TEMASEK JUNIOR COLLEGE

Temasek Junior College
JC1 H2 Computing
HTML

The **World Wide Web Consortium (W3C)** is an international community where Member organizations, a full-time staff, and the public work together to develop Web standards.

Design Principles
The following design principles guide W3C's work.

Web for All
The social value of the Web is that it enables human communication, commerce, and entopportunities to share knowledge. One of W3C's primary goals is to make these benefits available to all people, whatever their hardware, software, network infrastructure, native language, culture, geographical location, or physical or mental ability.

Web on Everything
The number of different kinds of devices that can access the Web has grown immensely. Mobile phones, smart phones, personal digital assistants, interactive television systems, voice response systems, kiosks and even certain domestic appliances can all access the Web.

Vision
W3C's vision for the Web involves participation, sharing knowledge, and thereby building trust on a global scale.

Web for Rich Interaction
The Web was invented as a communications tool intended to allow anyone, anywhere to share information. For many years, the Web was a "read-only" tool for many. Blogs and wikis brought more authors to the Web, and social networking emerged from the flourishing market for content and personalized Web experiences. W3C standards have supported this evolution thanks to strong architecture and design principles.

Web of Data and Services
Some people view the Web as a giant repository of linked data while others as a giant set of services that exchange messages. The two views are complementary, and which to use often depends on the application.

Web of Trust
The Web has transformed the way we communicate with each other. In doing so, it has also modified the nature of our social relationships. People now "meet on the Web" and carry out commercial and personal relationships, in some cases without ever meeting in person. W3C recognizes that trust is a social phenomenon, but technology design can foster trust and confidence. As more activity moves on-line, it will become even more important to support complex interactions among parties around the globe.

What is HTML?

HTML is the language for ***describing the structure of Web pages***. HTML gives authors the means to:

- Publish online documents with headings, text, tables, lists, photos, etc.
- Retrieve online information via hypertext links, at the click of a button.
- Design forms for conducting transactions with remote services, for use in searching for information, making reservations, ordering products, etc.
- Include spread-sheets, video clips, sound clips, and other applications directly in their documents.

With HTML, authors describe the structure of pages using *markup*. The *elements* of the language label pieces of content such as "paragraph," "list," "table," and so on.

HTML Element Reference
HTML Tags Ordered Alphabetically

Tag	Category	Description
<!--...-->	Basic	Defines a comment
<!DOCTYPE>	Basic	Defines the document type
<a>	**Links**	**Defines a hyperlink**
<b>	**Formatting**	**Defines bold text**
<body>	**Basic**	**Defines the document's body**
 	Basic	Defines a single line break (reference)
<button>	Forms and Input	Defines a clickable button (reference)
** **	**Styles and Semantics**	**Defines a section in a document**
<dl>	Lists	Defines a description list (reference)
<dt>	Lists	Defines a term/name in a description list (reference)
<form>	**Forms and Input**	**Defines an HTML form for user input**
<h1> to <h6>	Basic	Defines HTML headings (reference)
<head>	**Basic, Meta Info**	**Defines information about the document**
<hr>	**Basic**	**Defines a thematic change in the content**
<html>	**Basic**	**Defines the root of an HTML document**
<i>	**Formatting**	**Defines a part of text in an alternate voice or mood**
<img>	**Images**	**Defines an image**
<input>	**Forms and Input**	**Defines an input control**
	Lists	Defines a list item (reference)
<link>	**Links**	**Defines the relationship between a document and an external resource (most used to link to style sheets)**
<p>	**Basic**	**Defines a paragraph**
<select>	Forms and Input	Defines a drop-down list (reference)
****	**Styles and Semantics**	**Defines a section in a document**
<table>	**Tables**	**Defines a table**
<td>	**Tables**	**Defines a cell in a table**
<textarea>		**Defines a multiline input control (text area)**
<th>	**Tables**	**Defines a header cell in a table**
<title>	**Basic**	**Defines a title for the document**
<tr>	**Tables**	**Defines a row in a table**
	Lists	Defines an unordered list (reference)

HTML Introduction

What is HTML?

HTML is the standard markup language for *creating Web pages*.

- HTML stands for Hyper Text Markup Language
- HTML describes the structure of Web pages using markup
- HTML elements are the building blocks of HTML pages
- HTML elements are represented by tags
- HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
- Browsers do not display the HTML tags, but use them to render the content of the page

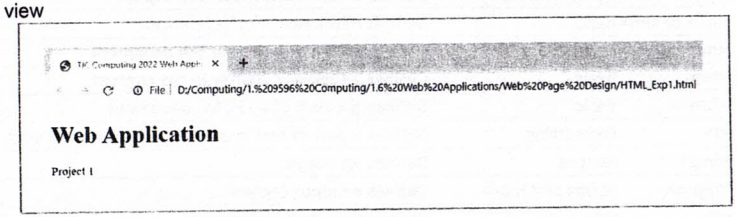
A Simple HTML Document

Example

```
<!DOCTYPE html>
<html>
<head>
<title>TJC Computing 2022 Web Application</title>
</head>
<body>
<h1>Web Application </h1>
<p>Project 1</p>
</body>
</html>
```

Web Browsers

The purpose of a web browser (Chrome, IE, Firefox, Safari) is to read HTML documents and display them.
The browser does not display the HTML tags, but uses them to determine how to display the document view



How to View HTML Source?

To find out, right-click in the page and select "View Page Source" (in Chrome) or "View Source" (in IE), or similar in other browsers. This will open a window containing the HTML source code of the page.

Inspect an HTML Element: Right-click on an element (or a blank area), and choose "Inspect" or "Inspect Element" to see what elements are made up of (you will see both the HTML and the CSS). You can also edit the HTML or CSS on-the-fly in the Elements or Styles panel that opens.

Explained

- The <!DOCTYPE html> ***declaration*** defines this document to be HTML5
- The <html> element is the ***root element*** of an HTML page
- The <head> element contains ***meta information*** about the document
- The <title> element specifies a ***title*** for the document
- The <body> element contains the ***visible page content***
- The <h1> element defines a large heading
- The element defines a paragraph

HTML Tags

HTML tags are ***element names*** surrounded by **angle brackets**:

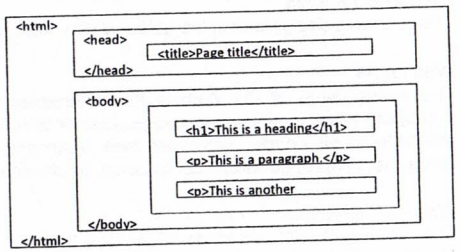
<tagname>content goes here...</tagname>

- HTML tags normally come **in pairs** like and
- The first tag in a pair is the **start tag,** the second tag is the **end tag**
- The end tag is written like the start tag, but with a **forward slash** inserted before the tag name

Tip: The start tag is also called the **opening tag**, and the end tag the **closing tag**.

HTML Page Structure

Below is a visualization of an HTML page structure:



Note: Only the content inside the <body> section (the white area above) is displayed in a browser.

The <!DOCTYPE> Declaration

It represents the document type, and helps browsers to display web pages correctly.
It must only appear once, at the top of the page (before any HTML tags).
The <!DOCTYPE> declaration is not case sensitive.
The <!DOCTYPE> declaration for HTML5 is: **<!DOCTYPE html>**

HTML Versions

Since the early days of the web, there have been many versions of HTML:

Version	Year
HTML	1991
HTML 4.01	1999
XHTML	2000
HTML5	2014

HTML Editors Notepad++

HTML Documents

All HTML documents must start with a document type declaration: **<!DOCTYPE html>**.
The HTML document itself begins with **<html>** and ends with **</html>**.
The visible part of the HTML document is between **<body>** and **</body>**.

HTML Elements

An ***HTML*** element usually consists of a **start** tag and **end** tag, with the content inserted in between:

<tagname> Content goes here... **</tagname>**

The HTML **element** is everything from the start tag to the end tag:

Example

```
<p> Computing</p>
```

Start tag	Element content	End tag
<h1>	Science stream	</h1>
	Computing.	

Nested HTML Elements
HTML elements can be nested (elements can contain elements).
All HTML documents consist of nested HTML elements.

This example contains ***four*** HTML elements:

Example

```
<!DOCTYPE html>
<html>
<body>
<h1>Science Stream</h1>
<p>Computing.</p>
</body>
</html>
```

Example Explained [Self reading]

The **<html>** element defines the **whole document**.
It has a **start** tag <html> and an **end** tag </html>.
The element **content** is another HTML element (the <body> element).
The **<body>** element defines the **document body**.
It has a **start** tag <body> and an **end** tag </body>.
The element **content** is two other HTML elements (<h1> and).
The **<h1>** element defines a **heading**.
It has a **start** tag <h1> and an **end** tag </h1>.
The element **content** is: Science Stream.
The ** ** element defines a **paragraph**.
It has a **start** tag and an **end** tag .
The element **content** is: Computing.

Do Not Forget the End Tag

Some HTML elements will display correctly, even if you forget the end tag:

Example

```
<html>
<body>
<p>This is a paragraph
<p>This is a paragraph
</body>
</html>
```

The example above works in all browsers, because the closing tag is considered optional.
Never rely on this. It might produce unexpected results and/or errors if you forget the end tag.

Empty HTML Elements
HTML elements with no content are called empty elements.

 is an empty element without a closing tag (the
 tag defines a line break).
Empty elements can be "closed" in the opening tag like this:
.

Use Lowercase Tags
HTML tags are not case sensitive: <P> means the same as .
W3C recommends lowercase in HTML, and **demands** lowercase for stricter document types like XHTML.
We always use lowercase tags.

HTML Comments

Comment tags are used to insert comments in the HTML source code.

HTML Comment Tags
You can add comments to your HTML source by using the following syntax:

Example

```
<!-- Write your comments here -->
```

Notice that there is an exclamation point (!) in the opening tag, but not in the closing tag.
Note: Comments are not displayed by the browser, but they can help document your HTML source code.

With comments you can place notifications and reminders in your HTML:

Example

```
<!-- This is a comment -->
<p>System Implementation.</p>
<!-- Remember to add more information here -->
```

Comments are also great for debugging HTML, because you can comment out HTML lines of code, one at a time, to search for errors:

Example

```
<!-- Do not display this at the moment
<img border="0" src="tjc.jpg" alt="TJC">
-->
```

HTML Head

The HTML <head> Element - The <head> element is a container for metadata (data about data) and is placed between the <html> tag and the <body> tag.

HTML metadata is data about the HTML document. Metadata is not displayed.
Metadata typically define the document title, character set, styles, links, scripts, and other meta information.
The following tags describe metadata: <title>, <style>, <meta>, <link>, <script>, and <base>.

The HTML <title> Element - The <title> element defines the title of the document, and is required in all HTML/XHTML documents.

The <title> element:

- defines a title in the browser tab
- provides a title for the page when it is added to favorites
- displays a title for the page in search engine results

A simple HTML document:

Example

```
<!DOCTYPE html>
<html>
<head>
  <title>Introduction Networking</title>
</head>
<body>
 Networking is.....
</body>
</html>
```

The HTML <link> Element - The <link> element is used to link to external style sheets

Example

```
<link rel="stylesheet" href="mystyle.css">
```

The HTML <meta> Element
The <meta> element is used to specify which character set is used, page description, keywords, author, and other metadata.
Metadata is used by browsers (how to display content), by search engines (keywords), and other web services.

Define the character set used: `<meta charset="UTF-8">`
Define a description of your web page: `<meta name="description" content="Free Web tutorials">`
Define keywords for search engines: `<meta name="keywords" content="HTML, CSS, XML, JavaScript">`
Define the author of a page: `<meta name="author" content="John Doe">`
Refresh document every 30 seconds: `<meta http-equiv="refresh" content="30">`

Example

```
<meta charset="UTF-8">
<meta name="description" content="Computing tutorial 4">
<meta name="keywords" content="HTML,CSS,XML,JavaScript">
<meta name="author" content="Paik Poh Leong">
```


HTML Encoding (Character Sets)

To display an HTML page correctly, a web browser must know which character set (character encoding) to use.

What is Character Encoding?
ASCII was the first character encoding standard (also called character set). ASCII defined 128 different alphanumeric characters that could be used on the internet: numbers (0-9), English letters (A-Z), and some special characters like ! $ + - ( ) @ < > .

Because ANSI was so limited, HTML 4 also supported UTF-8.
UTF-8 (Unicode) covers almost all of the characters and symbols in the world.
The default character encoding for HTML5 is UTF-8.

The HTML charset Attribute
To display an HTML page correctly, a web browser must know the character set used in the page.
This is specified in the <meta> tag:

For HTML4:
<meta http-equiv="Content-Type" content="text/html;charset=ISO-8859-1">

For HTML5:
<meta charset="UTF-8">

The ASCII Character Set
ASCII uses the values from 0 to 31 (and 127) for control characters.
ASCII uses the values from 32 to 126 for letters, digits, and symbols.
ASCII does not use the values from 128 to 255.

The UTF-8 Character Set
UTF-8 is identical to ASCII for the values from 0 to 127.
UTF-8 does not use the values from 128 to 159.
UTF-8 is identical to both ANSI and 8859-1 for the values from 160 to 255.
UTF-8 continues from the value 256 with more than 10 000 different characters.

HTML Attributes

Attributes provide additional information about HTML elements.

HTML Attributes

- All HTML elements can have **attributes**
- Attributes provide **additional information** about an element
- Attributes are always specified in **the start tag**
- Attributes usually come in name/value pairs like: **name="value"**

The href Attribute - HTML links are defined with the **<a>** tag. The link address is specified in the **href** attribute:

Example

```
<a href=" https://www.temasekjc.moe.edu.sg/ "><h4>Temasek Junior College</h4></a>
```

The src Attribute - HTML images are defined with the **<img>** tag.
The filename of the image source is specified in the **src** attribute:

Example

```
<img src="https://www.temasekjc.moe.edu.sg/qql/slot/u152/style/top/logo.png">
```

The width and height Attributes - Images in HTML have a set of size attributes, which specifies the width and height of the image:

Example

```
<img src="https://www.temasekjc.moe.edu.sg/qql/slot/u152/style/top/logo.png"
width="204" height="142">
```

The image size is specified in pixels: width="204" means 104 pixels wide.

The alt Attribute - The **alt** attribute specifies an alternative text to be used, when an image cannot be displayed.
The value of the attribute can be read by screen readers. This way, someone "listening" to the webpage, e.g. a blind person, can "hear" the element.

Example

```
<img src="https://www.temasekjc.moe.edu.sg/qql/slot/u152/style/top/logo.png" alt
="Temasek Junior College" >
```


Quote Attribute Values
The **href** attribute, demonstrated above, can be written as:

Example

```
<a href="https://www.temasekjc.moe.edu.sg/ "><h4>Temasek Junior College</h4></a>
```

W3C **recommends** quotes in HTML, and **demands** quotes for stricter document types like XHTML.
Sometimes it is **necessary** to use quotes. This example will not display the title attribute correctly, because it contains a space:

Example

```
<p title = Computer Science>
```

Using quotes are the most common. Omitting quotes can produce errors. Always use quotes around attribute values.

Single or Double Quotes?
Double quotes around attribute values are the most common in HTML, but single quotes can also be used.
In some situations, when the attribute value itself contains double quotes, it is necessary to use single quotes:

Example

```
<p title='I Love "Computing" Completely'>          <!-- what is title? -->
Or vice versa:
<p title="'I Love 'Computing' completely ">
```

HTML Headings

Headings are defined with the ***<h1>*** to ***<h6>*** tags.
<h1> defines the most important heading. ***<h6>*** defines the least important heading.

Example

```
<h1>Mathematics</h1>
<h2>Physics</h2>
<h2>Computing</h2>
```

Note: Browsers automatically add some white space (a margin) before and after a heading.

Headings Are Important
Search engines use the headings to index the structure and content of your web pages.
Users skim your pages by its headings. It is important to use headings to show the document structure.
<h1> headings should be used for main headings, followed by <h2> headings, then the less important <h3>, and so on.

Note: Use HTML headings for headings only. Don't use headings to make text BIG or bold.

Bigger Headings - Each HTML heading has a ***default size***. However, you can specify the size for any heading with the style attribute:

Example

```
<h1 style="font-size:60px;">Heading 1</h1>
```

HTML Horizontal Rules - The **<hr>** tag defines a thematic break in an HTML page, and is most often displayed as a horizontal rule.
The <hr> element is used to separate content (or define a change) in an HTML page:

Example

```
<h1>Science</h1>
<p>Computing</p>
<hr>
<h2>Art</h2>
<hr>
```


HTML Paragraphs

The HTML ** ** element defines a paragraph:

Example

```
<p>System Analysis.</p>
<p>System Design.</p>
```

Note: Browsers automatically add some white space (a margin) before and after a paragraph.

HTML Display
You cannot be sure how HTML will be displayed.
Large or small screens, and resized windows will create different results.
With HTML, you cannot change the output by adding extra spaces or extra lines in your HTML code.
The browser will remove any extra spaces and extra lines when the page is displayed:

Example

```
<p>WAN (Wide Area Network) is a computer network that covers a broad area. </p>
<p>Computers connected to a wide-area network are often connected through public networks, such as the telephone system. They can also be connected through leased lines or satellites </p>
```

HTML Line Breaks - The HTML **
** element defines a line break.
Use
 if you want a line break (a new line) without starting a new paragraph:

Example

```
<p> Have a large geographical range generally spreading <br> across boundaries and <br> need leased telecommunication lines.</p>
```

The
 tag is an empty tag, which means that it has no end tag.

HTML Text Formatting

Text Formatting - This text is bold, *This text is italic,* This is $_{\text{subscript}}$ and $^{\text{superscript}}$

HTML Formatting Elements - HTML defines special elements for defining text with a special meaning.
HTML uses elements like ****** and ****** for formatting output, like bold or italic text.
Formatting elements were designed to display special types of text:

HTML **** Element	HTML **** Element
The HTML **** element defines bold text, without any extra importance.	The HTML **** element defines italic text, without any extra importance.
Example	Example
This text is bold	This text is italic

HTML **<sub>** Element	HTML **<sup>** Element
The HTML **<sub>** element defines subscripted text.	The HTML **<sup>** element defines superscripted text.
Example	Example
This is _{subscripted} text.	This is ^{superscripted} text.

HTML Entities

Reserved characters in HTML must be replaced with character entities.
Characters that are not present on your keyboard can also be replaced by entities.

HTML Entities

Some characters are reserved in HTML.
If you use the less than (<) or greater than (>) signs in your text, the browser might mix them with tags.
Character entities are used to display reserved characters in HTML.

A character entity looks like this:

&*entity_name*;

OR

&#*entity_number*;

To display a less than sign (<) we must write: **<** or **<**

Advantage of using an entity name: An entity name is easy to remember.
Disadvantage of using an entity name: Browsers may not support all entity names, but the support for numbers is good.

Non-breaking Space

A common character entity used in HTML is the non-breaking space: ** **
A non-breaking space is a space that will not break into a new line.
Two words separated by a non-breaking space will stick together (not break into a new line).
This is handy when breaking the words might be disruptive.

Examples

- § 10
- 10 km/h
- 10 PM

Another common use of the non-breaking space is to prevent that browsers truncate spaces in HTML pages.
If you write 10 spaces in your text, the browser will remove 9 of them. To add real spaces to your text, you can use the ** ** character entity.
The non-breaking hyphen (‑) lets you use a hyphen character (-) that won't break.
Some Other Useful HTML Character Entities

Result	Description	Entity Name	Entity Number
	non-breaking space		
<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
"	double quotation mark	"	"
'	single quotation mark (apostrophe)	'	'
¢	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
€	euro	€	€
©	copyright	©	©
®	registered trademark	®	®

Note: Entity names are case sensitive.

ASCII Encoding Reference
Your browser will encode input, according to the character-set used in your page.
The default character-set in HTML5 is UTF-8.

Character	UTF-8	Character	UTF-8	Character	UTF-8	Character	UTF-8
space	%20	H	%48	p	%70	˜	%CB%9C
!	%21	I	%49	q	%71	™	%E2%84
"	%22	J	%4A	r	%72	š	%C5%A1
#	%23	K	%4B	s	%73	›	%E2%80
$	%24	L	%4C	t	%74	œ	%C5%93
%	%25	M	%4D	u	%75		%9D
&	%26	N	%4E	v	%76	ž	%C5%BE
'	%27	O	%4F	w	%77	Ÿ	%C5%B8
(	%28	P	%50	x	%78		%C2%A0
)	%29	Q	%51	y	%79	¡	%C2%A1
*	%2A	R	%52	z	%7A	¢	%C2%A2
+	%2B	S	%53	{	%7B	£	%C2%A3
,	%2C	T	%54	|	%7C	¤	%C2%A4
-	%2D	U	%55	}	%7D	¥	%C2%A5
.	%2E	V	%56	~	%7E	¦	%C2%A6
/	%2F	W	%57		%7F	§	%C2%A7
0	%30	X	%58	`	%E2%82%AC	¨	%C2%A8
1	%31	Y	%59		%81	©	%C2%A9
2	%32	Z	%5A	‚	%E2%80%9A	ª	%C2%AA
3	%33	[	%5B	ƒ	%C6%92	«	%C2%AB
4	%34	\	%5C	„	%E2%80%9E	¬	%C2%AC
5	%35	]	%5D	…	%E2%80%A6		%C2%AD
6	%36	^	%5E	†	%E2%80%A0	®	%C2%AE
7	%37	_	%5F	‡	%E2%80%A1	¯	%C2%AF
8	%38	`	%60	ˆ	%CB%86	°	%C2%B0
9	%39	a	%61	‰	%E2%80%B0	±	%C2%B1
:	%3A	b	%62	Š	%C5%A0	²	%C2%B2
;	%3B	c	%63	‹	%E2%80%B9	³	%C2%B3
<	%3C	d	%64	Œ	%C5%92	´	%C2%B4
=	%3D	e	%65		%C5%8D	µ	%C2%B5
>	%3E	f	%66	Ž	%C5%BD	¶	%C2%B6
?	%3F	g	%67		%8F	·	%C2%B7
@	%40	h	%68		%C2%90	¸	%C2%B8
A	%41	i	%69	‘	%E2%80%98	¹	%C2%B9
B	%42	j	%6A	’	%E2%80%99	º	%C2%BA
C	%43	k	%6B	“	%E2%80%9C	»	%C2%BB
D	%44	l	%6C	”	%E2%80%9D	¼	%C2%BC
E	%45	m	%6D	•	%E2%80%A2	½	%C2%BD
F	%46	n	%6E	–	%E2%80%93	¾	%C2%BE
G	%47	o	%6F	—	%E2%80%94	**More...**	

URL Encoding Reference
The ASCII control characters **%00-%1F** were originally designed to control hardware devices.
Control characters have nothing to do inside a URL.

ASCII Character	Description	URL-encoding
NUL	null character	%00
SOH	start of header	%01
STX	start of text	%02
ETX	end of text	%03
EOT	end of transmission	%04
ENQ	enquiry	%05
ACK	acknowledge	%06
BEL	bell (ring)	%07
BS	backspace	%08
HT	horizontal tab	%09
LF	line feed	%0A
VT	vertical tab	%0B
FF	form feed	%0C
CR	carriage return	%0D
SO	shift out	%0E
SI	shift in	%0F
DLE	data link escape	%10
DC1	device control 1	%11
DC2	device control 2	%12
DC3	device control 3	%13
DC4	device control 4	%14
NAK	negative acknowledge	%15
SYN	synchronize	%16
ETB	end transmission block	%17
CAN	cancel	%18
EM	end of medium	%19
SUB	substitute	%1A
ESC	escape	%1B
FS	file separator	%1C
GS	group separator	%1D
RS	record separator	%1E
US	unit separator	%1F

HTML Styles

Example

I am Red
I am Blue
I am Big

The HTML Style Attribute - Setting the style of an HTML element, can be done with the style attribute.

The HTML style attribute has the following syntax:

<tagname ***style*** = "property: value;">

The property is a CSS property. The value is a CSS value.

HTML Fonts - The font-family property defines the font to be used for an HTML element:

Example
```
<h1 style="font-family:verdana;"> Alfa testing </h1>
<p style="font-family:courier;"> Beta testing</p>
```

HTML Text Size - The font-size property defines the text size for an HTML element:

Example
```
<h1 style="font-size:300%;">Top-down testing</h1>
<p style="font-size:160%;">Bottom-up testing</p>
```

HTML Text Alignment - The text-align property defines the horizontal text alignment for an HTML element:

Example
```
<h1 style="text-align:center;">Unit testing</h1>
<p style="text-align:center;">Acceptance test</p>
```


HTML Colors

HTML colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

Color Names - In HTML, a color can be specified by using a color name:
Tomato, Orange, DarkBlue, LightGreen, Gray, Violet, LightGray
HTML supports 140 standard color names.

HTML Background Color - The ***background-color*** property defines the background color for an HTML element.

This example sets the background color for a page to darkblue:
Example
```
<body style="background-color: darkblue;">
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
```

Example
```
<h1 style="background-color:LightBlue;">Problem Definition</h1>
<p style="background-color:Silver;">Problem Statement</p>
```

HTML Text Color - The color property defines the text color for an HTML element:

Example
```
<h1 style="color:blue;">White box testing</h1>
<p style="color:red;">Black box testing</p>
```

Example
```
<h1 style="color:Tomato;"> Problem Definition </h1>
<p style="color:DodgerBlue;"> Problem Statement </p>
```

Border Color - You can set the color of borders:

Example
```
<h1 style="border:2px solid Red;"> User</h1>
```
User

Color Values
In HTML, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values:
Same as color name "Tomato":
rgb(255, 99, 71), #ff6347, hsl(9, 100%, 64%)
Same as color name "Tomato", but 50% transparent:
rgba(255, 99, 71, 0.5), hsla(9, 100%, 64%, 0.5)

Example
```
<h1 style="background-color:rgb(255, 99, 71);">...</h1>
<h1 style="background-color:#ff6347;">...</h1>
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>
<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>
```

RGB Value
In HTML, a color can be specified as an RGB value, using this formula: ***rgb(red, green, blue)***
Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.
For example, rgb(255, 0, 0) is displayed as red, because red is set to its highest value (255) and the others are set to 0.
To display the color black, all color parameters must be set to 0, like this: rgb(0, 0, 0).
To display the color white, all color parameters must be set to 255, like this: rgb(255, 255, 255).

Example

rgb(255, 0, 0), rgb(0, 0, 255), rgb(60, 179, 113), rgb(238, 130, 238), rgb(255, 165, 0), rgb(106, 90, 205)

Shades of gray are often defined using equal values for all the 3 light sources:

Example

rgb(0, 0, 0), rgb(60, 60, 60), rgb(120, 120, 120), rgb(180, 180, 180), rgb(240, 240, 240), rgb(255, 255, 255)

HEX Value - In HTML, a color can be specified using a hexadecimal value in the form: ***#rrggbb***
Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff) and the others are set to the lowest value (00).

Example

#ff0000, #0000ff, #3cb371, #ee82ee, #ffa500, #6a5acd

Shades of gray are often defined using equal values for all the 3 light sources:

Example

#000000, #3c3c3c, #787878, #b4b4b4, #f0f0f0, #ffffff

HTML Styles - CSS

CSS = Styles and Colors
Manipulate Text, Colors, Boxes

Styling HTML with CSS
CSS stands for **C**ascading **S**tyle **S**heets.
CSS describes *how HTML elements are to be displayed on screen, paper, or in other media.*
CSS **saves a lot of work**. It can control the layout of multiple web pages all at once.
CSS can be added to HTML elements in 3 ways:

- Inline - by using the style attribute in HTML elements
- Internal - by using a <style> element in the <head> section
- External - by using an external CSS file

The most common way to add CSS, is to keep the styles in separate CSS files.

Inline CSS - An inline CSS is used to apply a unique style to a single HTML element.
An inline CSS uses the style attribute of an HTML element.

This example sets the text color of the <h1> element to blue:

Example

```
<h1 style="color:blue;">Checking which part is blue</h1>
```

Internal CSS - An internal CSS is used to define a style for a single HTML page.
An internal CSS is defined in the ***<head>*** section of an HTML page, within a <style> element:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
background-color: lightblue;
}
h1 {
color: blue;
}
p {
color: red;
}
</style>
</head>
<body>
<h1>Networking</h1>
<p>Packet Switching</p>
</body>
</html>
```

External CSS
An external style sheet is used to define the style for many HTML pages.
With an external style sheet, you can change the look of an entire web site, by changing one file!
To use an external style sheet, add a link to it in the ***<head>*** section of the HTML page:

Example

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
<h1>Networking</h1>
<p>Packet Switching</p>
</body>
</html>
```

An external style sheet can be ***written in any text editor***. The file must not contain any HTML code, and must be saved with a ***.css*** extension.
Here is how the "styles.css" looks:

```
body {
  background-color: lightblue;
}
h1 {
  color: blue;
}
p {
  color: red;
}
```

External References - External style sheets can be referenced with a full URL or with a path relative to the current web page.

This example uses a full URL to link to a style sheet:
Example

```
<link rel="stylesheet" href="http://tjc.moe.edu.sg /html/styles.css">
```

This example links to a style sheet located in the **html** folder on the current web site:
Example

```
<link rel="stylesheet" href="/html/styles.css">
```

This example links to a style sheet located in the **same** folder as the current page:
Example

```
<link rel="stylesheet" href="styles.css">
```


CSS Fonts - The CSS color property defines the text color to be used.
The CSS font-family property defines the font to be used.
The CSS font-size property defines the text size to be used.

font–family	font-size	font-style	font-weight

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  color: blue;
  font-family: verdana;
  font-size: 300%;
}
p {
  color: red;
  font-family: courier;
  font-size: 160%;
}
</style>
</head>
<body>
<h1>Networking</h1>
<p>Packet Switching</p>
</body>
</html>
```

CSS Border - The CSS border property defines a border around an HTML element:
Example

```
p {
  border: 1px solid lightblue;
}
```

border-bottom	border-left	border-right	border-top

CSS Padding - The CSS padding property defines a padding (space) between the text and the border:
Example

```
p {
  border: 1px solid lightblue;
  padding: 30px;
}
```

padding-bottom	padding-left	padding-right	padding-top

CSS Margin - The CSS margin property defines a margin (space) outside the border:
Example

```
p {
  border: 1px solid lightblue;
  margin: 50px;
}
```

margin-bottom	margin-left	margin-right	margin-top

The id Attribute - To define a specific style for one special element, add an id attribute to the element:

```
<p id="p01">Circuit Switching</p>
```

then define a style for the element with the specific id:

Example

```
#p01 {
  color: blue;
}
```

Note: The id of an element should be unique within a page, so the id selector is used to select one unique element!

HTML Links

Links are found in nearly all web pages. Links allow users to click their way from page to page.

HTML Links – Hyperlinks - HTML links are hyperlinks.
You can click on a link and jump to another document.
When you move the mouse over a link, the mouse arrow will turn into a little hand.
Note: A link does not have to be text. It can be an image or any other HTML element.

HTML Links - Syntax
In HTML, links are defined with the <a> tag: **link text**
Example

```
<a href=" http://tjc.moe.edu.sg /html/">Visit our School</a>
```

The **href** attribute specifies the destination address (**http://tjc.moe.edu.sg** /html/) of the link.
The link text is the visible part.
Clicking on the link text will send you to the specified address.
Note: Without a forward slash on subfolder addresses, you might generate two requests to the server. Many servers will automatically add a forward slash to the address, and then create a new request.

Local Links
The example above used an absolute URL (A full web address).
A local link (link to the same web site) is specified with a relative URL (without http://www....).
Example

```
<a href="Chapter_5.asp">Chapter 5 Asynchronous Transmission</a>
```

HTML Link Colors
By default, a link will appear like this (in all browsers):

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

You can change the default colors, by using styles:
Example

```
<style>
a:link {
  color: green;
  background-color: transparent;
  text-decoration: none;
}
a:visited {
  color: pink;
  background-color: transparent;
  text-decoration: none;      }
a:hover {
  color: red;
  background-color: transparent;
  text-decoration: underline;}
a:active {
  color: yellow;
  background-color: transparent;
  text-decoration: underline;}
</style>
```

HTML Links - The target attribute - The target attribute specifies where to open the linked document.
The target attribute can have one of the following values:

- _blank - Opens the linked document in a new window or tab
- _self - Opens the linked document in the same window/tab as it was clicked (this is default)
- _parent - Opens the linked document in the parent frame
- _top - Opens the linked document in the full body of the window
- framename - Opens the linked document in a named frame

This example will open the linked document in a new browser window/tab:

Example

```
<a href=" http://tjc.moe.edu.sg/" target="_blank">Visit Parent Support Group! </a>
```

Tip: If your webpage is locked in a frame, you can use target="_top" to break out of the frame:
Example

```
<a href=" http://tjc.moe.edu.sg/html/" target="_top">Subject Combinations</a>
```

HTML Links - Image as Link
It is common to use images as links:

Example

```
<a href="default.asp">
 <img src="smiley.gif" alt="Computing Networking Tutorial" style="width:42px; height:42px; border:0;">
</a>
```

Note: border:0; is added to prevent IE9 (and earlier) from displaying a border around the image (when the image is a link).

HTML Links - Create a Bookmark
HTML bookmarks are used to allow readers to jump to specific parts of a Web page.
Bookmarks can be useful if your webpage is very long.
To make a bookmark, you must first create the bookmark, and then add a link to it.
When the link is clicked, the page will scroll to the location with the bookmark.

Example
First, create a bookmark with the id attribute:

```
<h2 id="T4"> Computing Networking Tutorial 4</h2>
```

Then, add a link to the bookmark ("Jump to Tutorial 4"), from within the same page:

```
<a href="#T4">Jump to Tutorial 4</a>
```

Or, add a link to the bookmark ("Jump to Tutorial 4"), from another page:

Example

```
<a href="html_demo.html#T4">Jump to Tutorial 4</a>
```


External Paths
External pages can be referenced with a full URL or with a path relative to the current web page.
This example uses a full URL to link to a web page:
Example

```
<a href="http://www.tjc.com/html/default.asp">Subject Combination</a>
```

This example links to a page located in the html folder on the current web site:
Example

```
<a href="/html/default.asp"> Subject Combination </a>
```

This example links to a page located in the same folder as the current page:
Example

```
<a href="default.asp"> Subject Combination </a>
```

HTML File Paths

HTML File Paths - A file path describes the location of a file in a web site's folder structure.
File paths are used when linking to external files like:

- Web pages
- Images
- Style sheets
- JavaScripts

Absolute File Paths - An absolute file path is the full URL to an internet file:
Example

```
<img src=" http://tjc.moe.edu.sg/images/ctgp03_21.jpg" alt="CTGP 03/21">
```

The <img> tag and the src and alt attributes are explained in the chapter about HTML Images.

Relative File Paths - A relative file path points to a file relative to the current page.
In this example the file path points to a file in the images folder located at the root of the current web:
Example

```
<img src="/images/ctgp03_21.jpg" alt="CTGP 03/21">
```

In this example the file path points to a file in the images folder located in the current folder:
Example

```
<img src="images/ctgp03_21.jpg" alt="CTGP 03/21"">
```

In this example the file path points to a file in the images folder located in the folder one level above the current folder:
Example

```
<img src="../images/ ctgp03_21.jpg" alt="CTGP 03/21">
```

Path	Description
<img src="picture.jpg"> picture.jpg	is located in the same folder as the current page
<img src="images/picture.jpg"> picture.jpg	is located in the images folder in the current folder
<img src="/images/picture.jpg"> picture.jpg	is located in the images folder at the root of the current web
<img src="../picture.jpg"> picture.jpg	is located in the folder one level up from the current folder

Best Practice
It is a best practice to use relative file paths (if possible).
When using relative file paths, your web pages will not be bound to your current base URL.
All links will work on your own computer (localhost) as well as on your current public domain and your future public domains.

HTML Images

Images can improve the design and the appearance of a web page.

Example

```
<img src= "scones.jpg" alt="Scones">
```

Example

```
<img src="SOC-Logo.jpg" alt="SOC-Logo">
```

HTML Images Syntax - In HTML, images are defined with the <img> tag.
The <img> tag is empty, it contains attributes only, and does not have a closing tag.
The src attribute specifies the URL (web address) of the image: <img src="url">

The alt Attribute - The alt attribute provides an alternate text for an image, if the user for some reason cannot view it (because of slow connection, an error in the src attribute, or if the user uses a screen reader).

The value of the alt attribute should describe the image:
Example

```
<img src= "scones.jpg" alt="Scones">
```

If a browser cannot find an image, it will display the value of the alt attribute

Note: The alt attribute is required. A web page will not validate correctly without it.

Image Size - Width and Height
You can use the style attribute to specify the width and height of an image.
Example

```
<img src="scones.jpg" alt="Scones" style="width:128px;height:128px;">
```

Alternatively, you can use the width and height attributes:
Example

```
<img src="scones.jpg" alt="Scones" width="500" height="600">
```

The width and height attributes always defines the width and height of the image in pixels.
Note: Always specify the width and height of an image. If width and height are not specified, the page might flicker while the image loads.

Width and Height, or Style?
Both the width, height, and style attributes are valid in HTML5.
However, we suggest using the style attribute. It prevents styles sheets from changing the size of images:

Example

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
   width:100%;
}
</style>
</head>
<body>
<img src="scones.jpg" alt="Scones" style="width:128px;height:128px;">
<img src="scones.jpg" alt="Scones" width="128" height="128">
</body>
</html>
```

Images in Another Folder - If not specified, the browser expects to find the image in the same folder as the web page. However, it is common to store images in a sub-folder. You must then include the folder name in the src attribute:

Example

```
<img src="/img/scones.jpg" alt="Scones" style="width:128px;height:128px;">
```

Animated Images - HTML allows animated GIFs:
Example

```
<img src="programming.gif" alt="Computer Man" style="width:48px;height:48px;">
```

Image as a Link - To use an image as a link, put the <img> tag inside the <a> tag:
Example

```
<a href="default.asp">
 <img src="Star_network.gif" alt="Star network" style="width:42px;height:42px;">
</a>
```

Image Floating - Use the CSS float property to let the image float to the right or to the left of a text:
Example

```
<p><img src=" Star_network.gif" alt=" Star network"
style="float:right;width:42px;height:42px;">

 The image will float to the right of the text.</p>
<p><img src=" Star_network.gif" alt=" Star network"
style="float:left;width:42px;height:42px;">
 The image will float to the left of the text.</p>
```

Background Image - To add a background image on an HTML element, use the CSS property background-image:

Example
To add a background image on a web page, specify the background-image property on the BODY element:

```
<body style="background-image:url(03_21.jpg')">
<h2>Background Image of CTGP 03/21</h2>
</body>
```

Example
To add a background image on a paragraph, specify the background-image property on the P element:

```
<body>
<p style="background-image:url(03_21.jpg')">
...
</p>
</body>
```


HTML Tables

HTML Table Example

Subject	Teachers	Civic Tutor Group
Computing	Fong Kwok Kwong	03/21
Physics	Chueng Jun Jie	10/21

Defining an HTML Table - An HTML table is defined with the ***<table>*** tag.
Each table row is defined with the ***<tr>*** tag. A table header is defined with the ***<th>*** tag. By default, table headings are bold and centered. A table data/cell is defined with the ***<td>*** tag.

Example

```
<table style="width:100%">
 <tr>
  <th>Subject</th>
  <th>Teachers</th>
  <th>Civic Tutor Group</th>
 </tr>
 <tr>
  <td>Computing</td>
  <td> Fong Kwok Kwong </td>
  <td>03/21</td>
 </tr>
 <tr>
  <td>Physics </td>
  <td>Chueng Jun Jie </td>
  <td>10/21</td>
 </tr>
</table>
```

Note: The <td> elements are the data containers of the table.
They can contain all sorts of HTML elements; text, images, lists, other tables, etc.

HTML Table - Adding a Border
If you do not specify a border for the table, it will be displayed without borders.
A border is set using the CSS ***border*** property:

Example

```
table, th, td {
  border: 1px solid black;
}
```

Remember to define borders for both the table and the table cells.

HTML Table - Collapsed Borders
If you want the borders to collapse into one border, add the CSS border-collapse property:

Example

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
```

HTML Table - Adding Cell Padding - Cell padding specifies the space between the cell content and its borders.
If you do not specify a padding, the table cells will be displayed without padding.
To set the padding, use the CSS padding property:

Example

```
th, td {
  padding: 15px;
}
```

HTML Table - Left-align Headings - By default, table headings are bold and centered.
To left-align the table headings, use the CSS text-align property:

Example

```
th {
  text-align: left;
}
```

HTML Table - Adding Border Spacing - Border spacing specifies the space between the cells.
To set the border spacing for a table, use the CSS border-spacing property:

Example

```
table {
  border-spacing: 5px;
}
```

Note: If the table has collapsed borders, border-spacing has no effect.

HTML Table - Cells that Span Many Columns - To make a cell span more than one column, use the ***colspan*** attribute:

Example

```
<table style="width:100%">
 <tr>
  <th>Name</th>
  <th colspan="2">Telephone</th>
 </tr>
 <tr>
  <td>Paik Poh Leong</td>  <td>1800-9992418</td>  <td>9999012</td>
 </tr>
</table>
```


HTML Table - Cells that Span Many Rows - To make a cell span more than one row, use the ***rowspan*** attribute:

Example

```
<table style="width:100%">
 <tr>
  <th>Name:</th>
  <td> Paik Poh Leong </td>
 </tr>
 <tr>
  <th rowspan="2">Telephone:</th>
  <td>1800-9992418</td>
 </tr>
 <tr>  <td>9999012</td> </tr>
</table>
```

HTML Table - Adding a Caption - To add a caption to a table, use the <caption> tag:

Example

```
<table style="width:100%">
 <caption>Monthly Expenses</caption>
 <tr>
  <th>Month</th>
  <th>Expenses </th>
 </tr>
 <tr>
  <td>January</td>
  <td>$1500</td>
 </tr>
 <tr>
  <td>February</td>
  <td>$5000</td>
 </tr>
</table>
```

Note: The <caption> tag must be inserted immediately after the <table> tag.

A Special Style for One Table - To define a special style for a special table, add an id attribute to the table:

Example

```
<table id="s01">
 <tr>
  <th>Subject</th>
  <th>Name</th>
  <th>CTGP</th>
 </tr>
 <tr>
  <td>Computing</td>
   <td>Lai Wai Liang</td>
  <td>03/22</td>
 </tr>
</table>
```

Now you can define a special style for this table:

```
table#s01 {
  width: 100%;
   background-color: #f1f1c1;
}
And add more styles:
table#s01 tr:nth-child(even) {
   background-color: #eee;
}
table#s01 tr:nth-child(odd) {
   background-color: #fff;
}
table#s01 th {
  color: white;
   background-color: black;
}
```


HTML Inline and Block Elements

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is inline or block.

Inline Elements - An inline element does not start on a new line and only takes up as much width as necessary.

Inline elements in HTML:

<a>	<abbr>	<acronym>	**<b>**	<bdo>	<big>	** **	<button>	<cite>
<code>	<dfn>		**<i>**	**<img>**	**<input>**	<kbd>	<label>	<map>
<object>	<q>	<samp>	<script>	<select>	<small>	****		<sub>
<sup>	**<textarea>**	<time>	<tt>	<var>				

The Element - The element is often used as a container for some text.
The ****** element has no required attributes, but both style and class are common.
When used together with CSS, the element can be used to style parts of the text:

Example

```
<span>Hello</span>
<span> Computing classes </span>
```

Example

```
<h1>Study <span style="color:red">Computing</span> is important</h1>
```

This is an inline ****** element inside a paragraph.

Block-level Elements -
A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

Block level elements in HTML:

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>	** **	<dl>
<dt>	<fieldset>	<figcaption>	<figure>	<footer>	**<form>**	<h1>-<h6>	<header>
<hr>		<main>	<nav>	<noscript>		<output>	<p>
<pre>	<section>	**<table>**	<tfoot>		<video>		

The Element - The element is often used as a container for other HTML elements.
The *** *** element has no required attributes, but both style and class are common.
When used together with CSS, the element can be used to style blocks of content:

The *** *** element is a block-level element.

Example

```
<div>Hello</div>
<div>Computing classes</div>
```

Example

```
<div style="background-color: black; color: white; padding:20px;">
  <h2>Networking</h2>
  <p>Network topology</p>
</div>
```

HTML Grouping Tags

Tag	Description
	Defines a section in a document (block-level)
	Defines a section in a document (inline)

HTML The class Attribute

Using the class attribute - The class attribute specifies one or more class names for an HTML element.
The class name can be used by CSS and JavaScript to perform certain tasks for elements with the specified class name.

Example
Using CSS to style all elements with the class name "ctgp":

```
<style>
•ctgp {
  background-color: tomato;
  color: white;
  padding: 10px;
}
</style>

<h2 class="ctgp">03/21</h2>
<p>Mathematics, Physics, Computing. </p>
<h2 class="ctgp">04/21</h2>
<p>Mathematics, Chemistry, Computing. </p>
<h2 class="ctgp">05/21</h2>
<p> Mathematics, Physics, Computing. </p>
```

The class attribute can be used on any HTML element.

Multiple Classes
HTML elements can have more than one class name, each class name must be separated by a space.

Example
Style elements with the class name "***ctgp***", also style elements with the class name "**stream** ":

```
<h2 class="ctgp stream">Science</h2>
<h2 class="ctgp">03/21</h2>
<h2 class="ctgp">04/21</h2>
```

In the example above, the first h2 element belongs to both the "ctgp" class and the "stream" class.

Same Class, Different Tag
Different tags, like <h2> and <p>, can have the same class name and thereby share the same style:
Example

```
<h2 class="ctgp">04/21</h2>
<p class="ctgp">04/21 is a chemistry class</p>
```


HTML Forms

HTML Form Example

Name:
Address:
Submit

The <form> Element - The HTML <form> element defines a form that is used to collect user input:

```
<form>
.
form elements
.
</form>
```

An HTML form contains form elements.
Form elements are different types of input elements, like text fields, checkboxes, radio buttons, submit buttons, and more.

The <input> Element
The <input> element is the most important form element.
The <input> element can be displayed in several ways, depending on the type attribute.
Here are some examples:

Type	Description
<input type="text">	Defines a one-line text input field
<input type="radio">	Defines a radio button (for selecting one of many choices)
<input type="submit">	Defines a submit button (for submitting the form)

Text Input - <input type=***"text"***> defines a one-line input field for text input:

Example

```
<form>
Name:<br>
 <input type="text" name="name"><br>
Address:<br>
 <input type="text" name="address">
</form>
```

This is how it will look like in a browser:

Name:
Address:

Note: The form itself is not visible. Also note that the default width of a text field is **20 characters**.

The Name Attribute - Each input field must have a name attribute to be submitted.

If the name attribute is omitted, the data of that input field will not be sent at all.
This example will only submit the "address" input field:

Example

```
<form action="/action_page.asp">
 Name:<br>
 <input type="text" value="Paik Poh Leong"><br>
 Address:<br>
 <input type="text" name="address" value="144 Bukit Timak"><br><br>
 <input type="submit" value="Submit">
</form>
```

Radio Button Input - <input type="***radio***"> defines a radio button.
Radio buttons let a user select ONE of a limited number of choices:

Example

```
<form>
 <input type="radio" name="gender" value="male" checked> Male<br>
  <input type="radio" name="gender" value="female"> Female<br>
</form>
```

This is how the HTML code above will be displayed in a browser:

⊙Male
○Female

The Submit Button - <input type="submit"> defines a button for submitting the form data to a form-handler.
The form-handler is typically a ***server page*** with a script for processing input data.
The form-handler is specified in the form's action attribute:

Example

```
<form action="/action_page.asp">
 Name:<br>
 <input type="text" name="name" value="Paik Poh Leong"><br>
 Address:<br> <input type="text" name="address" value="144 Bukit Timak"><br><br>
 <input type="submit" value="Submit">
</form>
```

The Action Attribute - The action attribute defines the action to be performed when the form is submitted.
Normally, the form data is sent to a web page on the server when the user clicks on the submit button.
In the example above, the form data is ***sent to a page on the server*** called "/action_page.asp".
This page contains a ***server-side script*** that handles the form data:

```
<form action="/action_page.asp">
```

If the action attribute is omitted, the action is set to the current page.

The Target Attribute
The target attribute specifies if the submitted result will open in a new browser tab, a frame, or in the current window.
The default value is "_self" which means the form will be submitted in the current window.
To make the form result open in a new browser tab, use the value "_blank":

Example

```
<form action="/action_page.asp" target="_blank">
```

Other legal values are "_parent", "_top", or a name representing the name of an iframe.

The Method Attribute
The method attribute specifies the HTTP method (GET or POST) to be used when submitting the form data:

Example

```
<form action="/action_page.asp" method="get">
```

or:

Example

```
<form action="/action_page.asp" method="post">
```

When to Use GET?
The default method when submitting form data is GET.
However, when GET is used, the submitted form data will be visible in the page address field:
/action_page.asp?name=Paik%20Poh%20Leong&address=144%20Bukit%20Timak
Notes on GET:

- Appends form-data into the URL in name/value pairs
- The length of a URL is limited (about 3000 characters)
- Never use GET to send sensitive data! (will be visible in the URL)
- Useful for form submissions where a user wants to bookmark the result
- GET is better for non-secure data, like query strings in Google

When to Use POST?
Always use POST if the form data contains sensitive or personal information. The POST method does not display the submitted form data in the page address field.
Notes on POST:

- POST has no size limitations, and can be used to send large amounts of data.
- Form submissions with POST cannot be bookmarked

HTTP Methods: GET vs. POST

The two most used HTTP methods are: GET and POST.

What is HTTP?
The Hypertext Transfer **Protocol** (HTTP) is designed to **enable communications** between **clients** and **servers**.

HTTP works as a **request-response protocol** between a client and server.
A web browser may be the client, and an application on a computer that hosts a web site may be the server.

Example: A client (browser) submits an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

Two HTTP Request Methods: GET and POST
Two commonly used methods for a request-response between a client and server are: GET and POST.

- GET - Requests data from a specified resource
- POST - Submits data to be processed to a specified resource

The GET Method

Note that the query string (***name/value pairs***) is sent in the URL of a GET request:
/test/demo_form.asp?name1=value1&name2=value2

Some other notes on GET requests:

- GET requests can be cached
- GET requests remain in the browser history
- GET requests can be bookmarked
- GET requests should never be used when dealing with sensitive data
- GET requests have length restrictions
- GET requests should be used only to retrieve data

The POST Method

Note that the query string (name/value pairs) is sent in the HTTP message body of a POST request:

```
POST /test/demo_form.asp HTTP/1.1
Host: tjc.moe.edu.sg
name1=value1&name2=value2
```

Some other notes on POST requests:

- POST requests are never cached
- POST requests do not remain in the browser history
- POST requests cannot be bookmarked
- POST requests have no restrictions on data length

Compare GET vs. POST

The following table compares the two HTTP methods: GET and POST.

	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL

Other HTTP Request Methods

The following table lists some other HTTP request methods:

Method	Description
HEAD	Same as GET but returns only HTTP headers and no document body
PUT	Uploads a representation of the specified URI
DELETE	Deletes the specified resource
OPTIONS	Returns the HTTP methods that the server supports
CONNECT	Converts the request connection to a transparent TCP/IP tunnel

HTML URL Encoding Reference

URL encoding converts characters into a format that can be transmitted over the Internet.

URL - Uniform Resource Locator
Web browsers request pages from web servers by using a URL.
The URL is the address of a web page, like: http://www.tjc.moe.edu.sg.

URL Encoding (% Percent Encoding)
URLs can only be sent over the Internet using the ASCII character-set.
Since URLs often contain characters outside the ASCII set, the URL has to be converted into a valid ASCII format.
URL encoding replaces unsafe ASCII characters with a "%" followed by two hexadecimal digits.
URLs cannot contain spaces. URL encoding normally replaces a space with a plus (+) sign or with %20.

HTTP Status Messages

When a browser requests a service from a web server, an error might occur.
This is a list of HTTP status messages that might be returned:

1xx: Information

Message:	Description:
100 Continue	The server has received the request headers, and the client should proceed to send the request body
101 Switching Protocols	The requester has asked the server to switch protocols
103 Checkpoint	Used in the resumable requests proposal to resume aborted PUT or POST requests

2xx: Successful

Message:	Description:
200 OK	The request is OK (this is the standard response for successful HTTP requests)
201 Created	The request has been fulfilled, and a new resource is created
202 Accepted	The request has been accepted for processing, but the processing has not been completed
203 Non-Authoritative Information	The request has been successfully processed, but is returning information that may be from another source
204 No Content	The request has been successfully processed, but is not returning any content
205 Reset Content	The request has been successfully processed, but is not returning any content, and requires that the requester reset the document view
206 Partial Content	The server is delivering only part of the resource due to a range header sent by the client

3xx: Redirection

Message:	Description:
300 Multiple Choices	A link list. The user can select a link and go to that location. Maximum five addresses
301 Moved Permanently	The requested page has moved to a new URL
302 Found	The requested page has moved temporarily to a new URL
303 See Other	The requested page can be found under a different URL
304 Not Modified	Indicates the requested page has not been modified since last requested
306 Switch Proxy	*No longer used*
307 Temporary Redirect	The requested page has moved temporarily to a new URL
308 Resume Incomplete	Used in the resumable requests proposal to resume aborted PUT or POST requests

4xx: Client Error

Message:	Description:
400 Bad Request	The request cannot be fulfilled due to bad syntax
401 Unauthorized	The request was a legal request, but the server is refusing to respond to it. For use when authentication is possible but has failed or not yet been provided
402 Payment Required	*Reserved for future use*
403 Forbidden	The request was a legal request, but the server is refusing to respond to it
404 Not Found	The requested page could not be found but may be available again in the future
405 Method Not Allowed	A request was made of a page using a request method not supported by that page
406 Not Acceptable	The server can only generate a response that is not accepted by the client
407 Proxy Authentication Required	The client must first authenticate itself with the proxy
408 Request Timeout	The server timed out waiting for the request
409 Conflict	The request could not be completed because of a conflict in the request
410 Gone	The requested page is no longer available
411 Length Required	The "Content-Length" is not defined. The server will not accept the request without it
412 Precondition Failed	The precondition given in the request evaluated to false by the server
413 Request Entity Too Large	The server will not accept the request, because the request entity is too large
414 Request-URI Too Long	The server will not accept the request, because the URL is too long. Occurs when you convert a POST request to a GET request with a long query information
415 Unsupported Media Type	The server will not accept the request, because the media type is not supported
416 Requested Range Not Satisfiable	The client has asked for a portion of the file, but the server cannot supply that portion
417 Expectation Failed	The server cannot meet the requirements of the Expect request-header field

5xx: Server Error

Message:	Description:
500 Internal Server Error	A generic error message, given when no more specific message is suitable
501 Not Implemented	The server either does not recognize the request method, or it lacks the ability to fulfill the request
502 Bad Gateway	The server was acting as a gateway or proxy and received an invalid response from the upstream server
503 Service Unavailable	The server is currently unavailable (overloaded or down)
504 Gateway Timeout	The server was acting as a gateway or proxy and did not receive a timely response from the upstream server
505 HTTP Version Not Supported	The server does not support the HTTP protocol version used in the request
511 Network Authentication Required	The client needs to authenticate to gain network access

HTML Form Elements

The <input> Element
The most important form element is the <input> element.
The <***input***> element can be displayed in several ways, depending on the type attribute.

Example
```
<input name="address" type="text">
```

If the type attribute is omitted, the input field gets the default type: "text".

Input Type Text
<input type="***text***"> defines a one-line text input field:

Example
```
<form>
  Name:<br>
  <input type="text" name="Paik Poh Leong"><br>
Address:<br>
  <input type="text" address="144 Bukit Timak ">
</form>
```

The maxlength Attribute - The maxlength attribute specifies the maximum allowed length for the input field:

Example
```
<form action="">
Name:<br>
<input type="text" name="name" value=" Paik Poh Leong " maxlength="10">
</form>
```
With a maxlength attribute, the input field will not accept more than the allowed number of characters.
Note: Input restrictions are not foolproof. To safely restrict input, it must be checked by the receiver (the server) as well!

The value Attribute - The value attribute specifies the initial value for an input field:

Example
```
<form action="">
 Name:<br><input type="text" name="name" value="Paik Poh Leong">
</form>
```

The size Attribute - The size attribute specifies the size (in characters) for the input field:

Example
```
<form action="">
Name:<br>
<input type="text" name="name" value=" Paik Poh Leong " size="40">
</form>
```

The readonly Attribute - The readonly attribute specifies that the input field is read only (cannot be changed):

Example

```
<form action="">
 Name:<br>
<input type="text" name="name" value=" Paik Poh Leong " readonly>
</form>
```

The disabled Attribute - The disabled attribute specifies that the input field is disabled.
A disabled input field is unusable and un-clickable, and its value will not be sent when submitting the form:

Example

```
<form action="">
Name:<br>
<input type="text" name="name" value=" Paik Poh Leong " disabled>
</form>
```

Input Type Password
<input type="***password***"> defines a password field:

Example

```
<form>
  User name:<br>
 <input type="text" name="username"><br>
  User password:<br>
 <input type="password" name="psw">
</form>
```

The characters in a password field are masked (shown as asterisks or circles).

The <textarea> Element
The <***textarea***> element defines a multi-line input field (a text area):

Example

```
<textarea name="message" rows="10" cols="30">Congratulation you are now in JC2!
</textarea>
```

The rows attribute specifies the visible number of lines in a text area.
The cols attribute specifies the visible width of a text area.

You can also define the size of the text area by using CSS:

Example

```
<textarea name="message" style="width:200px; height:600px">
Congratulation you are now in JC2!
</textarea>
```


Input Type Radio
<input type="***radio***"> defines a radio button.
Radio buttons let a user select ONLY ONE of a limited number of choices:

Example

```
<form>
 <input type="radio" name="gender" value="male" checked> Male<br>
 <input type="radio" name="gender" value="female"> Female<br>
 <input type="radio" name="gender" value="other"> Other
</form>
```

Input Type Checkbox
<input type="***checkbox***"> defines a checkbox.
Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

```
<form>
 <input type="checkbox" name="H21" value="Physics"> Physics <br>
 <input type="checkbox" name="H22" value="Chemistry"> Chemistry
</form>
```

The <select> Element
The <***select***> element defines a drop-down list:

Example

```
<select name="CivicTutorGroup">
 <option value="03/21">03/21</option>
 <option value="04/21">04/21</option>
 <option value="05/21">05/21</option>
</select>
```

The <*option*> elements defines an option that can be selected.
By default, the first item in the drop-down list is selected.

To define a pre-selected option, add the selected attribute to the option:

Example

```
<option value="03/21" selected>03/21</option>
```

Visible Values:
Use the ***size*** attribute to specify the number of visible values:

Example

```
<select name=" CivicTutorGroup " size="2">
 <option value="03/21">03/21</option>
 <option value="04/21">04/21</option>
 <option value="05/21">05/21</option>
</select>
```

Allow Multiple Selections:
Use the ***multiple*** attribute to allow the user to select more than one value:

Example

```
<select name="cars" size="3" multiple>
 <option value="03/21">03/21</option>
 <option value="04/21">04/21</option>
 <option value="05/21">05/21</option>
</select>
```

Input Type Submit
<input type="***submit***"> defines a button for submitting form data to a form-handler.
The form-handler is typically a server page with a script for processing input data.
The form-handler is specified in the form's action attribute:

Example

```
<form action="/action_page.php">
  Name:<br>
 <input type="text" name="Paik Poh Leong"><br>
Address:<br>
 <input type="text" address="144 Bukit Timak "><br>
 <input type="submit" value="Submit">
</form>
```

If you omit the submit button's value attribute, the button will get a default text:

Example

```
<form action="/action_page.php">
  Name:<br>
 <input type="text" name="Paik Poh Leong"><br>
Address:<br>
 <input type="text" address="144 Bukit Timak "><br>
 <input type="submit">
</form>
```

The height and width Attributes
The ***height*** and ***width*** attributes specify the height and width of an <input type="image"> element.
Always specify the size of images. If the browser does not know the size, the page will flicker while images load.

Example
Define an image as the **submit button**, with height and width attributes:

```
<input type="image" src="img_submit.gif" alt="Submit" width="48" height="48">
```

The multiple Attribute - The multiple attribute specifies that the user is allowed to enter more than one value in the <input> element.
The ***multiple*** attribute works with the following input types: email, and file.

Example
A file upload field that accepts multiple values:
Select images: <input type="file" name="img" ***multiple***>

Input Type Reset
<input type="***reset***"> defines a reset button that will reset all form values to their default values:

Example

```
<form action="/action_page.php">
  Name:<br>
 <input type="text" name="Paik Poh Leong"><br>
Address:<br>
 <input type="text" address="144 Bukit Timak "><br>
 <input type="submit" value="Submit">
 <input type="reset">
</form>
```

If you change the input values and then click the "Reset" button, the form-data will be reset to the default values.

The <button> Element - The <button> element defines a clickable button:
Example

```
<button type="button" onclick="alert('Hello Computing class!')">Click Me!</button>
```