HTML

The **language** specifies code for formatting, both the layout and style, within a text file. The code used to specify the formatting are called tags. HTMLis a an example of a widely known and used **markup language**.

what is use of HTML doctype

The **doctype** declaration is not an **HTML** tag; it is an instruction to the web browser about what version of the markup language the page is written in.

HTML is a document that contains text.

**HTML storage API:** An API is simply a messenger that takes the request and tells the system what to do and returns the response to you. Waiter

**HTML Local storage API:**

With the local storage web applications can store the data locally with in user’s browser.

Local storage is more secure, and large amounts of data can be stored locally, without affecting website performance.

HTML local storage provides two objects for storing data on the client:

* window.localStorage - stores data with no expiration date
* window.sessionStorage - stores data for one session (data is lost when the browser tab is closed)

## The localStorage Object

The localStorage object stores the data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week, or year. Count increases when the User click a button.

<!DOCTYPE html>

<html>

<body>

<div id="result"></div>

<script>

// Check browser support

if (typeof(Storage) !== "undefined") {

// Store

localStorage.setItem("lastname", "giri");

// Retrieve

document.getElementById("result").innerHTML = localStorage.getItem("lastname");

} else {

document.getElementById("result").innerHTML = "Sorry, your browser does not support Web Storage...";

}

</script>

</body>

</html>

The session storage is same as local storage but the only diff is it keeps the data for only one session.

SVG is a language for describing 2D graphics in XML.

Canvas draws 2D graphics, on the fly (with a JavaScript).

Document:



jQuery is a lightweight, "write less, do more", JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

There are several ways to start using jQuery on your web site. You can:

* Download the jQuery library from jQuery.com
* Include jQuery from a CDN, like Google

If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).Both Google and Microsoft host jQuery.

Jquery syntax is: **$(*selector*).*action*()**

* A $ sign to define/access jQuery
* A (*selector*) to "query (or find)" HTML elements
* A jQuery *action*() to be performed on the element(s)

jQuery selectors allow you to select and manipulate HTML element(s).

jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more.

Element Selector:

The jQuery element selector selects elements based on the element name. $("p")

The jQuery #id selector uses the id attribute of an HTML tag to find the specific element. $("#test")

The jQuery class selector finds elements with a specific class. $(".test")

An event represents the precise moment when something happens.

**$(document).ready()**

The $(document).ready() method allows us to execute a function when the document is fully loaded.

**click()**

The click() method attaches an event handler function to an HTML element.

$("p").click(function(){  
    $(this).hide();  
});

## jQuery DOM Manipulation

One very important part of jQuery is the possibility to manipulate the DOM.

jQuery comes with a bunch of DOM related methods that make it easy to access and manipulate elements and attributes.

**DOM = Document Object Model**  
  
The DOM defines a standard for accessing HTML and XML documents:  
  
*"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*

The Markup Validation Service is a [validator](https://en.wikipedia.org/wiki/Validator" \o "Validator) by the [World Wide Web Consortium](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium) (W3C) that allows [Internet](https://en.wikipedia.org/wiki/Internet) users to check [HTML](https://en.wikipedia.org/wiki/HTML) and [XHTML](https://en.wikipedia.org/wiki/XHTML) documents for [well-formed](https://en.wikipedia.org/wiki/Well-formed_element) markup. Markup validation is an important step towards ensuring the technical quality of web pages.

Get Content:

Three simple, but useful, jQuery methods for DOM manipulation are :

* text() - Sets or returns the text content of selected elements
* html() - Sets or returns the content of selected elements (including HTML markup)
* val() - Sets or returns the value of form fields

## Set Content - text(), html(), and val()

We will use the same three methods from the previous page to **set content**:

* text() - Sets or returns the text content of selected elements
* html() - Sets or returns the content of selected elements (including HTML markup)
* val() - Sets or returns the value of form fields

jQuery traversing, which means "move through", are used to "find" (or select) HTML elements based on their relation to other elements. Start with one selection and move through that selection until you reach the elements you desire.

An ancestor is a parent, grandparent, great-grandparent, and so on.  
A descendant is a child, grandchild, great-grandchild, and so on.  
Siblings share the same parent.

AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page.

AJAX = Asynchronous JavaScript and XML.

In short; AJAX is about loading data in the background and display it on the webpage, without reloading the whole page.

Examples of applications using AJAX: Gmail, Google Maps, Youtube, and Facebook tabs.

jQuery provides several methods for AJAX functionality.

With the jQuery AJAX methods, you can request text, HTML, XML, or JSON from a remote server using both HTTP Get and HTTP Post - And you can load the external data directly into the selected HTML elements of your web page!

## jQuery load() Method

The jQuery load() method is a simple, but powerful AJAX method.

The load() method loads data from a server and puts the returned data into the selected element.

$(*selector*).load(*URL,data,callback*);

The jQuery get() and post() methods are used to request data from the server with an HTTP GET or POST request.

wo commonly used methods for a request-response between a client and server are: GET and POST.

* **GET** - Requests data from a specified resource
* **POST** - Submits data to be processed to a specified resource

GET is basically used for just getting (retrieving) some data from the server. **Note:** The GET method may return cached data.

POST can also be used to get some data from the server. However, the POST method NEVER caches data, and is often used to send data along with the request.

## jQuery $.get() Method

The $.get() method requests data from the server with an HTTP GET request.

$.get(*URL,callback*);

AngularJS is a **JavaScript framework**.  It is a library written in JavaScript. It can be added to an HTML page with a <script> tag.

AngularJS extends HTML attributes with **Directives**, and binds data to HTML with **Expressions**.

AngularJS extends HTML with **ng-directives**.

The **ng-app** directive defines an AngularJS application.

The **ng-model** directive binds the value of HTML controls (input, select, textarea) to application data.

The **ng-bind** directive binds application data to the HTML view.

 AngularJS directives are HTML attributes with an **ng** prefix

AngularJS expressions are written inside double braces: **{{ expression }}**.

AngularJS expressions bind AngularJS data to HTML the same way as the **ng-bind** directive.

AngularJS modules define applications:

AngularJS Module

var app = angular.module('myApp', []);

AngularJS controllers control applications:

AngularJS Controller

app.controller('myCtrl', function($scope) {  
    $scope.firstName= "John";  
    $scope.lastName= "Doe";  
});

AngularJS has a set of built-in directives which you can use to add functionality to your application.

The **ng-app** directive defines the application, the **ng-controller** directive defines the controller.

## AngularJS Directives

AngularJS directives are extended HTML attributes with the prefix ng-.

The ng-app directive initializes an AngularJS application.

The ng-app directive defines the **root element** of an AngularJS application.

The ng-app directive will **auto-bootstrap** (automatically initialize) the application when a web page is loaded.

The ng-init directive initializes application data.

The ng-init directive defines **initial values** for an AngularJS application.

Normally, you will not use ng-init. You will use a controller or module instead.

The ng-model directive binds the value of HTML controls (input, select, textarea) to application data.

Data binding in AngularJS binds AngularJS expressions with AngularJS data.

{{ firstName }} is bound with ng-model="firstName".

The ng-repeat directive repeats an HTML element:

The ng-repeat directive actually **clones HTML elements** once for each item in a collection.

**Data binding** in AngularJS is the synchronization between the model and the view.

AngularJS applications usually have a data model. The data model is a collection of data available for the application.

Data model Example:

var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
    $scope.firstname = "John";  
    $scope.lastname = "Doe";  
});

The HTML container where the AngularJS application is displayed, is called the view.

The view has access to the model, and there are several ways of displaying model data in the view.

You can use the ng-bind directive, which will bind the innerHTML of the element to the specified model property:

You can also use double braces {{}} to display content from the model:

Or you can use the ng-model directive on HTML controls to bind the model to the view.

Use the ng-model directive to bind data from the model to the view on HTML controls (input, select, textarea) . The ng-model directive provides a two-way binding between the model and the view.

## Two-way Binding

Data binding in AngularJS is the synchronization between the model and the view.

When data in the model changes, the view reflects the change, and when data in the view changes, the model is updated as well. This happens immediately and automatically, which makes sure that the model and the view is updated at all times.

<div ng-app="myApp" ng-controller="myCtrl">  
  
First Name: <input type="text" ng-model="firstName"><br>  
Last Name: <input type="text" ng-model="lastName"><br>  
<br>  
Full Name: {{firstName + " " + lastName}}  
  
</div>  
  
<script>  
var app = angular.module('myApp', []);  
app.controller('myCtrl', function($scope) {  
    $scope.firstName = "John";  
    $scope.lastName = "Doe";  
});  
</script>

Application explained:

The AngularJS application is defined by  **ng-app="myApp"**. The application runs inside the <div>.

The **ng-controller="myCtrl"** attribute is an AngularJS directive. It defines a controller.

The **myCtrl** function is a JavaScript function.

AngularJS will invoke the controller with a **$scope** object.

In AngularJS, $scope is the application object (the owner of application variables and functions).

The controller creates two properties (variables) in the scope (**firstName** and **lastName**).

The **ng-model** directives bind the input fields to the controller properties (firstName and lastName).

In AngularJS, a service is a function, or object, that is available for, and limited to, your AngularJS application.

The $location service has methods which return information about the location of the current web page:

The AngularJS $http service makes a request to the server, and returns a response.

AngularJS is perfect for displaying data from a Database. Just make sure the data is in JSON format.

Angular 2 is an open source JavaScript framework to build web applications in HTML and JavaScript and has been conceived as a mobile first approach.

## Features

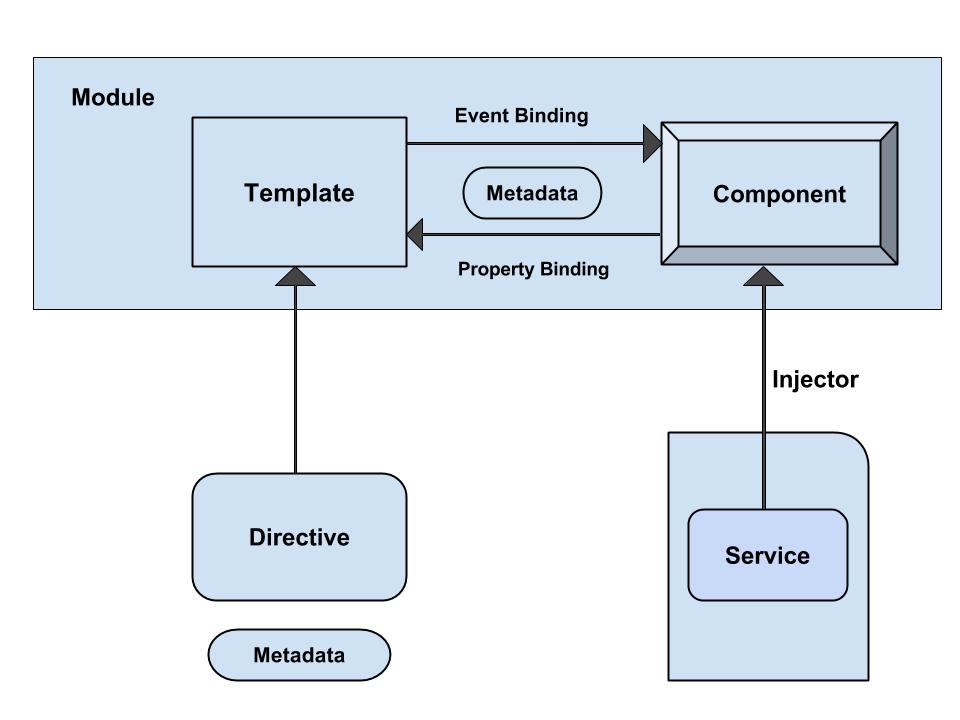
* Angular 2 is faster and easier than Angular 1.
* It supports latest the version of browsers and also supports old browsers including IE9+ and Android 4.1+.
* It is a cross platform framework.
* Angular 2 is mainly focused on mobile apps.
* Code structure is very simplified than the previous version of Angular.

## Advantages

* If an application is a heavy load, then Angular 2 keeps it fully UI responsive.
* It uses server side rendering for fast views on mobile.
* It works well with ECMAScript and other languages that compile to JavaScript.
* It uses dependency injection to maintain applications without writing too long code.
* Everything will be the component based approach.

Angular uses *TypeScript* which is a primary language for developing of Angular applications. The *TypeScript* is a super set of JavaScript which is migrated to TypeScript and code written in TypeScript makes less prone to run time errors.

 The following diagram shows architecture of Angular 2:



The architecture of Angular 2 contains following modules:

* **Module**
* **Component**
* **Template**
* **Metadata**
* **Data Binding**
* **Service**
* **Directive**
* **Dependency Injection**

The module component is characterized by a block of code which can be used to perform a single task.

A component is a controller class with a template which mainly deals with a view of the application and logic on the page. It is a bit of code that can be used throughout an application. Component knows how to render itself and configure dependency injection.

The component's view can be defined by using the *template* which tells Angular how to display the component.

**Data binding is a process of coordinating application data values by declaring bindings between sources and target HTML elements. It combines the template parts with components parts and template HTML is bound with markup to connect both sides. There are four types of data binding:**

* **Interpolation**: It displays the component value within the div tags.
* **Property Binding**: It passes the property from the parent to property of the child.
* **Event Binding**: It fires when you click on the components method name.
* **Two-way Binding**: This form binds property and event by using the *ngModel* directive in a single notation.

Web services: A web service is a way of calling a func which is inside of software from some other software.

s/w 1

Calls

Func()

s/w 2

It is a way of communication between softwares which is language interoperable.

Client(service consumer) would need a file(web services Descriptive language(WSDL🡪XML format )) in order to access the functions of server (service provider)

If the server knows the client they can directly provide the WSDL file to client. If the server doesn’t know the client then the server will keep all the files in UDDI (Universal Description Discovery and Integration).

Web services:

1. REST( Representation State Transfer): set of guide lines on which client should interact with server.

Total 8+yrs exp.

**Northwestern Mutual, Franklin, WI,** Senior Front-end Developer with node JS, Jan 2016 to Present Manager: Raja sekhar, Adam Curtright

Fair way meadows apartments, 7585 W

Drexel Ave, Franklin, WI 53132, W Drexel Ave

**Sterling bank, Houston, TX**,Senior Web/UI Developer, Oct 2014 to Dec 2015

**Belk, Inc. charlotte NC,**,Senior Web application Developer, Sep 2013 to Oct 2014

**Parkview Health System, Fort Way, Indiana**, UI Developer, Sep 2012 – Aug 2013

**India-4 years**

**Amar Raja group, Hyderabad,India , Web/UI Developer , July 2009 – Aug 2012**

**First Source Solution**,**UI Developer/ Designer, Sep 2008 - July 2009**

2012 – L1 visa

Sep, 2013- H1

2016- H1 ext (10/01/2016 to 09/17/2019)

AngularJS (less about Angular as a full blown language)

·         JSON

o   Karma / Jasmine (or Mocha)

**o   SystemJS / JsPM** (nice to haves)

o   **Node / NPM** (nice to haves)

·         Java (version 7 or 8)

·         SQL (ability to write queries to get/put/patch/etc. data)

·         REST Web Services (write end-points)

·         Python (would be a huge plus)