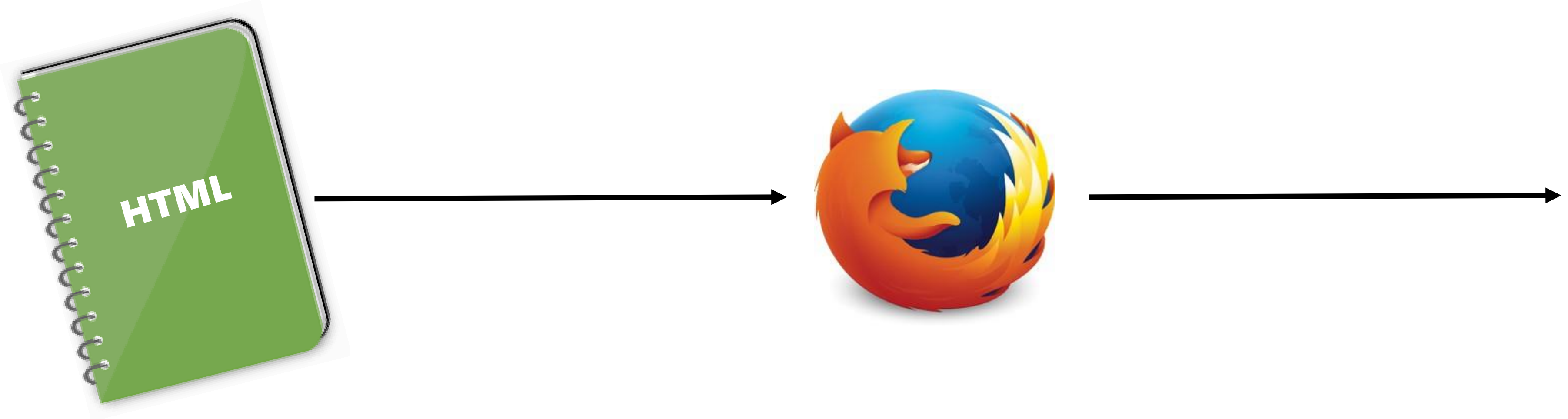
INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY

HYDERABAD

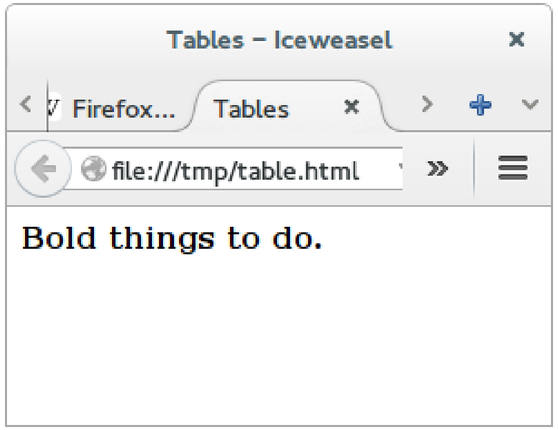
# Web Technologies

## HTML, CSS

# Web Browser



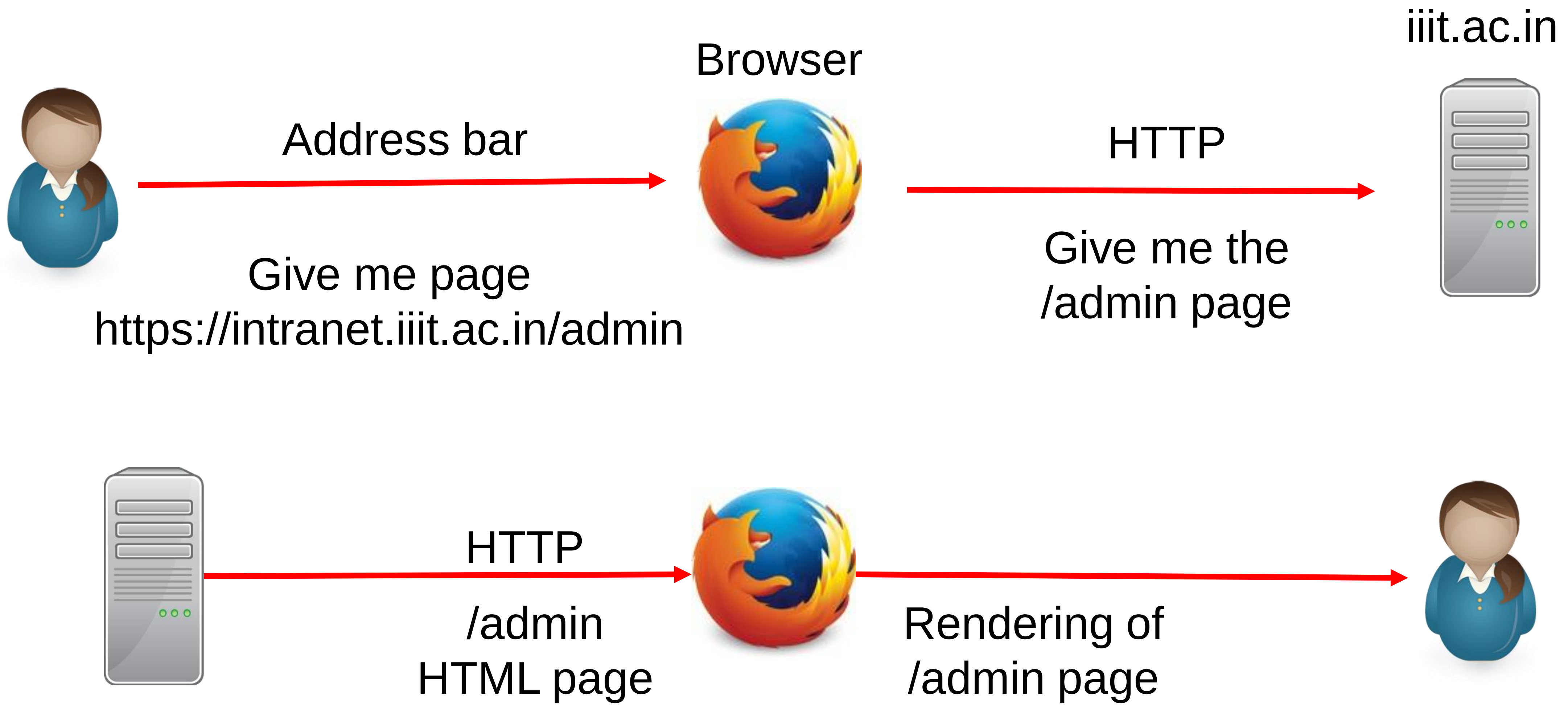
Instruction:
a paragraph
with bold text



**Bold things to do.**

Displays a paragraph
with bold text

# Web Page



Browser

iiit.ac.in

Address bar

Give me page
https://intranet.iiit.ac.in/admin

HTTP

Give me the
/admin page

HTTP

/admin
HTML page

Rendering of
/admin page

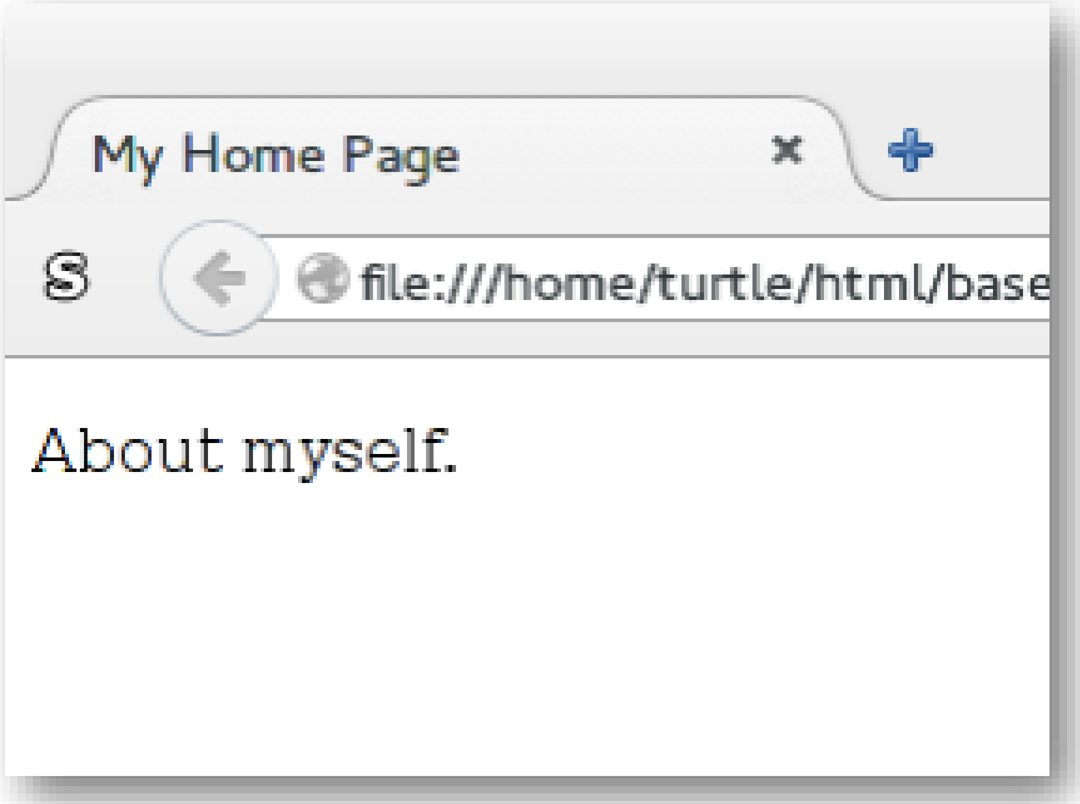
# HTML – Hypertext Markup Language

- Hyper Text Markup Language -lingua franca for publishing hypertext
- A simple text format to create web pages
- Specify structure, presentation and content of a web page
- Define tags <html><body> <head>….etc
- Allow to embed other scripting languages to manipulate design layout, text and graphics
- Platform independent
- No special software required. They are by default part of Internet browser application like Chrome, IE, Safari, Mozilla FireFox

You Write:

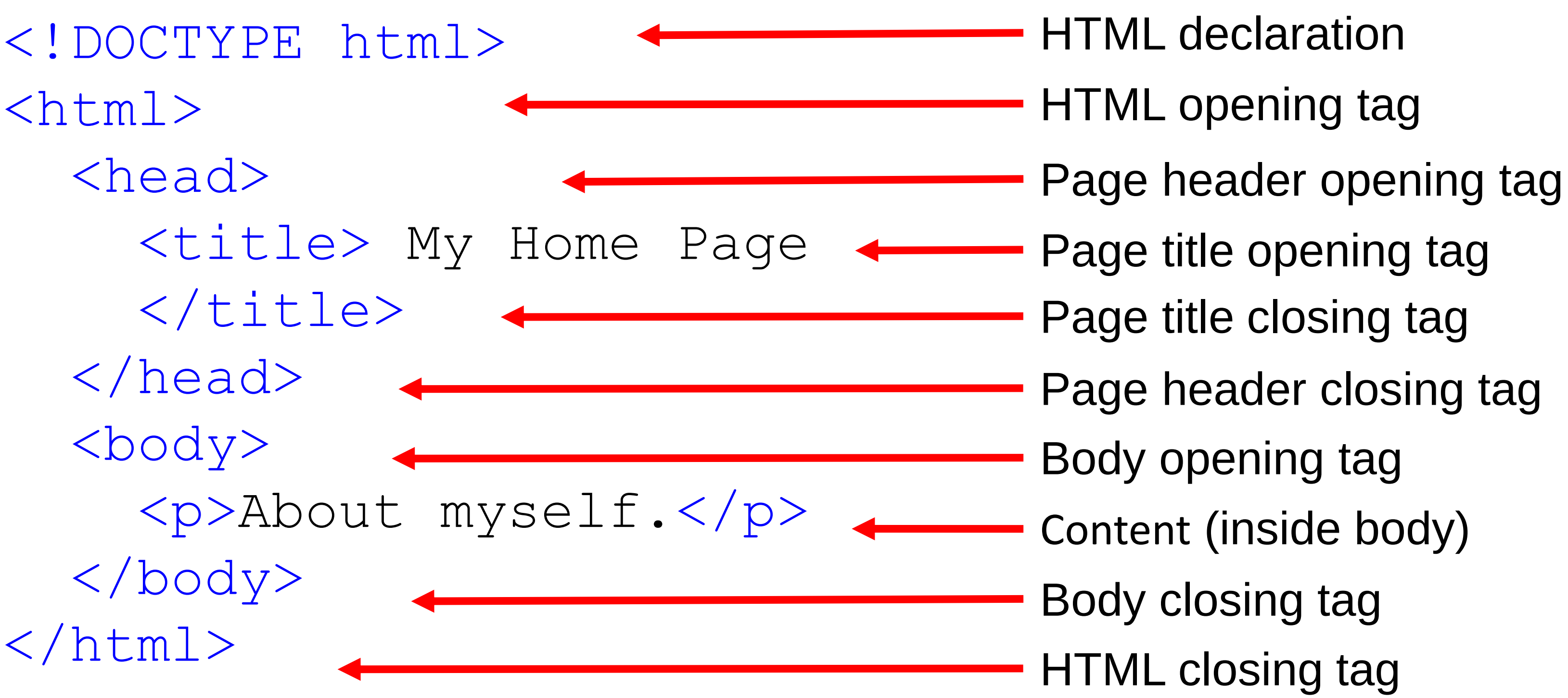
```
<!DOCTYPE html>
<html>
  <head>
    <title> My Home Page
    </title>
  </head>
  <body>
    <p>About myself.</p>
  </body>
</html>
```

You See:



About myself.

# Basic Structure



```
<!DOCTYPE html>
<html>
  <head>
    <title> My Home Page
    </title>
  </head>
  <body>
    <p>About myself.</p>
  </body>
</html>
```

HTML declaration

HTML opening tag

Page header opening tag

Page title opening tag

Page title closing tag

Page header closing tag

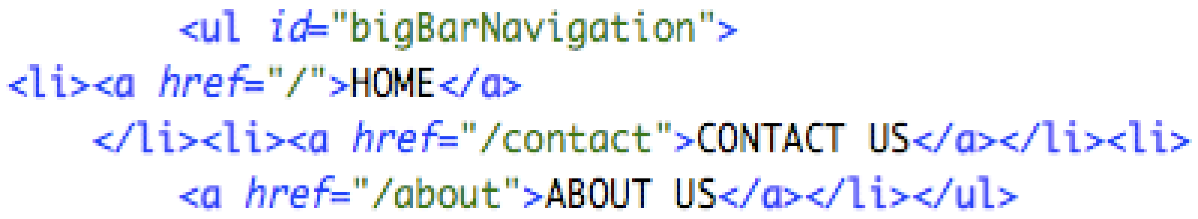
Body opening tag

Content (inside body)

Body closing tag

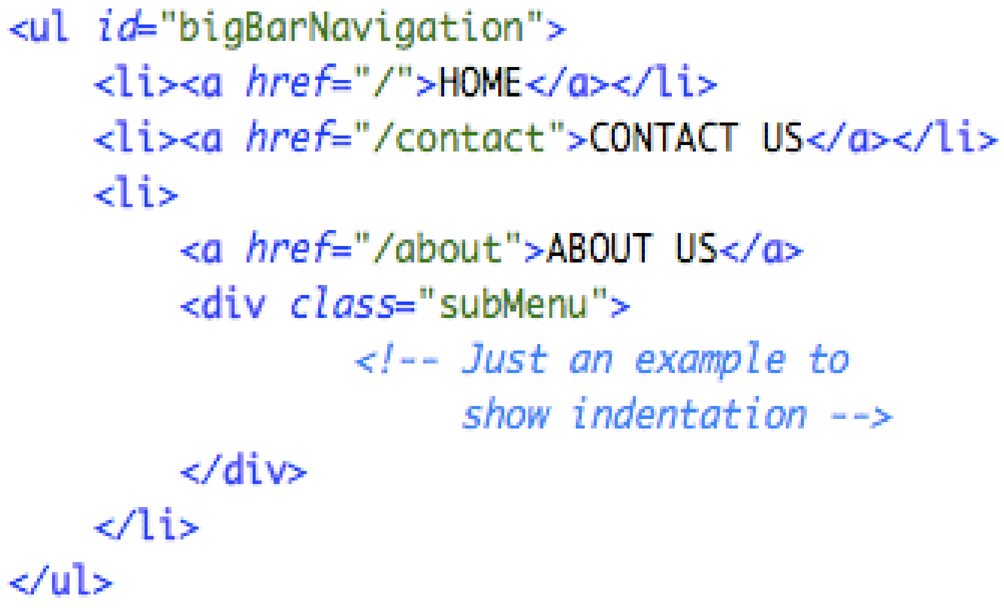
HTML closing tag

- Consider headers carefully, it helps to identify your page
- The title of a Web page determines its ranking in certain search engines
- The title will also appear on Favorite lists, History lists, and Bookmark lists to identify your page
- HTML Contents include paragraph, headings, hyperlinks, images, media, lists, tables, form
- HTML Tags are considered as markup to specify structure and style
- Opening tag **<tag>** and closing tag **</tag>,** Sometimes no closing tag Can have attributes to specify extra info **<html lang="en">**
- HTML Validator: http://validator.w3.org
- Checks your HTML page for errors
- Like a compiler checks for errors in your program
- Always write valid HTML pages even if wrong HTML page “works”.



```
        <ul id="bigBarNavigation">
<li><a href="/">HOME</a>
    </li><li><a href="/contact">CONTACT US</a></li><li>
        <a href="/about">ABOUT US</a></li></ul>
```

Confusing mess...



```
<ul id="bigBarNavigation">
    <li><a href="/">HOME</a></li>
    <li><a href="/contact">CONTACT US</a></li>
    <li>
        <a href="/about">ABOUT US</a>
        <div class="subMenu">
                <!-- Just an example to
                     show indentation -->
        </div>
    </li>
</ul>
```

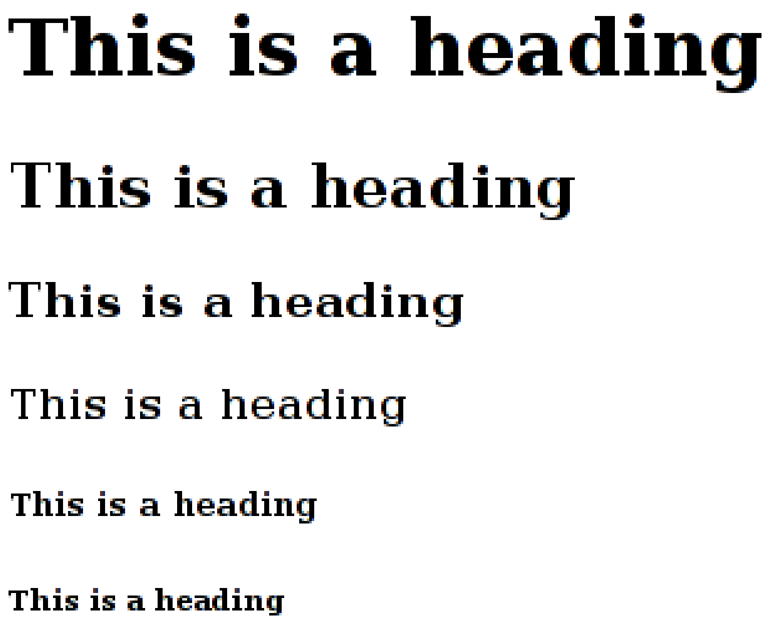
Nice and clean. mmmmmmm...

# Heading

You Write:

```
<h1>This is a heading</h1>
<h2>This is a heading</h2>
<h3>This is a heading</h3>
<h4>This is a heading</h4>
<h5>This is a heading</h5>
<h6>This is a heading</h6>
```

You See:



# This is a heading

## This is a heading

### This is a heading

#### This is a heading

##### This is a heading

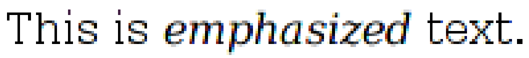
###### This is a heading

# Formatting

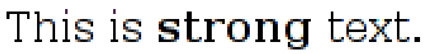
You Write:

```
This is <em>
emphasized </em>
text.
This is <strong>
strong </strong>
text.
This is
<strong><em> strong
and emphasized
</em></strong>
text.
```

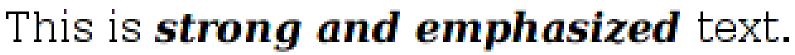
You See:



This is *emphasized* text.



This is **strong** text.



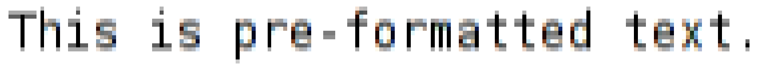
This is ***strong and emphasized*** text.

# Formatting

You Write:

```
<pre>This is pre-formatted text.
</pre>
```

You See:



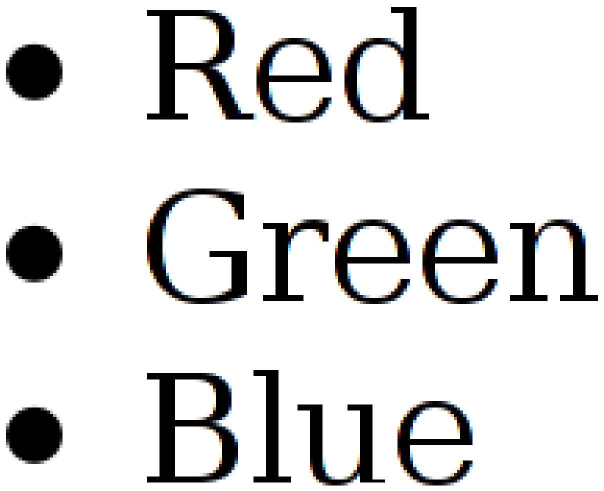
```
This is pre-formatted text.
```

# Unordered List

You Write:

```
<ul>
  <li>Red</li>
  <li>Green</li>
  <li>Blue</li>
</ul>
```

You See:



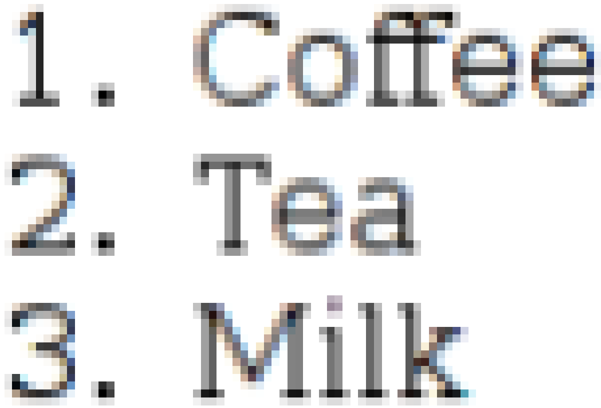
- Red
- Green
- Blue

# Ordered List

You Write:

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

You See:



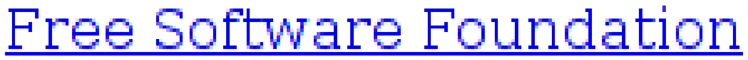
1. Coffee
2. Tea
3. Milk

# Link to Another Web Page

You Write:

```
<a href="https://fsf.org/">
Free Software Foundation </a>
```

You See:



Free Software Foundation

# Image

You Write:

```
<img src="gnu.png"
alt="GNU" />
```

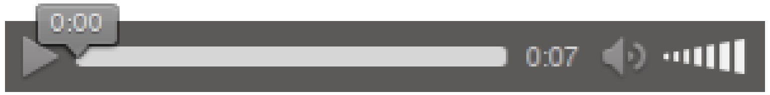
You See:

# Audio

You Write:

```
<audio width="320"
height="240"
controls>
  <source
src="audio.ogg"
type="audio/ogg">
</audio>
```

You Get:

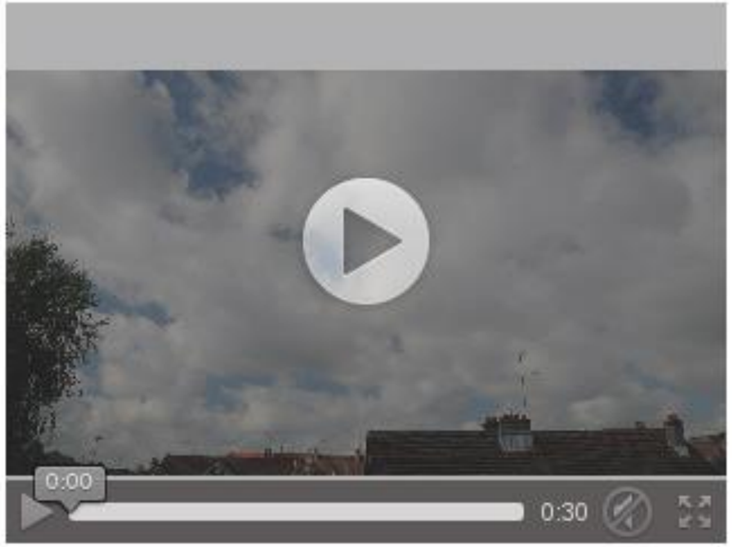


# Video

You Write:

```
<video width="320"
height="240"
controls>
  <source
src="video.ogx"
type="video/ogg">
</video>
```

You Get:

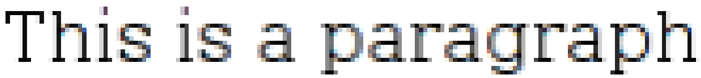


# Comments

You Write:

```
<!-- This is a
comment -->
<p> This is a
paragraph </p>
```

You See:



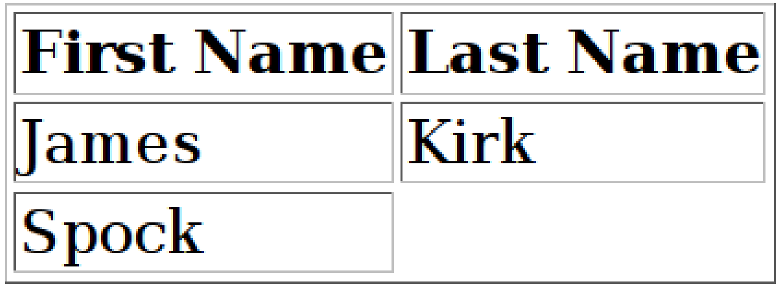
This is a paragraph

# Table

You Write:

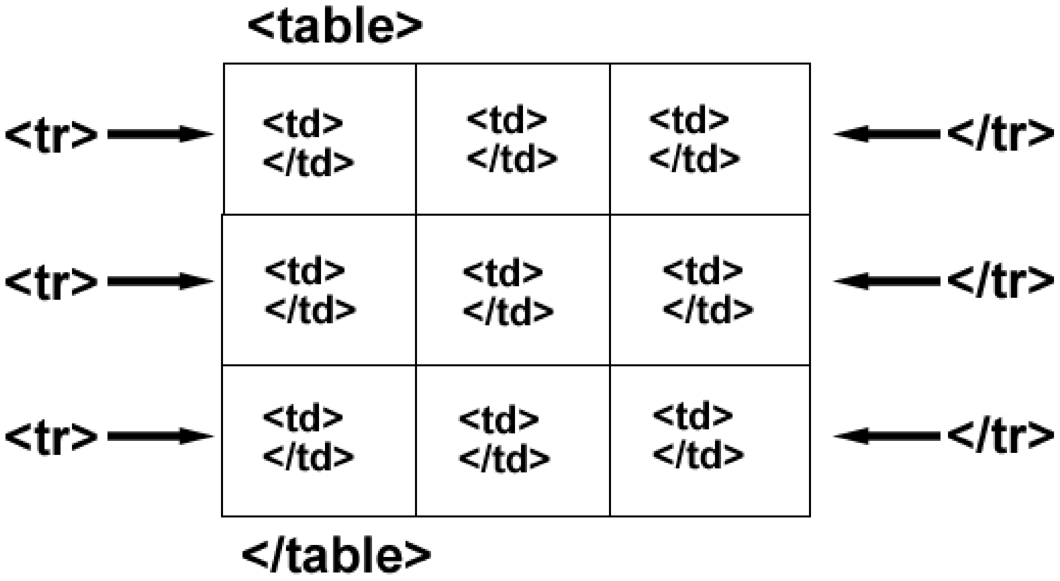
```
<table border="1">
  <tr>
    <th>First name</th>
    <th>Last name</th>
  </tr>
  <tr>
    <td>James</td>
    <td>Kirk</td>
  </tr>
  <tr>
    <td>Spock</td>
    <td></td>
  </tr>
</table>
```

You See:



| First Name | Last Name |
|---|---|
| James | Kirk |
| Spock | |

# Tables



<table>

| | | | | |
|---|---|---|---|---|
| <tr> | <td><br></td> | <td><br></td> | <td><br></td> | </tr> |
| <tr> | <td><br></td> | <td><br></td> | <td><br></td> | </tr> |
| <tr> | <td><br></td> | <td><br></td> | <td><br></td> | </tr> |

</table>

**Basic tags:**

**<table></table>**

**<caption>, <summary>**

**<tr> </tr>, <th></th>**

**<td> >/td>**

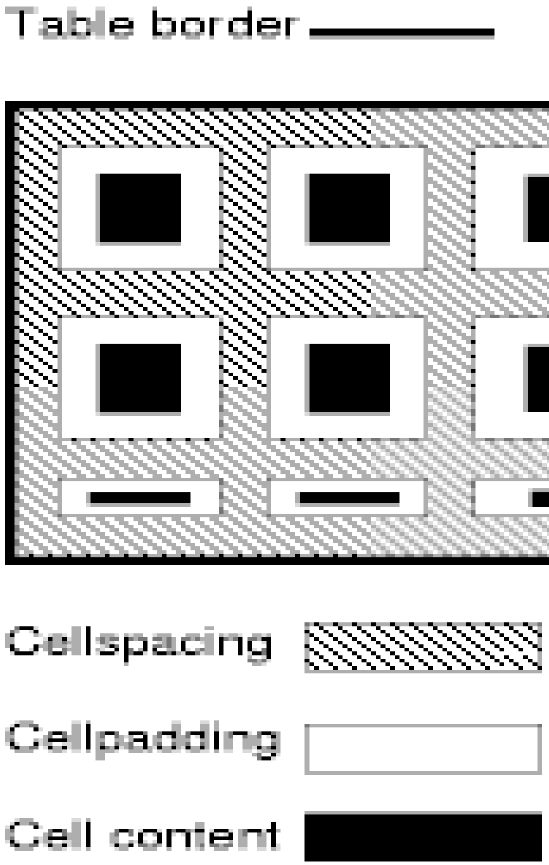
**<colspan>, <rowspan>**

**Attributes:**
**cellpadding, border,**
**bgcolor, width, align**

# How Tables are rendered

- Make the table <summary>, if any,  available to the user,
- Render the <caption>, if any
- Render the table header <thead>, and the footer <tfoot>, if any
- Calculate the number of columns <td> in the table
- Group the columns according to any column group <colgroup> specifications
- Render the cells, row by row and grouped in appropriate columns, between the header and footer
- format the table according to HTML attributes and style sheet specification

# Table Formatting



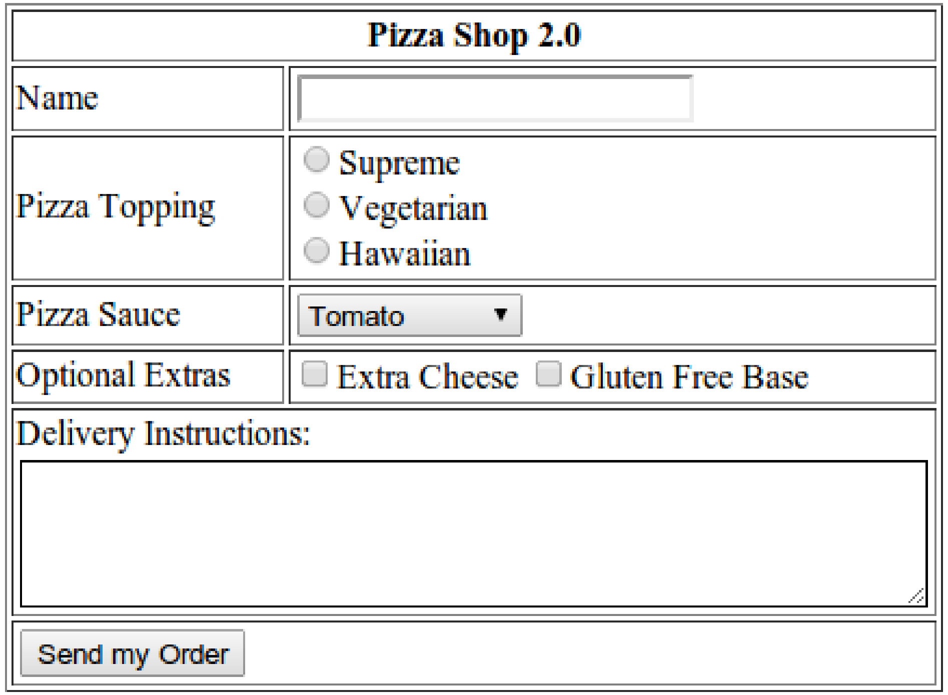
**Points to ponder about :**

**Accessibility,**

**readability**

**aesthetics**

# HTML Forms



| | |
|---|---|
| **Pizza Shop 2.0** | |
| Name | |
| Pizza Topping | ○ Supreme<br>○ Vegetarian<br>○ Hawaiian |
| Pizza Sauce | Tomato ▼ |
| Optional Extras | □ Extra Cheese □ Gluten Free Base |
| Delivery Instructions: | |
| Send my Order | |

**Basic tags:**

<input> <select> <option><button>

input type: text, password, submit, reset, radio, checkbox …

html5 input types: color, date, url, email, month,...

other attributes: action, validate,

img source: http://opentechschool.github.io/python-flask/core/images/sample_web_form.png

# HTML Form Action

**Uses HTTP protocol**

- enable communications between clients and servers
- A request response protocol
- Data goes as name value pair

**HTTP Request methods**

**GET** - Requests data from a specified resource

- query string (name/value pairs) is sent in the URL of a GET request:(/test/demo_form.php?name1=value1&name2=value2)
- Can be cached, bookmarked, should be used only for non-sensitive data, limits on length (2048)

**POST**- Submits data to be processed to a specified resource

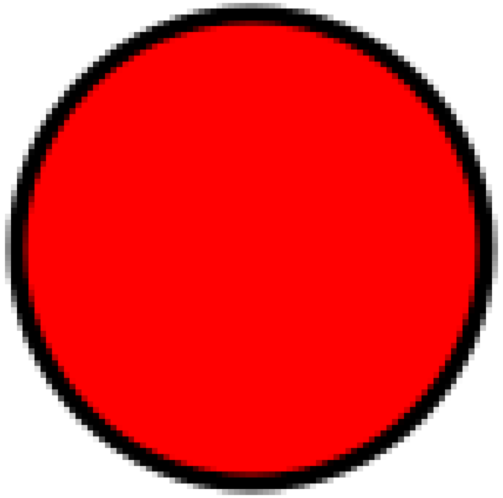
- query string (name/value pairs) is sent in the HTTP message body of a POST request: ( POST /test/demo.php HTTP/1.1, Host: iiit.ac.com, name1=value1&name2=value2)

# SVG – Scalable Vector Graphic

You Write:

```
<svg>
  <circle cx=”50”
        cy=”50”
r=”40”
    stroke=”black”
        stroke-
width=”3”
    fill=”red”/>
</svg>
```

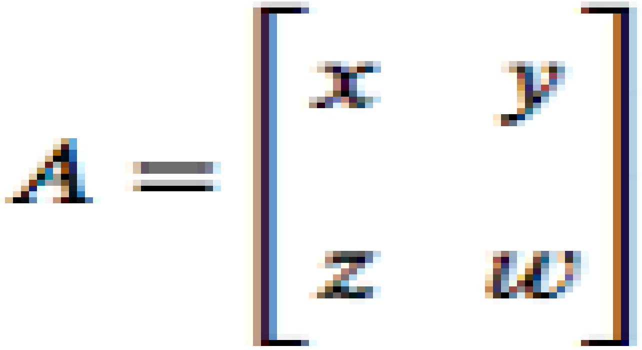
You See:



# MathML

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
        <mrow>
         <mi>A</mi>
         <mo>=</mo>
         <mfenced open="[" close="]">
               <mtable>
                  <mtr>
                     <mtd><mi>x</mi></mtd>
                     <mtd><mi>y</mi></mtd>
                  </mtr>

                  <mtr>
                     <mtd><mi>z</mi></mtd>
                     <mtd><mi>w</mi></mtd>
                  </mtr>
               </mtable>
         </mfenced>
     </mrow>
</math>
```



$$A = \begin{bmatrix} x & y \\ z & w \end{bmatrix}$$

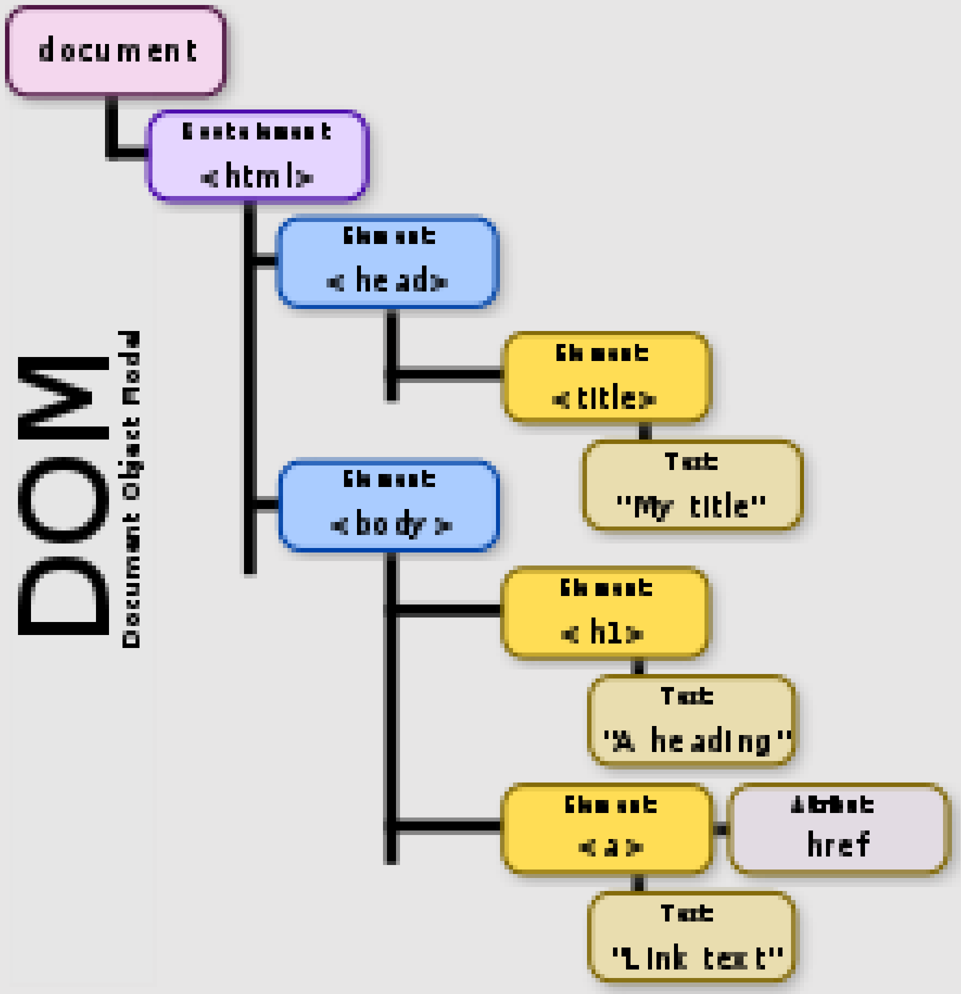
Note: Only supported in Firefox and Safari

# WebGL Demo

See more examples here: [http://webglsamples.org/](http://webglsamples.org/)

# DOM ( Document Object Modeling)

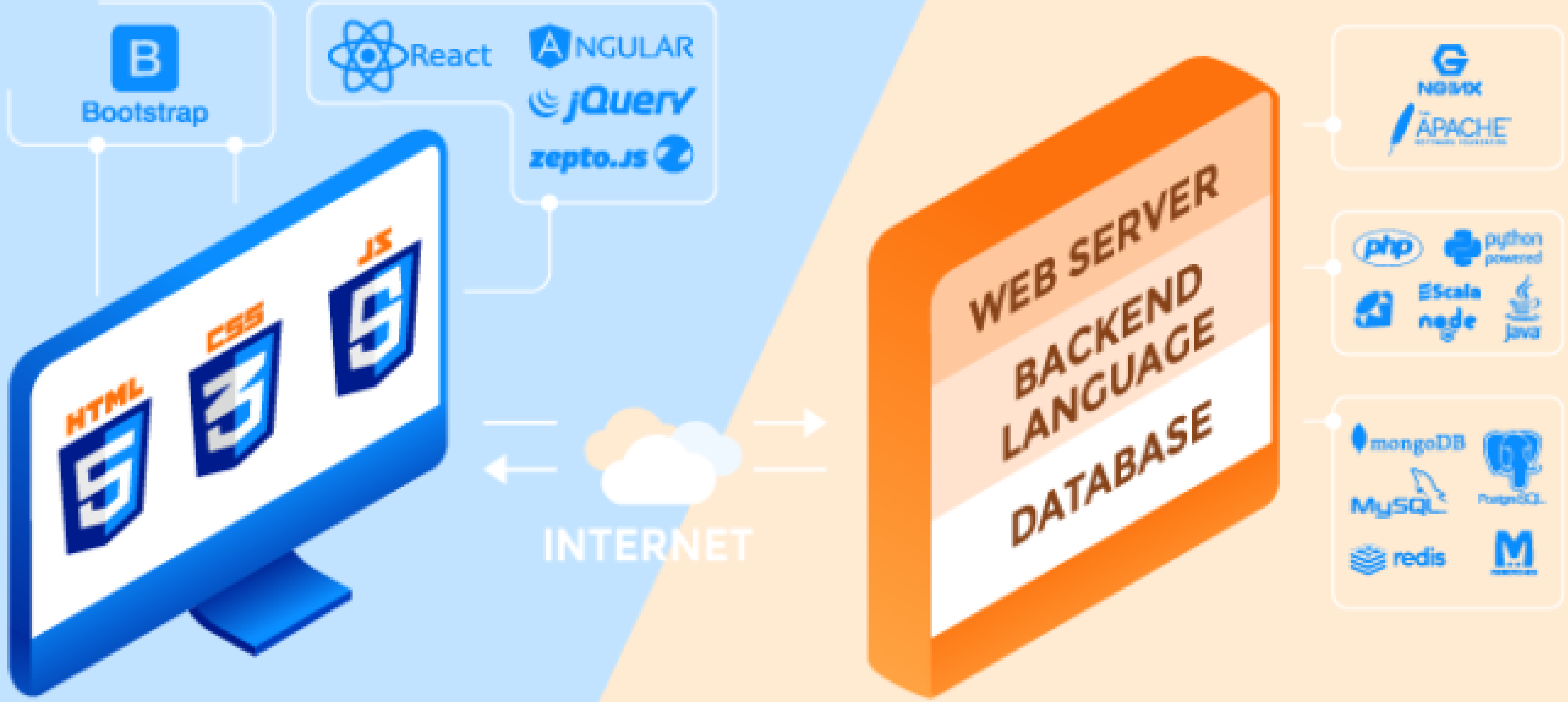
- The Document Object Model (DOM) is a programming API for HTML and XML documents.
- It defines the logical structure of documents and the way a document is accessed and manipulated.
- HTML, XML, XSLT, JSTL document as a tree structure wherein each node is an object representing a part of the document.



# XML - eXtensible Markup Language

```
<?xml version="1.0" encoding="UTF-8"?>
<note>
  <to>Ravi</to>
  <from>Sharat</from>
  <heading>Reminder</heading>
  <body>Don't forget to come to class on Monday</body>
</note>
```

```
<note>
  <to>Sharat</to>
  <from>Ravi</from>
  <heading>Mail</heading>
  <body>I am not coming to class</body>
</note>
```



CLIENT SIDE

SERVER SIDE