# HTML5

## Features

HTML5 comes with a lot of flexibility and it supports the following features –

- Uppercase tag names.
- Quotes are optional for attributes.
- Attribute values are optional.
- Closing empty elements are optional.

# The DOCTYPE

- DOCTYPEs in older versions of HTML were longer because the HTML language was SGML based and therefore required a reference to a DTD.

e.g.

```
<!DOCTYPE html>
```

The above syntax is case-insensitive.

# Character Encoding

```
<meta charset = "UTF-8">
```

The above syntax is case-insensitive.

## The <script> tag

- We commonly used script tag:

<script type = "text/javascript" src = "scriptfile.js"></script>

HTML 5 removes extra information required and you can use simply following syntax –

<script src = "scriptfile.js"></script>

# The <link> tag

- <link rel = "stylesheet" type = "text/css" href = "stylefile.css">

- HTML 5 removes extra information required and you can simply use the following syntax –

<link rel = "stylesheet" href = "stylefile.css">

## New added tags or elements

- **section** – This tag represents a generic document or application section. It can be used together with h1-h6 to indicate the document structure.
- **article** – This tag represents an independent piece of content of a document, such as a blog entry or newspaper article.
- **aside** – This tag represents a piece of content that is only slightly related to the rest of the page.

- **header** – This tag represents the header of a section.
- **footer** – This tag represents a footer for a section and can contain information about the author, copyright information, et cetera.
- **nav** – This tag represents a section of the document intended for navigation.
- **dialog** – This tag can be used to mark up a conversation.
- **figure** – This tag can be used to associate a caption together with some embedded content, such as a graphic or video.

## <figure> and <figcaption> Elements

- The purpose of a figure caption is to add a visual explanation to an image.
- In HTML5, an image and a caption can be grouped together in a <figure> element:

```
<figure>
  <img src="pic_trulli.jpg" alt="Trulli">
  <figcaption>Fig1. - Trulli, Puglia,
Italy.</figcaption>
</figure>
```

- The <section> tag:

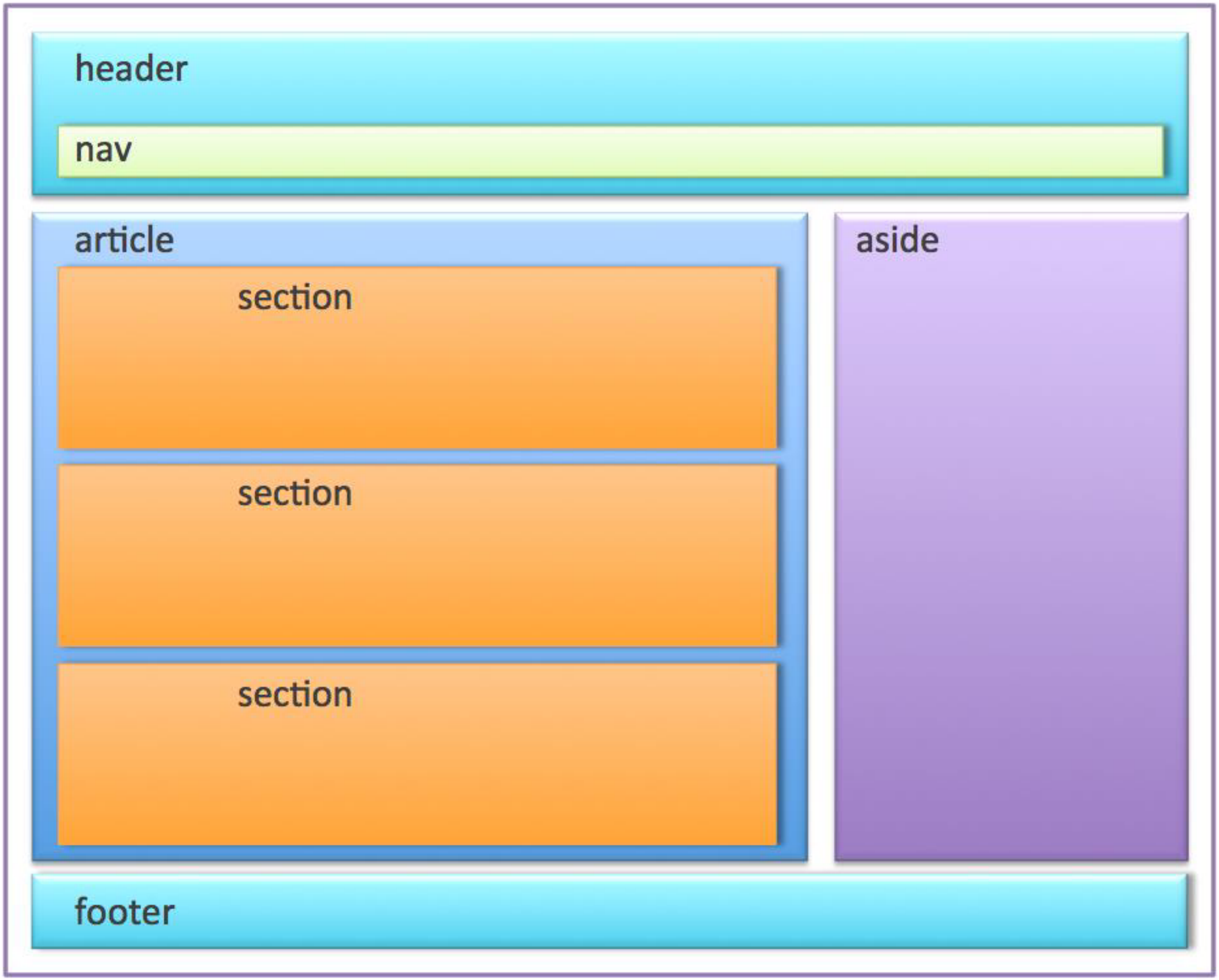
The section tag defines sections in a document, such as chapters, headers, footers, or any other sections of the document.

The <article> tag:

- The article tag specifies independent, self-contained content.
- An article should make sense on its own and it should be possible to distribute it independently from the rest of the site.

Potential sources for the article element:

- Forum post
- Blog post
- News story
- Comment


header
nav
article
section
section
section
aside
footer

## Forms in HTML5

- Web Forms 2.0 is an extension to the forms features found in HTML4. Form elements and attributes in HTML5 provide a greater degree of semantic mark-up than HTML4.

# The <input> element in HTML5

- HTML5 input elements introduced several new values for the **type** attribute. These are listed below.
- datetime
- Datetime-local
- Date
- Month
- Week
- Time
- Number
- Range
- Email
- url

## The <output> element

- HTML5 introduced a new element <output> which is used to represent the result of different types of output, such as output written by a script.

```
<!DOCTYPE HTML>

<html>
 <head>
<script type = "text/javascript">
 function showResult()
{ x = document.forms["myform"]["newinput"].value;
    document.forms["myform"]["result"].value = x;
 }
 </script>
</head>
<body>
<form action = "/cgi-bin/html5.cgi" method = "get" name = "myform">
    Enter a value : <input type = "text" name = "newinput" />
 <input type = "button" value = "Result" onclick = "showResult();" />
    <output name = "result"></output>
 </form>
 </body>
 </html>
```

# The placeholder attribute

- HTML5 introduced a new attribute called **placeholder**. This attribute on <input> and <textarea> elements provide a hint to the user of what can be entered in the field.

- <input type = "text" name = "search" placeholder = "search the web"/>

```
<!DOCTYPE HTML>
 <html>
<body>
 <form action = "/cgi-bin/html5.cgi" method = "get">
Enter email : <input type = "email" name = "newinput"
    placeholder = "email@example.com"/>
 <input type = "submit" value = "submit" />
 </form>
 </body>
 </html>
```

## The autofocus attribute

```
<input type = "text" name = "search"
  autofocus/>
```

## The required attribute

- Now you do not need to have JavaScript for client-side validations like empty text box would never be submitted because HTML5 introduced a new attribute called **required** which would be used as follows and would insist to have a value:

```
<input type = "text" name = "search" required/>
```

## CSS2

## CSS2(Level 2)

- CSS2 is a style sheet language that allows authors and users to attach style (e.g., fonts, spacing, and aural cues) to structured documents (e.g., HTML documents and XML applications).
- By separating the presentation style of documents from the content of documents, CSS2 simplifies Web authoring and site maintenance.

# Properties

1. Border: *border* is a shorthand property which sets the style, color, and width of the border around an element.

*object.style.border = "2px solid red";*

**Possible value is one or more of a color, a value for border-width, and a value for border-style.**

```
<html>
 <head>
 </head>
 <body>
<p style = "border:4px solid red;"> This example
   is showing shorthand property for border.
   </p>
</body> </html>
```

2. The border-bottom property allows you to specify color, style, and width of bottom lines in one property.

```
object.style.borderBottom = "2px solid red";
```

```
<body>
 <p style = "border-bottom:4px solid red;"> This
   example is showing shorthand property for
   border-bottom.
</p>
</body>
```

## borderBottomWidth

*object.style.borderBottomWidth = "2px";*

Possible values:

- **length**
- **Thin**
- **Medium**
- **thick**

```
<p style = "border-bottom-width:4px;
                border-top-width:10px;
                border-left-width: 2px;
                border-right-width:15px;
   border-style:solid;">   //border-style:dotted
 This is a a border with four different width.
</p>
```

## border-collapse

Possible Values

- **collapse** – Borders are collapsed to make a single border. Two adjacent cells will share a border.
- **separate** – Borders are separated. Every cell has its own border, and none of these borders are shared with other cells in the table.

# border-color

The border-color property allows you to change the color of the border surrounding an element. You can individually change the color of the bottom, left, top and right sides of an element's border using the properties –

- **border-bottom-color** changes the color of bottom border.
- **border-top-color** changes the color of top border.
- **border-left-color** changes the color of left border.
- **border-right-color** changes the color of right border.

## border-left

The border-left property allows you to specify color, style, and width of left lines in one property.

Example:

```
<p style = "border-left:4px solid red;"> This example is showing shorthand property for border-left.
</p>
```

## Border-left-width

- **length** – Any length unit. Length units for this property may not be negative(e.g. 4px).
- **thin** – A border which is thinner than a border set to medium.
- **medium** – A border which is thicker than a border set to thin, and thinner than a border set to thick.
- **thick** – A border which is thicker than a border set to medium.

- border-left-width: Sets the thickness of an element's left border.
- border-right: Sets an element's right border; value is one or more of a color, a value for border-right-width, and a value for border-style.
- border-right-width: Sets the thickness of an element's right border.

# border-spacing

border-spacing sets the distance between cells in a table.

Possible value:

**length** – Any length unit. If two values are declared, the first applies to spacing along the horizontal axis, and the second applies to the vertical axis.

*object.style.borderSpacing = "2px";*

## *caption-side*

- The *caption-side* property determines the placement of the element box of a table.

- **Top**
- **Bottom**
- Left
- right
- initial
- inherit

*object.style.captionSide = "left";*

## background property

background property consist of the following properties –

- background-attachment
- background-color
- background-image
- background-position
- background-repeat

background-attachment determines the tiling context and scroll state of a background image.

Possible Values:

- **scroll** – A background image set to scroll will scroll along with the rest of the document.
- **fixed** – A background image set to fixed will remain locked in place while the rest of the document scrolls.

<p **style = "background-image:url(any url);**
   **background-attachment:scroll ;    //fixed**

**</p>**

**<u>background-color:</u>**

***<p style = "background-color:yellow;">***

*</p>*

<u>background-image</u> defines a pointer to an image resource which is to be placed in the background of an element.

Possible Values

- **uri** – URL of the image.
- **none** – Setting background-image to none means that no background image should be used for matching elements.
- **transparent**

```
<html> <head>
<style>
body
{ background-image: url("/css/images/css.jpg");
   background-color: #cccccc; } </style>
 </head>
<body>
<h1>Hello World!</h1> </body> <html>
```

# background-position

- Sets the initial position of the element's background image, if specified; values normally are paired to provide x, y positions; default position is 0% 0%.

Possible Values

- percent
- none
- length
- top
- center
- bottom
- left
- right

- <p **style = "background-image:url(/css/images/logo.png); background-position:100px 200px;"**>
- The background image positioned 100 pixels away from the left and 200 pixels from the top.up </p>

background-repeat defines the directions in which a background image will be repeated (if any).

Possible Values

- **repeat** – Causes the background image to be repeated along both the horizontal and vertical axes.
- **repeat-x** – Causes the background image to be repeated along the x axis.
- **repeat-y** – Causes the background image to be repeated along the y axis.
- **no-repeat** – Prevents the background image from being repeated at all.

```
<style>
body
{ background-image: url("/css/images/css.jpg");
background-repeat: repeat; }
 </style>
```

# CSS 2.0

Font,

- The *font* property is a shorthand property used to affect the rendering of text.

- font-style
- font-variant
- font-weight
- font-size
- line-height
- font-family

## 1.Font-style(values)

- Normal
- Italic
- Oblique
- Initial
- inherit

2. font-variant: normal|small-caps|initial|inherit;

3. font-weight:

- normal
- Bold
- Bolder
- Lighter
- *Number (100-900)*
- Initial
- inherit;

4. **font-size**:medium|xx-small|x-small|small|large|x-large|xx-large|smaller|larger|*length*|initial|inherit;
5. **line-height**: normal|*number*|*length*|initial|inherit;

## Text Effects

- CSS filters can be used to add special effects to text, images and other aspects of a webpage without using images or other graphics.

1. **Alpha Channel:**

<img src = "/css/images/logo.png" alt = "CSS Logo“

style = "**Filter: Alpha**(Opacity=100, FinishOpacity = 0, Style = 2, StartX = 20, StartY = 40, FinishX = 0, FinishY = 0)" />

- **Opacity**- 0 to 100 (0 means transparent and 100 means opaque)
- **FinishOpacity**- 0 to 100 (optional)
- **Style**- 0 (uniform)

  1 (linear)

  2 (radial)

  3 (rectangular)
- **startX**

X coordinate for opacity gradient to begin.

- **startY**

Y coordinate for opacity gradient to begin.

- **finishX**

X coordinate for opacity gradient to end.

- **finishY**

Y coordinate for opacity gradient to end.

## Filter:blur

***style = "Filter: Blur(Add = 0, Direction = 225, Strength = 10)***

- **Add:** True or false. If true, the image is added to the blurred image; and if false, the image is not added to the blurred image.
- **Direction:**The direction of the blur, going clockwise, rounded to 45-degree increments. The default value is 270 (left).
- 0 = Top
- 45 = Top right
- 90 = Right
- 135 = Bottom right
- 180 = Bottom
- 225 = Bottom left
- 270 = Left
- 315 = Top left

- **Strength:** The number of pixels the blur will extend. The default is 5 pixels.

- Chroma Filter: Chroma Filter is used to make any particular color transparent and usually it is used with images.

```
<img src = "/images/css.gif" alt = "CSS Logo" style
   = "Filter: Chroma(Color = #FFFFFF)">
```

## Drop Shadow Effect

- Drop Shadow is used to create a shadow of your object at the specified X (horizontal) and Y (vertical) offset and color.

*drop-shadow(offset-x offset-y blur-radius spread-radius color)*

- **Color:** The color, in #RRGGBB format, of the dropshadow.
- **offX**

Number of pixels the drop shadow is offset from the visual object, along the x-axis. Positive integers move the drop shadow to the right, negative integers move the drop shadow to the left.

- **offY**

Number of pixels the drop shadow is offset from the visual object, along the y-axis. Positive integers move the drop shadow down, negative integers move the drop shadow up.

- blur-radius (optional):The shadow's blur radius, specified as a <length>. The larger the value, the larger and more blurred the shadow becomes. If unspecified, it defaults to 0, resulting in a sharp, unblurred edge. Negative values are not allowed.

- spread-radius (optional)The shadow's spread radius, specified as a <length>. Positive values will cause the shadow to expand and grow bigger, while negative values will cause the shadow to shrink. If unspecified, it defaults to 0, and the shadow will be the same size as the input image.

E.g.

```
<img src = "/css/images/logo.png" alt = "CSS
   Logo" style = "filter:drop-shadow(2px 2px 1px
   #FF0000);">
```

Grayscale Effect:Grayscale effect is used to convert the colors of the object to 256 shades of gray.

```
<img src = "/css/images/logo.png" alt = "CSS Logo" style = "filter: grayscale(50%)">
```

(if 100 % then totally grayscale image)

- Invert effect is used to map the colors of the object to their opposite values in the color spectrum, i.e., to create a negative image.

```
<img src = "/css/images/logo.png" alt = "CSS
   Logo" style = "filter: invert(100%)">
```

# CSS3

## border-radius property

- CSS3 Rounded corners are used to add special colored corner to body or text by using the border-radius property. The border-radius property can have from one to four values.

<u>Syntax:</u>

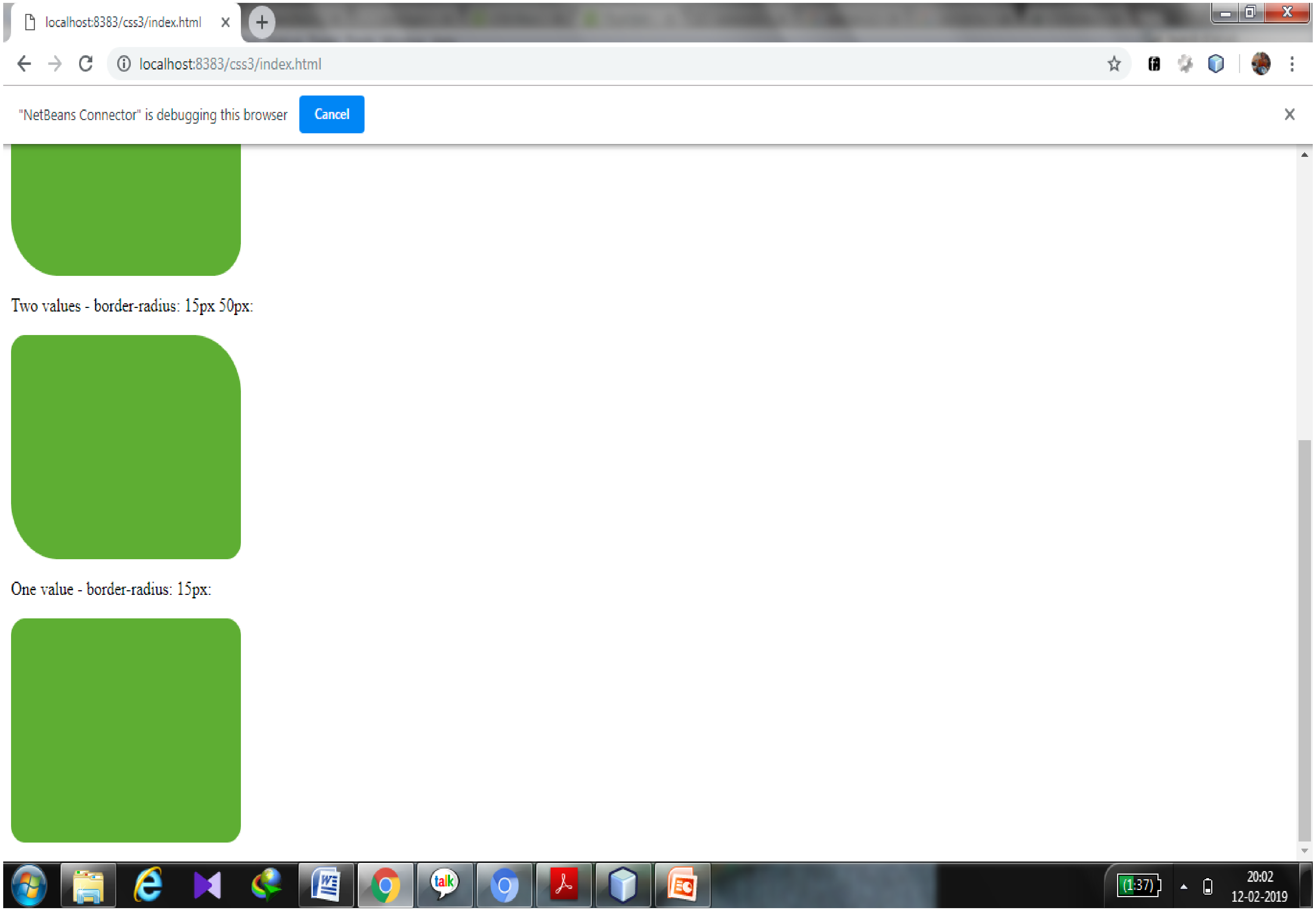
***border-radius: 15px 50px 30px 5px***

- **border-radius**
- **border-top-left-radius**
- **border-top-right-radius**
- **border-bottom-right-radius**
- **border-bottom-left-radius**

- first value applies to top-left corner,
- second value applies to top-right corner,
- third value applies to bottom-right corner, and
- fourth value applies to bottom-left corner

  – Some rules are defined:

- <u>Four values </u>- **border-radius: 15px 50px 30px 5px;** (first value applies to top-left corner, second value applies to top-right corner, third value applies to bottom-right corner, and fourth value applies to bottom-left corner)
- <u>Three values </u>- **border-radius: 15px 50px 30px;** (first value applies to top-left corner, second value applies to top-right and bottom-left corners, and third value applies to bottom-right corner)
- <u>Two values </u>- **border-radius: 15px 50px;** (first value applies to top-left and bottom-right corners, and the second value applies to top-right and bottom-left corners):
- <u>One value </u>- **border-radius: 15px;** (the value applies to all four corners, which are rounded equally:


localhost:8383/css3/index.html

"NetBeans Connector" is debugging this browser Cancel

Two values - border-radius: 15px 50px:

One value - border-radius: 15px:

## Border image property

The CSS border-image property allows you to specify an image to be used instead of the normal border around an element. The property has three parts:

- The image to use as the border
- Where to slice the image
- Define whether the middle sections should be repeated or stretched

Syntax:

*border-image: source slice width outset repeat|initial|inherit;*

- **border-image-source**
- **border-image-slice**
- **border-image-width**
- **border-image-repeat:stretch|repeat|round|space|initial|inherit**

```
<html>
<head>
<style>
body {
  background-color:#E7E9EB;
}
#myDIV {
  height:300px;
  background-color:#FFFFFF;
  border: 15px solid transparent;
  padding: 15px;
  border-image-source:url('border.png');
  border-image-repeat: stretch;
  border-image-slice: 30;
  border-image-width: 50px;
  border-image-outset: 50px;
  }
</style>
</head>
<body>

<h1>The border-image-outset Property</h1>

<div id="myDIV">
<p>A demonstration on how to set the border image.</p>
<p>We used this image:<br>
<img src="border.png">
</p>
</div>
```

## Texteffects

The following features are added later on in css3.0:

- text-overflow
- word-wrap
- word-break

## most commonly used property in CSS3

- **text-overflow**
- **word-break**
- **word-wrap**
- **Word-spacing**
- **text-align-last**

## Text-overflow

The text-overflow property specifies how overflowed content that is not displayed should be signaled to the user. It can be clipped, display an ellipsis (...), or display a custom string.

These following properties are required

- white-space: nowrap;
- overflow: hidden;

*text-overflow: clip|ellipsis|string|initial|inherit;*

- *Clip:Default value. The text is clipped and not accessible*
- *Ellipsis:Render an ellipsis ("...") to represent the clipped text.*
- *String:Render the given string to represent the clipped text.*
- *Initial:Sets this property to its default value.*
- *Inherit:Inherits this property from its parent element.*

## word-break property

The word-break property specifies how words should break when reaching the end of a line.

Syntax:

*word-break: normal|break-all|keep-all|break-word|initial|inherit;*

| normal | Default value. Uses default line break rules |
|---|---|
| break-all | To prevent overflow, word may be broken at any character |
| keep-all | This text will-break-at-hyphens |
| break-word | To prevent overflow, word may be broken at arbitrary points |
| initial | Sets this property to its default value. Read about *initial* |
| inherit | Inherits this property from its parent element. |

# word-wrap property

- The word-wrap property allows long words to be able to be broken and wrap onto the next line.

<u>Syntax:</u>

*word-wrap: normal|break-word|initial|inherit;*

## values

- Normal: Break words only at allowed break points
- break-word:Allows unbreakable words to be broken
- Initial:Sets this property to its default value.
- inherit:Inherits this property from its parent element.

## word-spacing property

- The word-spacing property increases or decreases the white space between words

Syntax:

*word-spacing: normal|length|initial|inherit;*

-

- Normal:Defines normal space between words (0.25em) .
- *Length:*Defines an additional space between words (in px, pt, cm, em, etc). Negative values are allowed.

```
p.a {
  word-spacing: normal;
}

p.b {
  word-spacing: 30px;
}

p.c {
  word-spacing: -1cm;
}
```

# CSS text-align-last Property

- The text-align-last property specifies how to align the last line of a text.

- text-align-last: *auto|left|right|center|justify|start|end|initial|inherit;*

## Multi Background

- CSS Multi background property is used to add one or more images at a time without HTML code, We can add images as per our requirement.

```
#multibackground
{ background-image: url(/css/images/logo.png),
   url(/css/images/border.png);
background-position: left top, left top;
background-size: 50px,180px;
 background-repeat: no-repeat, repeat;
 padding: 75px;
}
```

## Font property

Web fonts are used to allows the fonts in CSS, which are not installed on local system.Different web fonts format are:

- **TrueType Fonts (TTF)**
- **OpenType Fonts (OTF)**
- **The Web Open Font Format (WOFF)**
- **SVG Fonts/Shapes**
- **Embedded OpenType Fonts (EOT)**

Different prperties in css3 are:

- @font-face
- Font-size-adjust
- Font-stretch

<u>Syntax:</u>

```
@font-face
{
Font-properties
}
```

# Fonts description

- The following list contained all the fonts description which are placed in the @font-face rule –
- **font-family:**Used to defines the name of font
- **Src:**Used to defines the URL
- **font-stretch:**Used to find, how font should be stretched(values: condensed or expanded)
- **font-style:**Used to defines the fonts style

(font-style: normal|italic|oblique|initial|inherit;)

- **font-weight:**Used to defines the font weight(boldness)

(font-weight:

normal|bold|bolder|lighter|*number*|initial|inherit;)

# HTML5
# (Graphics)

## GRAPHICS

- HTML5 having two main graphic elements:

- Canvas
- SVG

## Canvas

The HTML <canvas> element is used to draw graphics on a web page. the <canvas> element is only a container for graphics. You must use the javascript for canvas

It shows four elements:

- a red rectangle
- a gradient rectangle
- a multicolor rectangle
- a multicolor text.

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid
    #d3d3d3;">
browser does not support the HTML5 canvas tag.
</canvas>
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.moveTo(0,0);
ctx.lineTo(200,100);
ctx.stroke();

</script>
</body>
</html>
```

# Draw Circle

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
 browser does not support the HTML5 canvas tag
</canvas>
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
ctx.stroke();
</script>

</body>
</html>
```

The beginPath() method begins a path, or resets the current path.

- Use moveTo(), lineTo(), quadricCurveTo(), bezierCurveTo(), arcTo(), and arc(), to create paths.
- Use the stroke() method to actually draw the path on the canvas.

- ctx.arc(x, y, radius, startAngle, endAngle, counterClockwise);

- In above example:

ctx.arc(95,50,40,0,2*Math.PI);

## Draw a Text

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
 browser does not support the HTML5 canvas tag.</canvas>

<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.font = "30px Arial";
ctx.fillText("Hello World",10,50);
</script>

</body>
</html>
```

```
var c =
document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.font = "30px Arial";
ctx.strokeText("Hello World", 10, 50);
```

```
<!DOCTYPE html>
<html>
<body>

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
// Create gradient
var grd = ctx.createLinearGradient(0,0,200,0);
grd.addColorStop(0,"red");
grd.addColorStop(1,"white");
// Fill with gradient
ctx.fillStyle = grd;
ctx.fillRect(10,10,150,80);
</script>

</body>
</html>
```

- The createLinearGradient() method creates a linear gradient object.

createLinearGradient(*x0,y0,x1,y1*)

- *X0:*The x-coordinate of the start point of the gradient
- *Y0:*The y-coordinate of the start point of the gradient
- *X1:*The x-coordinate of the end point of the gradient
- *Y1:*The y-coordinate of the end point of the gradient

- The gradient can be used to fill rectangles, circles, lines, text, etc.

Use this object as the value to the strokeStyle or fillStyle properties.

Use the addColorStop() method to specify different colors, and where to position the colors in the gradient object.

*gradient*.addColorStop(*offset*, *color*);

- Offset:A number between 0 and 1, inclusive, representing the position of the color stop.
- 0 represents the start of the gradient and 1 represents the end;
- Color reprsents the color

-
  ```
  var c = document.getElementById('myCanvas');
  var ctx = c.getContext('2d');

  var grd = ctx.createLinearGradient(0, 0, 170, 0);
  grd.addColorStop(0, "black");
  grd.addColorStop(1, "white");

  ctx.fillStyle = grd;
  ctx.fillRect(20, 20, 150, 100);
  ```

# Draw Image in canvas

```
<!DOCTYPE html>
<html>
<body>
<p>Image to use:</p>
<img id="scream" src="img_the_scream.jpg" alt="The Scream" width="220" height="277">
<p>Canvas to fill:</p>
<canvas id="myCanvas" width="250" height="300"
style="border:1px solid #d3d3d3;">
browser does not support the HTML5 canvas tag.</canvas>
<p><button onclick="myCanvas()">Try it</button></p>
<script>
function myCanvas() {
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  var img = document.getElementById("scream");
  ctx.drawImage(img,10,10);
}</script></body></html>
```

- void *ctx*.drawImage(*image*, *dx*, *dy*);
- void *ctx*.drawImage(*image*, *dx*, *dy*, *dWidth*, *dHeight*);
- void *ctx*.drawImage(*image*, *sx*, *sy*, *sWidth*, *sHeight*, *dx*, *dy*, *dWidth*, *dHeight*);

| *img* | **Specifies the image, canvas, or video element to use** |
|---|---|
| *sx* | Optional. The x coordinate where to start clipping |
| *sy* | Optional. The y coordinate where to start clipping |
| *swidth* | Optional. The width of the clipped image |
| *sheight* | Optional. The height of the clipped image |
| *x* | The x coordinate where to place the image on the canvas |
| *y* | The y coordinate where to place the image on the canvas |
| *width* | Optional. The width of the image to use (stretch or reduce the image) |

## SVG

- SVG is  Scalable Vector Graphics.
- <svg> element is a container for SVG graphics.

## SVG Circle

```
<!DOCTYPE html>
  <html>
  <body>

  <svg width="100" height="100">
    <circle cx="50" cy="50" r="40" stroke="green"
stroke-width="4" fill="yellow" />
  </svg>

  </body>
  </html>
```

- The cx and cy attributes define the x and y coordinates of the center of the circle.
- If cx and cy are omitted, the circle's center is set to (0,0)
- The r attribute defines the radius of the circle
- Stroke represents the border
- Fill attributes represents the color to fill in circle

## SVG Rectangle

```
<svg width="400" height="110">
<rect width="300" height="100" style="fill:rgb(0
,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
</svg>
```

- The width and height attributes of the <rect> element define the height and the width of the rectangle
- The style attribute is used to define CSS properties for the rectangle
- The CSS fill property defines the fill color of the rectangle
- The CSS stroke-width property defines the width of the border of the rectangle
- The CSS stroke property defines the color of the border of the rectangle

# Example-2(Rectangle)

```
<svg width="400" height="180“>

 <rect x="50" y="20" width="150" height="150"
    style="fill:blue;stroke:pink;stroke-width:5;fill-
   opacity:0.1;stroke-opacity:0.9" />

</svg>
```

- The x attribute defines the left position of the rectangle (e.g. x="50" places the rectangle 50 px from the left margin)

- The y attribute defines the top position of the rectangle (e.g. y="20" places the rectangle 20 px from the top margin)

- The CSS fill-opacity property defines the opacity of the fill color (legal range: 0 to 1)

- The CSS stroke-opacity property defines the opacity of the stroke color (legal range: 0 to 1)

```
<svg width="400" height="180">
<rect x="50" y="20" width="150" height="150"
    style="fill:blue;stroke:pink;stroke-
  width:5;opacity:0.5" />
  </svg>
```

# Rectangle with round corners

```
<svg width="400" height="180">
  <rect x="50" y="20" rx="20" ry="20" width="
 150" height="150"
  style="fill:red;stroke:black;stroke-
 width:5;opacity:0.5" />
 </svg>
```

## SVG Ellipse

```
<svg height="140" width="500">

  <ellipse cx="200" cy="80" rx="100" ry="50"
  style="fill:yellow;stroke:purple;stroke-
width:2" />

</svg>
```

- The cx attribute defines the x coordinate of the center of the ellipse
- The cy attribute defines the y coordinate of the center of the ellipse
- The rx attribute defines the horizontal radius
- The ry attribute defines the vertical radius

## Line, Polygon

```
<svg height="210" width="500">
   <line x1="0" y1="0" x2="200" y2="200" style
  ="stroke:rgb(255,0,0);stroke-width:2" />
  </svg>
```

```
<svg height="210" width="500">
   <polygon points="200,10 250,190
  160,210" style="fill:lime;stroke:purple;stroke-
  width:1" />
  </svg>
```

# Star

```
<svg height="210" width="500">
  <polygon points="100,10 40,198 190,78 10,78
160,198"
  style="fill:lime;stroke:purple;stroke-width:5;fill-
rule:nonzero;" />
</svg>
```

Fill-rule having two values: nonzero(default), evenodd

## polyline

```
<svg height="200" width="500">
  <polyline points="20,20 40,25 60,40 80,120
120,140 200,180"
  style="fill:none;stroke:black;stroke-
width:3" />
</svg>
```

## SVG-Path

The <path> element is used to define a path.
The following commands are available for path data:
M = moveto
L = lineto
H = horizontal lineto
V = vertical lineto
C = curveto
S = smooth curveto
Q = quadratic Bézier curve
T = smooth quadratic Bézier curveto
A = elliptical Arc
Z = closepath

-
  ```
  <svg height="210" width="400">
    <path d="M150 0 L75 200 L225 200 Z" />
  </svg>
  ```

## SVG-text

```
<svg height="30" width="200">
    <text x="0" y="15" fill="red">I love SVG!</text>
  </svg>
```

```
<svg height="60" width="200">
    <text x="0" y="15" fill="red" transform="rotate(
  30 20,40)">I love SVG</text>
  </svg>
```

# <tspan>

- The <text> element can be arranged in any number of sub-groups with the <tspan> element.

Text on several lines (with the <tspan> element):

- Several lines:
- First line.
- Second line.

## 5

```
<svg height="90" width="200">
    <text x="10" y="20" style="fill:red;">Several
  lines:
      <tspan x="10" y="45">First line.</tspan>
      <tspan x="10" y="70">Second line.</tspan>
    </text>
  </svg>
```

## Text as a link (with the <a> element)

```
<svg height="30" width="200" xmlns:xlink="http://
   www.w3.org/1999/xlink">
     <a xlink:href="https://www.w3schools.com/gra
   phics/" target="_blank">
       <text x="0" y="15" fill="red">I love
   SVG!</text>
     </a>
   </svg>
```

# SVG Stroke Properties

- stroke
- stroke-width
- stroke-linecap
- stroke-dasharray

```
<svg height="80" width="300">
    <g fill="none">
      <path stroke="red" d="M5 20 l215 0" />
<path stroke-width="2" d="M5 20 l215 0" />
<path stroke-linecap="butt" d="M5 20 l215 0" />
<path stroke-linecap="round" d="M5 40 l215 0" />
<path stroke-linecap="square" d="M5 60 l215 0" />
 </g>
   </svg>
```

```
<svg height="80" width="300">
   <g fill="none" stroke="black" stroke-width="4">
    <path stroke-dasharray="5,5" d="M5 20 l215
  0" />
    <path stroke-dasharray="10,10" d="M5 40 l215
  0" />
    <path stroke-
  dasharray="20,10,5,5,5,10" d="M5 60 l215 0" />
   </g>
  </svg>
```

# Blog Name

## TITLE HEADING

**Title description, Dec 7, 2017**

Image

Some text..

Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco.

## TITLE HEADING

**Title description, Sep 2, 2017**

Image

## About Me

Image

Some text about me in culpa qui officia deserunt mollit anim..

## Popular Post

Image

Image

Image

Some text..

Sunt in culpa qui officia deserunt mollit anim id est laborum consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco.

## Follow Me

Some text..

## Footer