Rendering Text: Code Along Exercises

In this lesson, we render text in a coding exercise.
Let's get started!

WE'LL COVER THE FOLLOWING

Code Along Exercise 02-04:

```
<!DOCTYPE html>
<html>
<head>
  <title>Common text markups</title>
</head>
<body>
  <strong>HTML5</strong>
  is a <dfn>markup language</dfn>
  for structuring and presenting content for the
  World Wide Web (<abbr>WWW</abbr>)
  and a core technology of the <em>Internet</em>.
</body>
</html>
```

The image below displays the expected output of the above code. We use four different options to play with our text namely:

1. **Line 7** `<strong>` tag
2. **Line 8** `<dfn>` tag
3. **Line 10** `<abbr>` tag
4. **Line 11** `<em>` tag

Common Text Markups

When you create text with markup, take care that you use the appropriate markup to tell the browser about the semantics of the highlighted text. According to Exercise 02-04, the *"markup language"* expression is a term definition, while Internet is an *emphasized text*, even if both are displayed in italic.

You could have used the markup shown in Exercise 02-05 to get exactly the same result as shown in the image above.

Code-along exercise 02-05:

```
<!DOCTYPE html>
<html>
<head>
  <title>Common text markups</title>
</head>
<body>
  <b>HTML5</b>
  is a <i>markup language</i>
  for structuring and presenting content for the
  World Wide Web (WWW)
  and a core technology of the <i>Internet</i>.
</body>
</html>
```

The main difference between the two Exercises is that while Exercise 2-4 adds semantics to the page (such as definition, abbreviation, etc.), Exercise 2-5 adds styling and does not mark semantic differences.

Wherever possible, use rich semantics instead of simple styling markups. It gives you two immediate advantages. **First**, semantics adds more information to your pages. **Second**, using CSS you can set how a certain semantic meaning should be visually represented.

For example, you can change definitions to use different background color, while emphasized text could still be shown in italics.

You can easily display technical text with source code, just use the `<pre>`, `<code>`, `<var>`, and `<kbd>` tags, as shown in Listing 2-6.

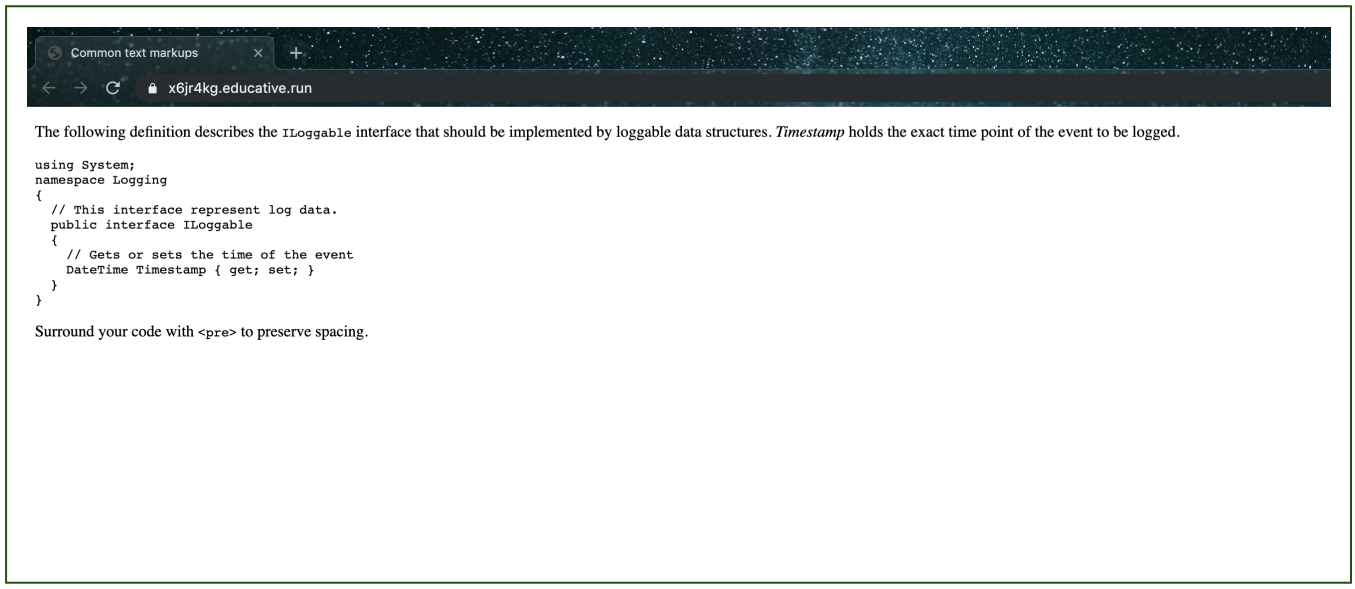
Code-along exercise 02-06:

```
<!DOCTYPE html>
<html>
<head>
  <title>Common text markups</title>
</head>
<body>
  <p>
    The following definition describes the
    <code>ILoggable</code> interface that
    should be implemented by loggable
    data structures. <var>Timestamp</var>
    holds the exact time point of the
    event to be logged.
  </p>
  <pre>using System;
namespace Logging
{
  // This interface represent log data.
  public interface ILoggable
  {
    // Gets or sets the time of the event
    DateTime Timestamp { get; set; }
  }
}</pre>
  <p>
    Surround your code with
    <kbd>&lt;pre&gt;</kbd>
    to preserve spacing.
  </p>
</body>
</html>
```

This markup renders the source code with monospace font, as it is shown in the figure below. You can observe that spaces and line breaks are preserved by `<pre>`. As you can see, the content directly follows the opening tag, and the

closing tag is put directly after the last character to display. Should you put a

line break after `<pre>` and before `</pre>`, an extra line break would occur before and after the rendered source code.

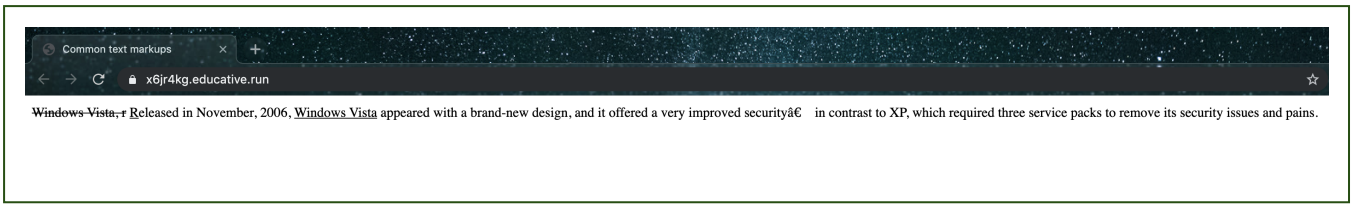


The markup renders the source code with monospace font

Code-along exercise 02-07:

```
<!DOCTYPE html>
<html>
<head>
  <title>Common text markups</title>
</head>
<body>
  <del cite="http://mybooks.com/Windows8"
    datetime="2012-06-22T08:45:13">
    Windows Vista, r</del>
  <ins cite="http://mybooks.com/Windows8"
    datetime="2012-06-22T08:45:15">R</ins>eleased
    in November, 2006,
  <ins cite="http://mybooks.com/Windows8"
    datetime="2012-06-22T08:45:15">
    Windows Vista</ins>
  appeared with a brand-new design, and it
  offered a very improved security --□in contrast
  to XP, which required three service packs
  to remove its security issues and pains.
</body>
</html>
```

As shown in the figure above, deleted text is represented with strikethrough characters, inserted text with underlined characters by default.



A reviewed document

The `<ins>` and `<del>` tags provide the cite and datetime attributes to give you extra information about the modification of the text. The cite attribute, a URL, refers to a document that explains why the text was deleted or inserted, while `datetime` specifies the date and time when the text was modified.

Although the information in these attributes is not rendered in the browser; there are techniques to display them. For example, with CSS and JavaScript (as you will learn later in this course) you can write code that pops up a little tooltip when the mouse moves over the modified text, and in the tooltip the information stored in these attributes is revealed.

In the *next lesson*, we learn the usage of reserved characters and symbols in HTML.