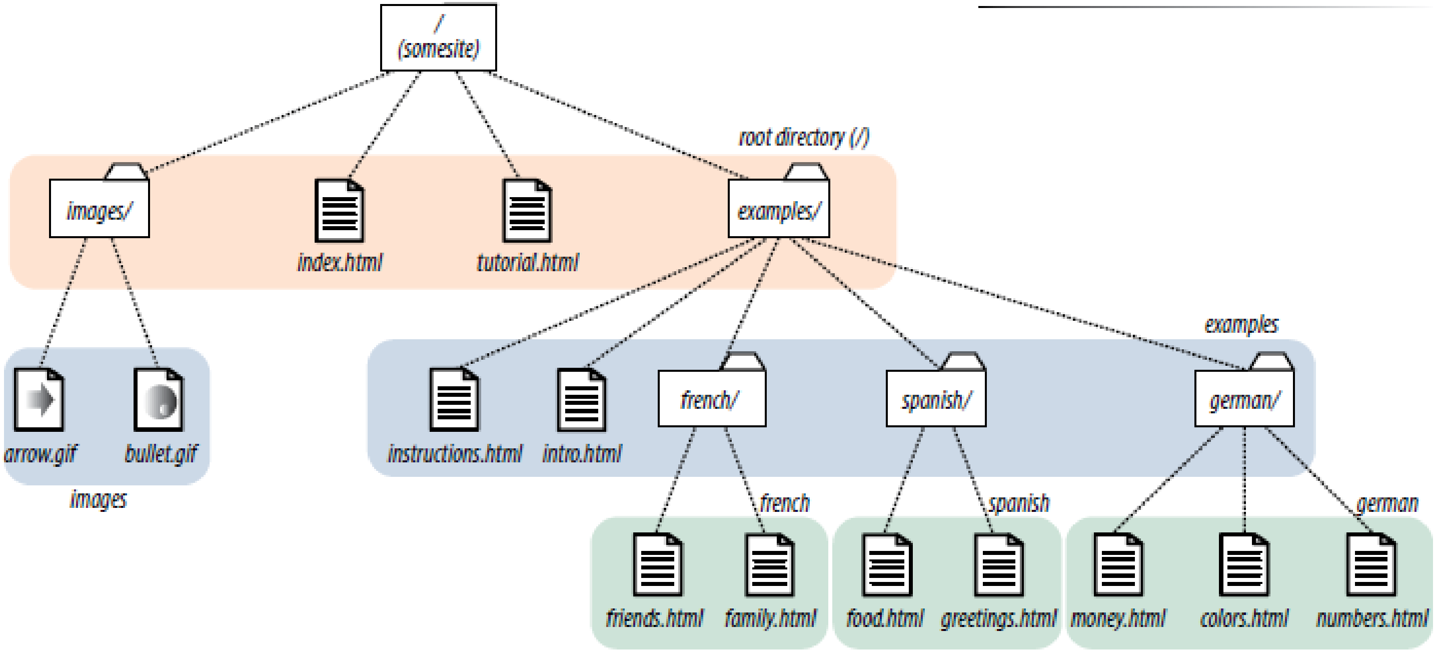
# CHAPTER 3 – HTML PT2: MULTIMEDIA & TABLES

# CHECK YOU OUT

Login to your LMS and Chap3 forum, look at the diagram below:



# TOPICS TO BE COVERED

- **HTML Semantic Elements**

  Study of the meanings of words and phrases in a language
- **HTML Tables**

  Providing a means of organizing the layout of data.
- **HTML Multimedia**

  Inserting Images and Videos unto a webpage.
- **HTML SVG**

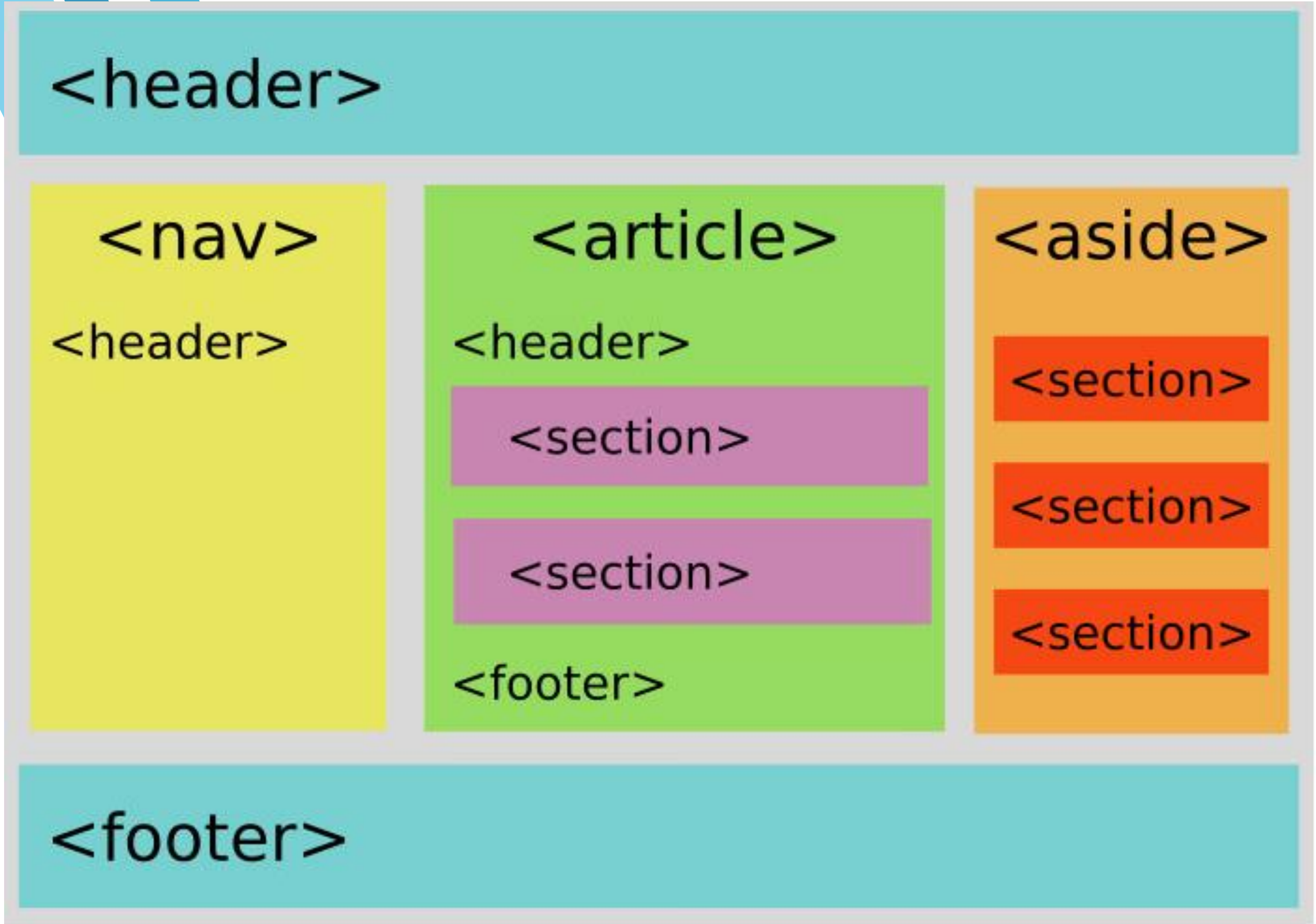
  Covering the basic 2D graphics to be used.

**More info on this topic can be found in e-books provided in LMS.**
Presentation slides made by Fifah S., M. Khalid.

# SEMANTIC ELEMENTS

- HTML5 introduced new elements that give semantic meaning to sections of a typical web page or application, which are:
  - Sections (**section**)
  - Articles (**article**)
  - Navigation (**nav**)
  - Tangentially related content (**aside**)
  - Headers (**header**)
  - Footers (**footer**)

- According to the W3C, a Semantic Web: “Allows data to be shared and reused across applications, enterprises and communities.”



<header>

<nav>

<header>

<article>

<header>

<section>

<section>

<footer>

<aside>

<section>

<section>

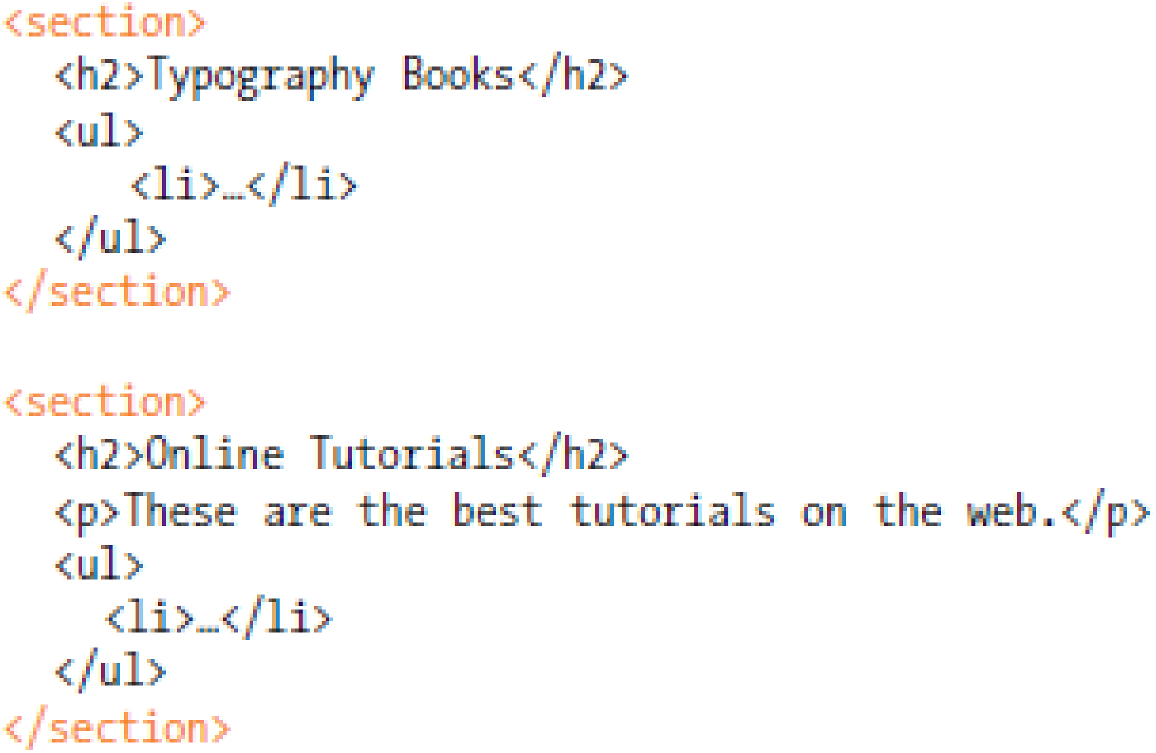
<section>

<footer>

# SECTIONS

- Long documents are easier to use when they are divided into smaller parts, like chapters in a book.
- To divide long web documents into thematic sections, use the **section** element.
- Sections typically have a heading and other content that has a meaningful reason to be grouped together.
- N/B. But if the purpose of the groupings the element is to provide a hook for styling, use **div** element instead.

# SECTIONS



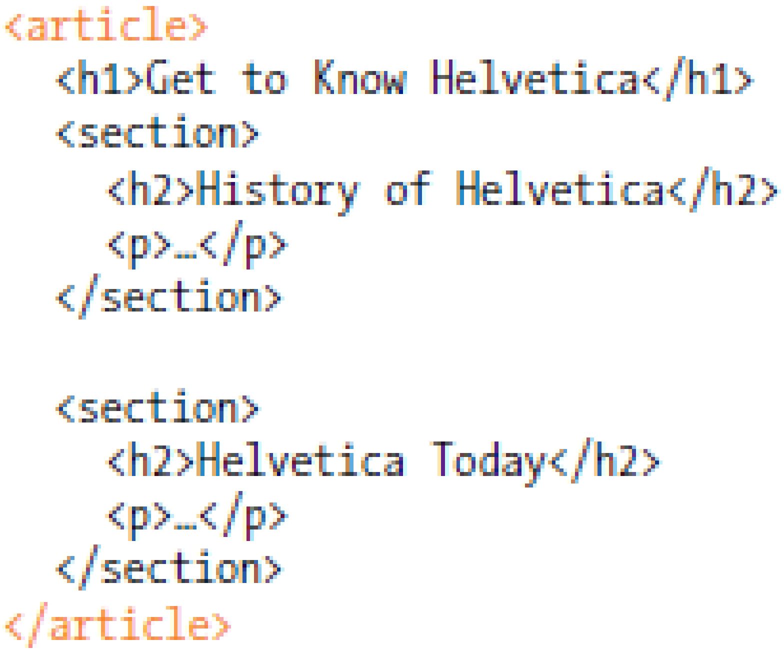
```
<section>
  <h2>Typography Books</h2>
  <ul>
      <li>…</li>
  </ul>
</section>

<section>
  <h2>Online Tutorials</h2>
  <p>These are the best tutorials on the web.</p>
  <ul>
     <li>…</li>
  </ul>
</section>
```

# ARTICLES

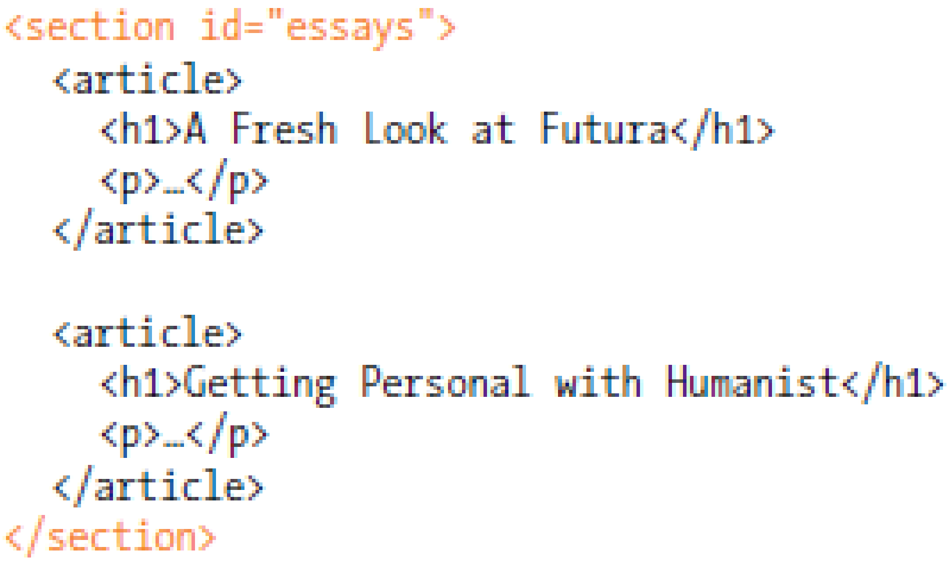
- Use **article** element for self-contained works that could stand alone or be reused in a different context.
- It is useful for magazine or newspaper aticles, blog posts, comments or other items that could be extracted for external use.
- It could be either broken by sections or a section of a web comprised of number of articles.

# SECTIONS AND ARTICLES



```
<article>
  <h1>Get to Know Helvetica</h1>
  <section>
    <h2>History of Helvetica</h2>
    <p>…</p>
  </section>

  <section>
    <h2>Helvetica Today</h2>
    <p>…</p>
  </section>
</article>
```



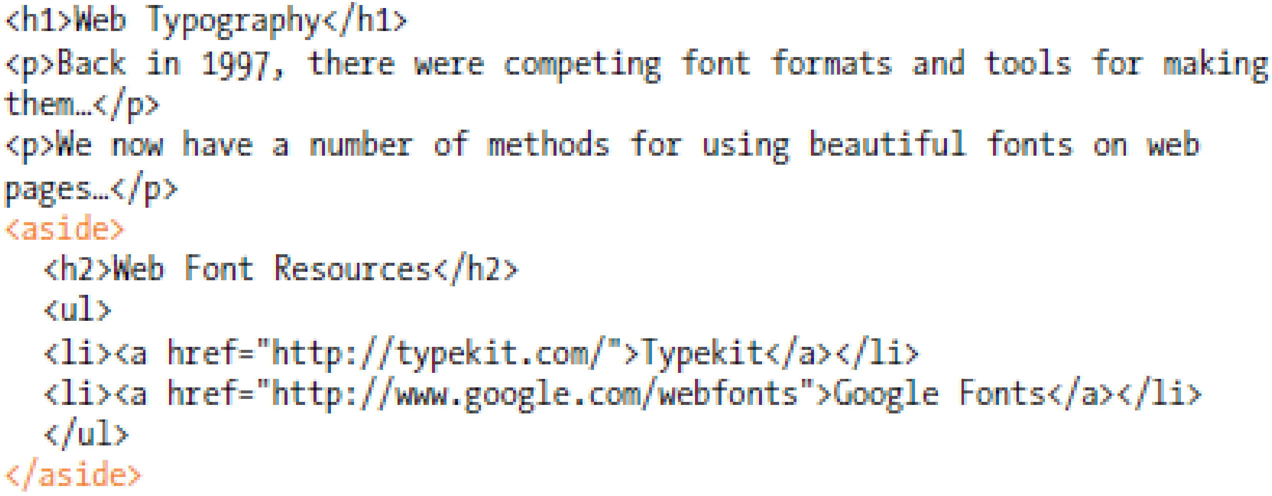
```
<section id="essays">
  <article>
    <h1>A Fresh Look at Futura</h1>
    <p>…</p>
  </article>

  <article>
    <h1>Getting Personal with Humanist</h1>
    <p>…</p>
  </article>
</section>
```

# SIDEBARS (ASIDE)

- The **aside** element identifies content that is related but tangential to the surrounding content – like sidebar.
- It can be used to pull quotes, background information, list of links, callouts or anything else that might be associated with a document.
- It has no default rendering, so you will need to make it a block element and adjust its appearance and layout with style sheet rules.

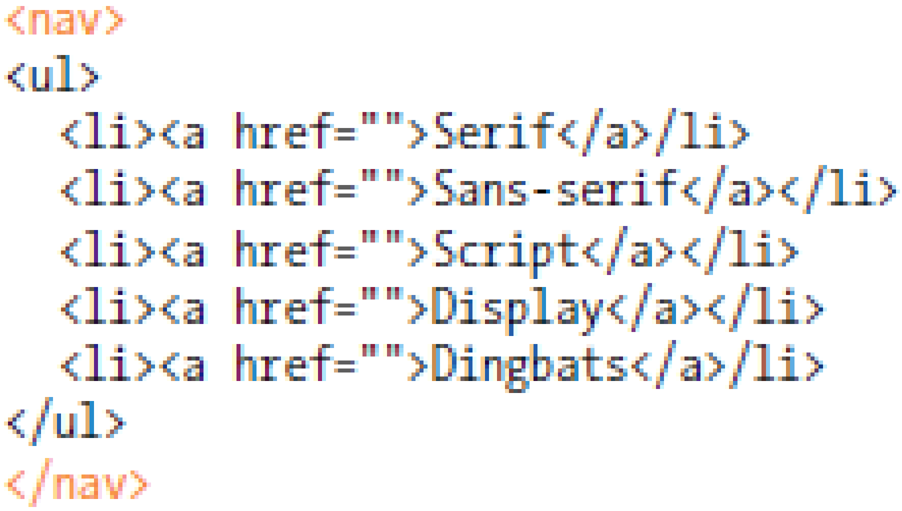
# ASIDE



```
<h1>Web Typography</h1>
<p>Back in 1997, there were competing font formats and tools for making
them…</p>
<p>We now have a number of methods for using beautiful fonts on web
pages…</p>
<aside>
  <h2>Web Font Resources</h2>
  <ul>
  <li><a href="http://typekit.com/">Typekit</a></li>
  <li><a href="http://www.google.com/webfonts">Google Fonts</a></li>
  </ul>
</aside>
```

# NAVIGATION

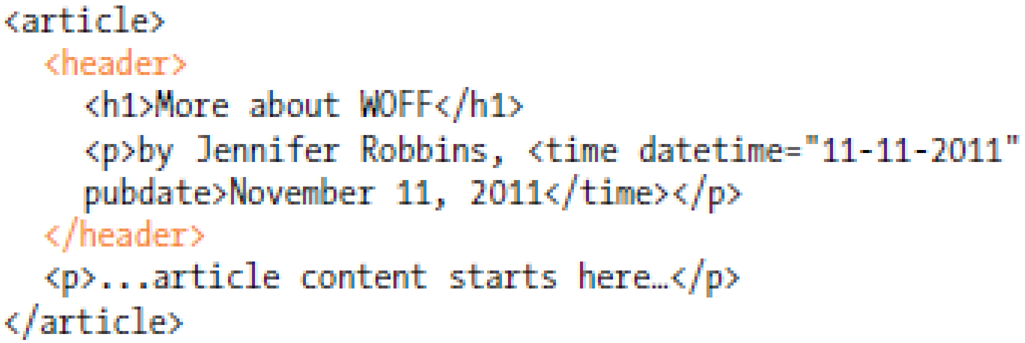
» The **nav** element gives developers a semantic way to identify navigation for a site.



```
<nav>
<ul>
   <li><a href="">Serif</a>/li>
   <li><a href="">Sans-serif</a></li>
   <li><a href="">Script</a></li>
   <li><a href="">Display</a></li>
   <li><a href="">Dingbats</a>/li>
</ul>
</nav>
```

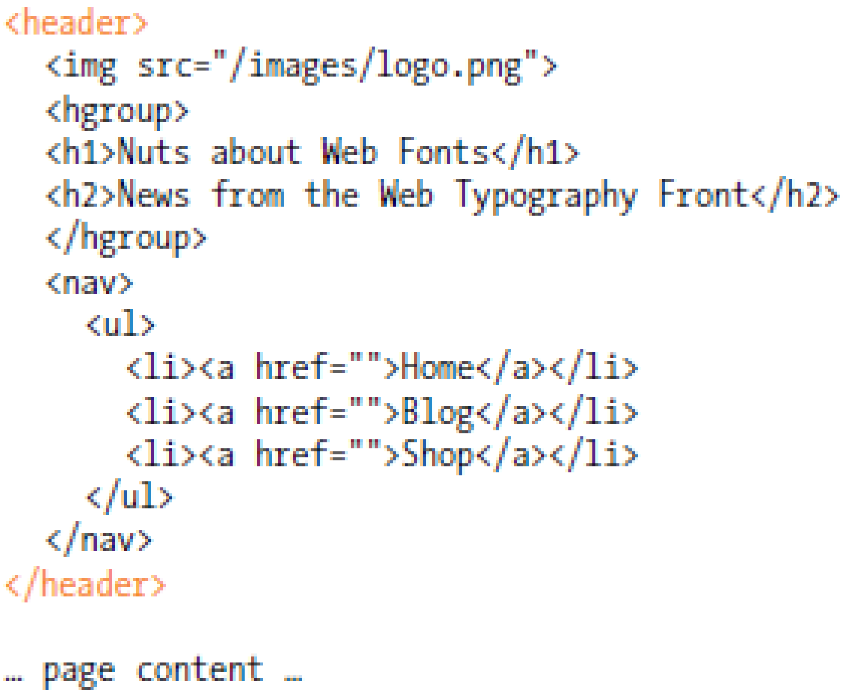
# HEADERS

- The **header** element is used for inductor material that typically appears at the beginning of a web page or at the top of a section / article.
- There is no specified list of what a header must contain; anything that makes sense as the introduction to a page or section is acceptable.



```
<article>
  <header>
    <h1>More about WOFF</h1>
    <p>by Jennifer Robbins, <time datetime="11-11-2011"
    pubdate>November 11, 2011</time></p>
  </header>
  <p>...article content starts here…</p>
</article>
```

# HEADERS



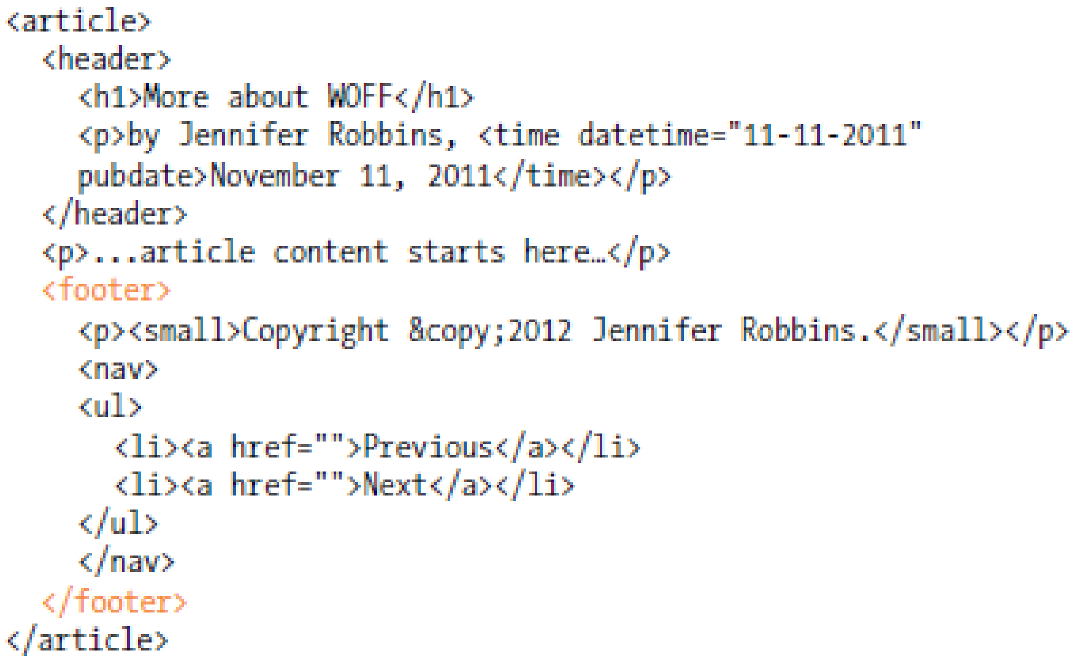
```
<header>
  <img src="/images/logo.png">
  <hgroup>
  <h1>Nuts about Web Fonts</h1>
  <h2>News from the Web Typography Front</h2>
  </hgroup>
  <nav>
    <ul>
      <li><a href="">Home</a></li>
      <li><a href="">Blog</a></li>
      <li><a href="">Shop</a></li>
    </ul>
  </nav>
</header>

… page content …
```

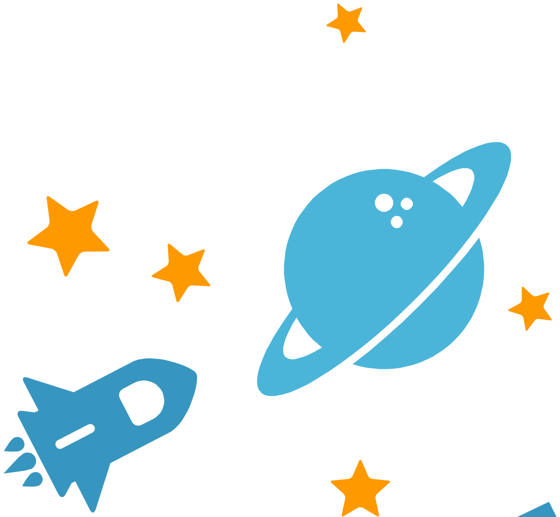
# FOOTERS

- The **footer** element is used to indicate the type of information that typically comes at the end of a page or an article, such as:
  - Author
  - Copyright information
  - Related documents
  - Navigation
- The element may apply to the entire document, or it could be associated with a particular section or article.
- Note that though it is called footer, there is no requirement that it come last in the document.

# FOOTERS



```
<article>
  <header>
    <h1>More about WOFF</h1>
    <p>by Jennifer Robbins, <time datetime="11-11-2011"
    pubdate>November 11, 2011</time></p>
  </header>
  <p>...article content starts here…</p>
  <footer>
    <p><small>Copyright &copy;2012 Jennifer Robbins.</small></p>
    <nav>
    <ul>
      <li><a href="">Previous</a></li>
      <li><a href="">Next</a></li>
    </ul>
    </nav>
  </footer>
</article>
```



# HTML TABLES

Providing a means of organizing the layout of data.

# TABLES

- Tables provide a means of organizing the layout of data.
- A table is divided into rows and columns.
  - Rows – Horizontal
  - Columns – Vertical

- Tables should **not** be used for layout.
  - Use CSS *floats and positioning* styles instead.

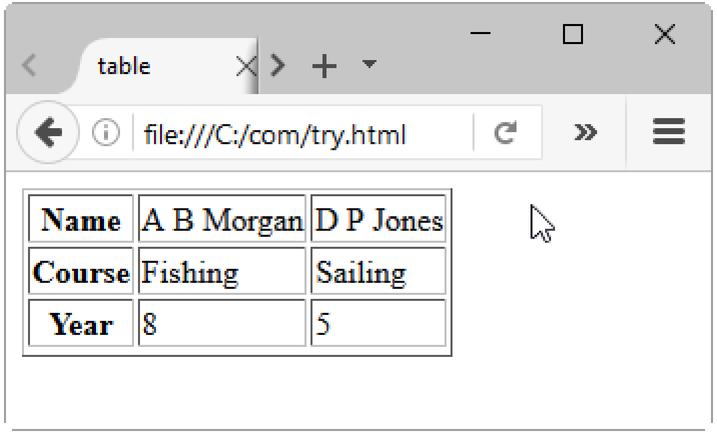
# TABLES

<table> - main element

<th> - table header

<td> - table data

<tr> - table row

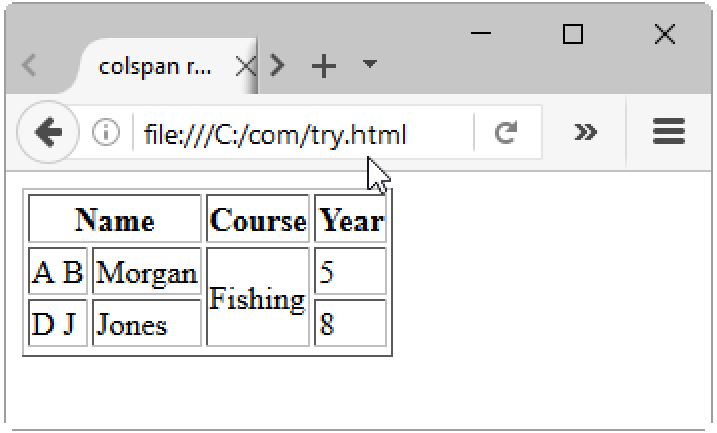


| | | |
|---|---|---|
| **Name** | A B Morgan | D P Jones |
| **Course** | Fishing | Sailing |
| **Year** | 8 | 5 |

```
<table border="1">
 <tr>
  <th>Name</th>
  <td>A B Morgan</td>
  <td>D P Jones</td>
 </tr>
 <tr>
  <th>Course</th>
  <td>Fishing</td>
  <td>Sailing</td>
</tr>
 <tr>
  <th>Year</th>
  <td>8</td>
  <td>5</td>
 </tr>
 <tr>
</table>
```

# ROWS AND COLUMNS

Cells can span multiple columns and multiple rows with the ***colspan*** and ***rowspan*** attributes.



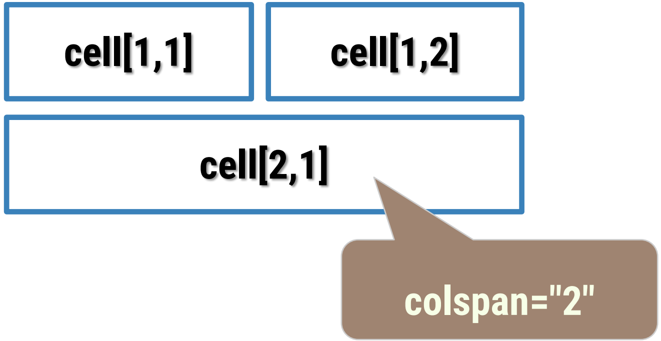
file:///C:/com/try.html

| Name | | Course | Year |
|---|---|---|---|
| A B | Morgan | Fishing | 5 |
| D J | Jones | | 8 |

```
<table border="1">
 <tr>
  <th colspan="2">Name</th>
  <th>Course</th>
  <th>Year</th>
 </tr>
 <tr>
  <td>A B</td>
  <td>Morgan</td>
  <td rowspan="2">Fishing</td>
  <td>5</td>
 </tr>
 <tr>
  <td>D J</td>
  <td>Jones</td>
  <td>8</td>
 </tr>
 <tr>
</table>
```

# ROWSPAN AND COLSPAN

- colspan



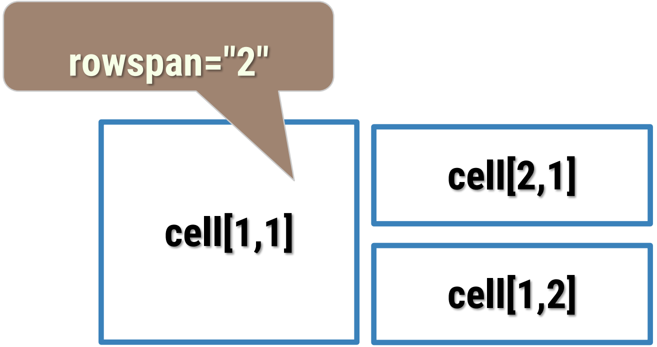
cell[1,1] cell[1,2]

cell[2,1]

colspan="2"

Defines how many columns the cell occupies

- Rowspan



rowspan="2"

cell[1,1]

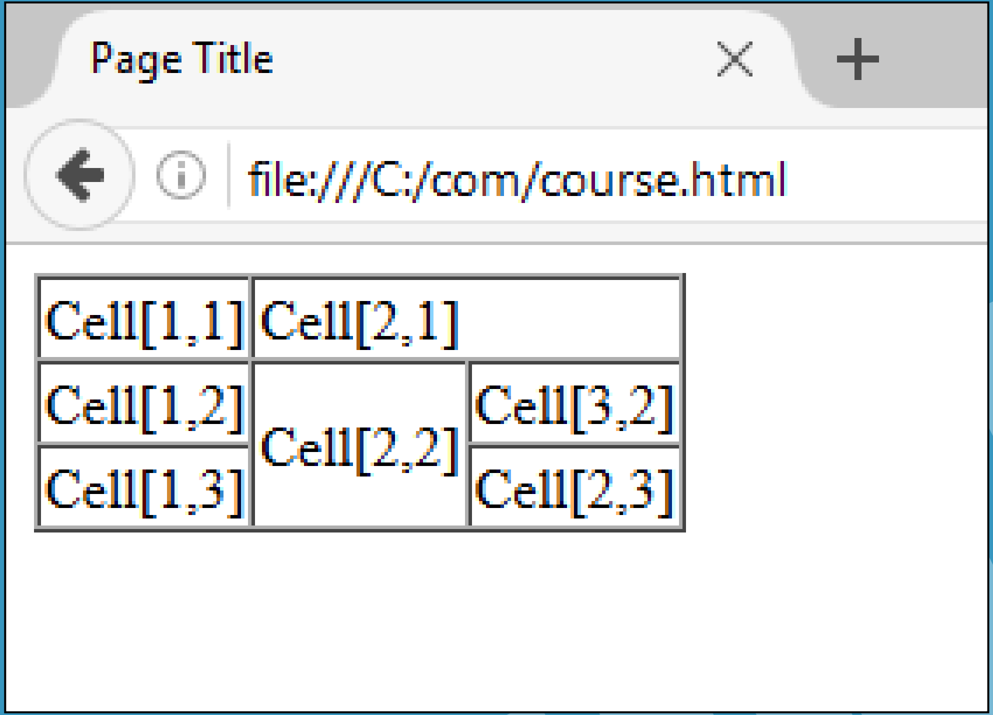
cell[2,1]

cell[1,2]

Defines how many rows the cell occupies.

# EXERCISE

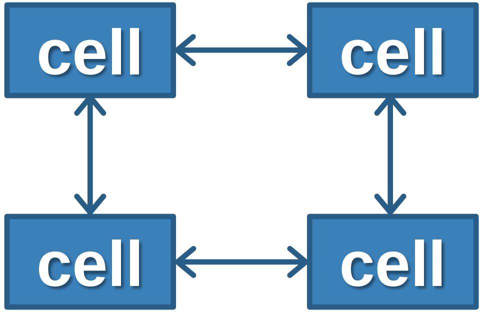
Create a web page as follows using Text Editor.



<table>
<tr><td>Cell[1,1]</td><td colspan="2">Cell[2,1]</td></tr>
<tr><td>Cell[1,2]</td><td rowspan="2">Cell[2,2]</td><td>Cell[3,2]</td></tr>
<tr><td>Cell[1,3]</td><td>Cell[2,3]</td></tr>
</table>

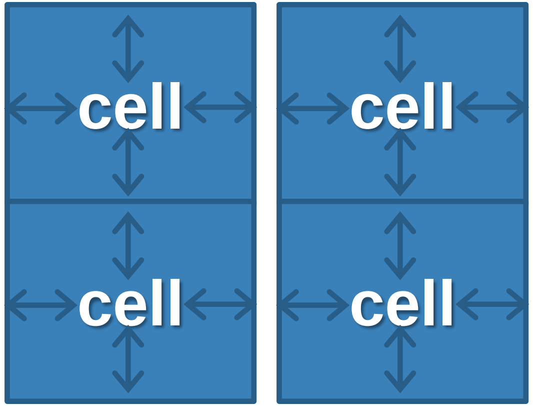
# CELLSPACING & CELLPADDING

- cellspacing



Defines the empty space between cells.

- cellpadding



Defines the empty space around the cell content.

# EXAMPLE

```
<html>
  <head><title>Table Cells</title></head>
  <body>
    <table cellspacing="15" cellpadding="0" border="0">
      <tr><td>First</td>
      <td>Second</td></tr>
    </table>
    <br/>
    <table cellspacing="0" cellpadding="10" border="0">
      <tr><td>First</td><td>Second</td></tr>
    </table>
  </body>
</html>
```

# ALIGN AND WIDTH ATTRIBUTES

- The **align** attribute determines *the position of the text within a cell.*

- The **width** attribute determines *the width of the row relative to the table.*

```
<table border="1" align="center">
 <tr>
  <th colspan="2" width="60%">Name</th>
  <th rowspan="2">Course</th>
  <th rowspan="2">Year</th>
 </tr>
 <tr>
  <th>Last</th>
  <th>Init.</th>
 </tr>
 <tr>
  <td>Morgan</td>
  <td>AB</td>
  <td>Fishing</td>
  <td align="center">5</td>
 </tr>
 </table>
```

# TABLE CHALLENGE!

Create your timetable using a table element to be shown in web browsers.

Take into accounts, the rowspan and colspan properties.

# HTML MULTIMEDIA

Inserting Images and Videos unto a webpage.

# IMAGES

- Images are included using the empty tag **<img>**.

  **<img src="mypicture.gif" alt="my picture">**

- Attribute = src, alt
  - src – specifies the file containing the image
    - To external website http://
    - To the local database
  - alt – specifies the text to be displayed if the image is not viewed.
    - Some users choose not to display images, for faster download.

# IMAGES FORMATTING

**For image formatting, CSS is highly recommended.**

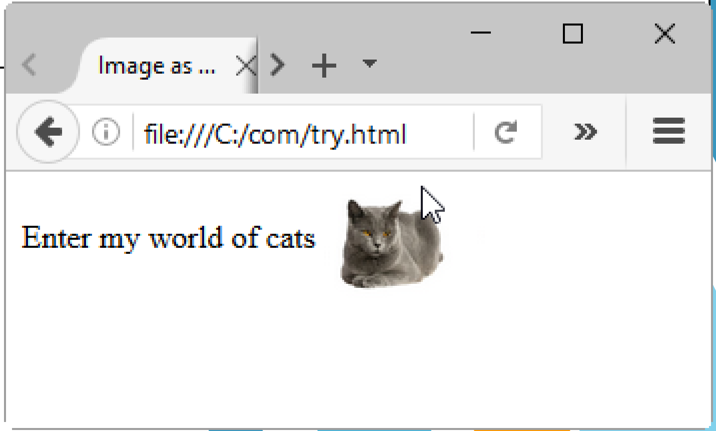
```
<img src=“cat.gif” height=“60” width=“90” align=“top” alt=“cat>
```

- Attribute: height, width, align.
  - The size attributes control the size of the image.
  - The align attribute controls the location of the image, relative to the line of text.
    - align=“top”
    - align=“middle”
    - align=“bottom”
    - align=“left”
    - align=“right”

# LINKS WITH IMAGES

- A link element can include an image instead of text.
  - Both images and text can be included if required.

```
<body>
Enter my world of cats <a href="cats.html"><img src="cat.jpg" height="60"
width=“80" align="middle" alt="cat"></a>
</body>
```



Enter my world of cats

# VIDEOS

- Videos are included using the empty tag **<video>**.

  **<video src=“myvideo.mp4” width=“320” height=“240” controls>**

- The current HTML5 draft specification does not specify which video formats browsers should support in the video tag. But most commonly used video formats are:
  - **Ogg –** Ogg files with Theodora video codec and Vorbis audio codec.
  - **Mpeg4 –** MPEG4 files with H.264 video codec and AAC audio codec.

# VIDEOS

- You can use <source> tag to specify media along with media type and many other attributes.

```
<!DOCTYPE html>
<html>
...
<body>
    <video width=“300” height=“200” controls autoplay>
        <source src=“/html5/intro.ogg” type=“video/ogg” />
        <source src=“/html5/intro.mp4” type=“video/mp4” />
    </video>
```

| Attribute | Description |
|---|---|
| autoplay | This video will automatically begin to play back as soon as it can do so without stopping to finish loading the data. |
| autobuffer | This video will automatically begin buffering even if it’s not set to automatically play. |
| controls | Allow the user to control video playback including volume, seeking, and pause/resume playback. |
| height | Specifies the height of the video’s display area, in CSS pixels. |
| loop | This will allow video automatically seek back to the start after reaching the end. |
| preload | The video will be loaded at page load, and ready to run. Ignored if autoplay is present. |
| Poster | A URL of an image to show until the user plays or seeks. |
| src | URL of the video to embed. |
| width | The width of the video’s display area, in CSS pixels. |

# AUDIO

- Audios are included using the empty tag **<audio>**.

  **<audio src="myaudio.mp3" controls autoplay>**

- The current HTML5 draft specification does not specify which audio formats browsers should support in the audio tag. But most commonly used video formats are:
  - **Ogg**
  - **Mp3**
  - **wav**

# VIDEOS

- You can use <source> tag to specify media along with media type and many other attributes.

```
<!DOCTYPE html>
<html>
…
<body>
    <audio controls autoplay>
        <source src=“/html5/audio.ogg” type=“video/ogg” />
        <source src=“/html5/audio.mp3” type=“video/mp3” />
    </audio>
```

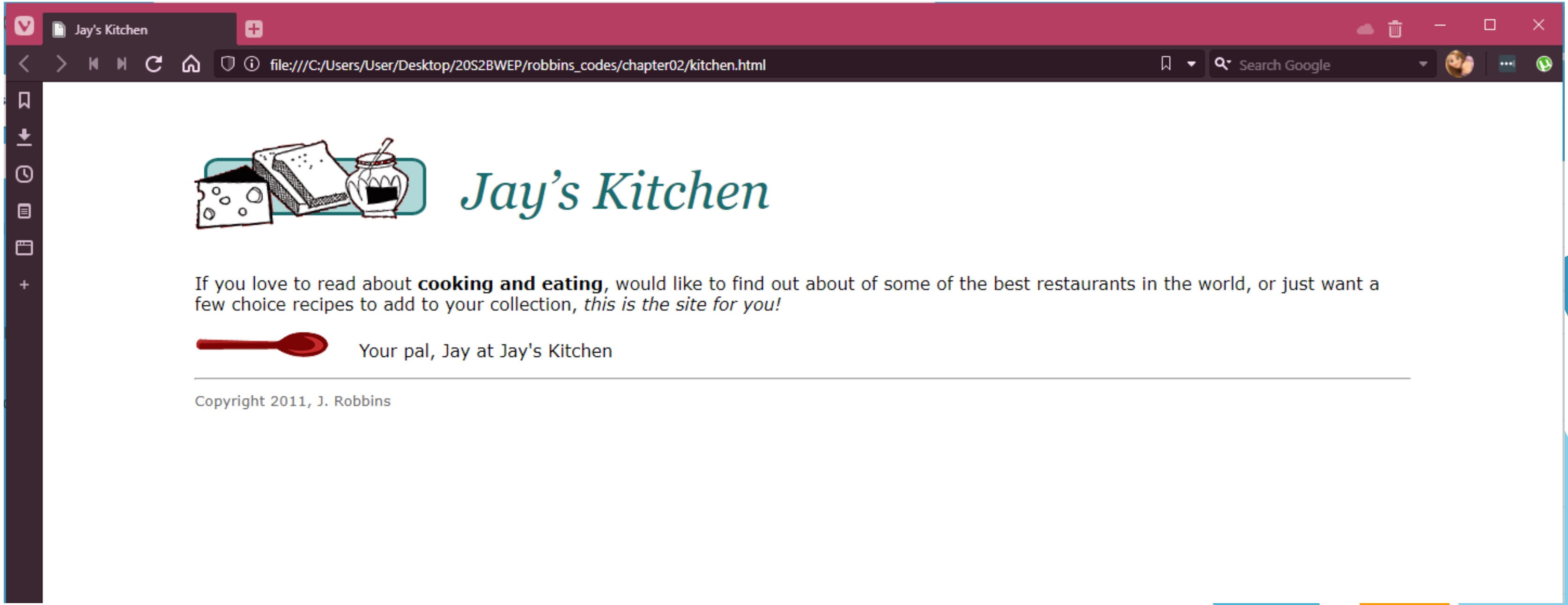
| Attribute | Description |
|---|---|
| autoplay | This audio will automatically begin to play back as soon as it can do so without stopping to finish loading the data. |
| autobuffer | This audio will automatically begin buffering even if it’s not set to automatically play. |
| controls | Allow the user to control audio playback including volume, seeking, and pause/resume playback. |
| loop | This will allow video automatically seek back to the start after reaching the end. |
| preload | The audio will be loaded at page load, and ready to run. Ignored if autoplay is present. |
| src | URL of the video to embed. |

# Activity! Try it out

Open the folder *Chap3* in your LMS and do the following:

- Add gif spoon and foods into the website.
- Create and link CSS file to index.html.
- Modify:
    1. All fonts should be in Verdana.
    2. H1 font should be in Georgia italic with size 3em and color hexacode 23, 109, 109.
    3. The footer should be in color #666.

# It should look something like this:



Jay's Kitchen
file:///C:/Users/User/Desktop/20S2BWEP/robbins_codes/chapter02/kitchen.html
Search Google

Jay's Kitchen

If you love to read about **cooking and eating**, would like to find out about of some of the best restaurants in the world, or just want a few choice recipes to add to your collection, *this is the site for you!*

Your pal, Jay at Jay's Kitchen

Copyright 2011, J. Robbins

# HTML SVG

Scalable Vector Graphics

# SVG

- It is a language for describing 2D-graphics and graphical application in XML, and then be rendered by an SVG viewer.
- Mostly useful for vector type diagrams like **Pie Charts, 2D graphic with x-y coordinates** etc.
- Most web browsers can display SVG just like they displayed images. IE users may have to install the **Adobe SVG Viewer**.
- Sample syntax:

```
<svg>
…
</svg>
```

# SVG ADVANTAGES

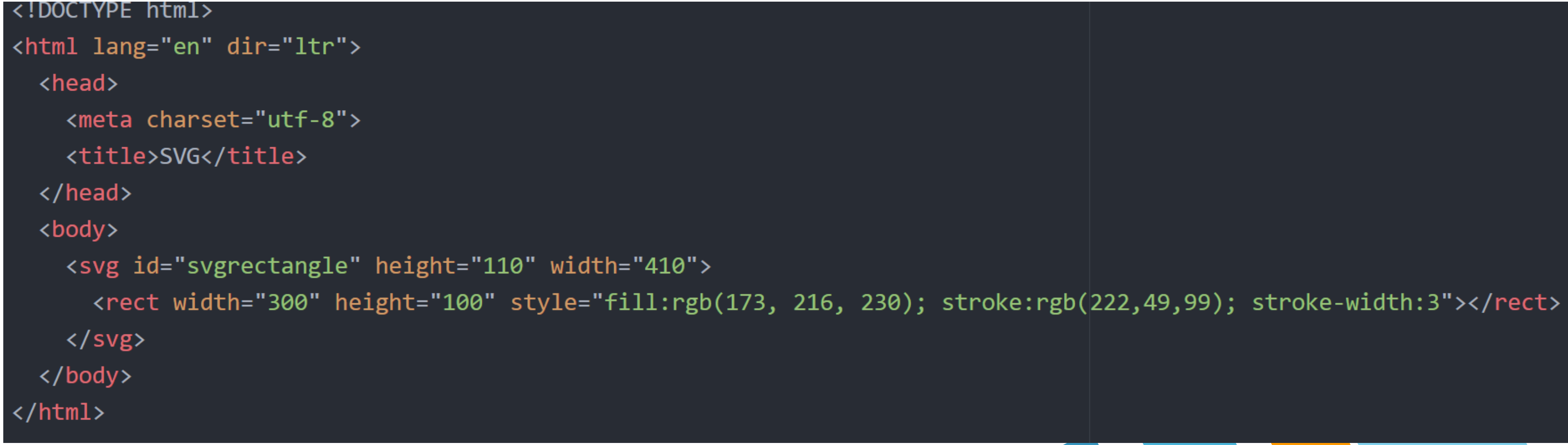
- Can be created and edited with any text editor.
- Can be searched, indexed, scripted and compressed.
- Are scalable.
- Can be printed with high quality at any resolution.
- Are zoomable.
- Do NOT lose any quality if they are zoomed or resized.
- Is an open standard.
- Are pure XML.

# SVG SHAPES

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

# SVG Rectangle

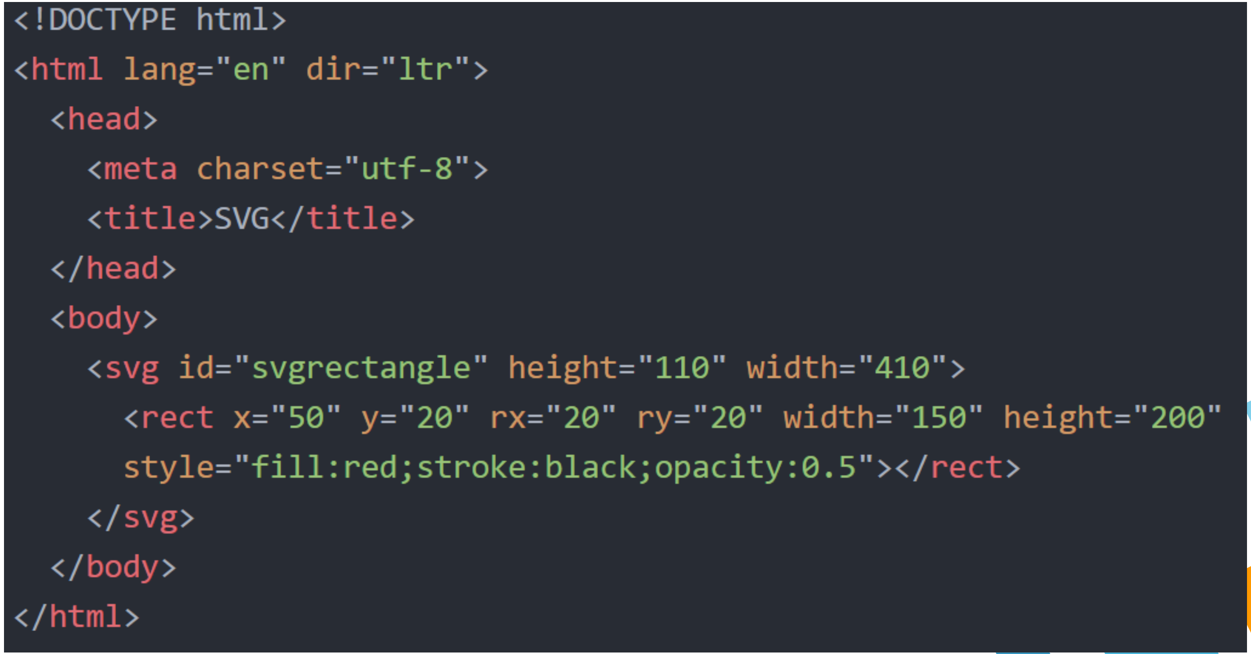
- Following is an example which would draw a rectangle using <rect> tag:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgrectangle" height="110" width="410">
      <rect width="300" height="100" style="fill:rgb(173, 216, 230); stroke:rgb(222,49,99); stroke-width:3"></rect>
    </svg>
  </body>
</html>
```

# SVG Rectangle

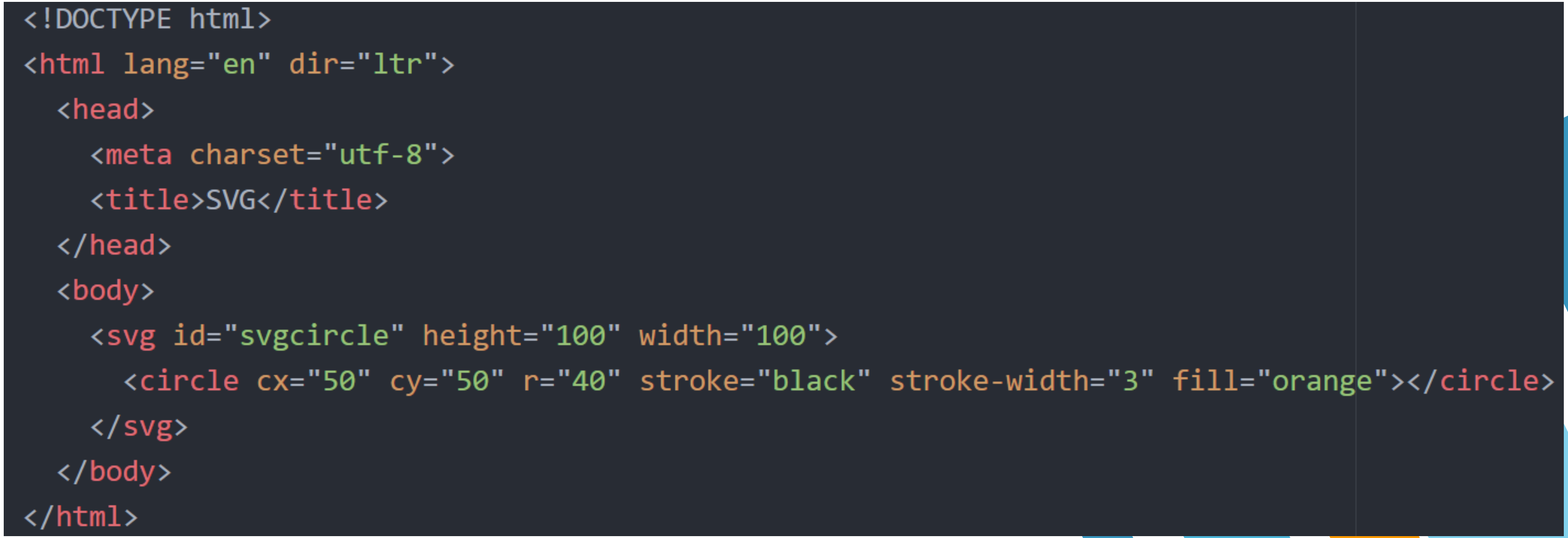
- Sample 2:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgrectangle" height="110" width="410">
      <rect x="50" y="20" rx="20" ry="20" width="150" height="200"
      style="fill:red;stroke:black;opacity:0.5"></rect>
    </svg>
  </body>
</html>
```

# SVG Circle

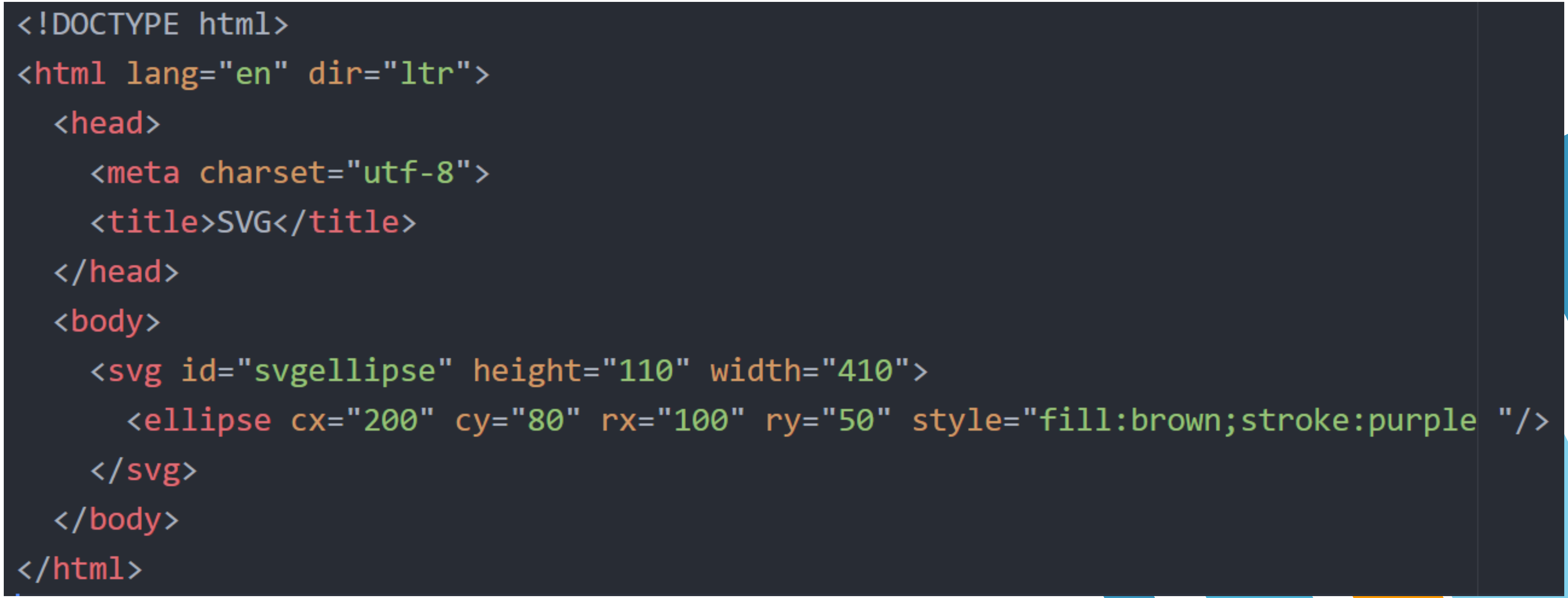
» Following is an example which would draw a circle using <circle> tag:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgcircle" height="100" width="100">
      <circle cx="50" cy="50" r="40" stroke="black" stroke-width="3" fill="orange"></circle>
    </svg>
  </body>
</html>
```

# SVG Ellipse

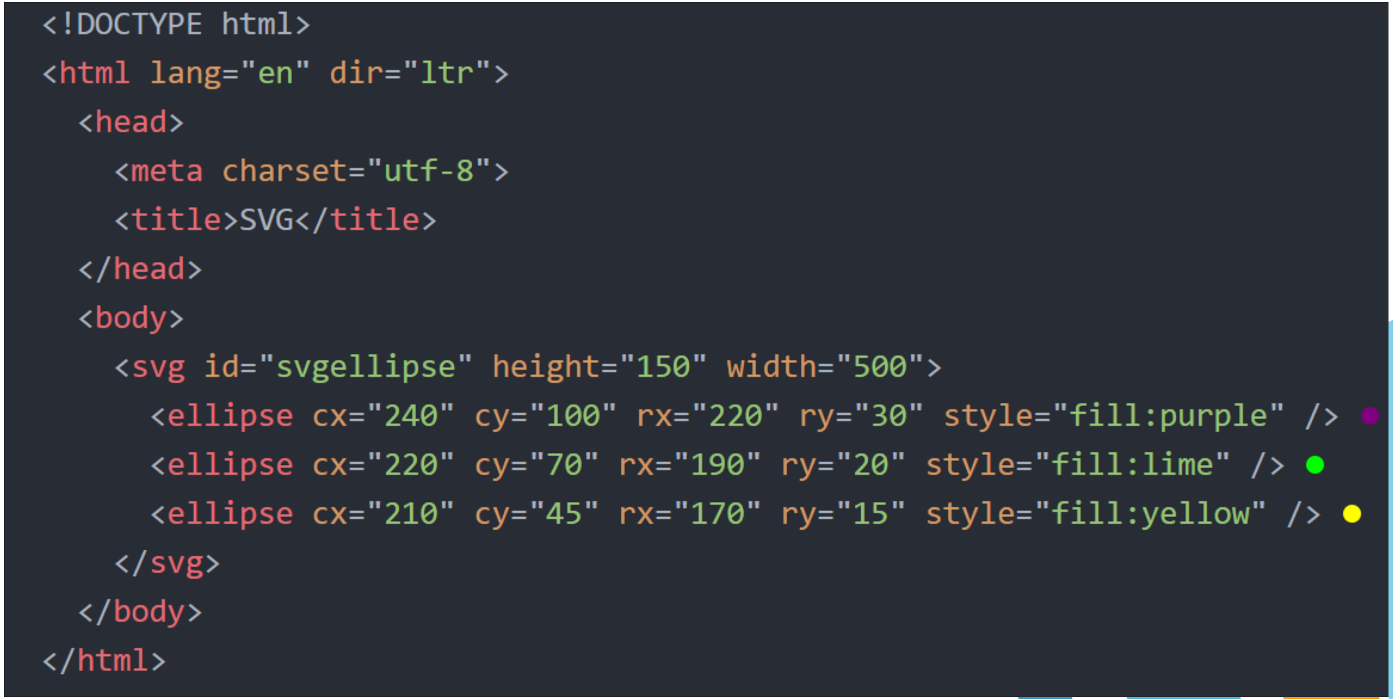
- Following is an example which would draw an ellipse (oval) using <ellipse> tag:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgellipse" height="110" width="410">
      <ellipse cx="200" cy="80" rx="100" ry="50" style="fill:brown;stroke:purple "/>
    </svg>
  </body>
</html>
```

# SVG Ellipse

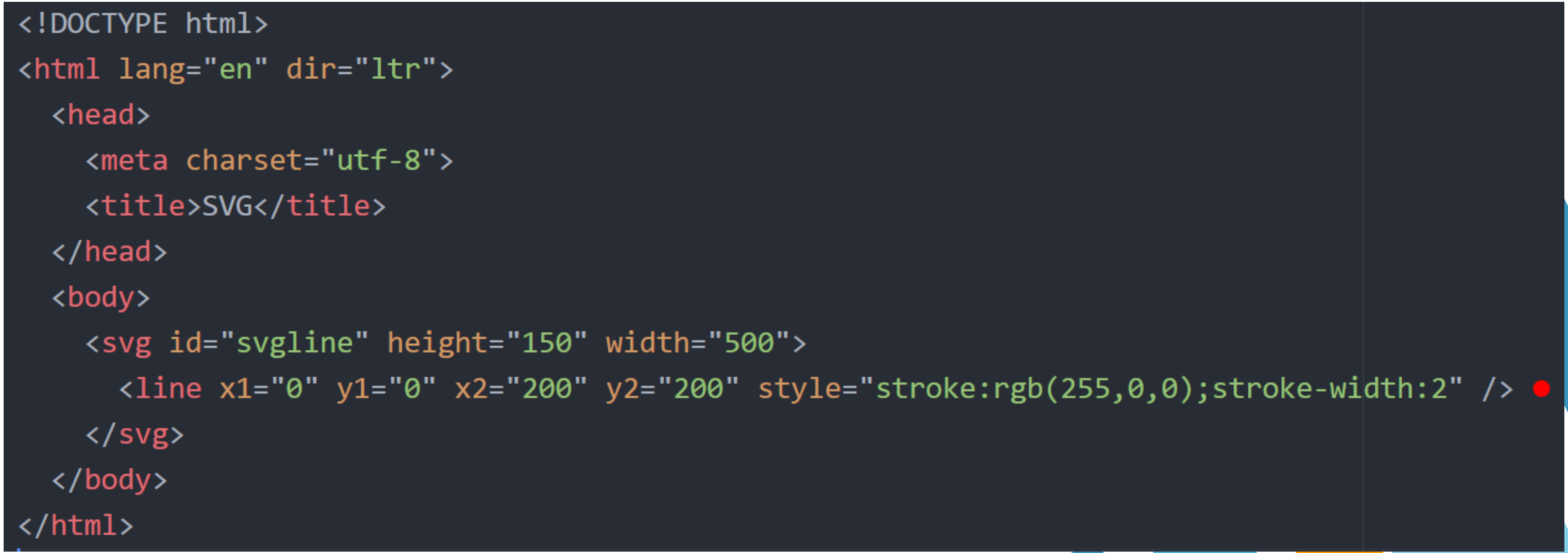
- Sample 2:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgellipse" height="150" width="500">
      <ellipse cx="240" cy="100" rx="220" ry="30" style="fill:purple" /> ●
      <ellipse cx="220" cy="70" rx="190" ry="20" style="fill:lime" /> ●
      <ellipse cx="210" cy="45" rx="170" ry="15" style="fill:yellow" /> ●
    </svg>
  </body>
</html>
```

# SVG Line

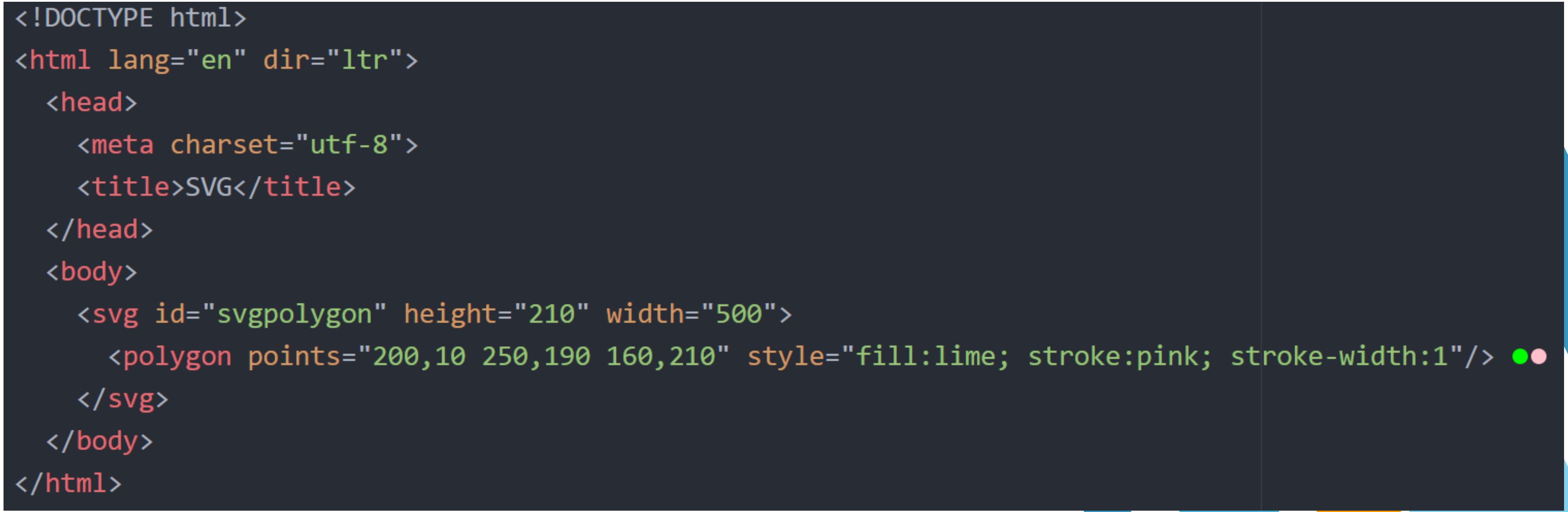
- Following is an example which would draw a line using <line> tag:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgline" height="150" width="500">
      <line x1="0" y1="0" x2="200" y2="200" style="stroke:rgb(255,0,0);stroke-width:2" />
    </svg>
  </body>
</html>
```

# SVG Polygon

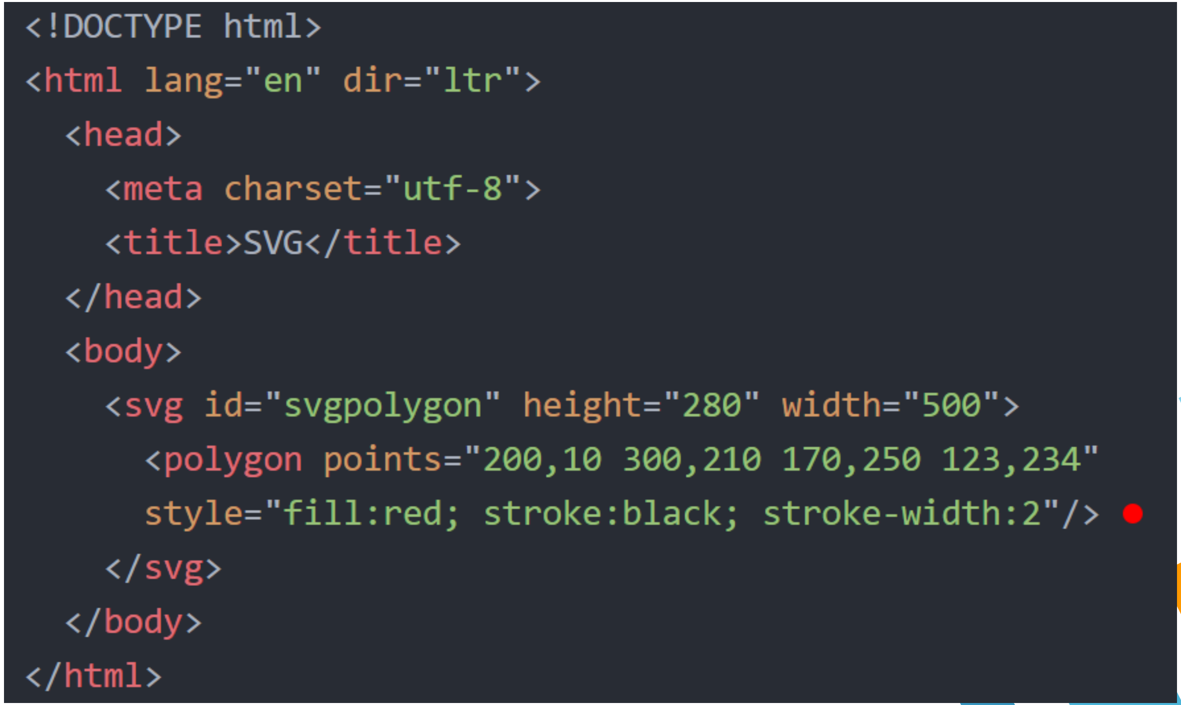
- Following is an example which would draw a polygon using <polygon> tag:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgpolygon" height="210" width="500">
      <polygon points="200,10 250,190 160,210" style="fill:lime; stroke:pink; stroke-width:1"/> ●●
    </svg>
  </body>
</html>
```

# SVG Polygon

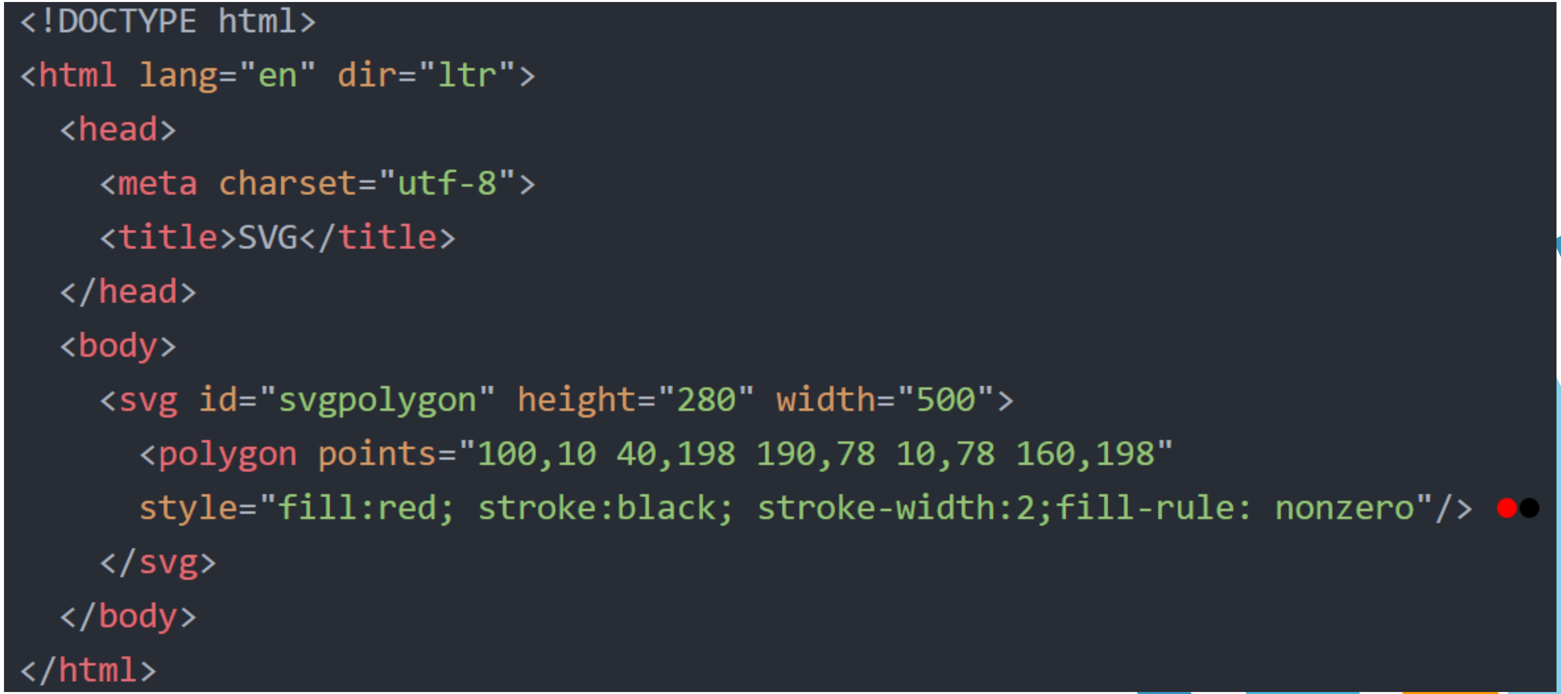
» Sample for creating 4 sided polygon:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgpolygon" height="280" width="500">
      <polygon points="200,10 300,210 170,250 123,234"
      style="fill:red; stroke:black; stroke-width:2"/>
    </svg>
  </body>
</html>
```

# SVG Polygon

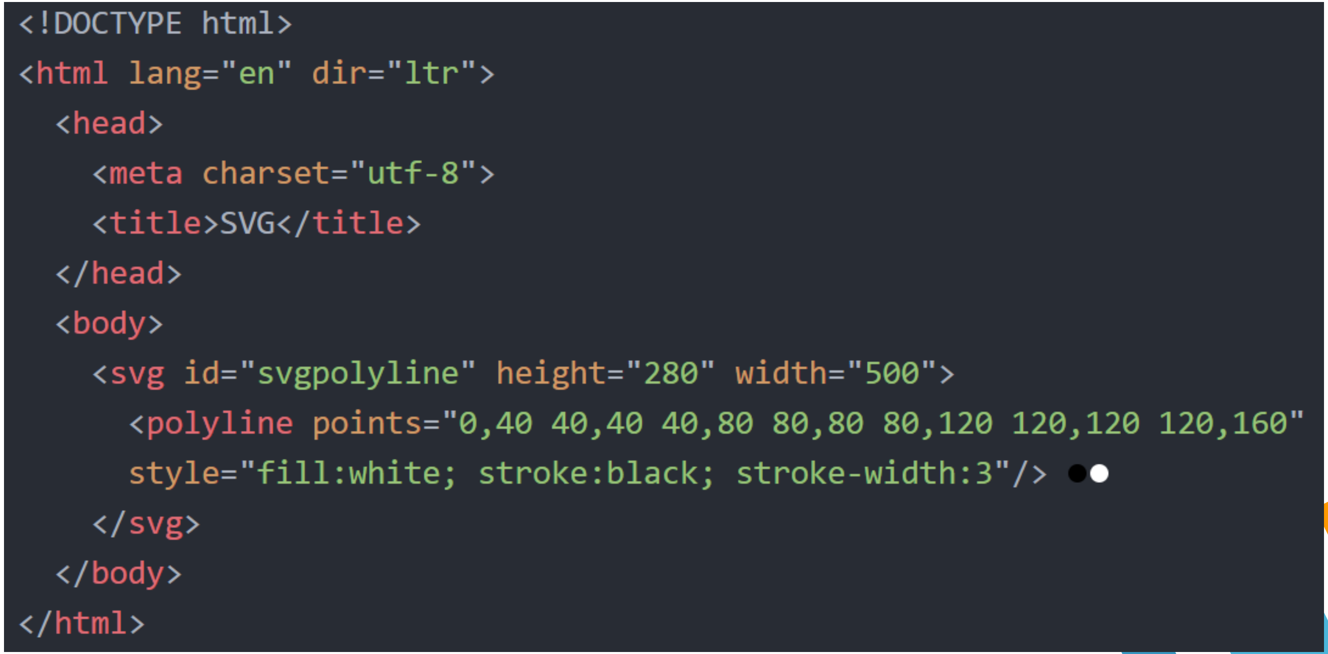
» Sample for creating a star:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgpolygon" height="280" width="500">
      <polygon points="100,10 40,198 190,78 10,78 160,198"
      style="fill:red; stroke:black; stroke-width:2;fill-rule: nonzero"/> ●●
    </svg>
  </body>
</html>
```

# SVG Polyline

» Following is an example which would draw a polyline using <polyline> tag:



```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>SVG</title>
  </head>
  <body>
    <svg id="svgpolyline" height="280" width="500">
      <polyline points="0,40 40,40 40,80 80,80 80,120 120,120 120,160"
      style="fill:white; stroke:black; stroke-width:3"/> ●○
    </svg>
  </body>
</html>
```