

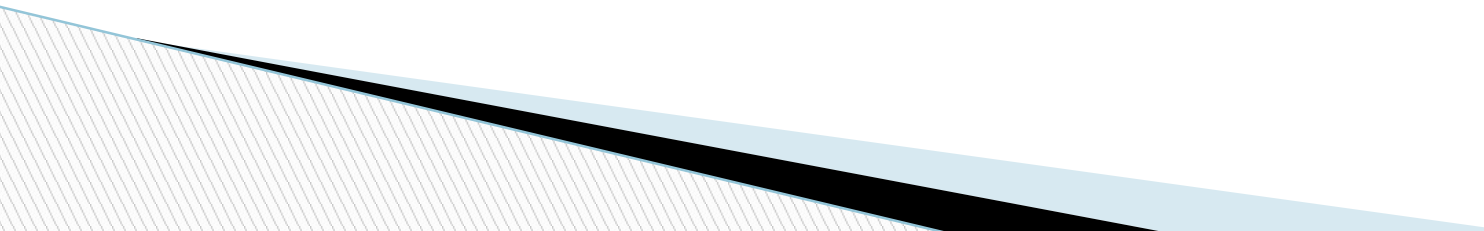
# IWT

## Unit 1

Jalpa Poriya

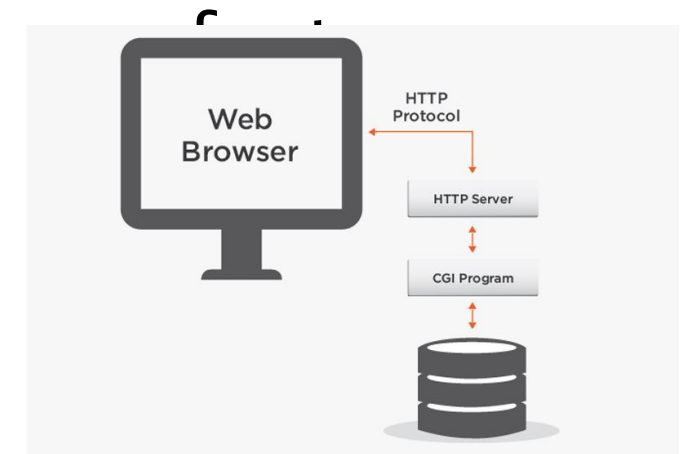


# Web Server

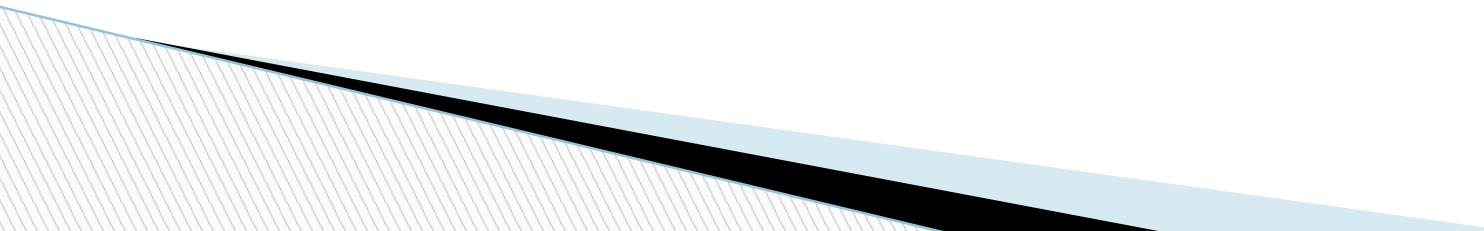
- The primary function of a web server is to store, process and deliver web pages to clients. The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP).
  - Pages delivered are most frequently HTML documents, which may include **images**, **style sheets** and **scripts** in addition to the text content.
- 

# Web Client/Browser

- The **client**, or user, side of the **Web**.
- It typically refers to the **Web browser** in the user's machine. It may also refer to plug-ins and helper applications that enhance the **browser** to support special services from the site.
- The term may imply the entire user machine handheld device that provides **Web** access.

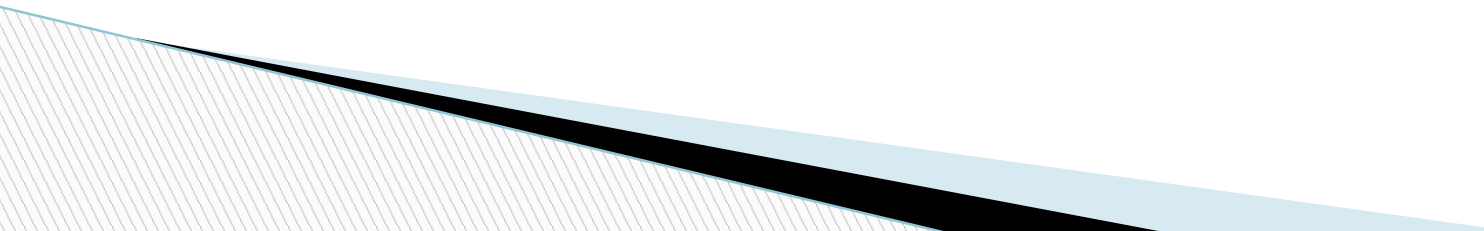


# Introduction of HTML

- HTML stands for **Hypertext Markup Language**, and it is the most widely used language to write Web Pages.
  - As its name suggests, HTML is a **Markup Language** which means you use HTML to simply "mark-up" a text document with **tags** that tell a Web browser how to structure it to display.
- 

# Basic HTML 5 Document

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="UTF-8">  
    <title>Title of the document</title>  
  </head>  
  <body>  
    Content of the document.....  
  </body>  
</html>
```

- **<meta>**
  - Metadata is data (information) about data.
  - The <meta> tag provides metadata about the HTML document. Metadata will not be displayed on the page, but will be machine parsable.
  - The metadata can be used by browsers (how to display content or reload page), search engines (keywords), or other web services.
- 

## ▣ **Setting The Viewport in HTML 5 with <meta>**

▣ The viewport is the user's visible area of a web page. It varies with the device, and will be smaller on a mobile phone than on a computer screen.

▣ Example :

```
<meta name="viewport" content="width=device-width,  
initial-scale=1.0">
```



**Without the viewport meta tag**



**With the viewport meta tag**



# Examples

- **Example 1 - Define keywords for search engines:**

- `<meta name="keywords" content="HTML, CSS, XML, XHTML, JavaScript">`

- **Example 2 - Define a description of your web page:**

- `<meta name="description" content="Free Web tutorials on HTML and CSS">`

- **Example 3 - Define the author of a page:**

- `<meta name="author" content="John Doe">`

- **Example 4 - Refresh document every 30 seconds:**

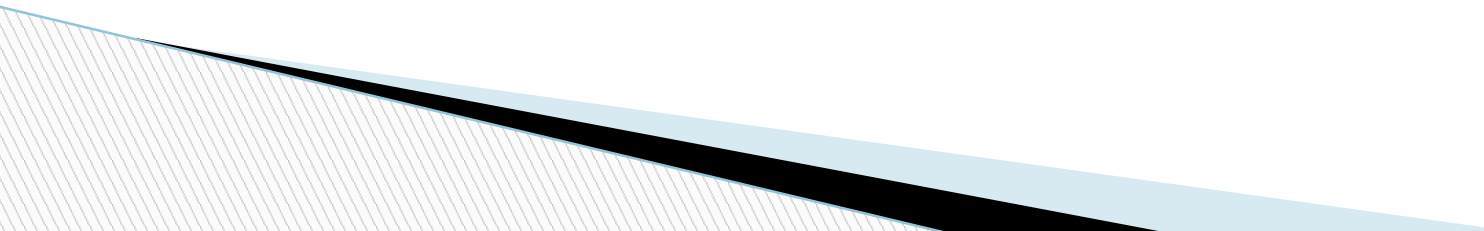
- `<meta http-equiv="refresh" content="30">`

- **Example 5 - Setting the viewport to make your website look good on all devices:**

- `<meta name="viewport" content="width=device-width, initial-scale=1.0">`



# New HTML5 Elements

- New semantic elements like **<header>**, **<footer>**, **<article>**, and **<section>**.
  - New attributes of **form elements** like **number**, **date**, **time**, **calendar**, and **range**.
  - New graphic elements: **<svg>** and **<canvas>**.
  - New multimedia elements: **<audio>** and **<video>**.
- 

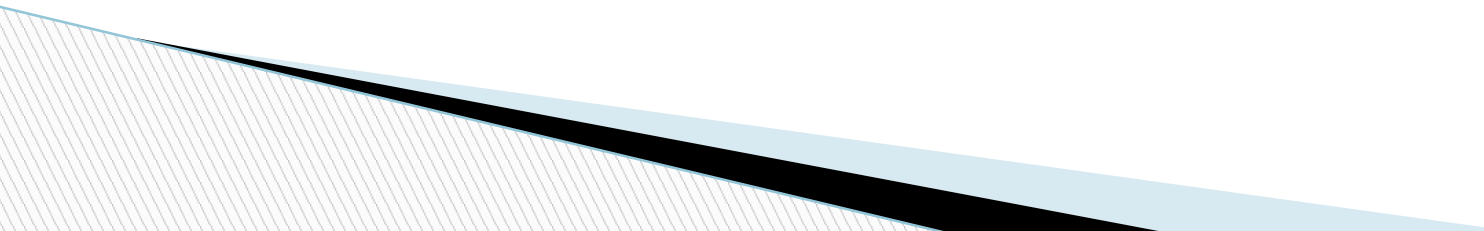
# Basic HTML Tags

1. HTML is a markup language and makes use of various tags to format the content.
2. These tags are enclosed within angle braces. Except few tags, most of the tags have their corresponding closing tags.

**Syntax:**        <Tag Name></TagName>

**Example:**        <H1>Hello World</H1>  
                  <B>This will print in bold</b>  
                  <B> Indus <U>University </U> </B>

# Basic HTML Tags

- Font Tag : `<FONT>`
  - Heading Tags : `<H1>` to `<H6>`
  - Paragraph Tag : `<P>`
  - Line Break `<BR/>`
- 

# Font tag

- The <font> tag specifies the font face, font size, and color of text.
  - Attribute: **color**
  - Value: color\_name , hex\_number
  - Example: “red”, “#800000”
  
  - Attribute: **Size**
  - Value: in number 1 to 7
  
  - Attribute: **face**
  - Value: font type
  - Example: “arial”, “Times new roman”, “arial “

▣ **Example:**

```
<FONT color = "blue" size = "7" face = "Lucida Sans Unicode">  
  Hello World  
</FONT>
```

# Heading Tags <h1>

- Any document starts with a heading. You can use different sizes for your headings. HTML also has six levels of headings, which use the elements **<h1>**, **<h2>**, **<h3>**, **<h4>**, **<h5>**, and **<h6>**. While displaying any heading, browser adds one line before and one line after that heading.
- For Example :
- `<h1>Hello Indus</h1>`

# Paragraph Tag `<p>`

- The `<p>` tag offers a way to structure your text into different paragraphs. Each paragraph of text should go in between an opening `<p>` and a closing `</p>` tag



# Line Break Tag `<br />`

- Whenever you use the `<br />` element, anything following it starts from the next line.
- If you miss the forward slash character and just use `<br>` it is not valid in XHTML.

# Centering Content `<center>`

- You can use `<center>` tag to put any content in the center of the page or any table cell.

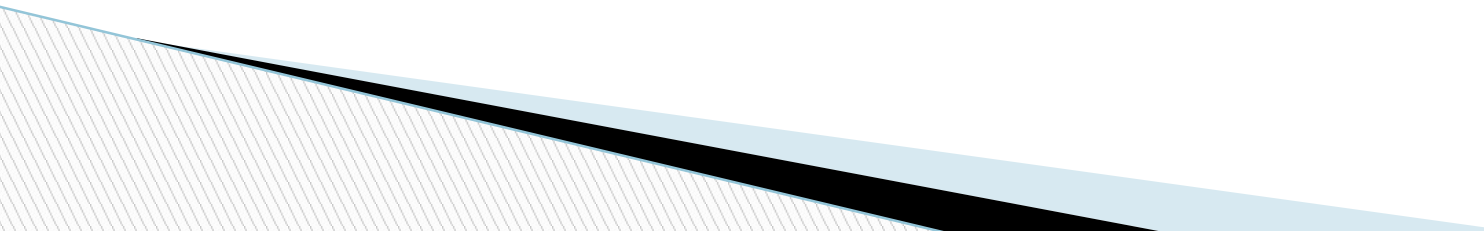
# Horizontal Lines <hr>

- Horizontal lines are used to visually break-up sections of a document. The **<hr>** tag creates a line from the current position in the document to the right margin and breaks the line accordingly.

# Preserve Formatting `<pre>`

- Sometimes, you want your text to follow the exact format of how it is written in the HTML document. In these cases, you can use the preformatted tag `<pre>`.

# List in Html

- Unordered List (list with symbols)
  - Ordered List (list with sequence number)
  - Definition List
- 

# Unordered List

□ **<ul>**

**<li>...</li>**

**<li>...</li>**

**<li>...</li>**

**</ul>**



# Attributes of UI tag

- Attribute:

- type – disc, circle, square, none

# ordered List

□ **<ol>**

**<li>...</li>**

**<li>...</li>**

**<li>...</li>**

**</ol>**

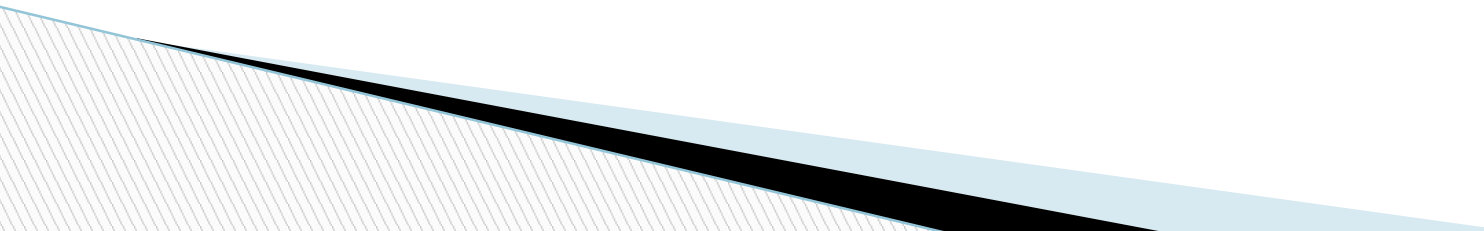


- Attribute of <OL>:
- Type - 1, A, a, l, i
- Start - in number
- Attribute of <LI>:
- Type - 1, A, a, l, i, disc, square, circle
- Value - in number

# definition list

- The definition list created using `<dl>` tag.
- The `<dl>` tag is used in conjunction with `<dt>` — defines the item in the list, and `<dd>` describes the item in the list:
- `<dl>`
- `<dt>What is html?<dt>`
- `<dd>Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. </dd>`
- `</dl>`

# Table Tag

- 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.
- 

# Table Caption

<caption>

Heading

Heading

<th>

<th>

Semester 1

Row 1

**Subject Code**

**Subject Name**

Row 2

1

IW

Row 3

2

CF

<tr>

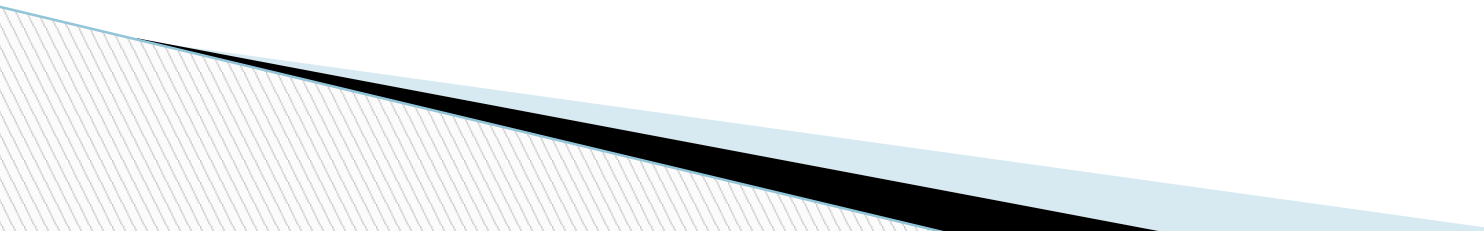
Data <td>

Data <td>

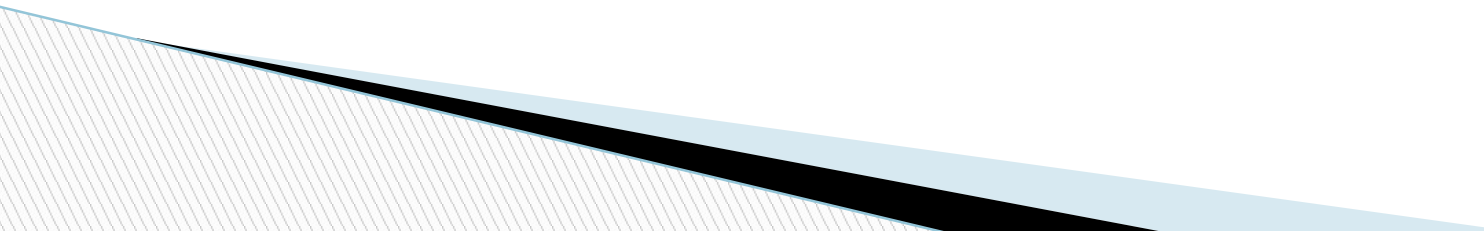
Subject Code	Subject Name
1	IW
2	CF

```
□ <table>
□   <caption>Semester 1 </caption>
□   <tr>
□     <th>Subject Code</th>
□     <th>Subject Name</th>
□   </tr>
□   <tr>
□     <td>1 </td>
□     <td>IW</td>
□   </tr>
□   <tr>
□     <td>2 </td>
□     <td>CF</td>
□   </tr>
□ </table>
```

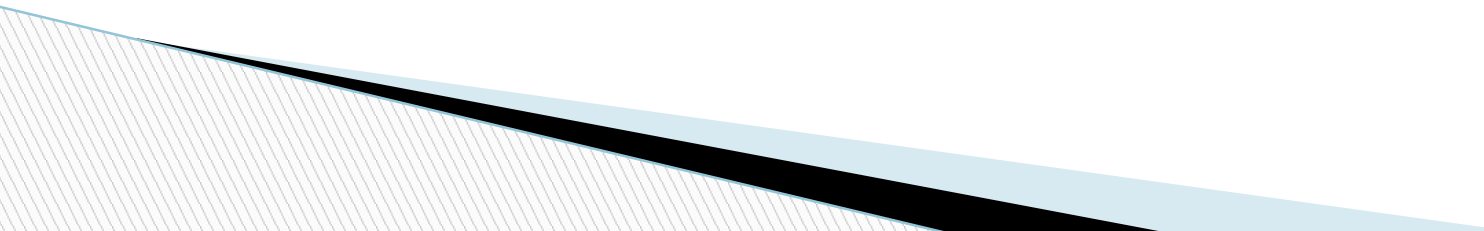
# Attributes of <Table>

- Border – in number
  - Align – left, right, center
  - Bgcolor – colorname, hex-code
  - Cellspacing – in px  
*Space between the order of a cell and the contents of the cell*
  - Cellpadding – in px  
*Space between individual cells in a table.*
  - Width – in px, in %
  - Height – in px, in %
- 

# Attributes of Table Row <TR>

- Align – left, right, center
  - Valign – baseline, bottom, middle, top
  - Bgcolor – colorname, hex-code
- 

# Attributes of Table Data `<td>`

- Align – left, right, center
  - Valign – baseline, bottom, middle, top
  - Bgcolor – colorname, hex-code
  - Colspan – in number
  - Rowspan – in number
- 



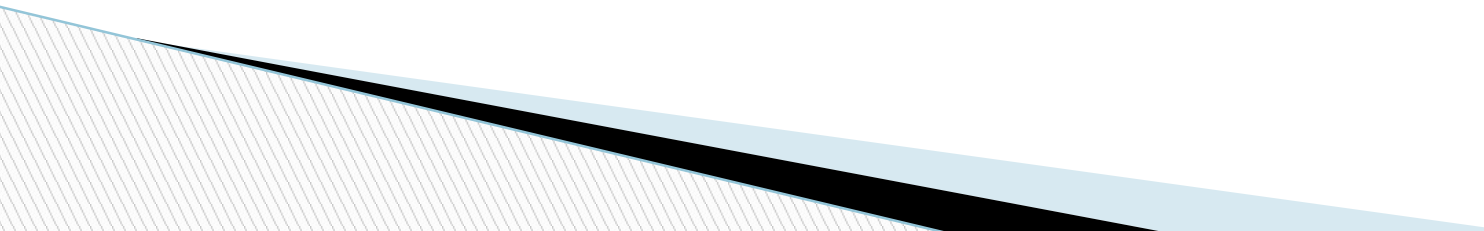
2 columns together  
"colspan"

a	
c	d
e	f
	g
h	

2 rows together  
"rowspan"

2 columns together

# HTML `<blockquote>` Tag

- The `<blockquote>` tag specifies a section that is quoted from another source.
  - Attribute
  - `cite`      URL      Specifies the source of the quotation
- 

## □ Example

```
<blockquote cite="www.wikipedia.com">
```

Wikipedia is a free online encyclopedia, created and edited by volunteers around the world and hosted by the Wikimedia Foundation.

```
</blockquote>
```



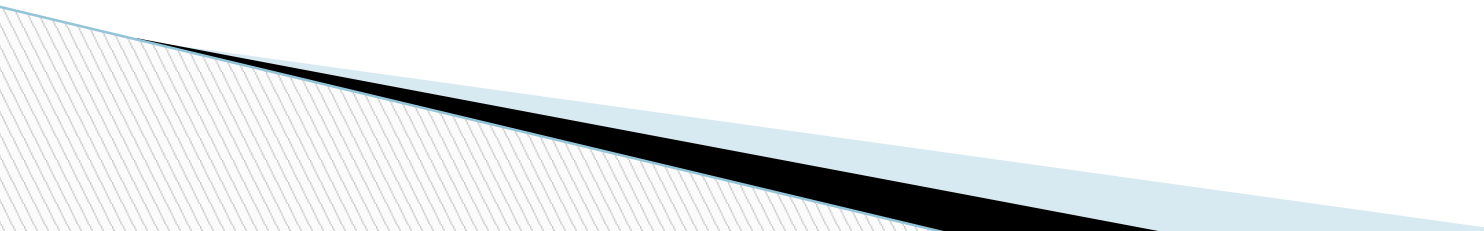
# HTML Entities/ Special character

- **Reserved characters** in HTML must be replaced with character entities.
- Characters that are not present on your keyboard can also be replaced by entities.
- For example, If you use the less than (<) or greater than (>) signs in your text, the browser might mix them with tags.
- A character entity looks like this:  
*&entity\_name;*

<u>Symbol</u>	<u>Description</u>	<u>Entity Name</u>
	non-breaking space	&nbsp;
<	less than	&lt;
>	greater than	&gt;
&	ampersand	&amp;
"	double quotation mark	&quot;
'	single quotation mark	&apos;

<u>Symbol</u>	<u>Description</u>	<u>Entity Name</u>
¢	cent	&cent;
£	pound	&pound;
¥	yen	&yen;
€	euro	&euro;
©	copyright	&copy;
®	registered trademark	&reg;

# Grouping of text

- `<DIV>` Tag
  - `<SPAN>` Tag
- 

# <SPAN> - Tag

- The HTML <span> tag is used for grouping and applying styles to inline elements.
- Example 1

```
<p>
```

This is a paragraph

```
<span style = "color:blue;border-style:solid">
```

This is a span

```
</span>
```

```
</p>
```





□ Example 2

```
<span style="background-color:gold">  
  <a href="ex1.html">  
    Click  
  </a>  
</span>
```

# HTML <center> Tag [align tag]

- The <center> tag is used to center-align text.

- **Example**

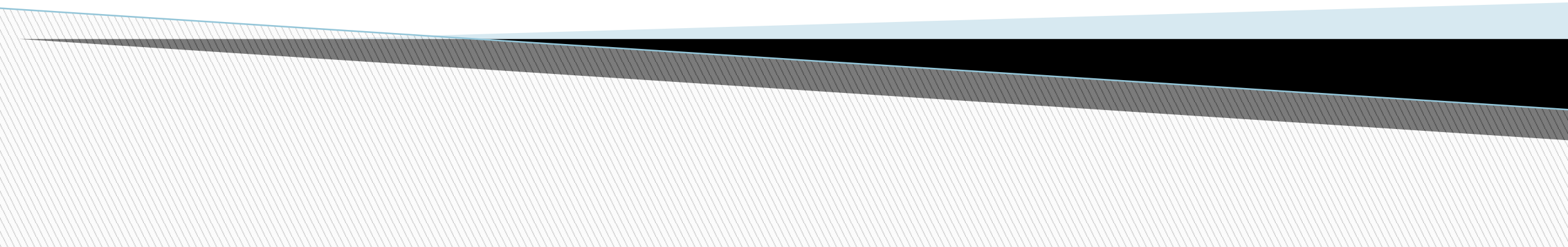
```
<center>
```

This text will be center-aligned.

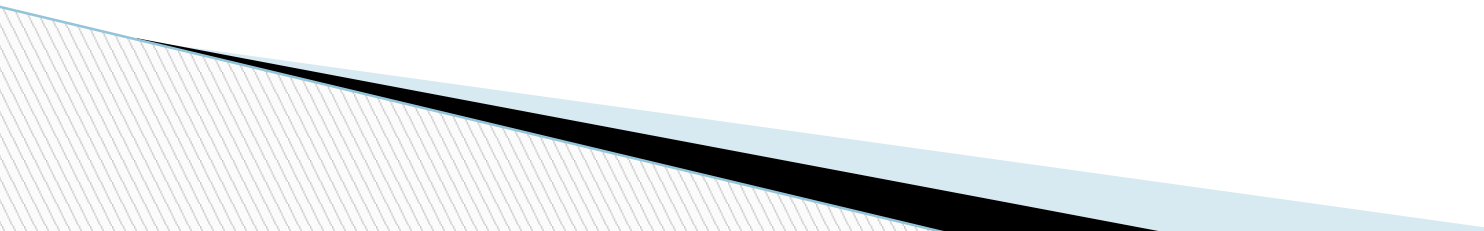
```
</center>
```



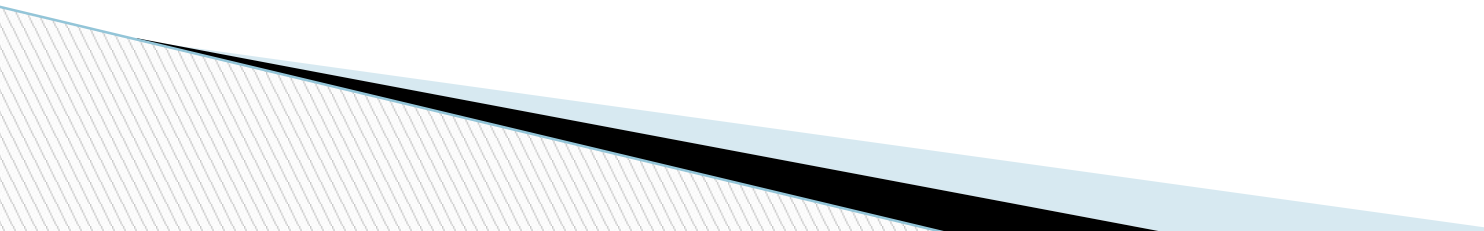
# Link in HTML



# Anchor tag for hyperlink

- The `<a>` tag defines a hyperlink, which is used to link from one page to another.
  - By default, links will appear as follows in all browsers:
  - An unvisited link is underlined and blue
  - A visited link is underlined and purple
  - An active link is underlined and red
- 

# <a> TAG ATTRIBUTE

- HREF – HREF stands for Hyper Reference.
  - The href attribute specifies the URL (web resource) to be loaded when the link is clicked.
  - The href or the name attribute must be present in the <a> tag.
- 
- **What is an URL**
  - A Uniform Resource Locator (URL) is used to address a document (or other data) on the World Wide Web.
- 

# Link another page in the same directory

- To link to another page in the **same directory**, simply include the **filename** in the href=" " attribute of the opening anchor tag.
- Example
- `<a href="tableDemo.html"> Table Demo 1 </a>`

- **Absolute Path**

- it simply refers to the ***exact path*** to the file's location, including all directories and subdirectories you have to go through to get there.

- Example:

- `<a href="d:\foldername\filename.html">Hyper Link</a>`

- **Relative Path:**

- A relative URL uses a kind of ***shorthand*** to tell the browser to go backward one or more directories.

- Example:

- `<a href="foldername\filename.html">hyper link</a>`

- `<a href="..\foldername\filename.html">hyper link</a>`

- `<a href="..\..\foldername\filename.html">hyper link</a>`



# Link on same document with name anchors

Attribute – Name

This attribute helps visitors to navigate a single long page.

Example

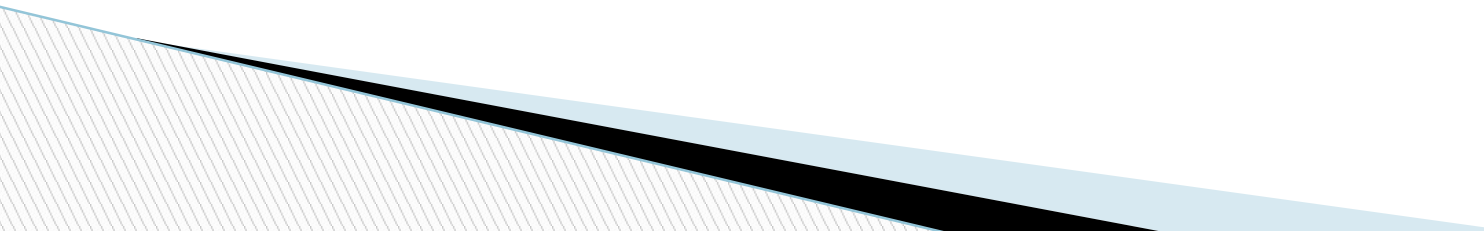
Place an anchor at the top of the page by typing

```
<a name="first"> </a>.
```

At the bottom of the page make a link to your anchor by typing the following:

```
<a href="#first">Top of Page</a>
```

# Attribute of <A>

- Rel - next, prev, index, chapter, alternate, stylesheet, contents, help, bookmark, appendix
  - Rev - next, prev, index, chapter, alternate, stylesheet, contents, help, bookmark, appendix
  - Target - \_blank, \_self, \_parent, \_top, framename
- 

# Forms in HTML

---

# Introduction

---

- HTML Forms are required, when you want to collect some data from the site visitor.
- For example, during user registration you would like to collect information such as name, email address, credit card, etc.

# Example of Form

First name:

Last name:

# The `<form>` Element

---

- The HTML `<form>` element defines a form that is used to collect user input:
- `<form>`  
.....
- *form elements*  
.....
- `</form>`

# <form> attributes

---

- Action:
  - The action attribute defines the action to be performed when the form is submitted.
  - Example: `<form action="/action_page.html`
- Target: `_blank`, **`_self`**, `_parent`, `_top`
  - The target attribute specifies if the submitted result will open in a new browser tab, a frame, or in the current window.
- Method: GET, POST
- `novalidate`
  - Specifies that the browser should not validate the form.

# Grouping Form Data with <fieldset>

---

- The <fieldset> element is used to group related data in a form.
- The <legend> element defines a caption for the <fieldset> element.



# Example

---

```
<form action="/action_page.php
```

```
<fieldset>
```

```
  <legend>Personal information:</legend>
```

```
  First name:<br> text" name="firstname" value="Mickey <br>
```

```
  Last name:<br> text" name="lastname" value="Mouse <br>
```

```
  submit" value="Submit
```

```
</fieldset>
```

```
</form>
```

---

Personal information:

First name:

Last name:

- Form elements are different types of input elements, like

- Text fields
- Multiple-Line Text Input
- Checkboxes
- Radio buttons
- Select Box Control
- Buttons

 Maths  Science MCA  MSC(IT & CA)

# <INPUT> Attributes

---

- HTML Input Types : **button**, **checkbox**, color, date, datetime-local, email, file, **hidden**, image, month, number, **password**, **radio**, range, **reset**, search, **submit**, tel, **text**, time, url, week

# <textarea> attribute

---

- Multiline Textbox
- Attributes:
- **Name**
- **Rows : in number**
- **Cols : in number**

# List / Select Box

## <select> Element

---

- **<select> tag**
- The SELECT tag defines a selection list on an HTML form.
- A selection list displays a list of options from which the user can select an item.
- **<option> tag**
- The OPTION tag specifies an option and placed inside a <select> tag.
- When the form containing the selection list is submitted to the server, a name/value pair is sent for each selected option in the list

## Attribute of <select>

- Name
- Size : in number
- Multiple

## Attributes of <option>

- Value
- Selected



# VIDEO AND SOUND IN HTML



# Video Player of HTML 5



# Properties of <Video>

- **Controls**

- Specifies that video controls should be displayed (such as a play/pause button etc).

- **Src** – url

- Specifies the URL of the video file

- **Poster** – image url (thumbnail of video)

- Specifies an image to be shown while the video is downloading, or until the user hits the play button

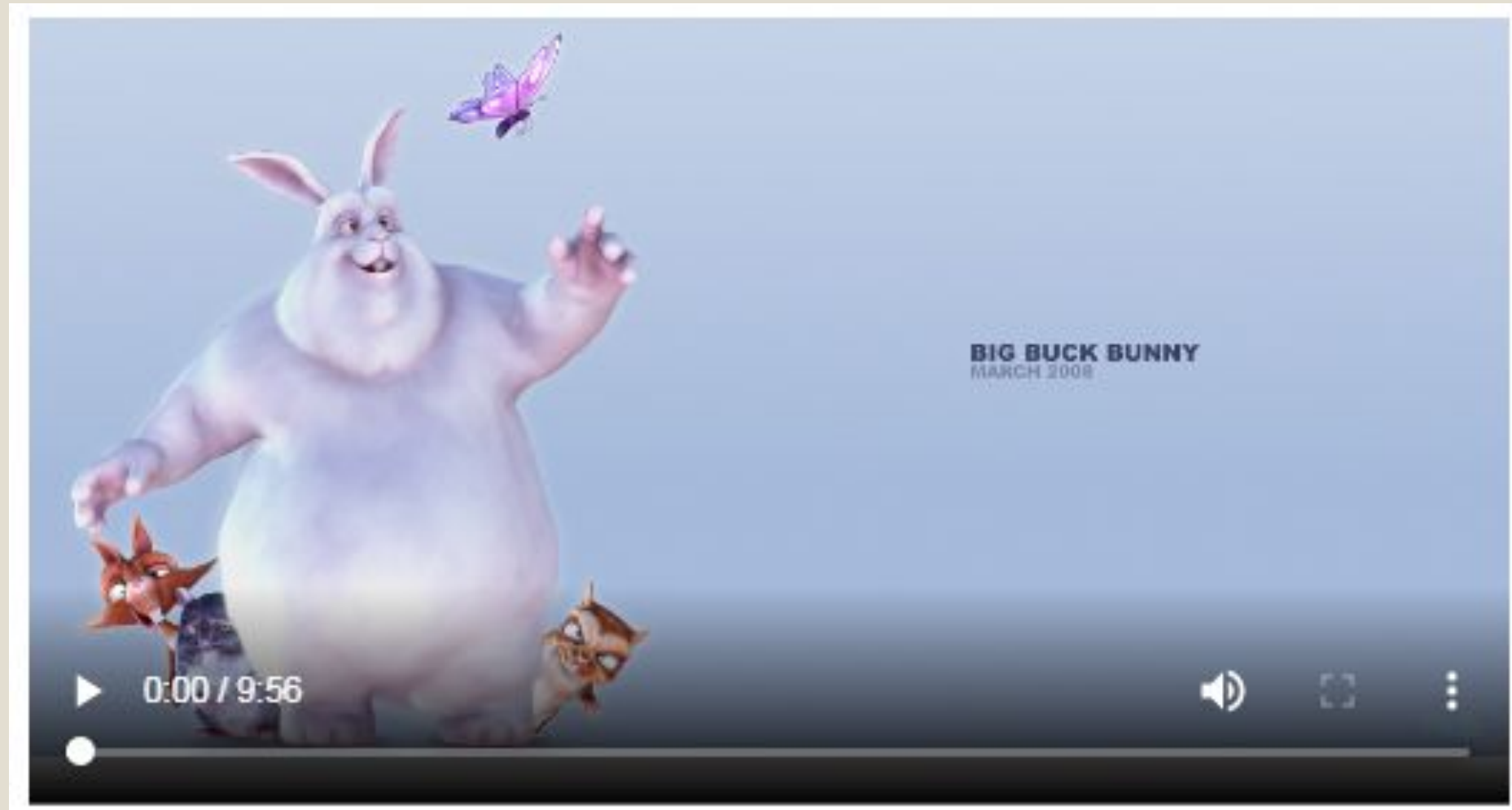
- **Height** – in px

- Sets the height of the video player

- **Width** – in px

- Sets the width of the video player

# Thumbnail of Video



- **Preload** – auto | metadata | none
- It may have one of the following values:
  - none: indicates that the video should not be preloaded
  - metadata: indicates that only audio metadata (e.g. length) is fetched
  - auto: indicates that the whole video file could be loaded, even if the user is not expected to use it

- **Muted (True | False)**

- which indicates whether the audio will be initially silenced. Its default value is false.

- **Loop (True | False)**

- if specified, will automatically seek back to the start upon reaching the end of the audio.

- **Autoplay (True | False)**

- if specified the audio will automatically begin playback as soon as it can do so, without waiting for the entire audio file to finish downloading.

# HTML YouTube Videos

- Converting videos to different formats can be difficult and time-consuming.
- An easier solution is to let YouTube play the videos in your web page.
- **YouTube Video Id**
- YouTube will display an id (like tgbNymZ7vqY), when you save (or play) a video.
- You can use this id, and refer to your video in the HTML code.

# Playing a YouTube Video in HTML

1. Upload the video to YouTube
2. Take a note of the video id
3. Define an `<iframe>` element in your web page
4. Let the `src` attribute point to the video URL
5. Use the `width` and `height` attributes to specify the dimension of the player

- URL of YouTube Video

- <https://www.youtube.com/watch?v=Ho508axF87Q>

- **<iframe> for youtube video**

```
<iframe width="420" height="345"  
src="https://www.youtube.com/embed/Ho508axF87Q">  
</iframe>
```



# **<embed > tag for audio and video play**

- **src**
- **autostart** - true | false
- **hidden** - true | false
- **volume** - 0 to 100
- **Loop** – true | false
- **playcount** - in number

# Sound in HTML

- The HTML `<audio>` element is used to embed sound content in documents.
- It may contain one or more audio sources, represented using the **src** attribute or the `<source>` element: the browser will choose the most suitable one.

# Attributes

- **controls**

- If this attribute is present, the browser will offer controls to allow the user to control audio playback, including volume, seeking, and pause/resume playback.

- **src**

- The URL of the audio to embed.

- This is optional; you may instead use the `<source>` element within the audio block to specify the audio to embed.

- **volume**

- The playback volume, in the range 0.0 (silent) to 1.0 (loudest).

- **preload**

- It may have one of the following values:

- none: indicates that the audio should not be preloaded

- metadata: indicates that only audio metadata (e.g. length) is fetched

- auto: indicates that the whole audio file could be downloaded, even if the user is not expected to use it

- **muted**

- **A Boolean attribute:** which indicates whether the audio will be initially silenced. Its default value is false.

- **loop**

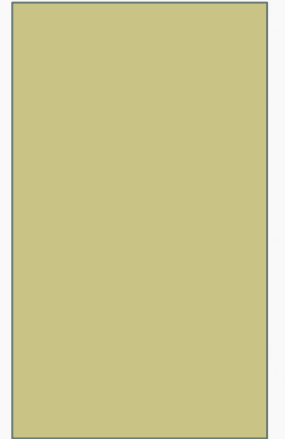
- **A Boolean attribute:** if specified, will automatically seek back to the start upon reaching the end of the audio.

- **autoplay**

- **A Boolean attribute:** if specified the audio will automatically begin playback as soon as it can do so, without waiting for the entire audio file to finish downloading.

# CANVAS AND SVG IN HTML 5

JALPA PORIYA



# INTRODUCTION

- The HTML `<canvas>` element is used to draw graphics, on the fly, via JavaScript.
- The `<canvas>` element is only a container for graphics. You must use JavaScript to actually draw the graphics.
- Canvas has several methods for drawing paths, boxes, circles, text, and adding images.

## <CANVAS> EXAMPLE

- Here is a simple <canvas> element which has only two specific attributes width and height plus all the core HTML5 attributes like id, name and class, etc.
- `<canvas id = "mycanvas" width = "100" height = "100"></canvas>`



# DRAWING RECTANGLES

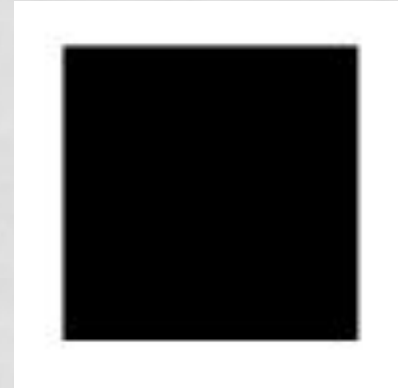
- **fillRect(x,y,width,height)**
  - This method draws a filled rectangle.
- **strokeRect(x,y,width,height)**
  - This method draws a rectangular outline.
- **clearRect(x,y,width,height)**
  - This method clears the specified area and makes it fully transparent

# HTML5 CANVAS - DRAWING LINES

- **beginPath()**
  - This method resets the current path.
- **moveTo(x, y)**
  - This method creates a new subpath with the given point.
- **lineTo(x, y)**
  - This method adds the given point to the current subpath, connected to the previous one by a straight line.

# EXAMPLE

- `ctx.beginPath();`
- `ctx.moveTo(25,25);`
- `ctx.lineTo(105,25);`
- `ctx.lineTo(105,105);`
- `ctx.lineTo(25,105);`
- `ctx.fill();`



# HTML5 CANVAS - TEXT AND FONTS

- **font [ = value ]**
  - This property returns the current font settings and can be set, to change the font.
  - The possible values are fontstyle (italic, underline, bold), font size (in pixel), fonttype (times new roman).
- **textAlign [ = value ]**
  - This property returns the current text alignment settings and can be set, to change the alignment.
  - The possible values are start, end, left, right, and center.
- **fillText(text, x, y [, maxWidth ] )**
  - This property fills the given text at the given position indicated by the given coordinates x and y.
- **strokeText(text, x, y [, maxWidth ] )**
  - This property strokes the given text at the given position indicated by the given coordinates x and y.

- `ctx.fillStyle = '#00F';`
- `ctx.font = 'Italic 30px Sans-Serif';`
- `ctx.textBaseline = 'Top';`
- `ctx.fillText('Hello world!', 40, 100);`

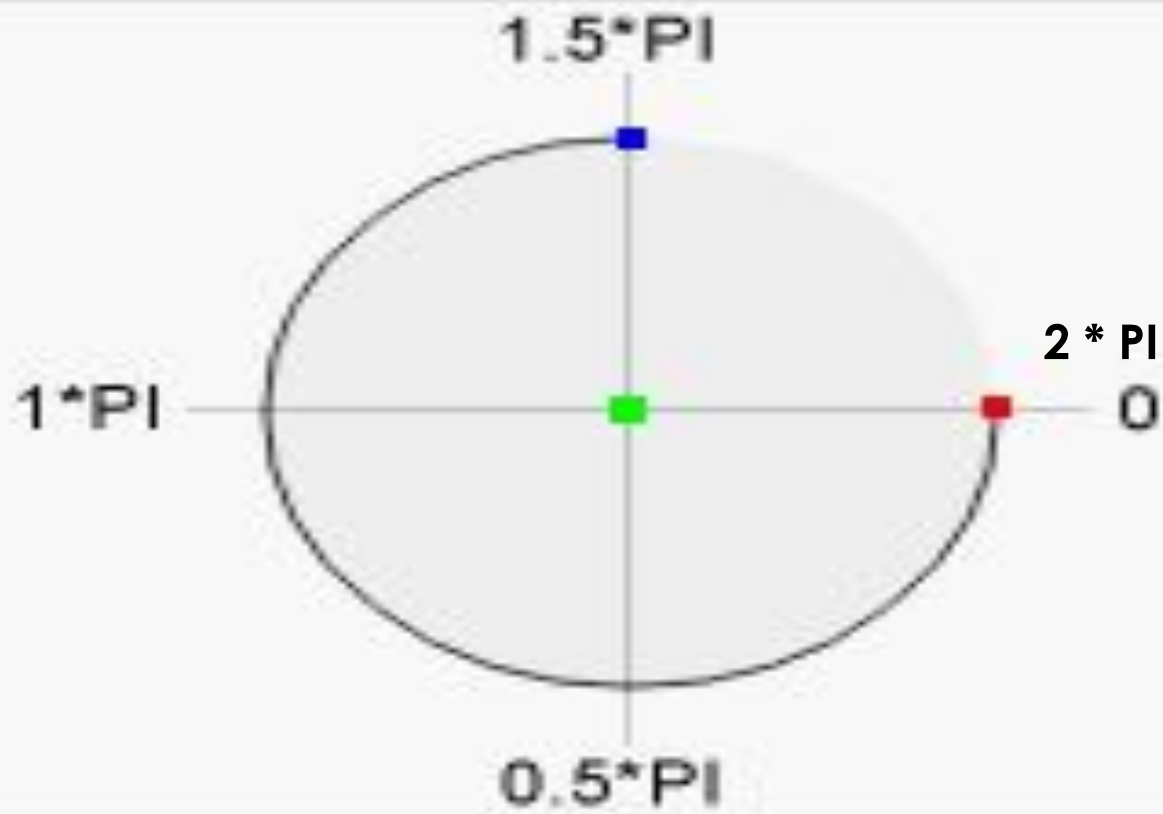
*Hello world!*

- `ctx.font = 'Bold 30px Sans-Serif';`
- `ctx.strokeText('Hello world!', 40, 50);`

Hello world!

# HTML 5 CANVAS - ARC METHOD

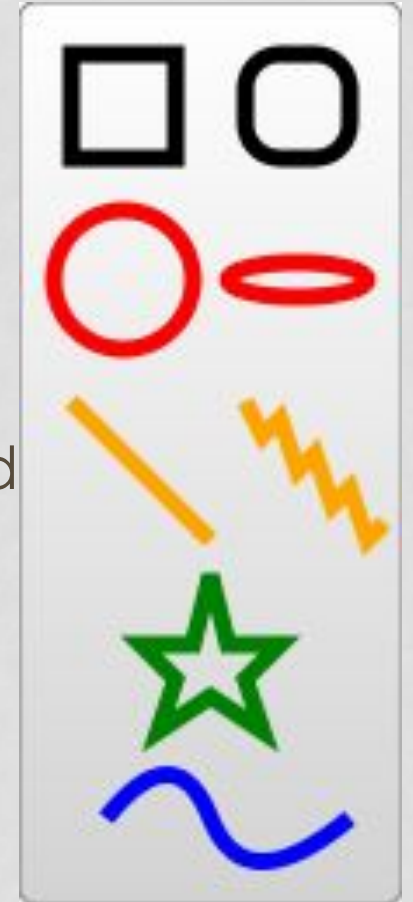
- `arc(x, y, r, sAngle, eAngle, counterclockwise);`
- **Parameters:**
- **x** The x-coordinate of the center of the circle
- **y** The y-coordinate of the center of the circle
- **r** The radius of the circle
- **sAngle** The starting angle, in radians (0 is at the 3 o'clock position of the arc's circle)
- **eAngle** The ending angle, in radians
- **counterclockwise** Optional. Specifies whether the drawing should be counterclockwise or clockwise. False is default, and indicates clockwise, while true indicates counter-clockwise.



- **Center**
  - `arc(100,75,50,0*Math.PI, 1.5*Math.PI)`
- **Start angle**
  - `arc(100,75,50,0,1.5*Math.PI)`
- **End angle**
  - `arc(100,75,50,0,1.5*Math.PI)`

# SVG IN HTML 5

- SVG stands for **Scalable Vector Graphics**
- SVG is used to define vector-based graphics for the Web.
- **What does Vector-based Graphic mean?**
  - A vector graphic is a type of image.
  - Vector images are graphical representations of mathematical objects such as lines, curves, polygons and its like.
  - These graphics are generated by computer and they follow x and y axis as their reference definition.
- SVG defines the graphics in XML format.
- Every element and every attribute in SVG files can be animated.

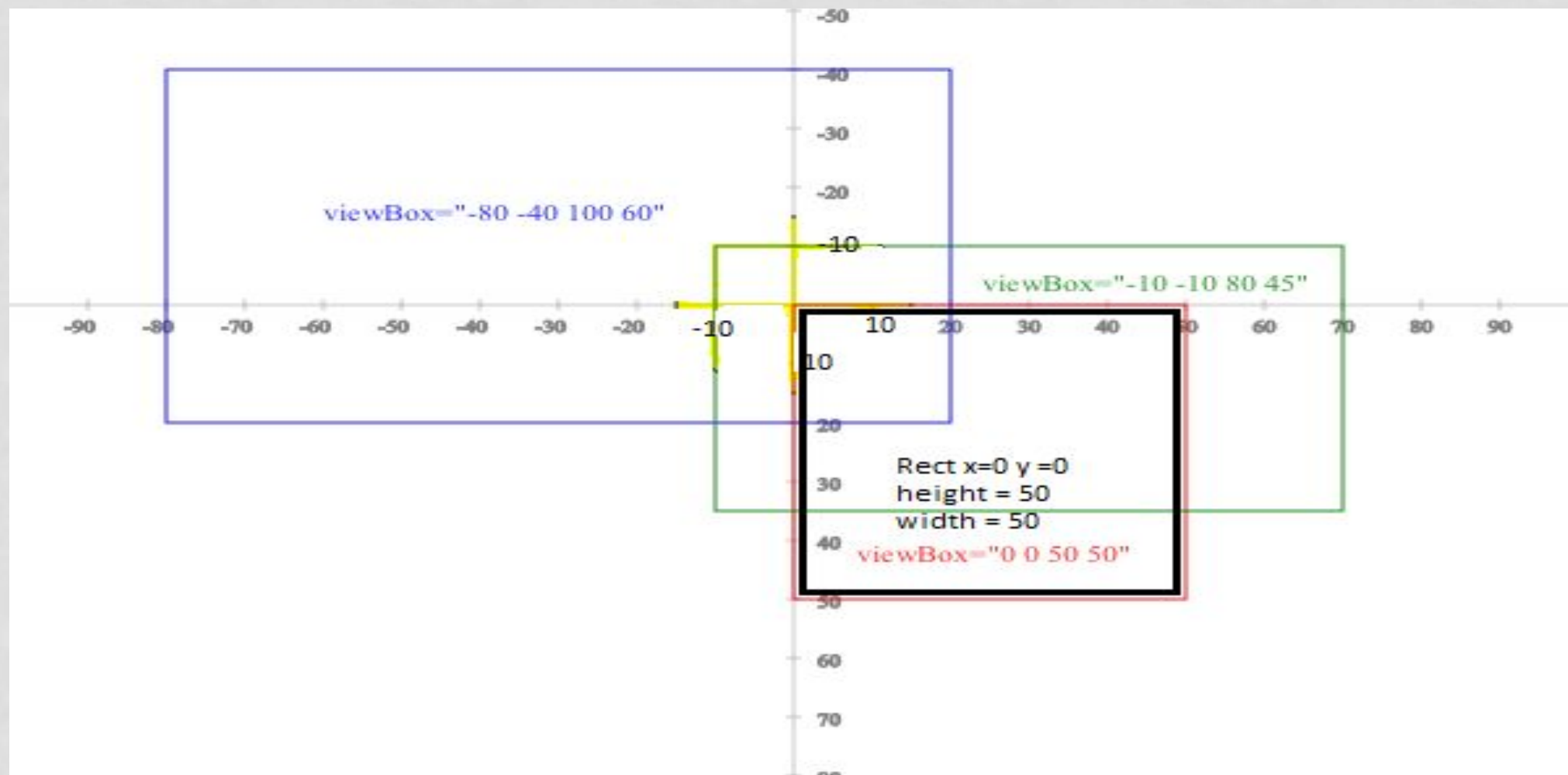




## <SVG> PROPERTIES

```
<svg width="100" height="100" viewBox = "0 0 50 50" >  
</svg>
```

# VIEW BOX IN SVG



# SVG SHAPES

- SVG has some predefined shape elements that can be used by developers:
  - Rectangle `<rect>`
  - Circle `<circle>`
  - Line `<line>`
  - Animation `<animate>`
  - Ellipse `<ellipse>`
  - Polyline `<polyline>`
  - Polygon `<polygon>`
  - Path `<path>`

# SVG RECTANGLE - <RECT> TAG PROPERTIES

- x - The x coordinate of the rect. Default value: 0;
- y - The y coordinate of the rect. Default value: 0;
- Width - The width of the rect.  
Value type: **auto** | <length> | <percentage> ;
- Height - The height of the rect.  
Value type: **auto** | <length> | <percentage> ;
- Rx - The horizontal corner radius of the rect.  
Value type: **auto** | <length> | <percentage> ;  
Default Value : Ry
- Ry - The vertical corner radius of the rect.  
Value type: **auto** | <length> | <percentage> ;  
Default Value : Rx

## PRESENTATION ATTRIBUTES OF <RECT>

- fill : none | color-name | hex-code | **black**
- fill-opacity : [0-**1**]
- stroke : color-name | hex-code | **fill-color**
- stroke-dasharray - **none** | <dasharray>
- stroke-opacity : [0-**1**]
- stroke-width : <length> | <percentage> | **1px**
- visibility : **visible** | hidden | collapse
- cursor : wait | text | **auto** | pointer | help | e-resize

# SVG ELLIPSE - <CIRCLE> TAG PROPERTIES

- cx - The x-axis coordinate of the center of the circle.
  - Default value: 0;
- cy - The y-axis coordinate of the center of the circle.
  - Default value: 0;
- r - The radius of the circle. A value lower or equal to zero disables rendering of the circle.
  - Value type: <length> ; Default value: 0;

# EXAMPLE

```
<svg height="200" width="200">  
  <circle cx="50" cy="50" r="50"/>  
</svg>
```

# SVG LINE - <LINE> TAG PROPERTIES

- **x1** - Defines the x-axis coordinate of the line starting point. Default value: 0;
- **y1** - Defines the y-axis coordinate of the line starting point. Default value: 0;
- **x2** - Defines the x-axis coordinate of the line ending point. Default value: 0;
- **y2** - Defines the y-axis coordinate of the line ending point. Default value: 0;
  
- **Example:**
- `<line x1="0" y1="80" x2="100" y2="20" style="stroke:black" />`



# SVG ANIMATION - <ANIMATE>

- attributeType - Information about target attribute
  - Values - CSS | XML | **auto**
- attributeName - Indicates the name of the CSS property or attribute of the target element that is going to be changed during an animation.
  - Example:
    - `<rect x="50" y="50" width="100" height="100">`
    - `<animate attributeType="XML" attributeName="y"/>`
    - `</rect>`

- from - The from attribute indicates the initial value of the attribute that will be modified during the animation. When used with the to attribute, the animation will change the modified attribute from the from value to the to value.
- to - The to attribute indicates the final value of the attribute that will be modified during the animation. The value of the attribute will change between the from attribute value and this value.
- dur - The dur attribute indicates the simple duration of an animation.
- repeatCount - The repeatCount attribute indicates the number of times an animation will take place. Values – number | indefinite

# HTML5 API

---



# Geo Location Introduction

---

The HTML Geolocation API is used to locate a user's position.

Since this can compromise privacy, the position is not available unless the user approves it.

Geolocation is most accurate for devices with GPS, like smartphone.

# Navigator in JS

---

The navigator object contains information about the browser.

## Properties of Navigator

**appName** Returns the name of the browser

**cookieEnabled** Determines whether cookies are enabled in the browser

**geolocation** Returns a Geolocation object that can be used to locate the user's position

# Method of geolocation

---

## 1. `getCurrentPosition()`

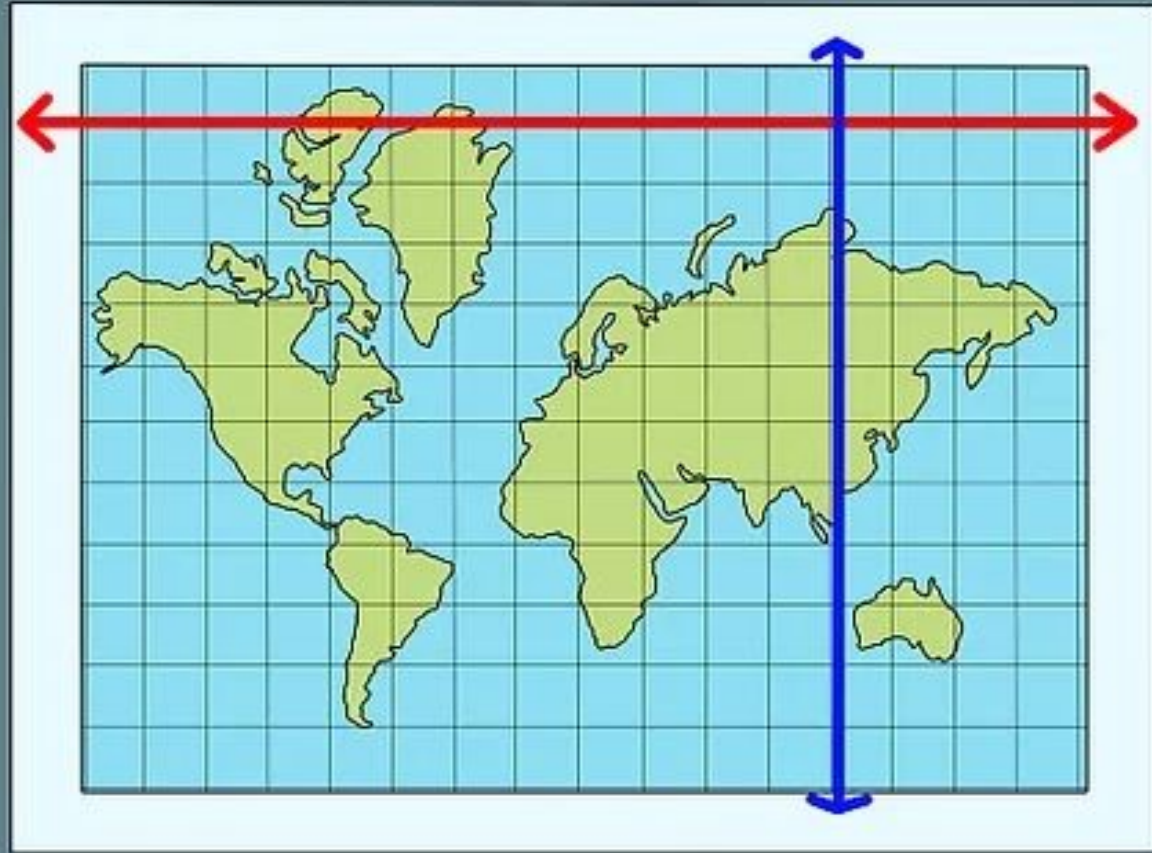
The `getCurrentPosition` method retrieves the current geographic location of the device.

The location is expressed as a set of geographic coordinates ([longitude, latitude](#)) together with information about heading and speed.

The location information is returned in a `Position` object.

North

Latitude



Longitude

# Syntax

```
getCurrentPosition(showLocation[, ErrorHandler, options]);
```

**showLocation** – This specifies the callback method that retrieves the location information.

This method is called asynchronously with an object corresponding to the Position object which stores the returned location information.

**ErrorHandler** – This specifies the callback method that is invoked when an error occurs in processing the asynchronous call.

**options** – This specifies a set of options for retrieving the location information.



# Drag and Drop

---

Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location.

In HTML5, drag and drop is part of the standard: Any element can be draggable.

# To Make an Element Draggable

---

First of all:

To make an element draggable, set the draggable attribute to true:

For Example: `<img draggable="true">`  
`<input type="text" draggable="true">`

# Drag Events

---

HTML drag-and-drop uses the DOM event model and *drag events* inherited from mouse events.

Event	Event Handler Name	Description
Drag	ondrag	a dragged item selection is dragged.
Drag start	ondragstart	the user starts dragging an item.
Drag end	ondragend	a drag operation ends such as releasing a mouse button or hitting the Esc key.
Drag enter	ondragenter	a dragged item enters a valid drop target.
Drag leave	ondragleave	a dragged item leaves a valid drop target.

# Drag Events

---

<b>Event</b>	<b>Event Handler Name</b>	<b>Description</b>
Drag over	ondragover	a dragged item is being dragged over a valid drop target, every few hundred milliseconds.
Drop	ondrop	an item is dropped on a valid drop target.

# When element Drag – ondragstart and setData()

---

## HTML Element:

```

```

## JS Function:

```
function drag(ev) {  
    ev.dataTransfer.setData("img", ev.target.id);  
}
```



# DataTransfer.setData()

---

Syntax:

```
void dataTransfer.setData(format, data);
```

Format - A representing the type of the drag data to add to the drag object.

Data - A representing the data to add to the drag object.

# Where element Drops - ondragover

---

The ondragover event specifies where the dragged data can be dropped.

By default, data/elements cannot be dropped in other elements. To allow a drop, we must prevent the default handling of the element.

## HTML Element:

```
<div id="div2" ondragover="allowDrop(event)"></div>
```

## JS Function:

```
function allowDrop(ev) {  
    ev.preventDefault();  
}
```



# Do the Drop - ondrop

---

When the dragged data is dropped, a drop event occurs.

## HTML Element:

```
<div id="div2" ondragover="allowDrop(event)" ondrop="drop(event)" ></div>
```

## JS Function:

```
function drop(ev) {  
    ev.preventDefault();  
    var data = ev.dataTransfer.getData("img");  
    ev.target.appendChild(document.getElementById(data)); }  
}
```





# HTML5 Web Storage

---

## What is HTML Web Storage?

With web storage, web applications can store data locally within the user's browser.

Before HTML5, application data had to be stored in cookies, included in every server request. Web storage is more secure, and large amounts of data can be stored locally, without affecting website performance.

Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.

Web storage is per origin (per domain and protocol). All pages, from one origin, can store and access the same data.

# HTML Web Storage Objects

---

HTML web storage provides two objects for storing data on the client:

1. `window.localStorage` - stores data with no expiration date
2. `window.sessionStorage` - stores data for one session

# localStorage

---

LocalStorage is a type of web storage that allows Javascript websites and apps to store and access data right in the browser with no expiration date.

This means the data stored in the browser will persist even after the browser window has been closed.

To use localStorage in your web applications, there are five methods to choose from:

---

**setItem():** Add key and value to localStorage

```
window.localStorage.setItem('class', 'IMSc & MScIT');
```

**getItem():** Retrieve a value by the key from localStorage

```
window.localStorage.getItem('class');
```

**removeItem():** Remove an item by key from localStorage

```
window.localStorage.removeItem('class');
```

---

**clear():** Clear all localStorage

```
window.localStorage.clear();
```

**key():** Passed a number to retrieve nth key of a localStorage

```
var KeyName = window.localStorage.key(index);
```

# LocalStorage JavaScript browser support

---

To be sure the browser supports localStorage, you can check using the following snippet:

```
if (typeof(Storage) !== "undefined")  
{  
    // Code for localStorage  
} else {  
    // No web storage Support.  
}
```

# LocalStorage JavaScript limitations

---

1. Do not store sensitive user information in localStorage.
2. It is not a substitute for a server based database as information is only stored on the browser.
3. LocalStorage is quite insecure as it has no form of data protection and can be accessed by any code on your web page.

# The sessionStorage Object

---

The sessionStorage object work in the same way as localStorage, except that it stores the data only for one session i.e. the data remains until the user closes that window or tab.

`setItem(key,value)`

`getItem(key)`



# **Image & Image Map in HTML**

# Image in HTML

- **Attribute of <IMG>**
- SRC – path of your image
- ALT – alternate text if image not found in string
- NAME – name of image in string
- BORDER – in px
- WIDTH – in %, in px
- HEIGHT – in %, in px
- ALIGN – Left, Right, Center, Top, Bottom
- HSPACE – in px
- VSPACE – in px

# Understand Image Map Elements

- `<map></map>`
  - Defines a ***client side*** image map on you web page.
- `<area/>`
  - Used for ***specifying the coordinates*** of hot spots in client-side image maps.
  - You will use one `<area />` element for ***each hot spot*** on the image.
- `<img/>`
  - You will use the image element to ***specify the image*** you want to use for your image map.

- Example
- `<map name="">`
- `<area> <area />`
- `</map>`

# <area> Tag attribute

- Shape : Ract
- Coords : Ract takes two pairs of numbers to define a rectangle.
- Href : URL of WebPage
- **Example:**

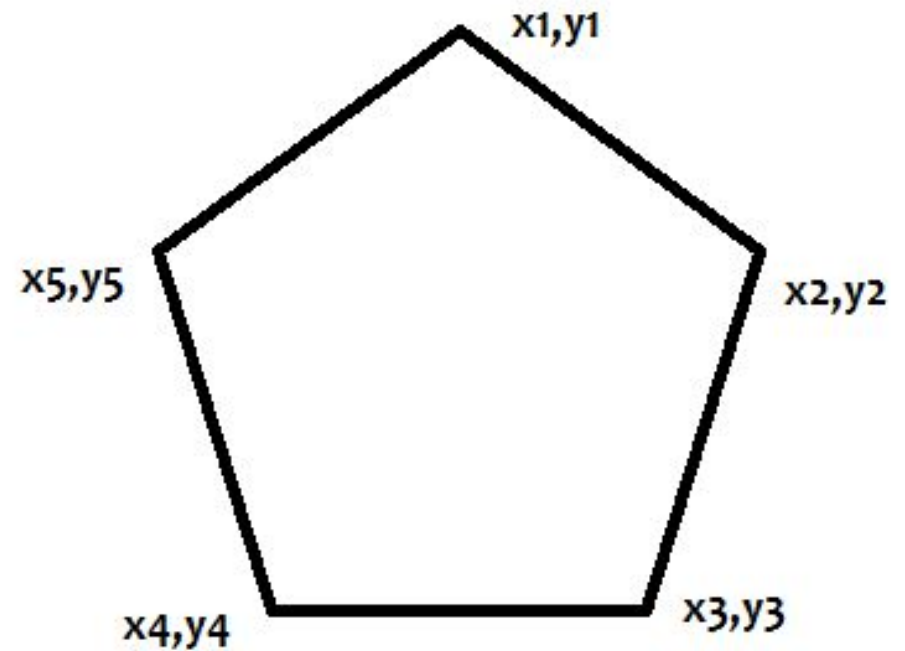
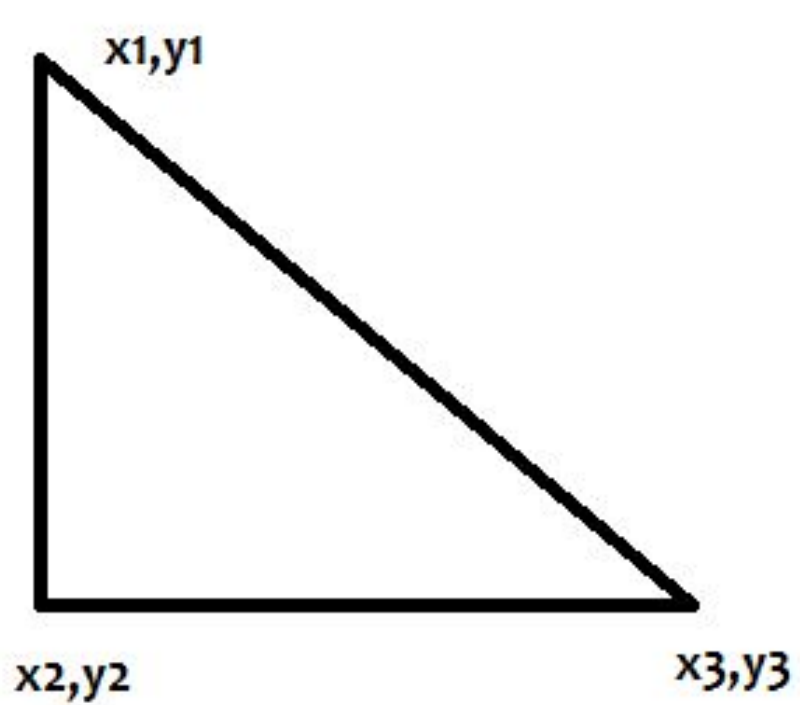
```
  
<map name="mapDemo">  
<area shape="ract"  
  coords="22,30,254,222"  
  href="1.html">  
</area></map>
```

X1,Y1



X2,Y2

- Shape : Poly
- Coords : the coords attribute takes three or more pairs of  $x, y$  coordinates to define a polygon.



- Shape : Circle
- Coords : The Coordinates attribute takes three numbers: the  $x$ ,  $y$  coordinates of the circle's center and the circle's radius

