

***HTML Essentials***

**Lab Guides**

|  |  |
| --- | --- |
| Document Code | 25e-BM/HR/HDCV/FSOFT |
| Version | 1.1 |
| Effective Date | 20/11/2012 |

**Hanoi, 04/2019**

RECORD OF CHANGES

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Effective Date | Change Description | Reason | Reviewer | Approver |
|  | 25/Jun/2018 | Create a new Lab | Create new | DieuNT1 | VinhNV |
|  | 01/May/2019 | Update Fsoft Template | Update | DieuNT1 | VinhNV |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Contents

[Unit 2 – Debugging HTML 4](#_Toc38547400)

[Lab Objectives: 4](#_Toc38547401)

[Problem Descriptions: 4](#_Toc38547402)

[Guidelines: 4](#_Toc38547403)

[Step 1: Study permissive code 4](#_Toc38547404)

[Step 2: Use DOM Inspector 4](#_Toc38547405)

[Step 3: Live-edit with DOM Inspector 4](#_Toc38547406)

[Step 4: Review and Fix 4](#_Toc38547407)

[Step 5: Use HTML Validator tools 4](#_Toc38547408)

[Step 6: Interpreting error messages 4](#_Toc38547409)

[Step 7: Summary 4](#_Toc38547410)

|  |  |
| --- | --- |
|  | **CODE: HTML-E.S.L201**  **TYPE: Short**  **LOC: N/A**  **DURATION: 30 MINUTES** |

# Unit 2 – Debugging HTML

## Lab Objectives:

* Understand errors code
* Able to use HTML Validator
* Understand error emitted from HTML Validator

## Problem Descriptions:

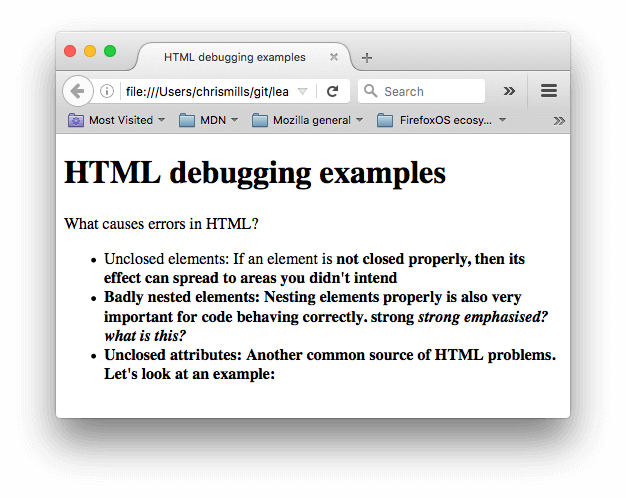
Given a project, you have to investigate and fix all errors.

## Guidelines:

Step 1: Study permissive code

It's time to study the permissive nature of HTML code.

First, download our [debug-example demo](https://github.com/mdn/learning-area/blob/master/html/introduction-to-html/debugging-html/debug-example.html) and save it locally. This demo is deliberately written to have some errors in it for us to explore (the HTML markup is said to be badly-formed, as opposed to well-formed).

Next, open it in a browser. You will see something like this:

This immediately doesn't look great; let's look at the source code to see if we can work out why (only the body contents are shown):

1. <h1>HTML debugging examples</h1>
2. <p>What causes errors in HTML?
3. <ul>
4. <li>Unclosed elements: If an element is <strong>not closed properly,
5. then its effect can spread to areas you didn't intend
6. <li>Badly nested elements: Nesting elements properly is also very important
7. for code behaving correctly. <strong>strong <em>strong emphasised?</strong>
8. what is this?</em>
9. <li>Unclosed attributes: Another common source of HTML problems. Let's
10. look at an example: <a href="https://www.mozilla.org/>link to Mozilla
11. homepage</a>
12. </ul>

Let's review the problems:

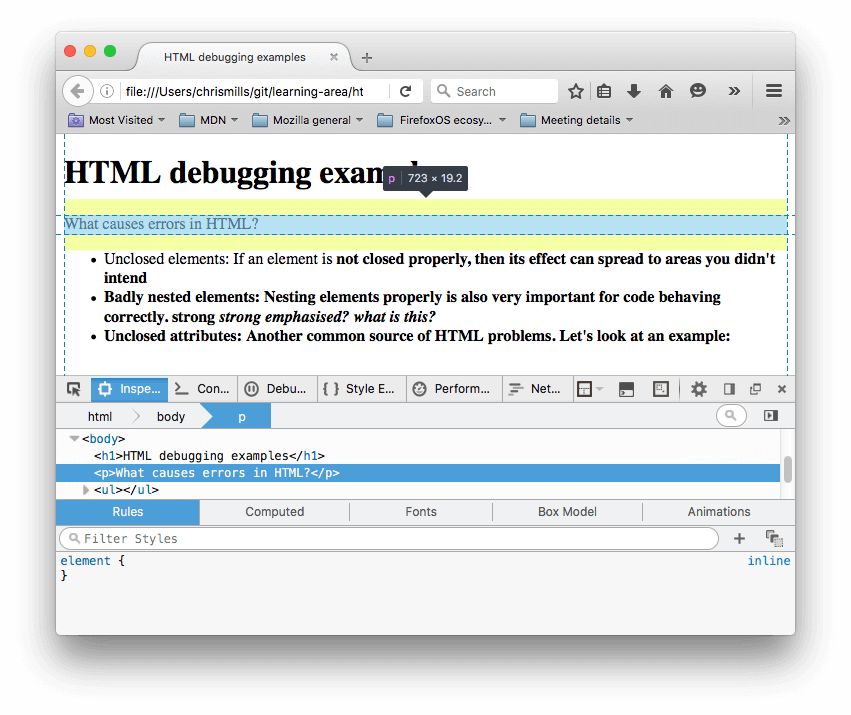
The [paragraph](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/p) and [list item](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/li) elements have no closing tags. Looking at the image above, this doesn't seem to have affected the markup rendering too badly, as it is easy to infer where one element should end and another should begin.

The first [<strong>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/strong) element has no closing tag. This is a bit more problematic, as it isn't easy to tell where the element is supposed to end. In fact, the whole of the rest of the text has been strongly emphasised.

This section is badly nested: <strong>strong <em>strong emphasised?</strong> what is this?</em>. It is not easy to tell how this has been interpreted because of the previous problem.

The [href](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a#attr-href) attribute value has a missing closing double quote. This seems to have caused the biggest problem — the link has not rendered at all.

Now let's look at the markup the browser has rendered, as opposed to the markup in the source code. To do this, we can use the browser developer tools. If you are not familiar with how to use your browser's developer tools, take a few minutes to review [Discover browser developer tools](https://developer.mozilla.org/en-US/docs/Learn/Discover_browser_developer_tools).

In the DOM inspector, you can see what the rendered markup looks like:

Using the DOM inspector, let's explore our code in detail to see how the browser has tried to fix our HTML errors (we did the review in Firefox; other modern browsers *should* give the same result):

The paragraphs and list items have been given closing tags.

It isn't clear where the first <strong> element should be closed, so the browser has wrapped each separate block of text with its own strong tag, right down to the bottom of the document!

The  incorrect nesting has been fixed by the browser like this:

1. <strong>strong
2. <em>strong emphasised?</em>
3. </strong>
4. <em> what is this?</em>

The link with the missing double quote has been deleted altogether. The last list item looks like this:

1. <li>
2. <strong>Unclosed attributes: Another common source of HTML problems.
3. Let's look at an example: </strong>
4. </li>

Step 2: Use DOM Inspector

Step 3: Live-edit with DOM Inspector

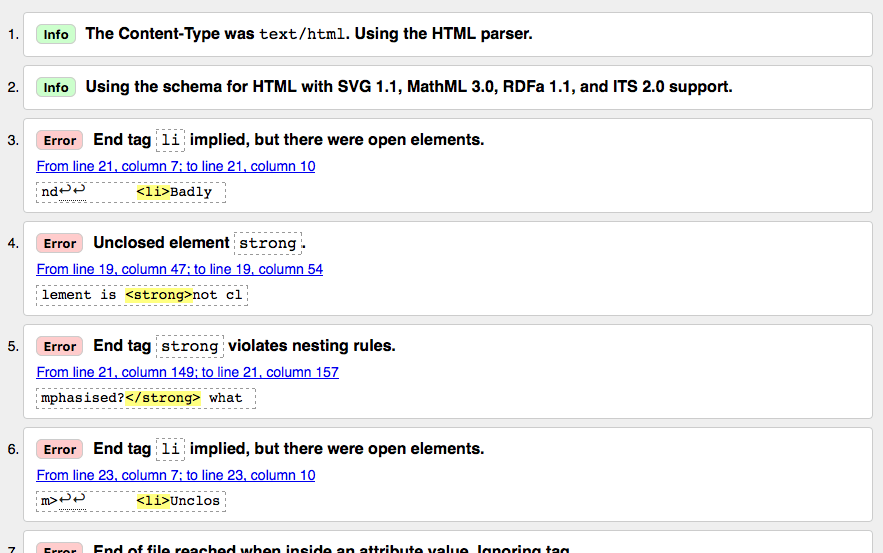
Step 4: Review and Fix

Step 5: Validating an HTML document

Let's try this with our [sample document](https://github.com/mdn/learning-area/blob/master/html/introduction-to-html/debugging-html/debug-example.html).

1. First, load up the [Markup Validation Service](https://validator.w3.org/) in one browser tab, if it isn't already.
2. Switch to the [Validate by Direct Input](https://validator.w3.org/#validate_by_input) tab.
3. Copy all the sample document's code (not just the body) and paste it into the large text area shown in the Markup Validation Service.
4. Press the *Check* button.

This should give you a list of errors and other information.



Step 6: Interpreting error messages

The error messages are usually helpful, but sometimes they are not so helpful; with a bit of practice you can work out how to interpret these to fix your code. Let's go through the error messages and what they mean. You'll see that each message comes with a line and column number to help you to locate the error easily.

"End tag li implied, but there were open elements" (2 instances): These messages indicate that an element is open that should be closed. The ending tag is implied, but not actually there. The line/column information points to the first line after the line where the closing tag should really be, but this is a good enough clue to see what is wrong.

"Unclosed element strong": This is really easy to understand — a [<strong>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/strong) element is unclosed, and the line/column information points right to where it is.

"End tag strong violates nesting rules": This points out the incorrectly nested elements, and the line/column information points out where it is.

"End of file reached when inside an attribute value. Ignoring tag": This one is rather cryptic; it refers to the fact that there is an attribute value not properly formed somewhere, possibly near the end of the file because the end of the file appears inside the attribute value. The fact that the browser doesn't render the link should give us a good clue as to what element is at fault.

"End of file seen and there were open elements": This is a bit ambiguous, but basically refers to the fact there are open elements that need to be properly closed. The lines numbers point to the last few lines of the file, and this error message comes with a line of code that points out an example of an open element:

1. example: <a href="https://www.mozilla.org/>link to Mozilla homepage</a> ↩ </ul>↩ </body>↩</html>

Note: An attribute missing a closing quote can result in an open element because the rest of the document is interpreted as the attribute's content.

"Unclosed element ul": This is not very helpful, as the [<ul>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/ul) element *is* closed correctly. This error comes up because the [<a>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a) element is not closed, due to the missing closing quote mark.

If you can't work out what every error message means, don't worry about it — a good idea is to try fixing a few errors at a time. Then try revalidating your HTML to show what errors are left. Sometimes fixing an earlier error will also get rid of other error messages — several errors can often be caused by a single problem, in a domino effect.

You will know when all your errors are fixed when you see the following banner in your output:

Banner that reads "The document validates according to the specified schema(s) and to additional constraints checked by the validator."

Step 7: Summary

**-- THE END --**