Do's and dont's

Do's and don'ts

The history of Web pages is such that browsers tend to be very forgiving of certain types of mistakes. If you miss a closing tag, it will often work the way you expect. It probably won't crash, or ignore the document, or give up completely, but it might not appear quite the way you meant it to. Or maybe it does look like you want, but you do not want to depend on that. In general, best practices would call for one to do it properly, and not depend on the browser to patch it for you..

Because an HTML file essentially represents a tree structure, the open and close tags should always match, and there should not be any overlap with other elements. That is, you can have an element that is enclosed in another element, or you can have two elements side-by-side, but you can never have a situation in which part of an element is in another, but the other part is not.

```
<p>This is a <em>paragraph</em></p>

<h1>Paragraph ahead</h1>
<p>And here it is.</p>
```

The two examples above are fine because in each case either an element is wholly contained in another (`<em>` in `<p>`) or they are completely separate (`<h1>` and `<p>`). This, on the other hand, is not valid:

```
<h1>Part of this header is<p>in the</h2> paragraph
below</p>
```

What happens in this case is what we call "undefined". That just means that there is no telling how the browser will decide to handle it. It might decide to automatically close the when it sees another close tag, or it could complain about an unexpected close tag at the header. Then it might complain again when there is a now unexpected close tag.

If you played around with the minimal HTML file from the previous section, you might have noticed that you can get more minimal than that. For example, if you take out the "head" section completely, the browser will still render the page without complaint (at least Chrome will; Firefox does complain in the debugging console, but we will save that for week 4). In fact, you can even take out the "body" open and close tags (not the content, of course) and it will still work as expected. Not only that, if you take out the <!doctype> statement, it still works (and Chrome still doesn't complain!).

What's actually happening is that the browser knows roughly what to expect in an HTML page, so if it sees a file ending in '.html' it will automatically stick some stuff in if it is not there already. It will typically make basic assumptions like: It is an HTML5 file, everything in there is content, so it goes in the <body> section, the <head> section is empty. If you right-click on an element and choose "Inspect", you will see that the browser has included the <html> section containing the <head> and <body> sections, even though it wasn't there in your file.

Note that we said "typically". The current behavior of most browsers will handle this, but it is "undefined" so there is no guarantee that next module's update won't break it. To be correct and complete, you need the <!doctype> section and the <html> section with <head> and <body>. In any case, it is a good idea (best practice).

Proper indentation is one way to make your code clearer and easier to understand:

```
1. <body>
2. <h1>Here is a heading</h1>
3. <p>
4. <ol><li>List Item 1</li></ol>
5. </p>
6. </body>
```

The code above doesn't give any sense of the structure of the document. By using indentation effectively, you can make it more clear, showing the nesting of elements:

```
<body>
  <h1>Here is a heading</h1>
  <p>
    <ol>
      <li>List Item 1</li>
    </ol>
  </p>
</body>
```

Consistent quoting of strings is also helpful, in part to avoid potential problems that can arise when you think something does not need quotes but it actually does.

Often, projects will have coding styles that everyone is expected to use so that everything looks consistent and developers can more easily read each other's code. If you are creating the project, you can decide what the rules are (how many spaces to indent, single or double-quotes, etc.) but unless there is a good reason to change away from typical practices, it is usually best to adopt them.