NOTES:

DIFFERENT ERRORS:-

a)A 404 is a Standard Response Code meaning that the server cannot find the requested resource.

b)An HTTP error code meaning "Bad Gateway".

A server can act as a gateway or proxy (go-between) between a client (like your Web browser) and another, upstream server. When you request to access a URL, the gateway server can relay your request to the upstream server. "502" means that the upstream server has returned an invalid response.

In computer networking, upstream server refers to a server that provides service to another server. In other words, upstream server is a server that is located higher in a hierarchy of servers. The highest server in the hierarchy is sometimes called the origin server.

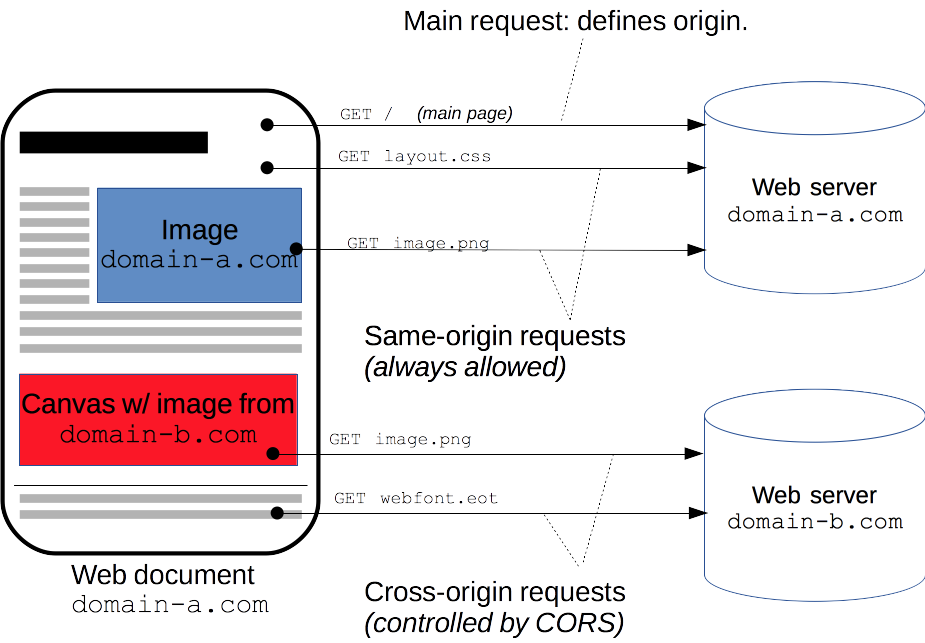
HTTP STORY:-

i)As part of a URI, the "http" within "http://example.com/" is called a "scheme". Resources using the "http" schema are typically transported over unencrypted connections using the HTTP protocol. The "https" scheme (as in "https://developer.mozilla.org") indicates that a resource is transported using the HTTP protocol, but over a secure TLS channel.HTTP transfers every resources of webpage to webserver even.

ii)The http within http means https or "Scheme"and only http means "Schema"

iii)Cross-Origin Resource Sharing (CORS) is a mechanism that uses additional HTTP headers to tell browsers to give a web application running at one origin, access to selected resources from a different origin. A web application executes a cross-origin HTTP request when it requests a resource that has a different origin (domain, protocol, or port) from its own.

An example of a cross-origin request: the front-end JavaScript code served from https://domain-a.com uses XMLHttpRequest to make a request for https://domain-b.com/data.json.<https://mdn.mozillademos.org/files/14295/CORS_principle.png>



WEB INFO,:-

"CLIENT" is a computer or a program that, as part of its operation, relies on sending a request to another

program(SERVER PROGRAM LIKE "PHP") or a computer hardware or software that accesses a service made available by a

server . For example, web browsers are clients that connect to web servers and retrieve web pages for display...

"WEB SERVER" is server software, or hardware dedicated to running this software, that can satisfy client requests

on the World Wide Web. A web server can, in general, contain one or more websites. A web server processes

incoming network requests over HTTP and several other related protocols.Servers can provide various functionalities,'

often called "services", such as sharing data or resources among multiple clients, or performing computation

for a client..

1) $\_GET is an array of variables passed to the current script via the

URL parameters (all variable names and values are displayed in the URL)

2)$\_POST is an array of variables passed to the current script via the

HTTP POST method(all names/values are embedded within the body of the HTTP request)

3)A)creating text box nd declraing post method

B)fetching values of text box nd storing in variable;

C)presenting a table format or statement format with the values,i.e.,fetched from the current file to another file

D)showing the output in the console,whic..h is returned from the server page

4)<a href='page2.php?id=2489&user=tom'>link to page2</a>

echo $\_GET['id']; // output 2489

echo $\_GET['user']; // output tom

5)POST supports advanced functionality such as support for multi-part binary input while uploading files to server.

ex.<input type="file">---will return the multiple files at once by clicking "CHOOSE" to the server..

6)echo $\_SERVER['PHP\_SELF'];//RETURNS the current page server URL link

7)The common gateway interface (CGI) is a standard way for a Web server to pass a Web user's request

to an application program and to receive data back to forward to the user.

8)$\_SERVER['HTTPS'] Is the script queried through a secure HTTP protocol,

So,this are the server requests made to the web server,and after the request is made it returns those data or values

related to it from the database,or sometime from itself..

9)<form method="post" action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]);?>">

EXPLAIN:-

a)The $\_SERVER["PHP\_SELF"] is a super global variable that returns the filename of the currently executing script.

b)The htmlspecialchars() function converts special characters to HTML entities. This means that it will replace HTML

characters like "<" and ">" with &lt and &gt. This prevents attackers from exploiting the code by injecting

HTML or Javascript code (Cross-site Scripting attacks) in forms.

10)In computing, the "Same-Origin Policy" is an important concept in the web application security model. Under the policy, a web browser permits scripts contained in a first web page to access data in a second web page, but only if both web pages have the same origin.It helps isolate potentially malicious documents, reducing possible attack vectors.

It is a critical security mechanism that decides if the origin or content or resource of the first page is same as the origin of 2nd page.

Web content's origin is defined by the scheme (protocol), host (domain), and port of the URL used to access it. Two objects have the same origin only when the scheme, host, and port all match.

Examples of SameOrigin:-

http://example.com/app1/index.html ----->HERE

http://example.com/app2/index.html same origin because same scheme (http) and host (example.com)

http://Example.com:80 ---->HERE

http://example.com same origin because a server delivers HTTP content through port 80 by default

Examples of different origin:-

http://example.com/app1 ----->HERE

https://example.com/app2 different schemes

http://example.com ----->HERE

http://www.example.com different hosts

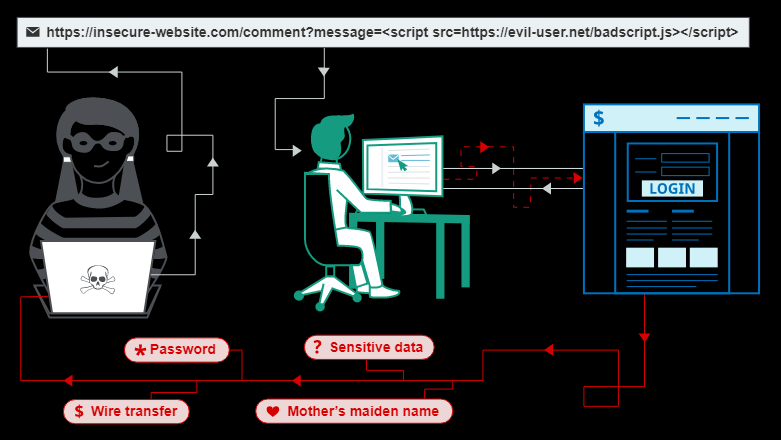
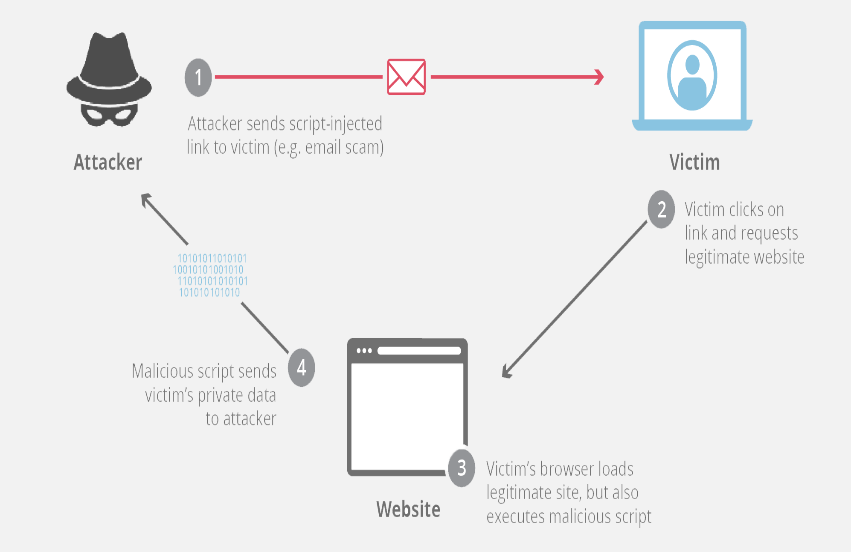
http://myapp.example.com

http://example.com ----->HERE

http://example.com:8080 different ports

CROSS-SITE SCRIPTING:-

Cross-site scripting is a type of computer security vulnerability typically found in web applications. XSS attacks enable attackers to inject client-side scripts into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy.



**Form validation** normally used to occur at the server, after the client had entered all the necessary data and then pressed the Submit button. If the data entered by a client was incorrect or was simply missing, the server would have to send all the data back to the client and request that the form be resubmitted with correct information. This was really a lengthy process which used to put a lot of burden on the server.

JavaScript provides a way to validate form's data on the client's computer before sending it to the web server. Form validation generally performs two functions.

* **Basic Validation** − First of all, the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for data.
* **Data Format Validation** − Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test correctness of data.

The **Htmlspecialchars()** function converts special characters to HTML entities. Now if the user tries to exploit the PHP\_SELF variable, it will result in the following output:

Example: <form method="post" action="test\_form.php/&quot;&gt;&lt;script&gt;alert('hacked')&lt;/script&gt;">

i)The first thing we will do is to pass all variables through PHP's htmlspecialchars() function.

ii)When we use the htmlspecialchars() function; then if a user tries to submit the following in a text field:

iii)<script>location.href('http://www.hacked.com')</script>

- this would not be executed, because it would be saved as HTML escaped code, like this:

iv)&lt;script&gt;location.href('http://www.hacked.com')&lt;/script&gt;

The code is now safe to be displayed on a page or inside an e-mail.

# MY EXPLANATION FOR ABOVE:when we run the code after using htmlspecial chars its showing an output: Access forbidden!

You don't have permission to access the requested object. It is either read-protected or not readable by the server.

,and without that it is showing no values…

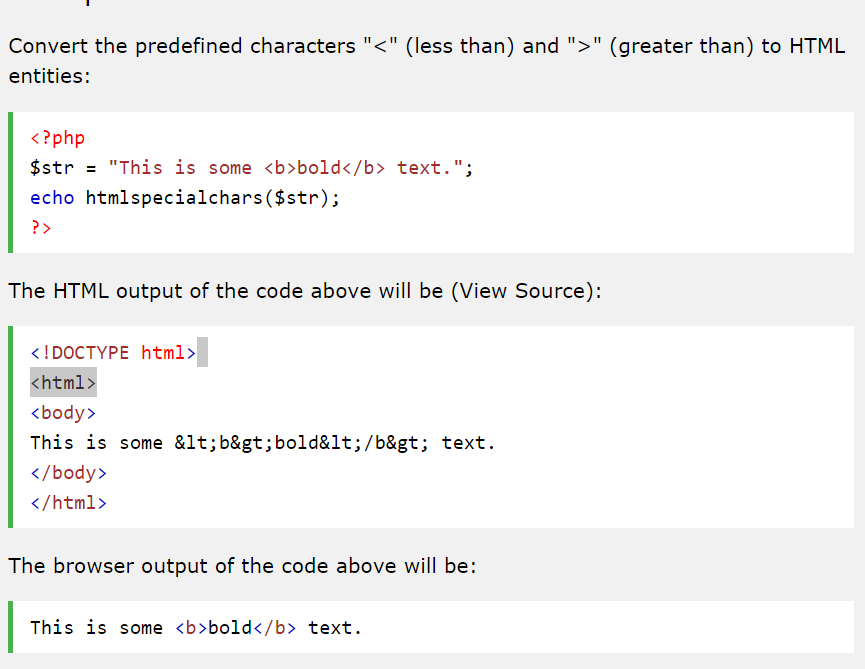
The next step is to create a function that will do all the checking for us (which is much more convenient than writing the same code over and over again).

We will name the function test\_input().

Now, we can check each $\_POST variable with the test\_input() function,

PAYLOAD- the actual information or message in transmitted data, as opposed to automatically generated metadata.The effects of a virus on a computer

T

his is the process of how htmlspecialchars shows the HTML output in a different way, Converting < and > into entities or symbols, are often used to prevent browsers from using it as an HTML element. This can be especially useful to prevent code from running when users or hackers have access to display input on your homepage using $SERVER[“POST\_SELF”],to get information or data from my website.

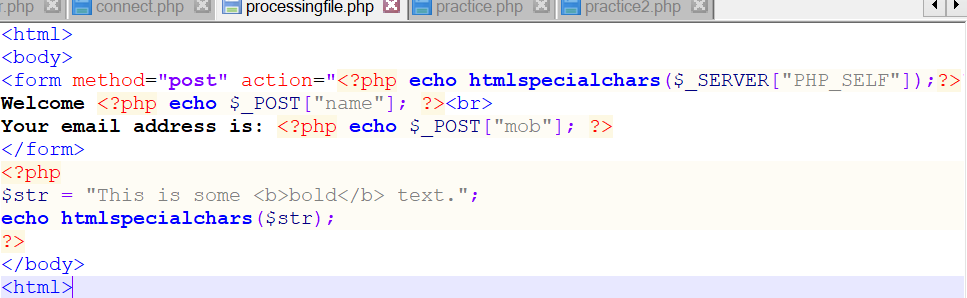
So when we open website and click right click>VIEW PAGE SOURCE,then we open the source code where the code is written in entity or symbol type as we used HTMLSPECIALCHARS in PHP to convert the code to that.thus hackers will get confused and if they copy the code and try to run that in notepad,then it will show error..

Fig:#ORIGINAL CODE

Example:see the difference between original code and source code:

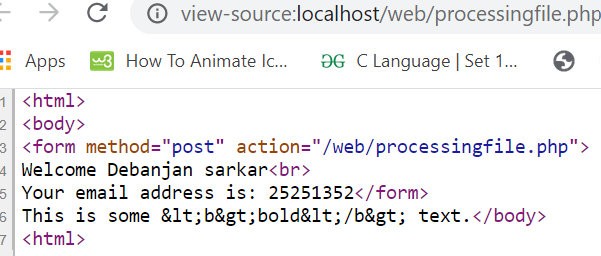


Fig:#SOURCE CODE

**URL parameters explained**

* **http://localhost/example.com**: Specifies the page request. It is the page the browser is requesting from the server.
* **?**: This character specifies where the query string for the requested page starts from. When it is omitted, the browser won't understand how to handle and send the query string (name/values pairs) to the server.
* **name1, name2**: Specifies the form field element's name attribute value. Each is assigned to its corresponding form field data, and it is used as the access key by the server script (PHP), to retrieve its data value when you fill out the form and submit it.
* **firstValue, secondValue**: These are the inputted values you entered before submitting the form. Each Value is assigned to its corresponding element's name attribute value.
* **&**: This character is used to concatenate the name/value pairs together as one long query string.
* **=**: This character is used to assign the name of the form field element to its data value as a name/value pair.

### **How to retrieve form data sent via GET**

When you submit a form through the GET method, PHP provides a superglobal variable, called $\_GET. PHP uses this $\_GET variable to create an associative array with keys to access all the sent information ( form data ). The keys is created using the element's name attribute values.

he PHP isset() function is used to determine if a variable is set and is not null.

Firstly, the isset() function checks if the form has been submitted by using the element's name attribute value "submit" (**name="submit"**) as key and pass it to the $\_GET[] superglobal variable. This is because the form data are stored in the $\_GET[] superglobal array by PHP when it is submitted through the GET method.

Then the form field, first name and last name form data are retrieved by using the same method, passing their respective name attribute values into the $\_GET['name as key'] array parameter, and each is assigned to a variable name that was used to display the results.

Using the POST

The form POST method sends information via HTTP header. All name/value pairs sent through this method is invisible to anyone else since all the information are embedded within the body of the HTTP request.

When you submit a form to a server through the POST method, PHP provides a superglobal variable called $\_POST. The $\_POST variable is used by PHP to create an associative array with an access key (**$\_POST['name as key']**). The key is created automatically by PHP when the form is submitted. PHP uses the form field element name attribute (**name="unique-name-here"**) to create the key.

Code explained

The $\_REQUEST variable script code works exactly the same way as the previous $\_GET and $\_POST code script above. The only task required is to replace the $\_GET and $\_POST with the $\_REQUEST variable.

The filter\_has\_var() function

It checks if a variable of a specified input type exists. It has two parameters, filter\_has\_var( type, variable\_name ), and both parameters are required. The first parameter **type** specifies the input type to check for, which can be any of the following constant values ( INPUT\_GET, INPUT\_POST, INPUT\_COOKIE, INPUT\_SERVER, INPUT\_ENV). The second parameter **input** specifies the variable name (the name attribute of the form input element, **name="unique-name-here"**) to check.

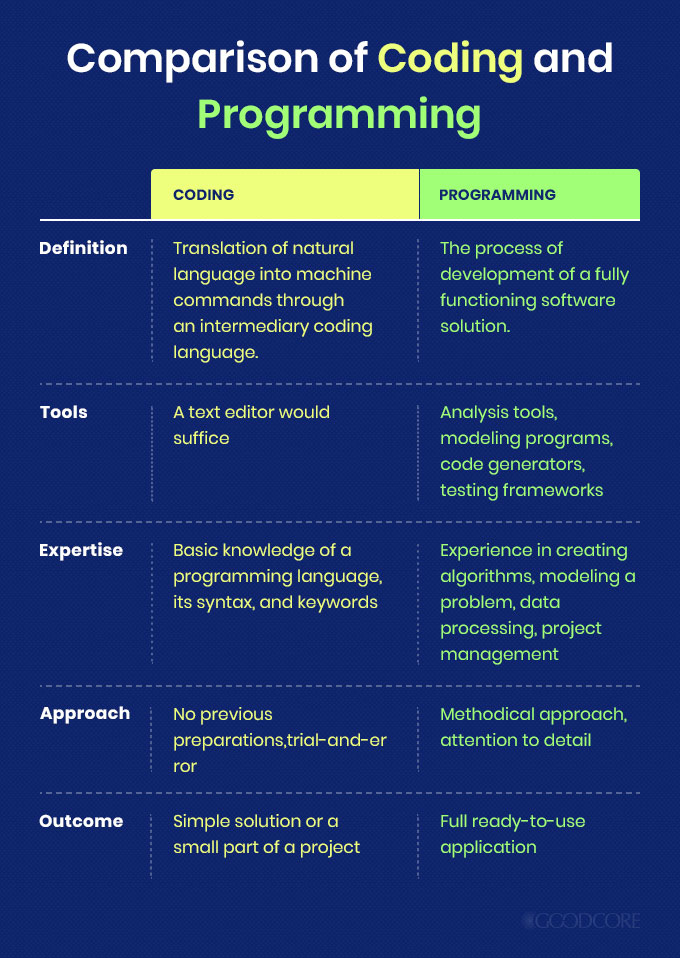
**Anatomy of the GET method**

* The GET method produces a long query string that is displayed in the browser's address bar when the form is submitted.
* The GET method should not be used to send sensitive content/information like password because all of the content/information is displayed in the browser's address bar.
* When sending form data through the GET method, you can only send a maximum of 2048 characters.
* The GET method cannot be used to send binary data like images, mp3 or pdf files to the server.
* When you submit a form through the GET method, PHP creates a $\_GET associative array in this format, $\_GET['name as key'] to enable you to retrieve the form data.
* The GET method is suitable to send non-sensitive content/information to the server.

**Anatomy of the POST method**

* The POST method sends information via HTTP header, all the information are embedded within the body of the HTTP request.
* The POST method can be used to send sensitive content/information since all data are sent through the HTTP header.
* This method has no limit on the amount of information to send to the server.
* The POST method provides support to send binary data like images, mp3 or pdf files, and also provides enhancement for file uploading to the server.
* When you submit a form through the POST method, PHP creates a $\_POST associative array in this format, $\_POST['name as key'] to enable you to retrieve the form data.

1. die() function prints a message in console an exits the current script..
2. echo can take multiple characters in the parameters which get concatenated
3. print can take only on eparameter that can be string or a integer..



**“Coding” and “Programming”** are the two most important approaches in Software Development Industries. Coding is basically the process of creating codes from one language to another one. It can also be called as a subset of Programming since it actually implements the initial steps of Programming. It involves writing codes in different languages as instructed. Programming is the process of developing an executable machine level program that can be implemented without any error. It is the process of formally writing codes so that the human inputs and corresponding machine outputs remain in sync.

* **User-level Threads**

The user-level threads are implemented by users and the kernel is not aware of the existence of these threads. It handles them as if they were single-threaded processes. User-level threads are small and much faster than kernel level threads. Also, there is no kernel involvement in synchronization for user-level threads.

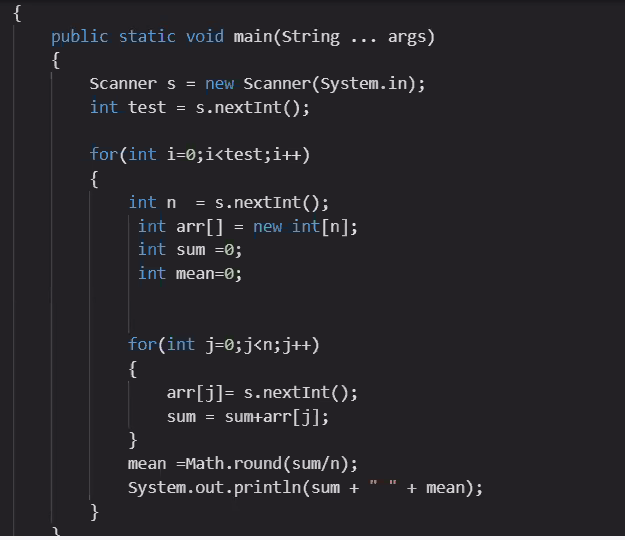
* **Kernel-level Threads**

Kernel-level threads are handled by the operating system directly and the thread management is done by the kernel. The context information for the process as well as the process threads is all managed by the kernel. Because of this, kernel-level threads are slower than user-level threads.

## Advantages of Multithreaded Processes

Some of the advantages of multithreaded processes are given as follows −

* All the threads of a process share its resources such as memory, data, files etc. A single application can have different threads within the same address space using resource sharing.
* It is more economical to use threads as they share the process resources. Comparatively, it is more expensive and time consuming to create processes as they require more memory and resources.
* Program responsiveness allows a program to run even if part of it is blocked using multithreading. This can also be done if the process is performing a lengthy operation.
* In a multiprocessor architecture, each thread can run on a different processor in parallel using multithreading. This increases concurrency of the system. This is in direct contrast to a single processor system, where only one process or thread can run on a processor at a time.



**BOOTSTRAP**

CODE:

.ml-1 {

margin-left: ($spacer \* .25) !important;

}

EXPLANATION:

The default value of $spacer is 1rem, i.e 16 pixels for most of the browsers. You can evaluate the outcome by multiplying 16 \* the fraction( .25, .5 etc),

2) <form class="form-inline ">

<input class="form-control mr-sm-2" type="search" placeholder="Search" aria-label="Search">

</form>

Explanation:

As we use ***mr-sm-2*** class on the input field, Which essentially says, set the right margin to size 2 on small breakpoint( between 576px to 767 px)

Now, if you check the right margin of the subject element, it will be 16 \* .5 = 8px.

ANOTHER EXAMPLE:

$spacer is a sass variable . its value is 16px in Bootstrap for example ml-1 mean margin-left with size 1 whose value is ($spacer \* 0.25) i.e 16px\*0.25 which is 1/4th value of 16px == 4px, therefore a size 1 for padding or margin stands for "$spacer \* .25" :4px. For size 2: $spacer \* .5== 16px\*0.5 == 8px. now.. size 3: $spacer\*1 [the exact value] == 16px. size 4: $spacer\*1.5==24px. size 5: $spacer\*3 == 48

PART 2:

## **Notation**

Spacing utilities that apply to all breakpoints, from xs to xl, have no breakpoint abbreviation in them. This is because those classes are applied from min-width: 0 and up, and thus are not bound by a media query. The remaining breakpoints, however, do include a breakpoint abbreviation.

The classes are named using the format {property}{sides}-{size} for xs and {property}{sides}-{breakpoint}-{size} for sm, md, lg, and xl.

Where property is one of:

* m - for classes that set margin
* p - for classes that set padding

Where sides is one of:

* t - for classes that set margin-top or padding-top
* b - for classes that set margin-bottom or padding-bottom
* l - for classes that set margin-left or padding-left
* r - for classes that set margin-right or padding-right
* x - for classes that set both \*-left and \*-right
* y - for classes that set both \*-top and \*-bottom
* blank - for classes that set a margin or padding on all 4 sides of the element

Where size is one of:

* 0 - for classes that eliminate the margin or padding by setting it to 0
* 1 - (by default) for classes that set the margin or padding to $spacer \* .25
* 2 - (by default) for classes that set the margin or padding to $spacer \* .5
* 3 - (by default) for classes that set the margin or padding to $spacer
* 4 - (by default) for classes that set the margin or padding to $spacer \* 1.5
* 5 - (by default) for classes that set the margin or padding to $spacer \* 3
* auto - for classes that set the margin to auto

(You can add more sizes by adding entries to the $spacers Sass map variable.)

* Add a dismiss button and the .alert-dismissible class, which adds extra padding to the right of the alert and positions the .close button.
* On the dismiss button, add the data-dismiss="alert" attribute, which triggers the JavaScript functionality. Be sure to use the <button> element with it for proper behavior across all devices.

**dropdown**-**toggle**-**split** for proper spacing around the **dropdown** caret. We use this extra class to reduce the horizontal padding on either side of the caret by 25% and remove the margin-left that's added for regular button **dropdowns**.

# What’s the difference between Scripting and Programming Languages?

Basically, all scripting languages are programming languages. The theoretical difference between the two is that scripting languages do not require the compilation step and are rather interpreted. For example, normally, a C program needs to be compiled before running whereas normally, a scripting language like JavaScript or PHP need not be compiled.  
   
Generally, compiled programs run faster than interpreted programs because they are first converted native machine code. Also, compilers read and analyze the code only once, and report the errors collectively that the code might have, but the interpreter will read and analyze the code statements each time it meets them and halts at that very instance if there is some error. In practice, the distinction between the two is getting blurred owing to improved computation capabilities of the modern hardware and advanced coding practices.  
   
Another point to be noted is that while classifying a language as scripting language or programming language, the environment on which it would execute must be taken into consideration. The reason why this is important is that we can design an interpreter for C language and use it as a scripting language, and at the same time, we can design a compiler for JavaScript and use it as a non-scripting(compiled language). A live example of this is V8, the JavaScript engine of Google Chrome, which compiles the JavaScript code into machine code, rather than interpreting it.  
   
Some scripting languages traditionally used without an explicit compilation step are JavaScript, PHP, Python, VBScript.  
   
Some programming languages traditionally used with an explicit compilation step are C, C++.  
   
   
**Applications of Scripting Languages :**

1. To automate certain tasks in a program

2. Extracting information from a data set

3. Less code intensive as compared to traditional programming languages

**Applications of Programming Languages :**

1. They typically run inside a parent program like scripts

2. More compatible while integrating code with mathematical models

3. Languages like JAVA can be compiled and then used on any platform