**html: hypertext markup language**

<!DOCTYPE html> declaration defines this document to be HTML5

<html lang = “en” > </html> root element of an html page

<head> </head> contains meta information about the document

<title> </title> specifies a title for the document

<body> </body> contains the visible page content

**Text tags:**

<p> </p> defines a paragraph

<br/> defines a line break

<h1> </h1>…<h6> </h6> denotes 1st…6th level of headers

<blockquote cite = “…”> </blockquote> defines a section that quoted from another source, indented

<strong> </strong> defines important text, bold

<em> </em> emphasis text, makes it italic

<sub> </sub> defines subscript text

<sup> </sup> defines supscript text

**Character entitles**

space: &nbsp

**Image tag:**

<img src = “…” alt = “…” width = “…” height = “…”>

must have src and alt attributes

**Links tags:**

<a href = “..”> </a> href attribute is required and indicates the resource

<a href = “..”> <img> </a> images as a link

target attribute:

target = “\_balnk” opens the link in a new window or tab

target = “\_self” opens the link in the same widow, default

title attribute:

title = “…” the information is most often shown as a tooltip text when the mouse over

email address:

<a href = “mailto: [xxx@xxx.xx](mailto:xxx@xxx.xx)” > </a>

**Lists tags:**

<li> </li> defines a list

<ul> </ul> unordered list

style = “list-style-type: xxx”

disc: bullet, default

circle: circle

square: square

none: will not be marked

<ol> </ol> ordered list

type = “xxx”

1: numbers, default

a: starts with lowercase letters

A: starts with uppercase letters

i: starts with lowercase roman letters

I: starts with uppercase roman letters

<dl> </dl> defines description list

<dt> </dt> defines description term

<dd> </dd> describe each term

**Table tags**

<table> </table> defines a table

<caption> </caption> add a caption

<th> </th> defines a table header

<tr> </tr> defines a row

<td> </td> defines a table data/cell

<th colspan = “x”> make a cell span more than one column

<th rowspan = “x”> make a cell span more than one row

**Forms tags:**

<form> </form> defines a form used to collect user input

<label> </label> sets a chunk of text as label

<input type = “xxx” name = “xxx” value = “xxx”> to collect data

name attribute: required

type = “text”: defines a one-line input field, default width is 20 characters

size = “x”: specifies the size for the input field

maxlength = “x”: specifies the maximum allowed length for input

type = “radio”: defines a radio button for selecting one of many choices

type = “checkbox” defines a checkbox let users select 0 to more options

type = “password”: defines a password field

type = “number”: defines a numeric input field, uses attributes to add restrictions. min/max = “x”

type = “submit”: defines a submit button for submitting the form

type = “reset”: defines a reset button that will reset all form values

type = “search”: used for search fields

type = “color”: show a color picker

type = “date”: show a date picker, and can use the min and max attribute to add restrictions to

date, max = “xxxx-xx-xx” min = “xxxx-xx-xx”

type = “time”: allow users to select a time

type = “email”: lets users input an email address

type = “file”: defines a file-select field and “Browse” button for file uploads

<textarea name = “xxx” rows = “x” cols = “x” > </textarea> defines a text area

<fieldset> </fieldset> group related data in a form

<legend> </legend> defines a caption for the fieldset

<select name = “xxx”> </select> defines a drop-down list

size = “x”: specifies the number of visible values

<option value = “xxx”> </option> defines an option that can be selected,

selected--attribute to define a pre-selected option

uses <p> tag to distinguish different parts of the form

**Media tags:**

<audio> </audio> specifies a standard way to embed audio in a web page

controls attribute: adds audio controls, like play, pause, and volume

<source src = “xxx” type = “audio/ogg(mpeg)”>

allows you to specify alternative audio file which the browser may choose form. The first recognized format will be played

<video width = “x” height = “x” > to show a video in html

controls attribute: same with audio

autoplay attribute: start a video antomatically

<source src = “xxx” type = “video/xxx”> same with audio

**HTML style attributes**

style = “attribute: xxx; ”

background-color: defines the background color for an html element

color: defines the text color

font-family: defines the font to be used

font-size: defines the text size

text-align: defines the horizontal text alignment

**CSS**: cascading style sheets, a language that describes the style of an html file

latter statements will override the formal

**Document-level style sheets:**

apply to the whole document is which they appear

styles defined in the head section

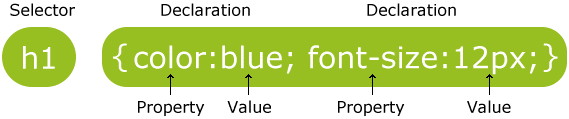
<style type = “text/css”>

**External style sheets:**

can be applied to any number of documents

specify style sheets to use in the head section

<link rel = “stylesheet” type = “text/css” href = “xxx.css”>

**CSS syntax:**

selector: html element that you want to style

**declaration:**

property and value

always ends with a semicolon ;

**frequently used patterns for selectors:**

#xx: selects the elements with id = “xx”—unique definition

.xxx: selects all elements with class = “xxx”—can be a group

\*: selects all elements

x: selects all <x> elements

x1, x2: selects all <x1> and <x2> elements

x1 x2: selects all <x2> inside <x1>

priorities: important! > inline style > id > class > tag > \*

the bottommost style will overlap the former styles

**Colors:**

color name, like gray, red, blue

HEX values: like #ff6347

RGB value: like rgb(255,0,0)

**Fonts:**

font-size: px – pixels; pt – points; em – iem is equal to the current font size; % - of the current size

font-family: holds several font names as a fallback system. If the browser doesn’t support the first, it

will try the next

font-style: used to specify italic text

normal: shown normally

italic: shown in italics

font-variant: specifies whether or not a text should be displayed in a small-caps (all lowercase letters

are converted upper, but the size is smaller than original upper) font

font-weight: specifies the weight of a font, can be normal, bold, bolder, and lighter

font shorthand: order must be: style, weight, size, name

**Text:**

color: set the color of text

text-align: set the horizontal alignment of a text, can be center, left, and right

text-decoration: set or remove decorations from text

none: often used to remove underlines from links

overline: add a line over the text

line-through: add a line through the text

underline: add an underline

text-transform:

uppercase: turn everything into uppercase

lowercase: turn everything int lowercase

capitalize: capitalizes the first letter of each word

text-indent: specifies the indentation of the first line

text-spacing: specifies the space between the characters

word-spacing: specify the space between the words

line-height: specifies the space between lines

**Links**

can be styled with any CSS property

a: link: a normal, unvisited link

a: visited: a link the user has visited

a: hover: a link when the user hovers (mouse over) over it, must come after link and visited

a: active: a link the moment it is clicked, must come after hover

color, background color

**List**

list-style-type:

unordered list: circle, square, none

ordered list: upper/lower-roman, upper/lower-alpha, numbers

list-style-image: url(‘xxx’): specifies an imagae as the list item marker

list-style-position:

outside: markers will be outside the list item, default

inside: markers will inside the list item

background, margin(-left/right ), padding(-top/bottom)

**Float:**

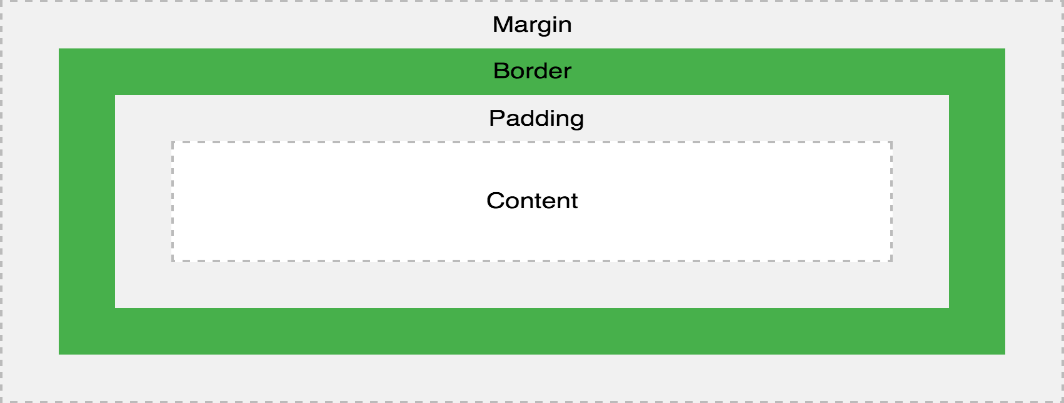
positions and formats content, text float around another elements (usually image or table)

left: floats to the left of its container

right: floats to the right of its container

none: the element does not float, default

inherit: inherits the float value of its parent

**Box model in CSS:**

**Border:**

4 values—top, right, bottom, left; 3 values—top, left/right, bottom

2 values—top/bottom, left/right; 1 value—all sides

border-top/bottom/left/right-xxx

border-style: specifies what kind of border to display

dashed, dotted, solid, double, groove, ridge, inset, outset, none, hidden, mix:

border-width: specifies the width of the four borders

thin, medium (default), thick, or a length value in pix

border-color: sets the color, if not set, inherits the element color

name, Hex, RGB, transparent

shorthand property:

border: width style(required) color

border-radius: add rounded borders

margin/padding(-top/bottom/left/right)

**Background:**

background-color: specifies the background color

background-image: url(“xxx”) : specifies an image to use as the background

background-repeat: sets the repeat way of image

repeat-x: repeated only horizontally

repeat-y: repeated only vertically

no-repeat: repeated once

background-position: sets the position of the image

background-attachment: fixed: specifies that the background image should be fixed

**Blocks tags (often used with CSS):**

<div> </div> used as a container for other html elements, to style blocks of content

useful for style specifications

<span> </span> used to style parts of the text

**Javascript:**

Javascript is the programming language of html and the web

the core of the language

client-side supports control of browser and interaction with users, more popular

server-side to control interaction with Web server

Inheritance: the capability of a class to rely on another class for some of its properties and methods

Polymorphism: the capability to write one function or method that works in a variety of ways

Javascript code referred to as a script, scripts can be explicitly or implicitly embedded in html

Explicit embedding in html code not always ideal

1. can be in the page’s <head> element: if is a script what reacts to user action or only when requested

2. can be in the page’s <body> element: when script that is interpreted only once

Implicit embedding in a separate file (.js)

1. hides the scripts from browsers

2. uses when JS code is meant for more than one page

**General syntax**

Import a JS file

<script type = “text/javaScript” src = “xxx.js”> </script>

Embed JS code

<script type = “text/javaScript”> </script>

**Output**

document.getElementByID(“id”). innerHTML=

to access an html element and writing into the element

document.write(“…”);

usually for testing purpose, using it after an html is loaded, will delete all existing html

window.alert(“…”);

displays data through an alert box

window.confirm(“…”)

used if you want the user to verify or accept something

window.prompt(“…”, “…”)

used if you want the user input value before entering a page

<button type = “button” onclick = “function”>xxx</button>

**Variables**

all JS variables must be identified with unique names called identifiers

names can contain letters, digits, underscores, and dollar signs, must begin with a letter

identify method: var name;

can store numbers, String, Boolean

null: a variable is null when not declared or not explicitly assigned a value, has only one value – null, cannot be used as the name of a function or variable, can erase the variables by assigning it to null

undefined: when declared but not assigned a value, has only one value – undefined

**lent:** have block{} scope, cannot be accessed from outside the block

**Numerical Operations:**

numeric operators: \_, -, \*, / , %

shortcut operators: ++, --, +=, -=, \*=, /=, %=

Math.max (x, y, z,…, n); returns the largest value

Math.ceil (x); returns x, rounded upwards to the nearest integer

Math.PI; returns pi

Math.sqrt(x); returns the square root of x

An arithmetic operation that creates overflow return NaN, which means not a number

isNaN(); check whether a value is NaN, if yes, then returns true, Boolean and null will be converted to 0 or 1, String will be converted to number if they can

toString(); converts a number to a String

String(num); converts num to a String with constructor

toFixed(a); return the number written with a specified number of decimals

parseInt(), parseFloat(); converts variables to a number

**String operations**

when both sides are numbers, + is addition; otherwise, String concatenation

when using non-string operator with strings, will try to convert string to number, or returns NaN

when using comparison operators

if one side is number, the other can be converted to a number, then number comparison

if one side is number, the other cannot be converted to a number, always false

if two strings, then string comparison based on the ASCII of the first different character

length; returns the length of a String

charAt(index); returns the character at the specified index

indexOf(st1); returns the index of the first found occurrence of the specified string; if not, -1

lastIndexOf(st1); returns the index of the last found occurrence of the specified string; if not. -1

endsWith(str1); checks whether a String contains the specified string

repeat(count); returns a new String with a specified number of copies

replace(str1,str2); replaces specified value str1 with str2, /str1/g-all values, /str1/gi-all values ignore case

substring(start,(end)); extract the characters form a String, including start but end

substr(start,length); extract form the start, but if a negative number, it will return from the last position

concat(str1); joins two or more strings

toLowerCase(); toUpperCase();

split(delimeter); split a string into an array of substrings

**Date**

var d =

new Date(); creates a new date object with the current date and time

new Date(year, month, day, hour, minute, second, millisecond)

new Date(dateString); creates a new date object from a date string

toLocalString(); returns a string of the date

getFullYear(); returns the year as a four digit number

getMonth(); returns the month as a number (0-11)

getDate(); returns the date as a number(1-31)

getHours(); returns the hour(0-23)

getMinutes(); returns the minute(0-59)

getSeconds(); returns the second(0-59)

getMilliSeconds(); returns the millisecond(0-999)

getTime(); returns the milliseconds since January 1, 1970

getDay(); returns the weekday(0-6)

also has the set mothods

**Arrays**

can have variables of different types in a same array, including objects, arrays, and functions

var a = new Array(x,x,x,x,x); new Array(x); creates an array with x undefined elements

var a = [x,x,x,x,x]; best format

accessing an undefined array entry gives the value undefined

assigning to an element beyond the end of the array increases its length

using a for loop to iterate an array

.length; returns the number of elements

toString(); converts an array to a string of array values; actually JS automatically converts

join(“\*”); like toString, but in addition you can specify the separator

pop(); removes and returns the last element from an array

shift(); removes and returns first element

unshift(x,x,…); inserts elements in front and returns new length

push(x,x,x,x); adds a new element to an array at the end, and returns the new array length

concat(arr2,…); returns a new array by joining the array with the given arrays

reverse(); changes the array itself to go backward

.sort(); sorts arrays alphabetically

split(delimeter); split a string into an array of substrings

splice(index, howmany, i1,i2,…); adds/removes items to/from an array, and returns the removed items

slice(index1.index2); slices out a piece of an array into a new array

forEach(function); calls a function once for each array element

**Comparison**

== equal to

==== equal value and equal type

!= not equal

!== not equal value or not equal type

>, < , >=, <=, &&, ||, !

**Control and flow statements**

for(statement 1; statement 2; statement 3){ codes}

for(var in arr){codes}

while(condition){}; do{}wihle();

if-else; switch

**Functions Syntax**

function name(param list) {codes}

can have parameters, no data type specified for params,

no check on the number of parameters received, missing params are set to undefined

JS functions have a built-in object called arguments object which is an array of params

function definitions placed in <head> of an html

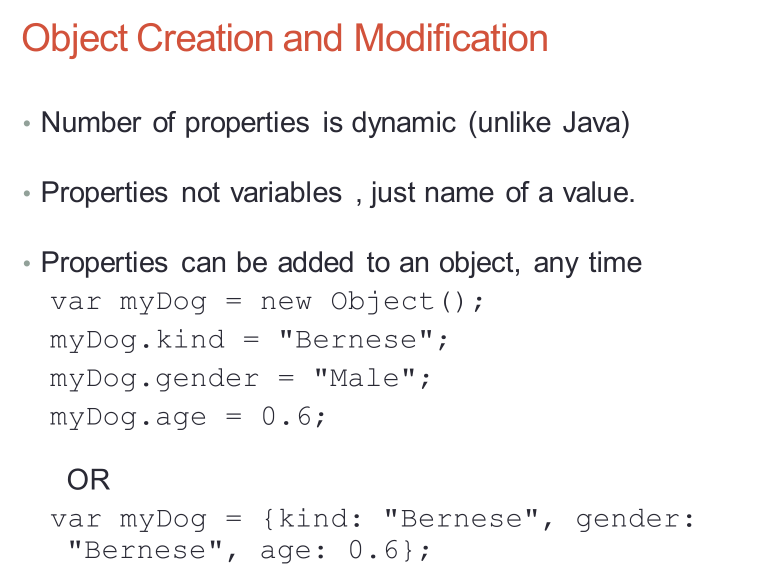
good habit to declare variables in a function

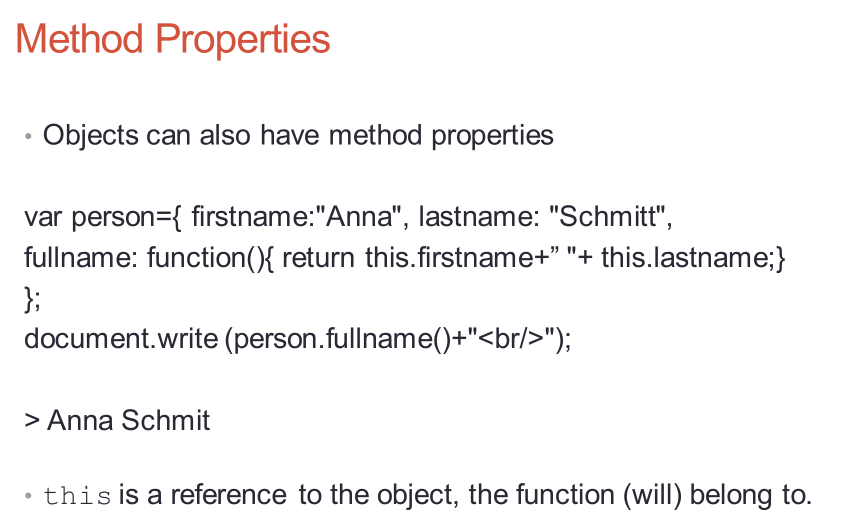
a variable declared with a var is local to that function and exists for the processing time of that function

a variable declared without a var is global, even if it is assigned inside a function

a variable declared at the top of a JS function lives on from the moment the file is loaded

**Object**

JS is not an OOP language



**Pattern matching**

using RegExp objects, a regular expression is a sequence of characters that forms a search pattern

syntax: /pattern/modifier

or using methods on string objects

normal characters (match themselves)

metacharacters (can have special meanings in patterns – do not match themselves)

\ | ( ) { } [ ] ^ $ \* + ? . period matches any character except new line

search(/pattern/)

returns the position in the object string of the pattern, returns -1 if it fails

**character classes**

put a sequence of characters in brackets [], and it defines a set of characters, any one of which matches

[abc] a single character of a, b, or c

[a-z] a single character in the range a-z

[^0-9] a single character not in the range 0-9

\d [0-9] a digit

\D [^0-9] not a digit

\w [A-Za-z\_0-9] a word character

\W [^A-Za-z\_0-9] not a word character

\s [\r\t\n\f] a whitespace character

\S [^\r\t\n\f] not a whitespace character

\d\* 0 or more digits

\d+ 1 or more digits

\d? 0 or 1 digit

/abc/ matches 1st occurrence of string ‘abc’

/ab\*b/ begins with one a, followed by any number of one b, followed by one c

/ab+c/ begins with one a, followed by at least one b, followed by one c

/ab{2,}c/ begins with one a, followed by two or more b followed by one c

/(ab)\*c/ begins with some ab, followed by one c

/(ab)?c/ begins with at most one ab, followed by one c

/[abc]/ matches 1st occurrence of a, b, or c

/^[abc]\d/ starts with one a, b, or c, followed by a digit

/^[abc]/ is the letter a, b or c, must be at the front of string

/[abc]\d$/ has the letter a, b, or c, followed by a digit at the end

/\d{1,2}/ matches 1st occurrence of one or two digits

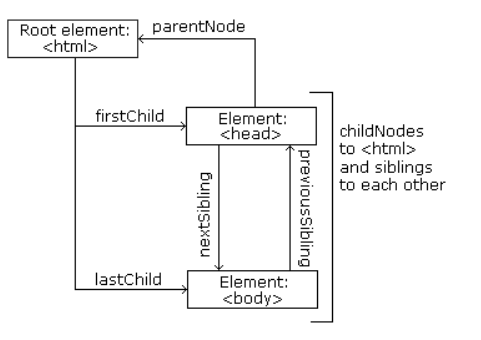
/\d[1-9]\d/ matches 1st occurrence of one 2 digits number not starting with 0

(?!.\*u) do not include letter u

^(?![0-9]+$)(?![a-zA-Z]+$)[0-9A-Za-z]{8,16}$

**DOM**

a standard for accessing documents: Document Object Model

a standard object model and programming interface for HTML. It defines: the HTML elements as objects, the properties of all HTML elements, the methods to access all HTML elements, the events for all HTML elements. In other words, DOM is a standard for how to get, change, add, or delete HTML elements

In the DOM, all HTML elements are defined as objects:

a property is a value that you can get or set

a method is an action you can do

getElemmentById(“id”); most common way to access an HTML element

finding an element by element id

getElementsByTagName(“tag”); find elements by tag name

getElementsByClassName(“class”); find elements by class name

return a Nodelist of all elements with the specified name or classname

element can be accessed through their node number

innerHTML; the easiest way to get and change the content of an element

setAttribut(“attribute”, “newvalue”); change the attribute of an element

style.attribute = newvalue; change the style of an element

document.createElement(element); create an html element

document.removeChild(element); remove an html element

document.appendChild(element); add an html element or element node

document.replaceChild(new, old); replace an html element

doucument.write(text); write in the html output stream

document.createTextNode(text); create an text node

document.forms.length; get the number of form elements

to add a new element to the HTML DOM, you must create the element (element node) first, and then append it to an existing element

node properties to navigate between nodes with js

parentNote, childNodes[nodenumber], firstChild, lastChild, nextSibling, previousSibling

**access elements:**

1. DOM address: use the forms and element arrays of the Document object

2. Element names: requires the element and all of its ancestors (except body) to have name attributes

3. getElementById method

**Navigation:**

nodeName—specifies the name of a node

nodeValue—specifies the value of a node

**event attributes:**

onclick, onload, onmouseover, onmouseout, onblur,

**PHP**

Php is a server scripting language, and a powerful tool for making dynamic and interactive web pages.

Internal: <?php ?>

External: include (“myScript.inc”)

**Variable:**

Name always begin with $, can contain letters, digits, underscore, cannot start with a digit, are case sensitive

An unassigned variable has the value, NULL

unset($var); //sets a variable to NULL

isset($var); //returns true if the variable is not null

**Boolean:**

FALSE and TRUE – case sensitive

If an integer, 0-false; if a string, empty or string “0” is false; if a double, 0.0 si false

**String:**

Defined with ‘’ or “”

With “” will substitude value of variable, while with ‘’ will not

strlen(str); //returns the number of characters in str

str\_word\_count(str);// counts the number of words

strrev(str); // reverses a string

strpos(str, str); //searcher for a specific text within a string and return the index

trim(str); //with spaces from both sides

ltrim(str); //removes the spaces from its beginning

str\_replace(str1,str2,str3); //replaces str1 of str3 with str2

substr(str,start,len);// returns a substring of str form start to end or to the length

strtolower, strtoupper

strcmp(str1,str2);//if str1 is greater than 2, reutrns positive integer

ucwords(str);//returns a given string with the first letter of each word in uppercase

explode(“delimeter”,str);//split string with the delimiter and store in an array

implode(“delimeter”,arr);//catenates the elements of array with delimeter

**scalar type conversion**

implicit: coercions

context of expression determines type that is expected or require

if string in a numeric context, it’s converted; otherwise, it’s 0

explicit: casts

(int)$var intval($var) settype($var, “integer”)

is\_int(var) is\_integer is\_long is\_double is\_float is\_bool is\_string

**output:**

print:

takes a string, but will coerce other values to strings

printf:

printf(literal\_string, param1,param2,…)

%10s-for string %6d-for integer %5.2f for float, 5 is whole spaces, 2 is the decimal part

**Arrays**

To create an array use the array() constructor, which takes one or more key=>value pairs as parameters, the keys are non-negative integer or string

If a key is omitted, 0 is default key, if more than a key omitted and there have been integer keys, the default key will be the largest current key +1. If a key appears that has already appeared, the new value will overwrite the old one.

$arr = array(key=>value,…); $arr[key]=value

for loop and for each loop:

foreach($arrayName as $elementName)

count(arr); or sizeof(arr)//returns the number of entries in array

array\_keys(arr);//returns an array of the keys of the given array

array\_values(arr);//returns an array of the values of the given array

is\_array(arr);//returns true if it’s an array

array\_pop(arr,$var);//deletes and returns last entry

array\_push(arr,$var1,$var2,…);//inserts one or more elements to the end of the array

array\_shift(arr,$var);//deletes and returns the first entry

array\_unshift(arr,$var,…);//inserts one or more elements at the beginning

array\_reverse(arr);// returns an array in the reversed order

sort(arr) or rsort(arr);// sorts the value in increasing or decreasing, replacing the keys with 0,1,2,…

asort(arr) or arsort(arr);// sort the value, but doesn’t change keys

ksort(arr) or krsort(arr);//sort the array with key order

array\_intersect(arr1,arr2);//compare arrays and return the common elements as an array

in\_array($var,arr);//return true if the var is in the array

**function**

similar to js

function names are not case sensitive

to specify pass-by-reference, prepend an ampersand & to the formal parameter

if a function returns a reference, the name of the function must have a prepended ampersand

an undeclared variable in a function has the scope of the function

to access a nonlocal variable, it must be declared to be global. Global $var;

**pattern matching**

preg\_match (regex, str);// returns a Boolean value

preg\_split (regex, str);// returns an array of the substrings split by regex

**form handling**

user enters information in a form and presses the submit button

need to validate form data to pretext malicious code

GET method

Can be cached, remain in the browser history, can be bookmarked, are visible to everyone, should never be used when dealing with sensitive data, have length restrictions (2000 characters), result in an implicit array $\_GET

POST method

Are never cached, do not remain in the browser history, cannot be bookmarked, have no restrictions on data length, result in an implicit array $\_POST.

empty(var);//returns false if the variable exists and has a non-empty, non-zero value

“”, 0, 0.0, “0”, NULL, FALSE, undefined variable—are considered empty

isset(var,var,…);// returns true if set to any value except NULL

**File I/O**

Provides a complete set of file and directory functions enabling you to easily access and manipulate files and folders on the local file system

Tests: file\_exists, is\_file, is\_dir, is\_readablle, is\_writable, is\_executable, filesize

Syntax: function\_name($filename)

fileatime—last access time of file

fileectime—inode change time of file

filegroup—file group

filemtime—file modification time

fileowner—file owner

fileperms—file permissions

filetype—file type

copy—copies a file

unlink—deletes a file

rename—renames a file

fopen—opens file for I/O, returns handle—mode: w, r, a

fclose—closes handle, returns a Boolean

feof—out put lines until reach the end of file

fflush—writes all buffered output to open file

fwrite(file, str)—writes to handle

fread(file, length)—reads from handle

fget(handle)—read file as lines

readfile(file)—sends file to output buffer

file(name)—returns the lines of the given file into an array

file\_get\_contents(name)—returns the file contents as a string

mkdir—creates a directory

rmdir—removes a directory

chdir—changes dir

to include other php files in a php script

include(file)—generates a warning, but the script will continue executing

require(file)—generates a fatal error, and the script will stop

**cookie**

cookies are used to often to perform session management, user identification, tracking, and analytics.

A cookie is a name/value pair that is passed between a browser and a server in the http header, a small file that the server embeds on the user’s computer, each time the same computer requests a page with a browser, it will send the cookie too. With php, you can both create and retrieve cookie values

setcookie(name, value, time()+(84600\*30), “/”);// creates and returns a Boolean

$\_COOKIE[“numOfVistits”] = 0; $count = $\_COOKIE[“numOfVistits”] + 1;

setcookie(“numOfVisits”, $count, expire)—set a cookie for number of visits, and increment this counter

isset($\_COOKIE[“numOfVistits”])—to see if a cookie has already been set

**sessions**

a session is a conversation between the server and a client, consisting of series of continuous request and response, stored on server side and closed once user closes browser.

Session variables hold information about one single user, and are available to all pages in one application

When a session is started:

-Php first creates a unique identifier for that particular session which is a random string of 32 hexadecimal.

-A cookie called PHPSESSID is automatically sent to the user’s computer to store unique session identification string

-A file is automatically created on the server in the designated temporary directory

-when a php file script wants to retrieve the value from a session variable, php automatically gets the unique session identifier string from the PHPSESSID cookie and then looks in its temporary directory for the file bearing that name and a validation can be done by comparing both values

-a session ends when the user closes the browser or after leaving the site, the server will terminate the session after a predetermined period time, commonly 30 mins duration

<?php session\_start(); ?>

Checks to see if a session has already been started, then if none currently exists starts one

Also alerts php engine to expect session variables

$\_SESSION: a global associative array which holds the session states/variables

$\_SESSION[‘count’] = $counter---to create/store a session state/variable

unset($\_SESSION[‘count’])—to delete/free a session state/variable

session\_unset()

session\_destroy()—to destroy a session