



# EECS E6893 Big Data Analytics

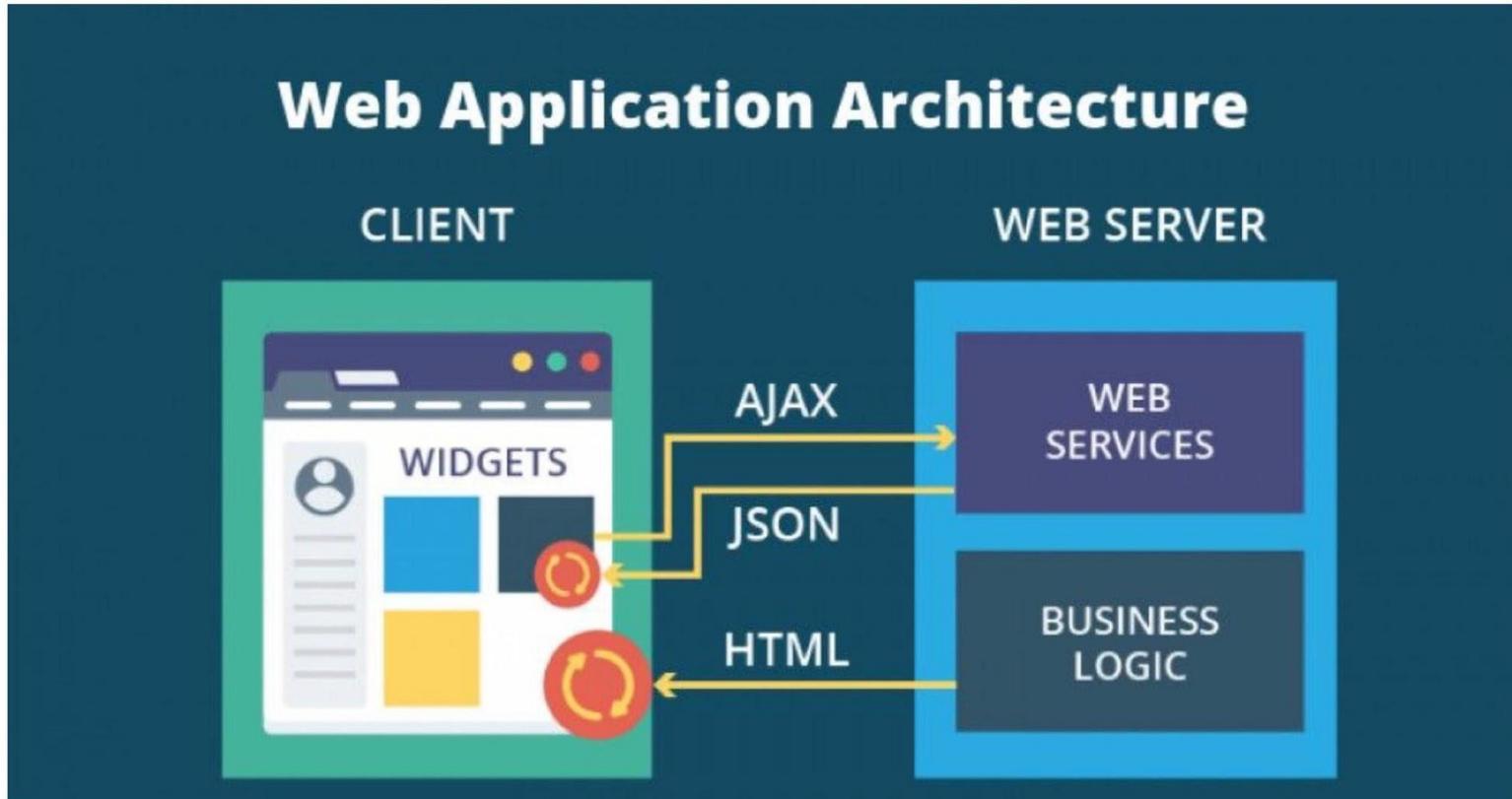
## HW3: Data visualization – Part I

Gudmundur Jonasson, [gmj2122@columbia.edu](mailto:gmj2122@columbia.edu)

# Agenda

- Introduction of Web Application
  - HTML, CSS and JavaScript
  - 2 important things to know: SVG and DOM
- Using D3.js to do data visualization
- Introduction of Apache HTTP Server

# Web Application

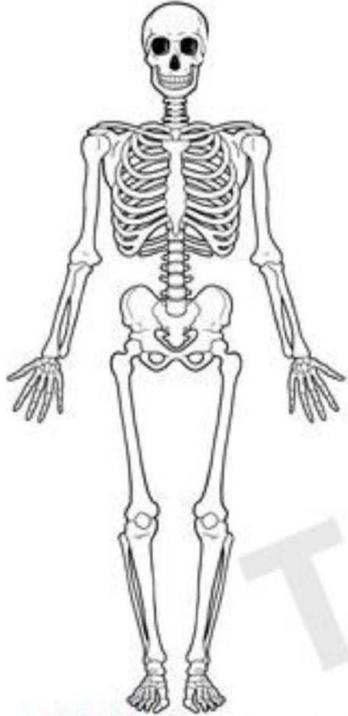


Reference: <https://hackr.io/blog/web-application-architecture-definition-models-types-and-more>

# Client-side of Web

- HTML, CSS and JS are the parts of all websites that users directly interact with.
- HTML provides the *basic structure* of sites, which is enhanced and modified by other technologies like CSS and JavaScript.
- CSS is used to control *presentation, formatting, and layout*.
- JavaScript is used to control the *behavior* of different elements.

# WEB DESIGNING



**HTML** ( Structure)



**CSS** ( Presentation)



**Javascript** ( functionality)  
[tutorial.techaltum.com](http://tutorial.techaltum.com)

# What and Why?

- JavaScript is a programming language
- used by Web browsers to create a dynamic and interactive experience for the user.
- Most of the functions and applications that make the Internet indispensable to modern life are coded in some form of JavaScript.
- Some of the dynamic website enhancements performed by JavaScript are:  
Loading new content or data onto the page without reloading the page,  
Rollover effects and dropdown menus etc.
- Some of its most powerful features involve asynchronous interaction with a remote server.

# Common Uses of JavaScript

- Form validation
- Page embellishments and special effects
- Navigation systems
- Basic math calculations
- Dynamic content manipulation
- Sample applications
  - Dashboard widgets in Mac OS X, Google Maps, Philips universal remotes, Writely word processor, hundreds of others...

# JavaScript in Web Pages

- Embedded in HTML page as `<script>` element
  - JavaScript written directly inside `<script>` element
    - `<script> alert("Hello World!") </script>`
  - Linked file as `src` attribute of the `<script>` element
    - `<script type="text/JavaScript" src="functions.js"></script>`
- Event handler attribute
  - `<a href="http://www.yahoo.com" onmouseover="alert('hi');">`
- Pseudo-URL referenced by a link
  - `<a href="JavaScript: alert('You clicked');">Click me</a>`

# Example 1: Add Two Numbers

```
<> jsexample.html > ...
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>Adding two numbers</title>
5    </head>
6    <body>
7      <script>
8        var num1, num2, sum
9        num1 = prompt("Enter first number")
10       num2 = prompt("Enter second number")
11       sum = parseInt(num1) + parseInt(num2)
12       alert("Sum = " + sum)
13     </script>
14
15   </body>
16 </html>
17
```

This page says

Enter first number

Cancel

OK

This page says

Enter second number

Cancel

OK

This page says

Sum = 15

OK

# Example 2: Page Manipulation

```
<> jsexample.html > ...
1  <!DOCTYPE html>
2  <html>
3    <body>
4      <h1>Element Object</h1>
5      <h2>appendChild() Method</h2>
6
7      <ul id="myList">
8        <li>Car</li>
9        <li>Bike</li>
10     </ul>
11
12     <p>Click "Append" to append an item to the end of the list:</p>
13
14     <button onclick="myFunction()">Append</button>
15
16     <script>
17       function myFunction() {
18         const node = document.createElement("li");
19         const textnode = document.createTextNode("Bus");
20         node.appendChild(textnode);
21         document.getElementById("myList").appendChild(node);
22       }
23     </script>
24
25   </body>
26 </html>
```

## Element Object

### appendChild() Method

- Car
- Bike

Click "Append" to append an item to the end of the list:

Append

## Element Object

### appendChild() Method

- Car
- Bike
- Bus

Click "Append" to append an item to the end of the list:

Append

# Language Basics

- JavaScript is case sensitive
  - `onClick`, `ONCLICK`, ... are HTML, thus not case-sensitive
- Statements terminated by returns or semi-colons
  - `x = x+1;` same as `x = x+1`
- “Blocks” of statements enclosed in `{ ... }`
- Variables
  - Define using the `var` statement
  - Define implicitly by its first use, which must be an assignment
    - Implicit definition has global scope, even if occurs in nested scope!

# JavaScript Primitive Data types

- Boolean: true and false
- Number: 64-bit floating point
  - Similar to Java double and Double
  - No integer type
  - Special values NaN (not a number) and Infinity
- String: sequence of zero or more Unicode chars
  - No separate character type (just strings of length 1)
  - Literal strings using ' or " characters (must match)
- Special objects: null and undefined

# Objects

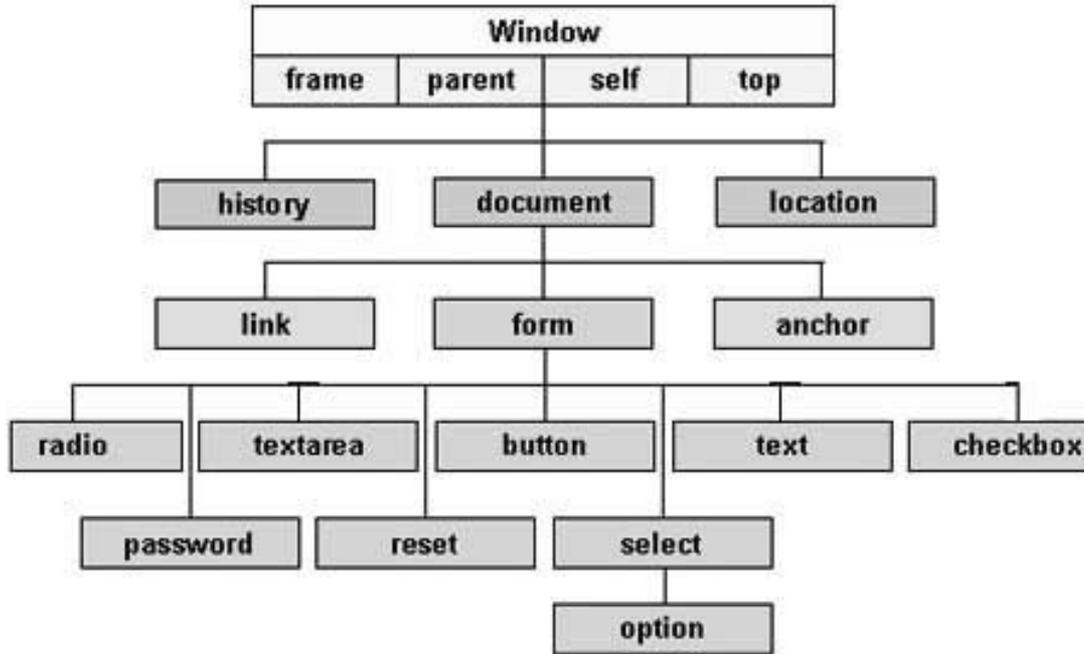
- An object is a collection of named properties
- Think of it as an associative array or hash table
  - Set of name:value pairs
    - objBob = {name: "Bob", grade: 'A', level: 3};
  - Play a role similar to lists in Lisp / Scheme
- New members can be added at any time
  - objBob.fullname = 'Robert';
- Can have methods

# Functions

- Functions are objects with method called “( )”
  - A property of an object may be a function (=method)
    - function `max(x,y) { if (x>y) return x; else return y;}`;
    - `max.description` = “return the maximum of two arguments”;
  - Local declarations may appear in function body
- Call can supply any number of arguments
  - `functionname.length` : # of arguments in definition
  - `functionname.arguments.length` : # arguments in call
  - Basic types are passed by value, objects by reference
- “Anonymous” functions
  - `(function (x,y) {return x+y})(2,3);`

# Document Object Model (DOM)

- HTML page is structured data
- DOM provides representation of this hierarchy
- Examples
  - Properties: `document.alinkColor`, `document.URL`, `document.forms[ ]`, `document.links[ ]`, `document.anchors[ ]`, ...
  - Methods: `document.write(document.referrer)`
    - These change the content of the page!
- Also Browser Object Model (BOM)
  - Window, Document, `Frames[ ]`, History, Location, Navigator (type and version of browser)



JavaScript Document Object Model (DOM) hierarchy

Ref: [http://www.tutorialspoint.com/javascript/javascript\\_html\\_dom.htm](http://www.tutorialspoint.com/javascript/javascript_html_dom.htm)

# Document Object Model (DOM)

The way a document content is accessed and modified is called the Document Object Model, or DOM. The Objects are organized in a hierarchy. This hierarchical structure applies to the organization of objects in a Web document.

- Window object – Top of the hierarchy. It is the outmost element of the object hierarchy.
- Document object – Each HTML document that gets loaded into a window becomes a document object. The document contains the contents of the page.
- Form object – Everything enclosed in the `<form>...</form>` tags sets the form object.
- Form control elements – The form object contains all the elements defined for that object such as text fields, buttons, radio buttons, and checkboxes.

# Introduction to HTML

- HTML is a language for describing web pages.
- HTML stands for **H**yper **T**ext **M**arkup **L**anguage
- HTML is not a programming language, it is a **markup language**
- A markup language is a set of **markup tags**
- HTML uses **markup tags** to describe web pages

# Objectives of HTML

- create, save and view a HTML document
- format a web page using section heading tags
- describe Ordered and Unordered lists
- explain graphics in HTML document
- describe hypertext links and making text/image link

# World Wide Web

- The **World Wide Web** (abbreviated as **WWW** or **W3** and commonly known as **the Web**) is a system of interlinked hypertext documents accessed via the Internet. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks.

# HTML Tools

- a) HTML Editor: it is the program that one uses to create and save HTML documents. They fall into two categories:
  - Text based or code based which allows one to see the HTML code as one is creating a document.e.g. Notepad.
  - Netscape composer
  
- b) Web Browser: program to view and test the HTML documents. They translate Html encoded files into text, image, sounds and other features user see. Microsoft Internet Explorer, Netscape, Chrome are examples of browsers that enables user to view text and images and many more other World Wide Web features.

# HTML Terminology

- Tag: Tags are always written within angles brackets. it is a piece of text is used to identify an element so that the browser realizes how to display its contents.e.g.<HTML> tag indicates the start of an HTML document . HTML tag can be two types. They are:-
  - Paired Tags :A tag is said to be a paired tag if text is placed between a tag and its companions tag.In paired tag ,the first tag is referred to as opening tag and the second tag is referred to as closing tag.
  - Unpaired Tags: An unpaired tag does not have a companion tag .unpaired tag also known as singular or Stand-Alone tags.e.g:<br>,<hr> etc.

# HTML Terminology

- **Attribute:** Attribute is the property of an tag that specified in the opening angle brackets. It supplies additional information like color,size,home font-style etc to the browser about a tag. E.g. most of the common attributes are height, color,width,src,border,align etc.
- **DTD: Document Type Definition** is a collection of rules written in standard Generalized Markup Language(SGML).HTML is define in terms of its DTDS. All the details of HTML tags, entities and related document structure are defined in the DTDS.
- **ELEMENT:** Element is the component of a document's structure such as a title, a paragraph or a list. It can include an opening and a closing tag and the contents within it.

# Steps to create a HTML file and view in browser

- Step-1: Open a text editor or notepad on your machine.
- Step-2: Enter the following lines of code:

```
<> myfirstpage.html >  html
1  <!DOCTYPE html>
2  <html>
3
4      <head>
5          <title>Page Title</title>
6      </head>
7
8      <body>
9          <h1>This is a Heading</h1>
10         <p>This is a paragraph.</p>
11     </body>
12 </html>|
```

- Step-3: Save the file as myfirstpage.html (go to File-Save As give File name: myfirstpage.html-choose save as type: All Files-click save)
- Step-4: Viewing document in web browser (go to folder where file is saved and open it in any browser of your choice)



## **This is a Heading**

This is a paragraph.

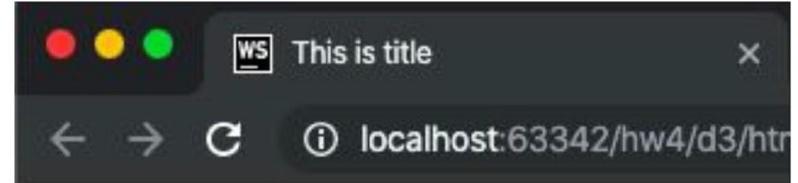
# HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>This is title</title>
</head>
<body>

  <h1>Header 1</h1>
  <h2>Header 2</h2>
  <h3>Header 3</h3>

  <p>Paragraph 1</p>

</body>
</html>
```



**Header 1**

**Header 2**

**Header 3**

Paragraph 1

# Styles

Rough timeline of HTML / CSS history:

Early 90s:

```
<h1>This is an h1 header.</h1>
```

**This is an h1 header.**

<http://www.pmichaud.com/toast/>

# Styles

## Mid 1990s

```
<p>This method of <font color="green" face="Times New Roman">  
styling</font> was deprecated in 1998--but it still works :-).</p>
```

This method of **styling** was deprecated in 1998—but it still works :-).

HTML tag history

<http://www.martinrinehart.com/frontend-engineering/engineers/html/html-tag-history.html>

# Styles: External style sheet (preferred method)

Late 1990s - present: efforts to separate *style* from *content*

```
<head>  
  <link rel="stylesheet" href="style.css">  
</head>
```

style.css:

```
.formal {color: red;  
  font-size: 30px;  
  font-family: Lucida Calligraphy;  
}
```

# Styles: Internal style sheet

**<style> tag in <head> section:**

```
<head>
  <style type="text/css">
    .formal {color: red;
      font-size: 30px;
      font-family: Lucida Calligraphy;
    }
  </style>
</head>

<body>
  <h2 class="formal">Styled with CSS</h2>
</body>
```

**Styled with CSS**

# Styles: External style sheet

Preferred method of adding styles

Body of html file:

```
<body>  
  <h2 class="formal">Styled with CSS</h2>  
</body>
```

**Styled with CSS**

<http://www.csszengarden.com/> (started 2003)

# Styles: Inline style attributes

- Not recommended if you are adding styling manually
- However, JavaScript/D3 add styling *inline*

```
<h1 style="font-family: Bookman;">The word  
<span style="color: blue;">blue</span>  
has four letters.</h1>
```

The word **blue** has four letters.

[view-source:http://www.dolekemp96.org/agenda/issues/education.htm](http://www.dolekemp96.org/agenda/issues/education.htm)

# SVG in HTML

- SVG stands for Scalable Vector Graphics. It is used to define vector-based graphics for the Web
- **Every element and every attribute in SVG files can be animated**
- SVG integrates with other W3C standards such as the **DOM** and XSL

```
<p id="p1">Paragraph 1</p>
<button type="button" onclick=myfunction()>Click here to change Paragraph 1</button>

<svg id='svg1' width="400" height="200">
  <rect id='r1' width="300" height="100" fill="red"/>
</svg>
```

# SVG in HTML

---

**Header 1**

**Header 2**

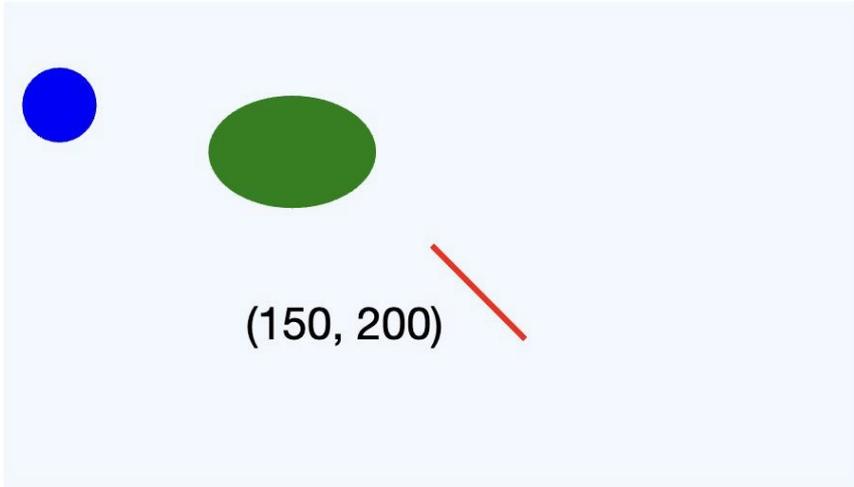
**Header 3**

Paragraph 1



# SVG

```
<svg width="500" height="300">  <!-- some SVG -->
  <rect x="20" y="20" width="460" height="260" fill="aliceblue"></rect>
  <circle cx="50" cy="75" r="20" fill="blue"></circle>
  <ellipse cx="175" cy="100" rx="45" ry="30" fill="green"></ellipse>
  <text x="150" y="200">(150, 200)</text>
  <line x1="250" y1="150" x2="300" y2="200" stroke="red" stroke-width="3"></line>
</svg>
```



What if we create a SVG elements and use DOM in Javascript to access its attributes?

```
var r1 = document.getElementById('r1');  
r1.setAttribute('fill', 'blue');
```

If we draw a series of SVG and texts based on data, and use DOM to control their attributes, then we get a simple charts!

- D3.js, a library to do this in a simple way

What if we create a SVG elements and use DOM in Javascript to access its attributes?

**Header 1**

**Header 2**

**Header 3**

Paragraph 1



If we draw a series of SVG and texts based on data, and use DOM to control their attributes, then we get a simple charts!

- D3.js, a library to do this in a simple way

# Hosting webpage on Apache web server using virtual host (MAC users)

## Step 1: Install xcode

```
(base) srividyanampudi@Srividyas-Air ~ % xcode-select --install
xcode-select: error: command line tools are already installed, use "Software Update" to install updates
(base) srividyanampudi@Srividyas-Air ~ % █
```

## Step 2: Install Homebrew

Go to [brew.sh](https://brew.sh) in your browser and copy the command there to your terminal-  
`/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"`

```
(base) srividyanampudi@Srividyas-Air ~ % /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
=> Checking for `sudo` access (which may request your password)...
=> This script will install:
/opt/homebrew/bin/brew
/opt/homebrew/share/doc/homebrew
/opt/homebrew/share/man/man1/brew.1
/opt/homebrew/share/zsh/site-functions/_brew
/opt/homebrew/etc/bash_completion.d/brew
/opt/homebrew
```

⇒ Installation successful!

⇒ Homebrew has enabled anonymous aggregate formulae and cask analytics.

Read the analytics documentation (and how to opt-out) here:

<https://docs.brew.sh/Analytics>

No analytics data has been sent yet (nor will any be during this `install` run).

⇒ Homebrew is run entirely by unpaid volunteers. Please consider donating:

<https://github.com/Homebrew/brew#donations>

⇒ Next steps:

- Run these three commands in your terminal to add Homebrew to your **PATH**:

```
echo '# Set PATH, MANPATH, etc., for Homebrew.' >> /Users/srividyanampudi/.zprofile  
echo 'eval "$(opt/homebrew/bin/brew shellenv)"' >> /Users/srividyanampudi/.zprofile  
eval "$(opt/homebrew/bin/brew shellenv)"
```

- Run **brew help** to get started

- Further documentation:

<https://docs.brew.sh>

## Step 3: Add Homebrew to your PATH

Follow instructions after installation

```
(base) srividyanampudi@Srividyas-Air ~ % echo '# Set PATH, MANPATH, etc., for Homebrew.' >> /Users/srividyanampudi/.zprofile  
(base) srividyanampudi@Srividyas-Air ~ % echo 'eval "$(opt/homebrew/bin/brew shellenv)"' >> /Users/srividyanampudi/.zprofile  
(base) srividyanampudi@Srividyas-Air ~ % eval "$(opt/homebrew/bin/brew shellenv)"  
(base) srividyanampudi@Srividyas-Air ~ % █
```

## Step 4: Install apache2

Command - brew install apache2

```
(base) srividyanampudi@Srividyas-Air ~ % brew install apache2
=> Downloading https://ghcr.io/v2/homebrew/core/apr/manifests/1.7.0_3
##### 100.0%
=> Downloading https://ghcr.io/v2/homebrew/core/apr/blobs/sha256:02e6b44b3284fa471cce15592a8666
=> Downloading from https://pkg-containers.githubusercontent.com/ghcr1/blobs/sha256:02e6b44b328
##### 100.0%
```

## Step 5: Start apache server

```
(base) srividyanampudi@Srividyas-Air ~ % sudo apachectl start
(base) srividyanampudi@Srividyas-Air ~ % █
```

This will start Apache HTTP server which can be tested by visiting localhost on the browser. The localhost 8080 gives the response as shown below:

A screenshot of a web browser's address bar. It shows navigation icons (back, forward, refresh) on the left and the text 'localhost:8080' on the right, indicating the current page is being accessed at that local address.

**It works!**

## Step 6: Open httpd config file

- If you're using Intel-based Mac: `vim /usr/local/etc/httpd/httpd.conf`
- If you're using Mac with Apple Silicon: `vim /opt/homebrew/etc/httpd/httpd.conf`

## Step 7: Update these lines

Listen 8080 to Listen 80

DocumentRoot `"/usr/local/var/www"` to DocumentRoot `"/Users/your_account/Sites"`  
<Directory `"/usr/local/var/www"`> to <Directory `"/Users/your_account/Sites"`>

AllowOverride None to AllowOverride All

```
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:80
Listen 80
```

```
#
DocumentRoot "/Users/srividyanampudi/Sites"
<Directory "/Users/srividyanampudi/Sites">
#
# Possible values for the Options directive are "None", "All",
# or any combination of:
#   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
# The Options directive is both complicated and important. Please see
# http://httpd.apache.org/docs/2.4/mod/core.html#options
# for more information.
#
Options Indexes FollowSymLinks

#
# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
#   AllowOverride FileInfo AuthConfig Limit
#
AllowOverride All

#
# Controls who can get stuff from this server.
#
```

Step 8: Uncomment this line

LoadModule rewrite\_module lib/httpd/modules/mod\_rewrite.so

Update these lines

User \_www to User your\_account

Group \_www to Group staff

```
LoadModule alias_module lib/httpd/modules/mod_alias.so
LoadModule rewrite_module lib/httpd/modules/mod_rewrite.so

<IfModule unixd_module>
#
# If you wish httpd to run as a different user or group, you must run
# httpd as root initially and it will switch.
#
# User/Group: The name (or #number) of the user/group to run httpd as.
# It is usually good practice to create a dedicated user and group for
# running httpd, as with most system services.
#
User srividyaianampudi
Group staff
```

## Step 9: Update this line

ServerName www.example.com:8080 to ServerName localhost

```
# If your host doesn't have a registered DNS name, enter its IP address here.
#
ServerName localhost

#
# Deny access to the entirety of your server's filesystem. You must
# explicitly permit access to web content directories in other
# <Directory> blocks below.
#
```

## Step 10: Create Sites folder and add your html file to the folder

```
(base) srividyanampudi@Srividyas-Air ~ % mkdir Sites
(base) srividyanampudi@Srividyas-Air ~ %
(base) srividyanampudi@Srividyas-Air ~ %
(base) srividyanampudi@Srividyas-Air ~ % cd Sites
(base) srividyanampudi@Srividyas-Air Sites % ls
jsexample.html
```

## Step 11. Restart apache

```
(base) srividyanampudi@Srividyas-Air Sites % sudo apachectl stop
(base) srividyanampudi@Srividyas-Air Sites %
(base) srividyanampudi@Srividyas-Air Sites %
(base) srividyanampudi@Srividyas-Air Sites %
(base) srividyanampudi@Srividyas-Air Sites % sudo apachectl start
```

Step 12: Test localhost to see your hosted website

<http://localhost>

A screenshot of a web browser's address bar. It features navigation icons (back, forward, refresh) on the left, an information icon, and the text 'localhost' in the center. The address bar is highlighted with a blue border.

## Sample HTML to host on apache webserver

- Car
  - Ford

Step 13: Stop apache

```
(base) srividyanampudi@Srividyas-Air Sites % sudo apachectl stop
Password:
(base) srividyanampudi@Srividyas-Air Sites %
```

# Hosting webpage on Apache web server using virtual host (Windows users)

Step 1: Download Apache lounge zip file -

<https://www.apachelounge.com/download/>



The screenshot shows the Apache Lounge website's download page for Apache 2.4 VS16 Windows Binaries and Modules. The page includes a navigation menu on the left with links for Home, VS16, and Additional. A sidebar on the left lists recent updates, including a new update for httpd 2.4.54 on November 02, 2022, and a dropped VC15 download on November 01, 2022. The main content area features a title "Apache 2.4 VS16 Windows Binaries and Modules" and a paragraph explaining that Apache Lounge provides up-to-date Windows binaries and popular third-party modules for more than 15 years. It also mentions that the binaries are built with the latest Windows Visual Studio C++ 2019 (VS16) and are backward compatible. Below this, there are two download options: "Apache 2.4.54 Win64" and "Apache 2.4.54 Win32", each with a corresponding download link, date, and size. The page also includes links for PGP signatures and SHA1-SHA512 checksums for each download.

← → ↻ apachelounge.com/download/

## Apache 2.4 VS16 Windows Binaries and Modules

Apache Lounge has provided up-to-date Windows binaries and popular third-party modules for more than 15 years. We have hundreds of thousands of satisfied home users. Always built with up to date dependencies and latest compilers, and tested thorough. The binaries are referenced by the ASF, Microsoft, PHP etc. a with our binaries and modules.

The binaries, are build with the sources from ASF at [httpd.apache.org](http://httpd.apache.org), contains the latest patches and latest dependencies like zlib, openssl etc. which makes it downloads from other places. The binaries **do not run** on XP and 2003. Runs on: 7 SP1, Vista SP2, 8/8.1, 10, 11 Server 2008 SP2 / R2 SP1, Server 2012 / R2, Server 2016 / R2.

Build with the latest Windows® Visual Studio C++ 2019 aka VS16. VS16 has improvements, fixes and optimizations over VC15 in areas like Performance, Memory features, Code generation and Stability. For example code quality tuning and improvements done across different code generation areas for "speed". And makes supported Windows editions (win7 and up) internal features.

**VS16 is backward compatible**, see [Compatibility VS16](#). You can use a VC15/14 module inside a VS16 binary, for example PHP VC15/14 as module,

**Be sure** you installed latest 14.32.31332 Visual C++ Redistributable Visual Studio 2015-2022 : [vc\\_redist\\_x64](#) or [vc\\_redist\\_x86](#) see [Redistributable](#)

### Apache 2.4 binaries VS16

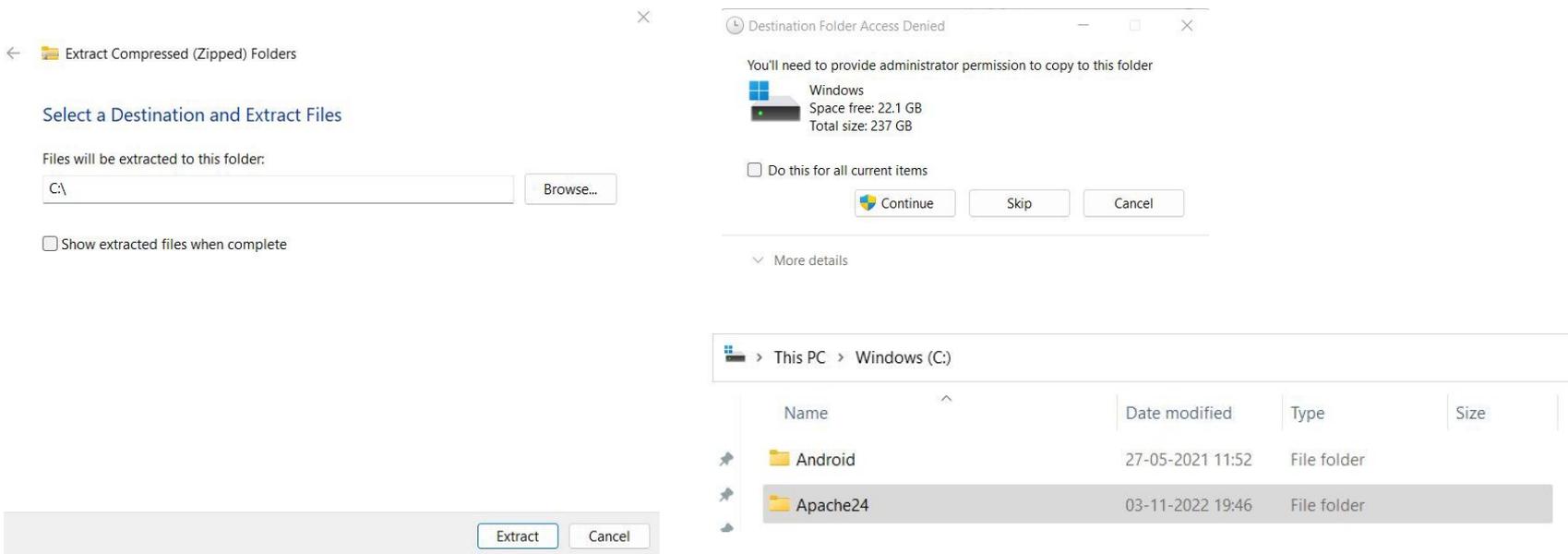
#### Info & Changelog

|  |                    |
|--|--------------------|
| <b>Apache 2.4.54 Win64</b>   |                    |
| ● <a href="#">httpd-2.4.54-win64-VS16.zip</a>  | 02 Nov '22 10.642k |
| <a href="#">PGP Signature</a> (Public <a href="#">PGP key</a> ), <a href="#">SHA1-SHA512</a> <a href="#">Checksums</a> |                    |
| <b>Apache 2.4.54 Win32</b>   |                    |
| ● <a href="#">httpd-2.4.54-win32-VS16.zip</a>  | 02 Nov '22 9.714k  |
| <a href="#">PGP Signature</a> (Public <a href="#">PGP key</a> ), <a href="#">SHA1-SHA512</a> <a href="#">Checksums</a> |                    |

To be sure that a download is intact and has not been tampered with, use PGP, see [PGP Signature](#)

Download 64bit if your system is 64bit else download 32bit zip file

Step 2: Once zip file is finished downloading unzip it to C drive. Click Yes for administrative privileges.



Step 3: Go to **"C:\Apache24\bin\"** and find httpd and we need to install this httpd. This is what is actually going to run apache. So we will go to start menu and type cmd to get command prompt. Right click as select **Run as administrator**. Click yes if it asks you for permission.

Step 4: Change directory to **C:\Apache24\bin\** by typing **cd C:\Apache24\bin** in the terminal

Step 5: Now that we are in bin directory we will install httpd by typing the command **httpd -k install**. Allow access

> This PC > Windows (C:) > Apache24 > bin

| Name                     | Date modified    | Type                  | Size   |
|--------------------------|------------------|-----------------------|--------|
| iconv                    | 03-11-2022 19:46 | File folder           |        |
| ab                       | 03-11-2022 19:46 | Application           | 97 KB  |
| abs                      | 03-11-2022 19:46 | Application           | 108 KB |
| ApacheMonitor            | 03-11-2022 19:46 | Application           | 42 KB  |
| apr_crypto_openssl-1.dll | 03-11-2022 19:46 | Application extens... | 19 KB  |
| apr_dbd_odbc-1.dll       | 03-11-2022 19:46 | Application extens... | 31 KB  |
| apr_idap-1.dll           | 03-11-2022 19:46 | Application extens... | 15 KB  |
| dbmmanage.pl             | 03-11-2022 19:46 | PL File               | 9 KB   |
| htcacheclean             | 03-11-2022 19:46 | Application           | 100 KB |
| htdbm                    | 03-11-2022 19:46 | Application           | 121 KB |
| htdigest                 | 03-11-2022 19:46 | Application           | 84 KB  |
| htpasswd                 | 03-11-2022 19:46 | Application           | 117 KB |
| httpd                    | 03-11-2022 19:46 | Application           | 30 KB  |

cmd

All Apps Documents Web More ▾

Best match

Command Prompt  
App

Apps

- Git CMD (Deprecated) >
- Anaconda Prompt >
- mkTO.cmd >
- Google Cloud SDK Shell >

Search the web

cmd - See web results >

Command Prompt  
App

- Open
- Run as administrator
- Open file location
- Pin to Start
- Pin to taskbar

C:\> Administrator: Command Prompt

```
Microsoft Windows [Version 10.0.22000.1098]
(c) Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\Apache24\bin

C:\Apache24\bin>
```

C:\> Administrator: Command Prompt

```
Microsoft Windows [Version 10.0.22000.1098]
(c) Microsoft Corporation. All rights reserved.

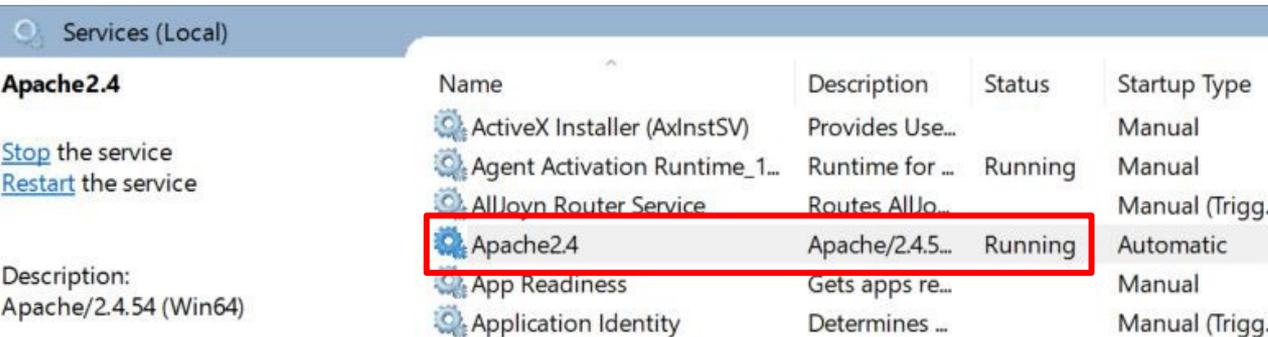
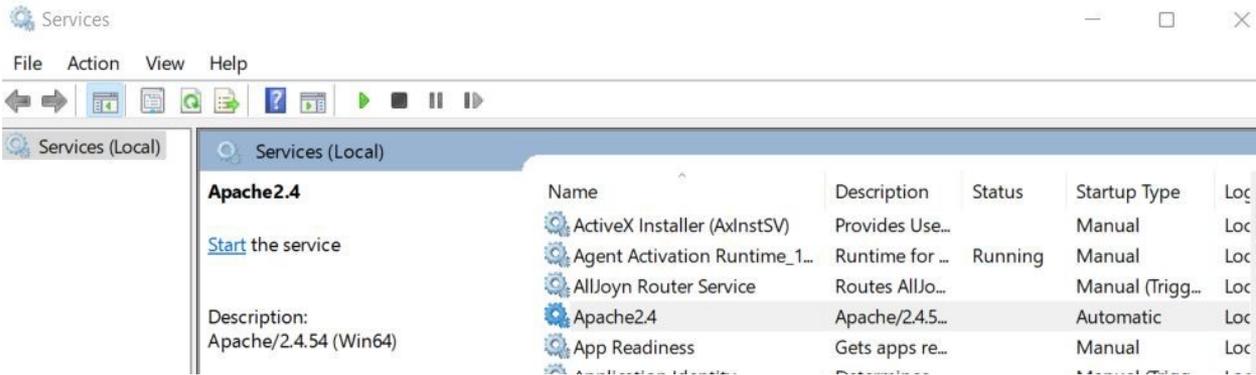
C:\WINDOWS\system32>cd C:\Apache24\bin

C:\Apache24\bin>httpd -k install
Installing the 'Apache2.4' service
The 'Apache2.4' service is successfully installed.
Testing httpd.conf...
Errors reported here must be corrected before the service can be started.

C:\Apache24\bin>
```

Step 6: Now Apache is installed as a service. If we go to Windows and type in Services in search bar and open Services we can see that Apache2.4 is installed.

Step 7: Now we will start the APache service by clicking start on left side. And then it will say that it is running. We can stop or restart the service on the left side if we need to.



Step 8: Now to check if it is running. We will go to 127.0.0.1 in your browser it shows that it works!



Step 9: If you have any errors you can check them in the logs folder in **C:\Apache24\logs** in errors.txt file

Step 10: Now to host our webpage we will go to **C:\Apache24\htdocs** and change the index.html to be your required html file and then again open 127.0.0.1 to see our webpage.

I have hosted the basic histogram html file for example.

```

EXPLORER
  index.html X
  index.html > script
  1 <!DOCTYPE html>
  2 <meta charset="utf-8">
  3
  4 <!-- Loading v4 d3.js -->
  5 <script src="https://d3js.org/d3.v4.js"></script>
  6
  7 <!-- Creating a div where the graph will be plotted -->
  8 <div id="histogram"></div>
  9
  10 <script>
  11 // setting the dimensions and margins of the graph
  12 var margin = {top: 10, right: 30, bottom: 30, left: 40},
  13     width = 400 - margin.left - margin.right,
  14     height = 400 - margin.top - margin.bottom;
  15
  16 // appending the svg object to the body of the page
  17 var svg = d3.select("#histogram")
  18     .append("svg")
  19     .attr("width", width + margin.left + margin.right)
  20     .attr("height", height + margin.top + margin.bottom)
  21     .append("g")
  22     .attr("transform",
  23         "translate(" + margin.left + "," + margin.top + ")");
  24
  25 // getting the data in csv format
  26 d3.csv("https://raw.githubusercontent.com/holtzy/data_to_viz/master/Example_dataset/1_OneNum.csv", function(data) {
  27
  28     // X axis: scale and draw:
  29     var x = d3.scaleLinear()
  30         .domain([0,1000])
  31         // .domain([0, d3.max(data, function(d) { return +d.price })])
  32         // this will set the scale of x from 0 to max of the price column
  33         .range([0, width]);
  34     svg.append("g")
  35         .attr("transform", "translate(0," + height + ")")
  36         .call(d3.axisBottom(x));
  37
  38     // setting the parameters for the histogram
  39     var histogram = d3.histogram()
  40         .value(function(d) { return d.price; }) // giving a vector of value
  41         .domain(x.domain()) // then setting the domain of the graphic
  42         .thresholds(x.ticks(30)); // then the numbers of bins
  43
  44     var bins = histogram(data);
  45
  46     // Y axis: scale and draw:
  47     var y = d3.scaleLinear()
  48         .range([height, 0]);
  49     y.domain([0, d3.max(bins, function(d) { return d.length; })]);
  50     svg.append("g")
  51         .call(d3.axisLeft(y));
  52
  53     // append the bar rectangles to the svg element
  54     svg.selectAll("rect")
  55         .data(bins)
  56         .enter()
  57         .append("rect")
  58         .attr("x", 1)
  59         .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
  60         .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
  61         .attr("height", function(d) { return height - y(d.length); })
  62         .style("fill", "#69b3a2");
  63
  64 });
  65 </script>

```

```

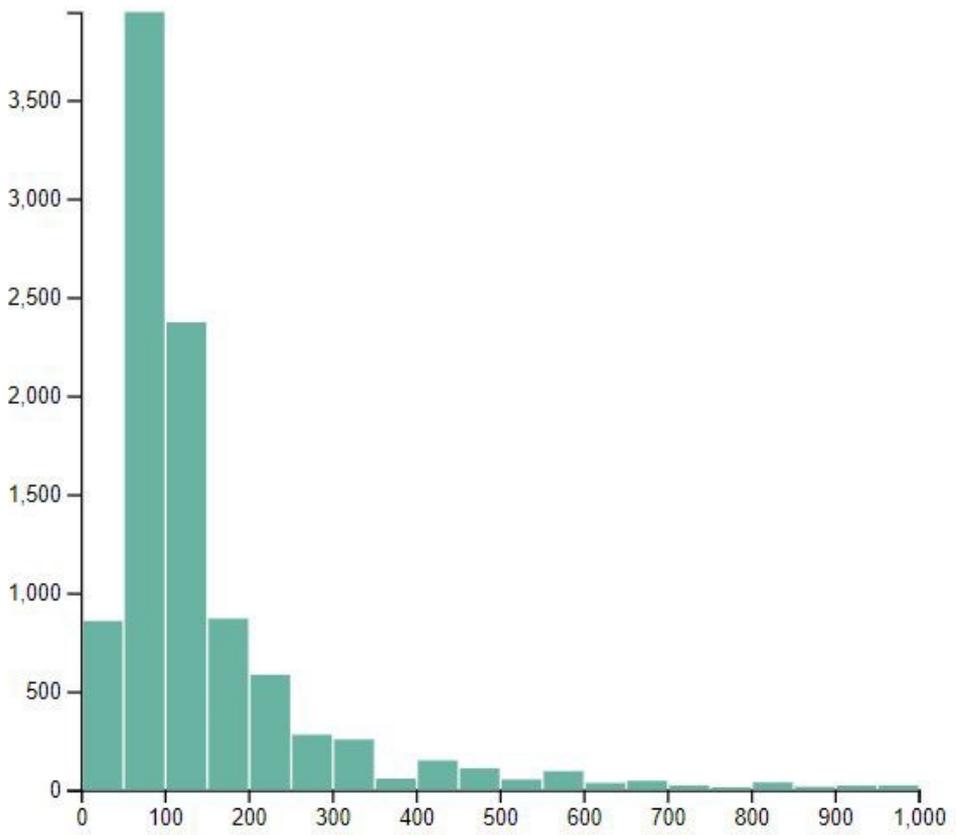
var bins = histogram(data);

// Y axis: scale and draw:
var y = d3.scaleLinear()
    .range([height, 0]);
y.domain([0, d3.max(bins, function(d) { return d.length; })]);
svg.append("g")
    .call(d3.axisLeft(y));

// append the bar rectangles to the svg element
svg.selectAll("rect")
    .data(bins)
    .enter()
    .append("rect")
    .attr("x", 1)
    .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
    .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
    .attr("height", function(d) { return height - y(d.length); })
    .style("fill", "#69b3a2");

});
</script>

```



We can see that our d3.html file is hosted on Apache web server when we go to [127.0.0.1](http://127.0.0.1) on browser

# References

- [https://www.w3schools.com/js/js\\_htmlDOM.asp](https://www.w3schools.com/js/js_htmlDOM.asp)
- [Vitaly Shmatikov CS 345 Introduction to JavaScript](#)
- [Sarbjit Kaur Introduction to HTML](#)
- [D3.js](#)
- <https://d3js.org/>
- <https://developer.mozilla.org/en-US/docs/web/SVG>
- <https://livebook.manning.com/book/d3js-in-action-second-edition/chapter-7/70>
- <https://d3-graph-gallery.com/index.html>

# Acknowledgement

Part of the slides is credited to Juncai Liu and Yiwen Fang

**THANK YOU**