a new file that contains the text from the previous page (remember – do not type in the line numbers). I called my version of this file: “markup-xhtml.html”, but you can call it whatever you would like. Next, open your Web browser and load the file “markup-xhtml.html” that you just created. The screen shot below, shows what the file looks like in Internet Explorer on my computer at home.



Differences Between HTML And XHTML

- Looking at the two examples on pages 35 and 37, you probably noticed that the code is very similar. Both HTML and XHTML are platform- and vendor-independent. Both used elements and tags to describe pieces of data within a document.
- As we mentioned earlier, they use the same set of element names based on the element set from HTML 4. However, XHTML has much stricter syntax and is more powerful and flexible than HTML.
- The following three pages list some of the major differences between XHTML and HTML. We'll explore all of these differences and many more as the semester progresses.



Differences Between HTML And XHTML

1. XHTML documents contain the XML and DOCTYPE declarations at the top of the document. The XML declaration is optional but the DOCTYPE declaration is required. The DOCTYPE declaration was optional in HTML. These are lines 1 and 2 in the XHTML document on page 37.
2. XHTML documents must be well formed, meaning that they need to adhere to the syntax rules for the language. HTML, however, does not strictly require that documents be well formed.
3. XHTML is not dependent on a single document type or set of markup elements, like HTML. XHTML can be extended or used in conjunction with other markup languages.



Differences Between HTML And XHTML

4. XHTML element and attribute names must be lowercase. XHTML elements and attributes are case sensitive, while HTML elements and attributes are not. In our examples on pages 35 and 37, notice that the HTML elements are all uppercase: `<HTML>`, ``, `</BODY>`. This was done simply as a matter of style. These tags could have been written in lowercase or a combination of upper and lowercase, and HTML would still have interpreted them correctly: `<html>`, ``, `</BoDY>`. The XHTML document, on the other hand, must have all of its tags and attributes in lowercase.
5. For nonempty elements, XHTML requires end tags. An empty element is an element that does not contain an end tag. This is not a requirement for HTML, as the HTML element set contains a subset of elements that do not have end tags. In the XHTML code on page 37, notice that these elements are written a little differently: `
` and `<p />`, In XHTML, all elements must either have an end tag or end in `/>`.



Differences Between HTML And XHTML

6. Attribute values must always be quoted in XHTML. This was not a requirement in HTML. The following is valid in HTML:

```
<img src=picture.gif>
```

The attribute src has a value of picture.gif assigned to it. However, the same line in an XHTML document places quotes around the value of the attribute:

```

```

- The W3C provides a great deal of information about the differences between HTML and XHTML. See <http://www.w3.org/TR/xhtml1/#diffs>



Rules and Tools For Building XHTML Documents

- Because XHTML is an XML application, you must adhere to the same syntax rules as any XML language.
- If you are used to writing HTML code, these rules may seem a bit intimidating, as HTML does not force developers to adhere so strictly to syntax rules. If you are new to writing markup code, that's fine...you'll be learning the correct way to markup code from the start.
- An XHTML document that adheres to XML syntax is said to be **well-formed**. A document that is not well-formed will generate an error in a parser program.



Syntax Rules For Well-formed XML Documents

1. All XHTML documents must contain the root element `<html>` and cannot contain more than one root element.
2. All elements must have a start and end tag, such as:

```
<h1> . . . </h1>
```

The exception is an empty tag, which must have a forward slash (/) before the end tag. An example of the shorthand notation is: `
`.

3. Elements must be nested properly and cannot overlap. Each element must be contained completely inside of its parent element. This rule is the same as for mathematical functions such as: $(x * [y + z])$. Notice that there are two parts to this equation, one is surrounded by the parentheses and the other by brackets. The subequation $[y + z]$ is entirely contained within the outer equation, which is delineated by the parentheses. It is illegal to write the equation like this: $(x * [y + z])$.

Example of illegally nested elements:

```
<h1> <strong> . . .</h1> </strong>
```

Example of legally nested elements:

```
<h1> <strong>. . . </strong> </h1>
```



Syntax Rules For Well-formed XML Documents

4. All attributes must have a value, and that value must be enclosed in quotes. Both double and single quotes are allowed and can be nested, as long as you are consistent and you do not use the same type of quote twice. Here are some valid examples:

Single quotes: ``

Double quotes: ``

Nested quotes:

```

```

5. Attributes must be placed in the start tag of an element, and no attribute can appear more than once. Here is an invalid example:

```
<a href="page2.html" href="page3.html"> Click here</a>
```

6. Element names are case sensitive, for example, `<TITLE>` and `</title>` are both legal beginning and ending tags in HTML, because it is not case sensitive. In XHTML, all tags must be lowercase, so `<TITLE>` would produce an error.



Well-Formed Versus Valid Documents

- The terms well-formed and valid in reference to writing good XHTML are not the same. They are both very important pieces of developing code that will be compatible with future tools, but they are distinctly different in the kinds of error they test for.
- A well-formed document, as we've just seen, must adhere to a set of structural rules. These rules apply to the syntax of how the code is written, but does not pay attention to the code itself.
 - For example, if you include the names of elements or attributes that are not part of the HTML 4.01 specification, but you follow the syntax rules we just covered, your document will pass the well-formed test, even though you have used elements that are not part of the specification.
- For a document to be valid, it must not only be well-formed, but also conform to the rules of the DTD that it lists in its DOCTYPE declaration. The DTD contains the set of all valid element and attribute names that can be used for that document type. Recall the XHTML Strict DTD:

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



Validating An XHTML Document

- While there are many free and commercial programs available, one of the simplest ways to check if your XHTML documents are well-formed and valid is to use the free on-line validating tool from the W3C Web site. It can check your document either from your computer or from a website.
- The W3C markup validation service is available at: <http://validator.w3.org>.
- The next page shows the first step in using the file upload version of the validator.



The W3C Markup Validation Service - Windows Internet Explorer

http://validator.w3.org/#validate_by_upload+with_options

File Edit View Favorites Tools Help

Google www.validator.w3.org

The W3C Markup Validation Service

W3C[®] Markup Validation Service

Check the markup (HTML, XHTML) of Web documents

Validate by URI **Validate by File Upload** Validate by Direct Input

Validate by File Upload

Upload a document for validation:

File:

▼ More Options

Character Encoding (detect automatically) Only if missing

Document Type (detect automatically) Only if missing

List Messages Sequentially Group Error Messages by type

Show Source Clean up Markup with HTML Tidy

Show Outline Validate error pages Verbose Output

Internet 100%

I used the markup-xhtml.html version of our document from XHTML Part 1 page 37 with the DTD set to Transitional not Strict as it appeared in the notes.

Once you enter the filename of your XHTML document, click "Check", and you should see the next page.



[Valid] Markup Validation of E:\Courses\CGS 3175 - Internet Applications\markup-xhtml.html - W3 - Windows Internet Explorer

http://validator.w3.org/check

File Edit View Favorites Tools Help

Google www.validator.w3.org Go 57 blocked Check AutoLink AutoFill Send to www Settings

[Valid] Markup Validation of E:\Courses\CGS 3175 - In...

W3C® Markup Validation Service
Check the markup (HTML, XHTML) of Web documents

Jump To: Important Warnings Congratulations - Icons

This Page Is *Tentatively Valid XHTML 1.0 Transitional*

Result:	Tentatively passed validation
File:	E:\Courses\CGS 3175 - Internet Applications\markup-xhtml.html
Encoding:	utf-8
Doctype:	XHTML 1.0 Transitional
Root Element:	html
Root Namespace:	http://www.w3.org/1999/xhtml

Important Warnings

The validator has found the following problem(s) prior to validation, which should be addressed in priority:

⚠ No Character Encoding Found! Falling back to UTF-8.
I was not able to extract a character encoding labeling from any of the valid sources for such information. Without encoding

Internet 100%

Since our xml declaration did not include an encoding attribute, we get this warning – to remove this warning and get a complete validation add the encoding attribute to the XHTML document as shown on the next page.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Internet Applications Fall 2007</title>
  </head>
  <body>
    <strong>Course Name: </strong> Internet Applications
    <strong>Course Number: </strong> CGS 3175 <br />
    <strong>Instructor: </strong> Dr. Mark Llewellyn <br />
    <strong>Class Meets: </strong> Tuesday and Thursday, 1:30pm-
2:45pm, HEC 104 <br />
    <p />
    <strong>Course Description: </strong> This course covers
Internet applications including how to write XHTML Web documents.
    <p />
    <strong>Prerequisites: </strong>
    <ul>
      <li> CGS 1060C or,</li>
      <li> CGS 2100C</li>
    </ul>
  </body>
</html>
```

Add the encoding attribute to the xml declaration – then rerun the validation service on the revised document and you should receive a valid document message as shown on the next page.



[Valid] Markup Validation of E:\Courses\CGS 3175 - Internet Applications\markup-xhtml.html - W3 - Windows Internet Explorer


http://validator.w3.org/check

File Edit View Favorites Tools Help

Google G Go Bookmarks 57 blocked Check AutoLink AutoFill Send to Settings

[Valid] Markup Validation of E:\Courses\CGS 3175 - In...

Page Tools



Markup Validation Service

Check the markup (HTML, XHTML) of Web documents

Jump To: [Congratulations](#) - [Icons](#)

This Page Is Valid XHTML 1.0 Transitional!

Result:	Passed validation
File:	E:\Courses\CGS 3175 - Internet Applications\markup-xhtml.html
Encoding:	utf-8
Doctype:	XHTML 1.0 Transitional
Root Element:	html
Root Namespace:	http://www.w3.org/1999/xhtml

Congratulations

The uploaded document "E:\Courses\CGS 3175 - Internet Applications\markup-xhtml.html" was checked and found to be valid XHTML 1.0 Transitional. This means that the resource in question identified itself as "XHTML 1.0 Transitional" and that we successfully performed a formal validation using an SGML or XML Parser (depending on the markup language used).

Done Internet 100%



XHTML Code With Errors – Can You Find Them?

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Internet Applications Fall 2007
  </head>
  <body>
    <strong>Course Name: </strong> Internet Applications <br />
    <strong>Course Number: </strong> CGS 3175 <br />
    <strong>Instructor: </strong> Dr. Mark Llewellyn <br />
    <strong>Class Meets: </strong> Tuesday and Thursday, 1:30pm-
2:45pm, HEC 104<BR />
    <p />
    <strong>Course Description: </strong> This course covers
Internet applications including how to write XHTML Web documents.
    <p />
    <strong attribute="yes">Prerequisites: </strong>
    <ul>
      <li> CGS 1060C or,</li>
      <li> CGS 2100C</li>
    </ul>
  </body>
</html>
```

Missing end tag </title>

end tag is capitalized

No attribute named "attribute"



[Invalid] Markup Validation of upload://Form Submission - W3C Markup Validator - Windows Internet Explorer

http://validator.w3.org/check

File Edit View Favorites Tools Help

Google www.validator.w3.org

Bookmarks 57 blocked Check AutoLink AutoFill Send to www Settings

[Invalid] Markup Validation of upload://Form Submissi...

Page Tools

Jump To: Validation Output

This page is not Valid XHTML 1.0 Transitional!

Result:	Failed validation, 6 Errors
File:	upload://Form Submission
Encoding:	utf-8
Doctype:	XHTML 1.0 Transitional
Root Element:	html
Root Namespace:	http://www.w3.org/1999/xhtml

↑ TOP

Validation Output: 6 Errors

✖ Line 7, Column 9: end tag for "title" omitted, but OMITTAG NO was specified.

```
</head>
```

Done Internet 100%



[Invalid] Markup Validation of upload://Form Submission - W3C Markup Validator - Windows Internet Explorer

http://validator.w3.org/check

File Edit View Favorites Tools Help

Google www.validator.w3.org

Bookmarks 57 blocked Check AutoLink AutoFill Send to www Settings

[Invalid] Markup Validation of upload://Form Submissi...

Page Tools

◆ Line 6, Column 5: start tag was here.

```
<title>Internet Applications Fall 2007
```

✖ Line 12, Column 86: element "BR" undefined.

```
...Thursday, 1:30pm-2:45pm, HEC 104<BR />...
```

You have used the element named above in your document, but the document type you are using does not define an element of that name. This error is often caused by:

- incorrect use of the "Strict" document type with a document that uses frames (e.g. you must use the "Frameset" document type to get the "<frameset>" element),
- by using vendor proprietary extensions such as "<spacer>" or "<marquee>" (this is usually fixed by using CSS to achieve the desired effect instead).
- by using upper-case tags in XHTML (in XHTML attributes and elements must be all lower-case).

✖ Line 16, Column 24: there is no attribute "attribute".

Done Internet 100%

