

1

Getting started

9

Introducing HTML5	10
Addressing web pages	12
Defining document structure	14
Creating a document	16
Validating documents	18
Employing an HTML editor	20
Summary	22

2

Providing page information

23

Bestowing a title	24
Specifying a character set	26
Refreshing the page	28
Describing the document	30
Incorporating scripts	32
Incorporating style sheets	34
Linking more resources	36
Summary	38

3

Creating body content

39

Working the body	40
Inserting paragraphs	42
Including quotations	44
Emphasizing text	46
Reading the small print	48
Marking text	50
Keeping preformatted text	52
Modifying text	54
Including code in text	56
Giving tooltip advice	58
Adding images	60
Directing languages	62
Summary	64

4

Inserting hyperlinks**65**

Creating hyperlinks	66
Accessing links via keys	68
Linking to page fragments	70
Linking to protocols	72
Using images as hyperlinks	74
Producing image map links	76
Generating popups	78
Summary	80

5

Arranging content sections**81**

Proclaiming headings	82
Grouping headers	84
Providing navigation	86
Writing articles	88
Standing aside	90
Footing the page	92
Positioning content	94
Denying anonymity	96
Summary	98

6

Writing lists and tables**99**

Creating unordered lists	100
Creating ordered lists	102
Creating definition lists	104
Producing a simple table	106
Spanning cells over rows	108
Spanning cells across columns	110
Adding a header and footer	112
Aligning cell content	114
Grouping columns	116
Setting column widths	118
Utilizing templates	120
Summary	122

7

Embedding media content**123**

Referencing figures	124
Selecting pictures	126
Embedding objects	128
Specifying parameters	130
Embedding in frames	132
Embedding plugin movies	134
Embedding audio	136
Embedding video	138
Indicating progress	140
Summary	142

8

Building input forms**143**

Submitting forms	144
Gathering text input	146
Controlling input type	148
Providing text areas	150
Checking boxes	152
Choosing radio buttons	154
Selecting options	156
Utilizing hidden data	158
Pushing buttons	160
Using images for submission	162
Adding logos to buttons	164
Labeling form controls	166
Listing data options	168
Uploading files	170
Summary	172

9

Painting on canvas**173**

Generating a canvas	174
Painting shapes	176
Stroking borders	178
Filling options	180
Writing text	182
Drawing lines	184
Swerving curves	186
Translating coordinates	188
Transforming shapes	190
Animating the canvas	192
Summary	194

10

Employing APIs**195**

Measuring fractions	196
Dragging and dropping objects	198
Manipulating pixels	200
Storing user data	202
Editing live content	204
Cross-document messaging	206
Pinpointing the user	208
Summary	210

+

Handy reference**211**

Global HTML5 attributes	212
HTML5 element tags	214
CSS properties & values	232

Index**233**

The creation of this book has been for me, Mike McGrath, an exciting opportunity to discover how the latest specification of HTML5 can be implemented in today's web browsers. Whereas my previous books on web page markup described the HTML 4.01 and HTML 5.0 specifications, it has been fascinating to see how the HTML 5.1 specifications are supported in Microsoft Edge, Internet Explorer, Firefox, Google Chrome, Safari, and Opera web browsers.

The “Handy Reference” section at the end of the book lists the HTML5 tags and attributes that are included in the HTML 5.1 specifications. All the examples I have given in this book demonstrate HTML5 features that are supported by leading web browsers, and the screenshots illustrate the actual results produced by the listed code. I truly believe that now, more than ever, authors can integrate HTML5 content markup, JavaScript functionality, and CSS presentation, to produce stunning interactive web pages.

Conventions in this book

In order to clarify the code listed in the steps given in each example I have adopted certain colorization conventions. Those parts of the HTML language itself are colored blue, like this:

```
<html>
```

Values assigned to HTML attributes are colored red, like this:

```
<html lang="en">
```

Literal content that is marked up by HTML tags is colored black, like this:

```
<title>HTML5 in easy steps</title>
```

Similarly, for style sheet code listed in the steps, those parts of the CSS language itself are colored blue and values assigned to properties are colored red, like this:

```
h1 { color : red ; background : yellow ; }
```

Additionally, in order to identify each source code file described in the steps, a colored icon and the file name appears in the margin alongside the steps, such as these:



page.html



style.css



script.js



vector.svg



embed.pdf



audio.ogg



video.mp4

For convenience I have placed all the source code files and associated files featured in this book into a single ZIP archive file, which you can obtain by following these easy steps:

-
- The screenshot shows a Windows File Explorer window titled 'Sources'. The address bar indicates the current location is 'Sources'. The search bar contains the text 'Search Sources'. The main area displays a grid of 24 items, each with a file icon and a name. The items are arranged in four rows and six columns. The first row includes 'dom.png', 'dragdrop.css', 'dragdrop.html', 'dragdrop.js', 'dragdrop.png', 'echo.css', 'echo.pl', and 'emphasis.html'. The second row includes 'enhance.html', 'favicon.ico', 'figure.css', 'figure.html', 'fish.png', 'footer.css', 'footer.html', and 'format.html'. The third row includes 'fragment.css', 'fragment.html', 'front.png', 'geo.css', 'geo.html', 'geo.js', 'geo.png', and 'go.png'. The fourth row includes 'grn.png', 'heading.css', 'heading.html', 'hello.html', 'hidden.html', 'hidden.js', 'HTML5_Logo_64.png', and 'hyperlink.css'. The bottom status bar shows '219 items'.
- | File Name | File Type |
|-------------------|-------------------|
| dom.png | Image (PNG) |
| dragdrop.css | Text (CSS) |
| dragdrop.html | Text (HTML) |
| dragdrop.js | Text (JavaScript) |
| dragdrop.png | Image (PNG) |
| echo.css | Text (CSS) |
| echo.pl | Text (Perl) |
| emphasis.html | Text (HTML) |
| enhance.html | Text (HTML) |
| favicon.ico | Image (ICO) |
| figure.css | Text (CSS) |
| figure.html | Text (HTML) |
| fish.png | Image (PNG) |
| footer.css | Text (CSS) |
| footer.html | Text (HTML) |
| format.html | Text (HTML) |
| fragment.css | Text (CSS) |
| fragment.html | Text (HTML) |
| front.png | Image (PNG) |
| geo.css | Text (CSS) |
| geo.html | Text (HTML) |
| geo.js | Text (JavaScript) |
| geo.png | Image (PNG) |
| go.png | Image (PNG) |
| grn.png | Image (PNG) |
| heading.css | Text (CSS) |
| heading.html | Text (HTML) |
| hello.html | Text (HTML) |
| hidden.html | Text (HTML) |
| hidden.js | Text (JavaScript) |
| HTML5_Logo_64.png | Image (PNG) |
| hyperlink.css | Text (CSS) |

Mike McGrath

1

Getting started

*Welcome to the exciting
world of the HTML5 web.*

*This chapter introduces
HTML5 and demonstrates
how to create a “barebones”
valid HTML5 document.*

- 10** Introducing HTML5
- 12** Addressing web pages
- 14** Defining document structure
- 16** Creating a document
- 18** Validating documents
- 20** Employing an HTML editor
- 22** Summary

HTML



Sir Tim Berners-Lee,
creator of
the World Wide Web.



The W3C is the recognized body that oversees the HTML standard and other standards on the web. See the latest developments on their informative website at www.w3.org

Introducing HTML5

Historically, the desire to have text printed in specific formats meant that original manuscripts were “marked up” with annotation to indicate to the book-printer how the author would like sections of text laid out. This annotation had to be concise and needed to be easily understood by both the printer and the author. A series of commonly recognized abbreviations therefore formed the basis of a standard markup language.

HyperText Markup Language (HTML) is a modern standard markup language that uses common abbreviations called “tags” to indicate to the web browser how the author would like to have sections of a web page laid out. It was first devised in March 1989 by British physicist Tim Berners-Lee at CERN in Switzerland (the European organization for nuclear research) to share all computer-stored information between the CERN physicists. Berners-Lee created a text browser to transfer information over the internet using hypertext to provide point-and-click navigation. In May 1990 this system was named the World Wide Web, and was enhanced in 1993 when college student Marc Andreessen added an image tag. Now that HTML could display both text and images, the World Wide Web quickly became hugely popular.

As various web browsers were developed, their makers began to add individual proprietary tags – effectively creating their own versions of HTML! The World Wide Web Consortium (W3C) standards organization recognized the danger that HTML could become fragmented, so they created a standard specification to which all web browsers should adhere. This successfully encouraged the browser makers to support the standard tags. The W3C’s HTML specification was continually revised to introduce new features until the publication of HTML version 4.01 at the turn of the century. At that time the W3C also published a specification for XHTML (eXtensible HTML), which strictly required all code to be “well-formed”, to comply with the rules of eXtensible Markup Language (XML). This attempt to coerce web authors into adopting rigorous syntax, as Berners-Lee admits, did not work. So the W3C returned to HTML and finally produced a specification for version 5 in October 2014. This was updated on November 1, 2016 to version 5.1. The supported features of the HTML 5.1 version are described and demonstrated in this book, where it is generically referred to as “HTML5” or just plain “HTML”.

...cont'd

What's new in HTML5?

Placing great emphasis on backward compatibility, HTML5 is largely a superset of the previous versions, but it introduces some new features that let authors create more meaningful web pages:

- **<main>** – a structural element to contain the main content
- **<article>** – a structural element to contain stand-alone items of content, such as a self-contained topic
- **<section>** – a structural element to group together associated content, such as articles related to a common topic
- **<aside>** – a structural element to contain related content
- **<figure>**, **<figcaption>** – structural elements to contain stand-alone illustrations, diagrams, or photos for reference
- **<header>** – a structural element to contain page header content, such as a title, logo, and navigation
- **<footer>** – a structural element to contain page footer content, such as copyright information and contact details
- **<ruby>**, **<rt>**, and **<rp>** – semantic elements to indicate pronunciation for East Asian languages, such as Japanese
- **<audio>**, **<video>**, and **<source>** – embedding elements to incorporate audio and video media, such as MP3 music files
- **<embed>** – an embedding element to incorporate media that does require an external plug-in, such as movies in SWF format
- **<canvas>** – an embedding element to create an area in which to dynamically draw bitmap graphics, such as graphs
- **<details>**, **<summary>** – interactive elements to contain additional information that users can choose to read
- **<menu>**, **<menuitem>** – elements to add functionality to the web browser's context menu

Drawing on the area provided by the **<canvas>** element is accomplished exclusively using JavaScript and the Canvas 2D API (Application Programming Interface). HTML5 also includes Drag and Drop, Web Storage, and Messaging APIs with which JavaScript can provide dynamic web page functionality.



Much effort has been made in HTML5 so that it does not “break the web” – by continuing to define how browsers should deal with legacy markup code.



HTML5 finally brings intrinsic support for audio and video content with codec support built into the browsers.



The New icon pictured above indicates new features introduced in the very latest W3C HTML 5.1 specifications.

Addressing web pages

The World Wide Web comprises a series of large-capacity computers, known as “web servers”, which are connected to the internet via telephone lines and satellites. The web servers each use the HyperText Transfer Protocol (HTTP) as a common communication standard to allow any computer connected to any web server to access files across the web.

HTML web pages are merely plain text files that have been saved with a “.htm” or “.html” file extension, such as **index.html**.

In order to access a file across the web, its web address must be entered into the address field of the web browser. The web address is formally known as its “Uniform Resource Locator” (URL) and typically has three parts:

- **Protocol** – any URL using the HTTP protocol begins by specifying the protocol as **http://** or secure **https://**
- **Domain** – the host name of the computer from which the file can be downloaded. For instance, **www.example.com**
- **Path** – the virtual path to the file on the named domain, including any parent directory names where applicable. For instance, **/htdocs/index.html**

A URL describing the location of a file by protocol, domain, and path is stating its “absolute address”. So the absolute address of the file described by the protocol, domain, and path components above is **http://www.example.com/htdocs/index.html**

Code contained within an HTML web page can reference other HTML files in any domain by their absolute address. HTML files resident within the same domain can also be referenced more simply by their “relative address”, which means that files located within the same directory can be referenced just by their file name. For instance, a file named “adjacent.html” located in the same directory can be referenced simply as **adjacent.html**.

Additionally, a relative address can reference a file within the parent directory by prefixing its name with “../”. For instance, a file named “higher.html” in the parent directory can be referenced from the current directory as **../higher.html**.



A web page address (URL) cannot contain any blank spaces.



Where an address states only the HTTP protocol and a domain name, most web servers are configured to seek a file named **index.html** in their default directory.

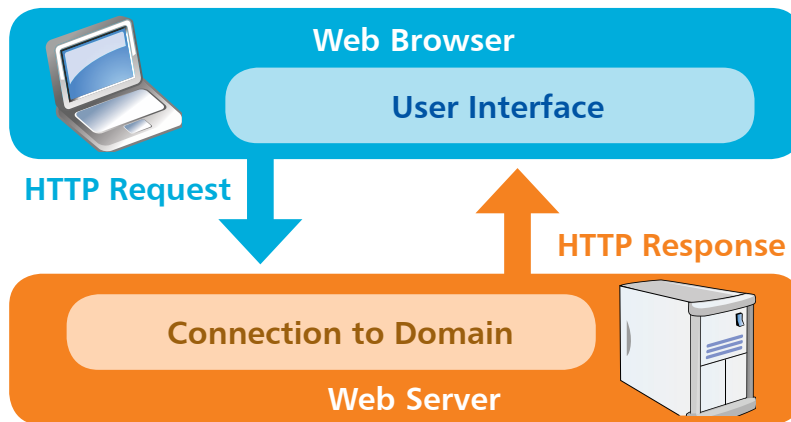
...cont'd

How do web servers work?

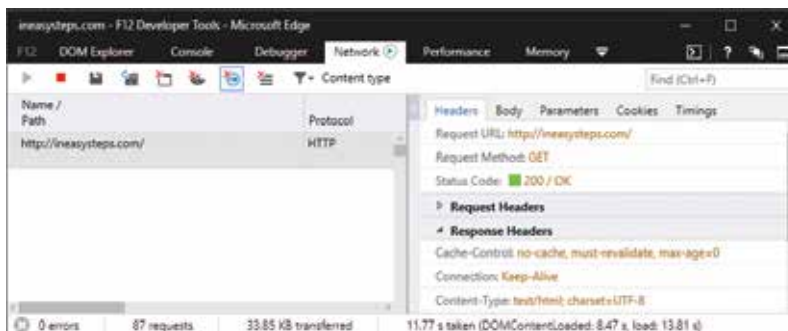
When you enter a URL into the browser address field, the browser first examines the protocol. Where the protocol is specified as HTTP, or assumed to be HTTP if unspecified, the browser recognizes that a file is being sought from a web server. It then contacts a Domain Name Server (DNS) to look up the numerical Internet Protocol (IP) address of the specified domain name. Next, a connection is established with the web server at that IP address to request the file at the specified path. When the file is successfully located it is copied back to the browser, otherwise the web server sends an error code, such as “404 – Page Not Found”.



The Domain Name Server is typically run by your Internet Service Provider or by your company.



A successful response sends HTTP headers to the web browser, describing the nature of the response, along with a copy of the requested file. The HTTP headers are not normally visible but can be examined using various development tools, such as the F12 Developer Tools feature in the Microsoft Edge web browser:



Notice in the headers that the Content-Type is “text/html” – the MIME type used by all web servers to describe plain text HTML files.

Defining document structure

The structure of an HTML5 document has these three parts:

- **Document type declaration** – declaring precisely which version of HTML is used to mark up the document
- **Head section** – providing descriptive data about the document itself, such as the document’s title and the character set used
- **Body section** – containing the content that is to appear when the document gets loaded into a web browser

Document type declaration

The document type declaration must appear at the start of the first line of every HTML5 document to ensure the web browser will “render” (display) the document in “Standards Mode” – following the HTML5 specifications. The document type declaration tag for all HTML5 documents looks like this:

```
<!DOCTYPE HTML>
```

It is important to note that HTML5 is not a case-sensitive language – so the document type declaration tag, and all other tags, may alternatively be written in any combination of uppercase and lowercase characters. For example, the following are all valid:

```
<!DOCTYPE html>
```

```
<!Doctype Html>
```

```
<!doctype html>
```

The choice of capitalization is yours, but it is recommended you adhere consistently to whichever style you choose. The document type declaration tag capitalization style favored throughout this book uses all uppercase to emphasize its prominence as the very first tag on each page – but all other tags are in all lowercase.

Those familiar with previous versions of HTML may be surprised at the simplicity of the HTML5 document type declaration. In fact, the document type declaration in earlier versions was not actually part of the HTML language – so required lengthy references to schema documents. By contrast, the HTML5 document type declaration is an intrinsic part of HTML itself.



The document type declaration in earlier versions of HTML was part of the Standard Generalized Markup Language (SGML) from which HTML is derived.

...cont'd

The entire document head section and body section can be enclosed within a pair of `<html>` `</html>` tags to contain the rest of the document. The HTML5 specification actually states that these are optional, but it is logical to provide a single “root” element. Most HTML tags are used in pairs like this to act as “containers” with the syntax `< tagname > data </ tagname >`.

Head section

The document’s head section begins with an HTML opening `<head>` tag and ends with a corresponding closing `</head>` tag. Data describing the document can be added later between these two tags to complete the HTML document’s head section.

Body section

The document’s body section begins with an HTML opening `<body>` tag and ends with a corresponding closing `</body>` tag. Data content to appear in the browser can be added later between these two tags to complete the HTML document’s body section.

Code comments

Comments can be added at any point within both the head and body sections between a pair of `<!--` and `-->` tags. Anything that appears between the comment tags is ignored by the browser.

Fundamental structure

So, the markup tags that create the fundamental structure of every HTML5 document look like this:

```
<!DOCTYPE HTML>

<html>
  <head>
    <!-- Data describing the document to be added here -->
  </head>

  <body>
    <!-- Data content to appear in the browser to be added here -->
  </body>
</html>
```



An HTML “element” is any matching pair of opening and closing tags, or any single tag not requiring a closing tag – as described in the HTML5 element tags list on the inside front cover of this book.



The “invisible” characters that represent tabs, newlines, carriage returns, and spaces are collectively known as “whitespace”. They may optionally be used to inset the tags for clarity.



The `<meta>` tag is a single tag – it does not have a matching closing tag. See the element tags list on the inside front cover of this book to find other single tags.

Creating a document

The fundamental HTML5 document structure, described on page 15, can be used to create a simple HTML5 document in any plain text editor – such as Windows' Notepad application. In order to create a valid “barebones” HTML5 document, information must first be added defining the document's primary written language, its character encoding format, and its title.

The document's primary language is defined by assigning a standard language code to a **lang** “attribute” within the opening `<html>` root tag. For the English language the code is **en**, so the complete opening root element looks like this: `<html lang="en">`.

The document's character encoding format is defined by assigning a standard character-set code to a **charset** attribute within a `<meta>` tag placed in the document's head section. The recommended encoding is the popular 8-bit Unicode Transformation Format for which the code is **UTF-8**, so the complete element looks like this: `<meta charset="UTF-8">`.

Finally, the document's title is defined by text between a pair of `<title>` `</title>` tags placed in the document's head section.

Follow these steps to create a valid “barebones” HTML5 document:



hello.html



HTML documents should not be created in word processors, such as MS Word, as they include additional information in their file formats.

1

Launch your favorite plain text editor, then start a new document with the HTML5 document type declaration
`<!DOCTYPE HTML>`

2

Below the document type declaration, add a root element that defines the document's primary language as English
`<html lang="en">`
`<!-- Head and Body sections to be added here -->`
`</html>`

3

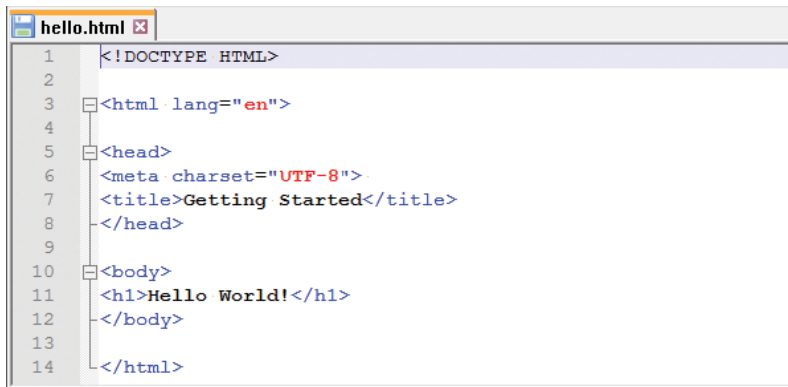
Within the root element, insert a document head section
`<head>`
`<!-- Descriptive information to be added here -->`
`</head>`

4

Within the head section, insert an element defining the document's encoding character set
`<meta charset="UTF-8">`

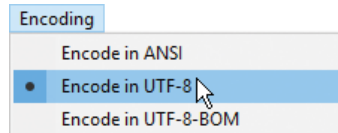
...cont'd

- 5 Next, within the head section, insert an element defining the document's title
`<title>Getting Started</title>`
- 6 After the head section, insert a document body section
`<body>`
`<!-- Actual document content to be added here -->`
`</body>`
- 7 Within the body section, insert a size-one large heading
`<h1>Hello World!</h1>`

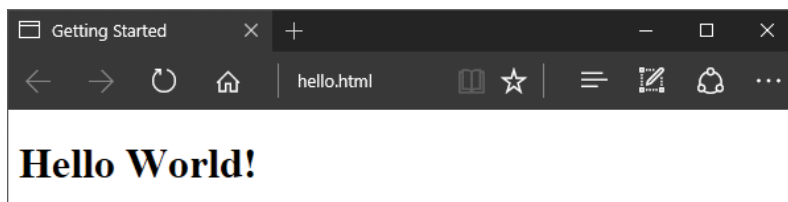


```
1 <!DOCTYPE HTML>
2
3 <html lang="en">
4
5 <head>
6   <meta charset="UTF-8">
7   <title>Getting Started</title>
8 </head>
9
10 <body>
11   <h1>Hello World!</h1>
12 </body>
13
14 </html>
```

- 8 Save the document as "hello.html", setting the encoding to the popular "UTF-8" format



- 9 Now, open the HTML5 document in a modern web browser to see the title displayed on the title bar or tab, and the document content displayed as a large heading



The quotation marks around an attribute value are usually optional, but are required for multiple values. For consistency, attribute values in the examples throughout this book are all surrounded by quotation marks.



Windows' Notepad automatically adds a hidden "Byte Order Mark" (BOM) to the file, while other editors (such as Notepad++ shown here) allow this to be omitted. Notepad++ can be freely downloaded from notepad-plus-plus.org



The W3C's online HTML validator can be found at validator.w3.org

Validating documents

Just as text documents may contain spelling and grammar errors, HTML documents may contain various errors that prevent them from conforming to the specification rules. In order to verify that an HTML document does indeed conform to the rules of its specified document type declaration, it can be tested by a validator tool. Only HTML documents that pass the validation test successfully are sure to be valid documents.

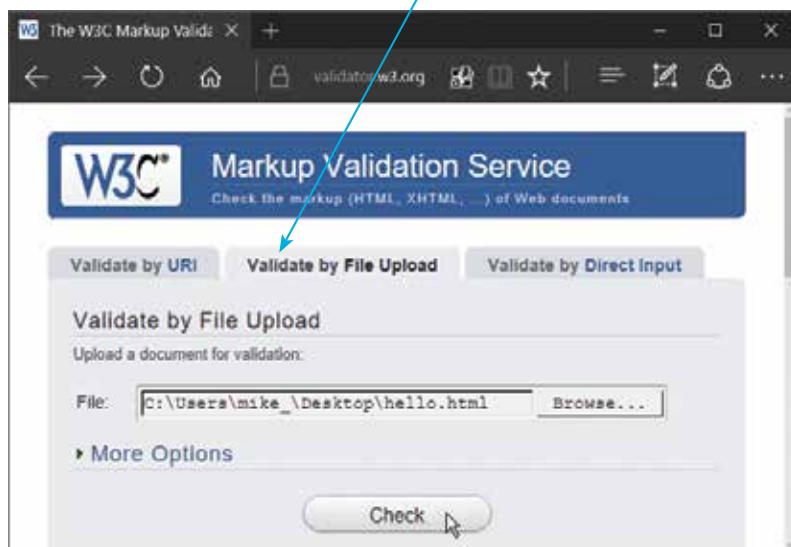
Web browsers make no attempt at validation, so it is well worth verifying every HTML document with a validator tool before it is published, even when the content looks fine in your web browser. When the browser encounters HTML errors it will make a guess at what is intended – but different browsers can make different interpretations so may display the document incorrectly. Conversely, valid HTML documents should always appear correctly in any standards-compliant browser.

The World Wide Web Consortium (W3C) provides a free online validator tool that checks the syntax of web documents:

- 1 With an internet connection, open your web browser and navigate to the W3C Validator Tool at **validator.w3.org**, then click on the “Validate by File Upload” tab

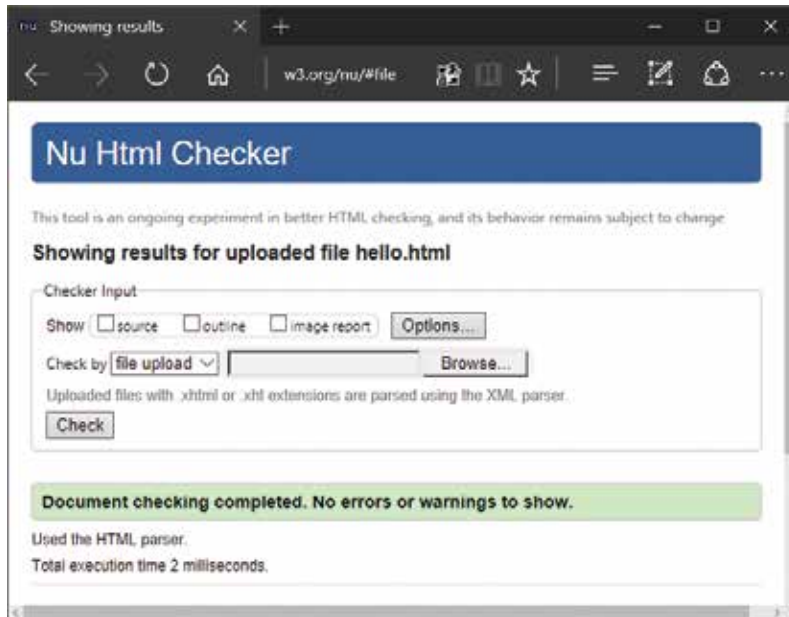


Other tabs in the validator allow you to enter the web address of an HTML document located on a web server to “Validate by URI” or copy and paste all code from a document to “Validate by Direct Input”.



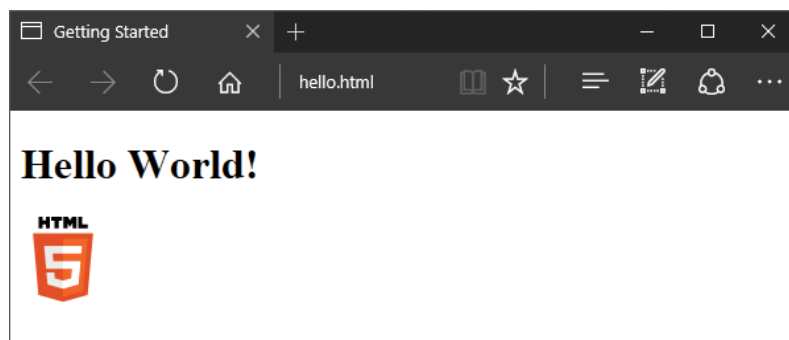
...cont'd

- 2 Click the “Browse” button, then navigate to the HTML document you wish to validate – once selected, its local path appears in the validator’s “File” field
- 3 Next, click the validator’s “Check” button to upload a copy of the HTML document and run the validation test – the results will then be displayed

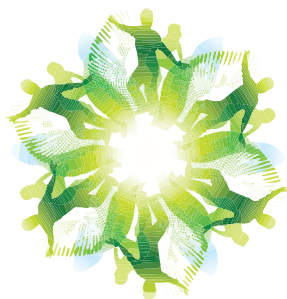


The validator automatically detects the document’s character set and HTML version.

If validation fails, the errors are listed so you may easily correct them. When validation succeeds, you may choose to include an icon at the end of the document demonstrating HTML5 support:



The HTML5 support logo is available in several sizes and formats – find more details online at w3.org/html/logo



At the time of writing, Expression Web 4 is available free for download at microsoft.com/en-us/download/details.aspx?id=36179

Employing an HTML editor

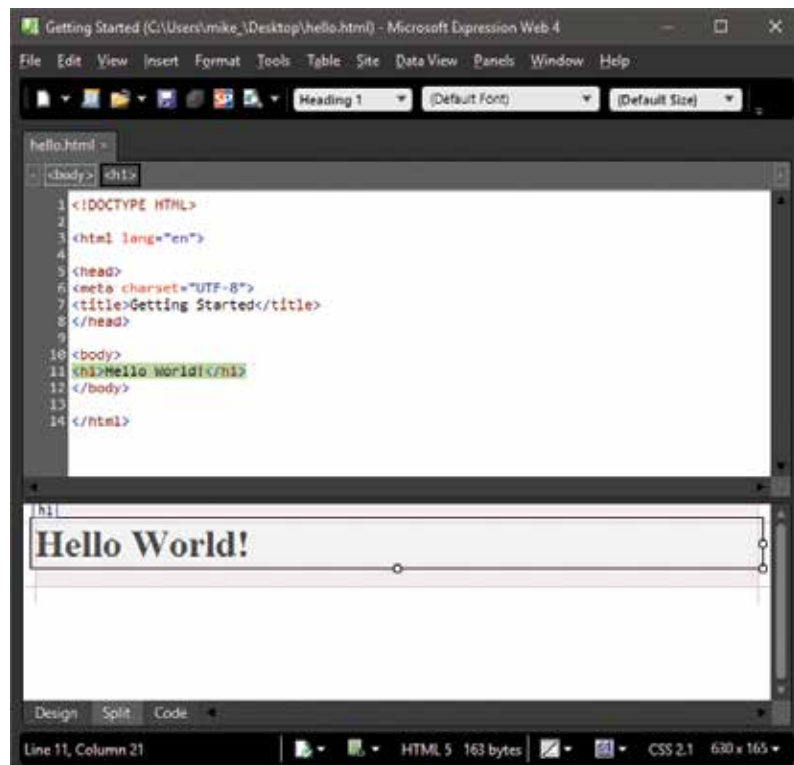
HTML code can be created in any plain text editor that provides encoding in the recommended UTF-8 format. As long as the code in the new text file is saved with a file extension of “.html” or “.htm”, an HTML document is created. This file can then be opened in any web browser, such as Microsoft Edge, to see how the HTML code is interpreted to “render” (display) the content on the screen.

Older web browsers, such as Internet Explorer 8 or earlier, will not fully recognize the modern HTML5 markup code. It is best, therefore, to view HTML5 web pages in the latest version of the Microsoft Edge, Internet Explorer, Firefox, Google Chrome, Opera, and Safari web browsers.

Some HTML authors prefer to use specialized HTML editors that colorize the various parts of the source code for greater clarity, and offer further features. Microsoft’s Expression Web editor, shown below, is a popular choice for some HTML authors.

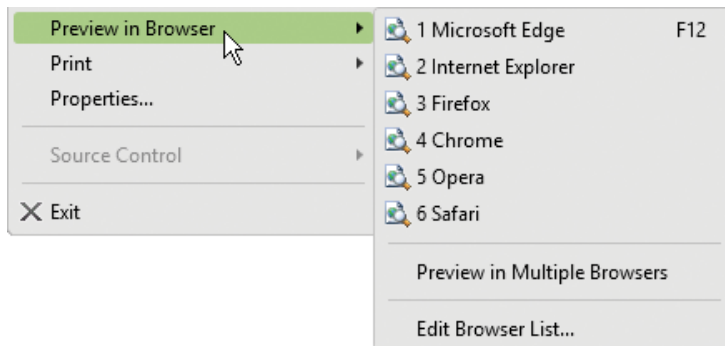


The Expression Web interface also allows web pages to be created visually by dragging components onto its Design window – but knowledge of HTML is often helpful in fine-tuning the web page.



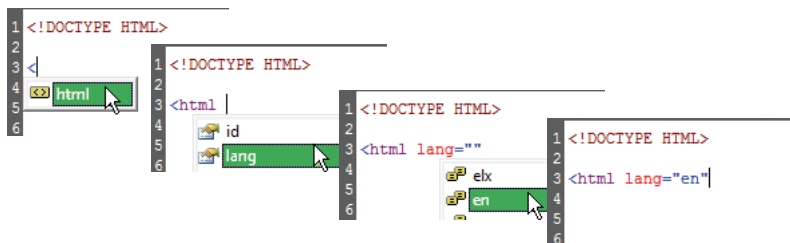
...cont'd

The Split view in Expression Web provides a live preview of what the HTML document will look like in a web browser. Additionally, a built-in menu lets you quickly view the document in any web browser installed on your computer.



The W3C online validator tool can also be used to verify correct HTML code, along with Expression Web's Compatibility Checker tool.

Expression Web's "Compatibility Checker" tool lets you easily locate code errors that do not conform to the declared document type. Additionally, like other Microsoft development tools, the Expression Web editor has "IntelliSense", which identifies syntax errors live as you type the HTML code. This feature also provides context-sensitive menus that can insert HTML tags compatible with the current point in the document. For example, when you type a "<" in the head section, IntelliSense presents a list of tags that may be inserted at that point. After selecting a tag and typing a space, IntelliSense then presents a list of attributes that may be inserted within that tag.



Expression Web also provides a customizable Code Snippet facility to quickly insert frequently used chunks of code – press **Ctrl + Enter** to see the Code Snippet list.

By default, IntelliSense automatically inserts a matching closing tag, if appropriate, whenever you type an opening tag. Some authors dislike this ability, however, but the Expression Web options allow IntelliSense features to be turned on and off individually to customize the editor to your personal taste.

Summary

- The World Wide Web Consortium (W3C) is the recognized body that oversees standards on the web.
- HTML5 introduces intrinsic support for audio and video media content.
- JavaScript can draw on canvas areas and provide dynamic web page functionality using the new HTML5 APIs.
- HyperText Transfer Protocol (HTTP) is the common communication standard used by web servers.
- Uniform Resource Locator (URL) is an absolute web address comprising protocol, domain, and path components.
- A relative address can reference an adjacent file by its file name and may use the syntax “../” to reference a parent directory.
- Web servers send response headers back to the requesting computer and a copy of the file requested, or an error code.
- Each HTML5 document should have a document type declaration, head section, and body section.
- Information about the document itself is contained within the head section, and content is contained within the body section.
- The document’s written language is specified to the **lang** attribute in the opening **<html>** root element tag.
- The document’s character-set encoding is specified to the **charset** attribute in a **<meta>** tag, within the head section.
- The document’s title is specified between **<title>** **</title>** tags, within the head section.
- The free online W3C validator tool should be used to verify that the HTML5 document is free of errors.
- HTML5 documents can be created in a plain text editor or a specialized HTML editor such as Microsoft Expression Web.