**Node.js**

Node.js is the latest in the programming languages ,API’s ,and toolkits’ that sense, it lands equally in the tradition of Ajax , Hadoop,and even to some degree of IPhone programming and HTML5 . If you go to some technical conference , and you’ll almost certainly find a few talks on Node.js, most of them will fly far over the head of the common mortal programmer .

Lets know more about it , and you will hear that Node.js or many simply call it “NODE” . It’s a server-side solution for javaScript and in genral for receving and responging to HTTP requests. The questions arise is it really a java script and why anyone wants to run javascript outside the javaScript why not the alone at the server side .

If you thinking about this it means you going along the right things . Node is really is concerned about network programming and server-side requests/response processing . Node.js applications are written in javaScript, and can be run within the Node.js runtime on OS X ,Microsoft ,Windows and Linux.

Node.js also provides a rich library of various javaScript modules which simplifies the development of web application using Node.js to a great extent .

**Node.js = Runtime Enviorment + JavaScript Library**

**Features of Node.js**

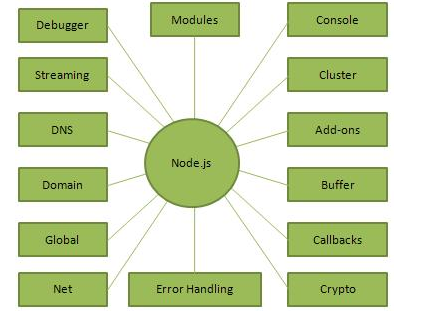
Following are some of the important features that make Node.js the first choice of software architects .

* Asynchronous and event driven - All the APIs of the Node.js library are asynchronous , that is non-blocking . It essentially means a Node.js based server never waits for an APIs to return data.The server moves to the next API after calling the notification mechanism of events of Node.js helps the server to get a response from the previous API call .
* No Buffering – There is never buffering in Node.js application. These application simply output data .
* Very Fast – Its build on google chrome V8 javaScript engine , that why its very fast in code execution .
* Single Threaded but highly scalable – Node.js uses a single threaded model with event looping . Event mechanism helps the server to respond in a non-blocking way and makes it very highly scalable as compared to other servers which creates limited threads to handle requests . Node.js uses a single threaded and at the same time same program can be used to provide to a much larger number of other requests than other servers like Apache and HTTP Servers .
* Licence – Node.js is released to under MIT license .

**Where its Used**

Now days big IT companies uses Node.js . This includes eBay, General Electric , GodDaddy ,Microsoft,Paypal , Uber,Wikipins,Yahoo, and Yammer.

**Concepts**

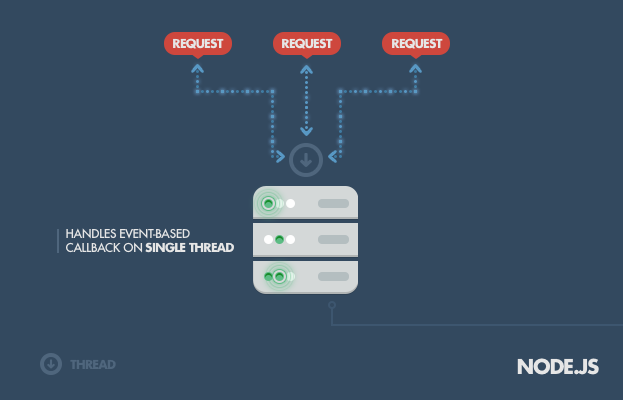
****

**How does it work**

The main concept of Node.js is to use the non – blocking , event-driven I/O to remain lightweight and efficient in the face of data-intensive real time application that run across distributed devices .

That means Node.js is a platform that fills out a need of a particular application. Furthermore, understanding of it is absolutely essential. You definitely don’t want to use Node.js for CPU-intensive operations, infact using it for heavy computation will annual nearly all of its advantages . Node.js is really fast and compatible in building fast ans scalable network application, as its capable of handling very high number of throughput , which leads it to very high scalable.

How it works under-the-hood seems very interesting . Compared to other traditional web-serving techniques where each request handled by a new thread , taking upi system RAM and eventually maxing-out the amount of the RAM available, Node.js is operate on a single-thread, using non-blocking I/O call , allowing it to support thousands of connection at a time .



**Why we used Node.js**

Node.js has been enjoying developer’s attention ever since its launched . Node.js is a open source cross-platform run time environment has been written in javaScript, Which is why it makes it an exception choice for real time applications. Node.js comes with many features which helps developers to use for web development. Just because effective use of javaScript Node allows developers to create servers.

**It’s Fast -** Node.js has been written in javaScript and has phenomenal running speed and it uses a V8 engine by google that makes it running code much faster.

**It’s good for real time application –** Node.js has a phenomenal speed for real time applications such as chats , gaming apps and event based server as well as non-blocking driven servers. It is worth mentioning that there is some example websites we have been using in every day life like PayPal,eBay ,LinkedIn etc and they all have been developed using Node.

**Data streaming –** In real time requests and responses are taken as isolated events , although they are actually data streaming in real and it reduces the complete processing time.

**Node.js server as a proxy server –** Node.js has effective feature that may be employed to proxy a number of services with difference In their response time. It can be used to stream data from from different sources for eg : if a server side application is employed for communicating with third party resources , storing images or collecting data additionally, at that time node.js can be used and in our case we pulling information from API .

**Effective single codebase –** Node.js has proved as a game changing. As we able to write javaScript server side and client side as well. It became easy to send and synchronize the data between these two points automatically helped us to save time .

Express.js

**Expressjs**.com. **Express**.**js**, or simply **Express**, is a web application framework for Node.**js**, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It is the de facto standard server framework for Node.**js**.

Middleware is the main concept behind Express.js request processing and routing . If you understand how middle ware works we can create a more applications by writing less code. The moment a request is received by an Express.js app it invokes various functions referred to as middleware.

Middleware

Middleware is any number of functions that are invoked by Express.js routing layer before your final request handler is, and its sits in the middle between a raw request (made from front end) and final intended route. We often refer to these functions as the middleware stack they are always in order they are added.

Take a simple Express.js example

Var app = express();

app.get(‘/’ , function (req , res) {

res.send(‘Hello World’);

});

app.get(‘/help’,function(req,res){

res.send(‘Nothing here’);

});

The above Express.js app will respond with Hello World to a request for / and Nothing here for requests to /help.

This is only a simple basic example showing how long every request your applications receive. Middleware can be as simple as one line or it can be complex as a session handling. Middleware is commonly used to perform tasks like body parsing for url-encoding or json request ,cookie parsing for basic cookie handling or even building javaScript modules on the fly.

The reason we used it because **Easy learning curve**.There are lot of inbuilt, third-party middleware compatible with it. It overall provides complete structure in clean syntax.It is just a set of ready functions and modules that can be used easier than writing code from 0. It saves time.

AngularJS

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML’s syntax to express your application components clearly and succinctly. AngularJs is a data binding and dependency injection eliminate much of the code we have to write otherwise. And all it happens within the browser, making it ideal partner with any server technology.

AngularJs is what html was before, It had been designed for application. HTML is a declarative language for static documents. AngularJs actually teaches the web browser through new syntax through a construct which are directives.

AngularJs is not a little piece of overall puzzle of building the client-side of a web application. AngularJs handles all the DOM and Ajax code well-defined structure and together. This makes AngularJs opinionated about how a CRUD(Create,Read,Update,Delete) application should be build. It also make sure that it’s viewpoint is a just the point you can easily change.

AngularJs comes with loads many things out of the box but we gonna define some of the main one we used in our project and they are some of the main ones which can be used in every web application development .

Modules - Module is a container for different parts of your app – controllers, services, filters, directives etc.

Most of the application have the main method to instantiates and write together the different parts of application. In the case of AngularJs don’t have a main method , Instead of them modules declaratively specify how an application should be bootstrapped.

Directives – Those are the markers on a DOM element (Such as attribute, element name, comment or CSS class) that tell AngularJs’s compiler to attach a specified behaviour to that DOM element , for eg via event listeners , or event to transform the DOM element and its children . AngularJs comes with a some of these different directives build in like , ngBind, ngModel and ngClass.

Controllers - In the AngularJS, a controller is defined by a javaScript constructor function that is used to augment the AngularJS Scope. You can attach controller to the DOM via ng-controller directive, AngularJS will instantiate a new controller object, by using the specified controller’s finction. A new child scope will be created and made available as an injectable parameter to the controller’s constructor function as a $scope.

Scope – The scope is the binding part between the HTML (view) and the javaScript (controller). The scope object is available between properties and methods and it is available for both view and controller.

Data binding - Data binding is the one of the automatic synchronization of data between the model and view components. AngularJS implements data-binding lets you treat the model as the single source of truth of your application. When the model changes the view reflects the changes and vice versa. AngularJS applications usually have a data model. The data model is a collection of data available for the application.

Validation – AngularJS offers a client-side form validation. It lets you monitors the state of the form input fields and lets you notify the user about the current state. It also holds the information about whether they have been touched or modified or not .

Event Handling – AngularJS has its own HTML event directives.

The event directive allows us to run functions at user events. AngularJS packed of my events.

For eg : ng-mouseenter, ngmouseover, ng-click .

**Why We used AngularJs**

AngularJS is being used by a lot of applications, ranging from hobby to commercial products. Adoption of AngularJS as a viable framework for client-side development is quickly becoming known to entire web development community. Secondly AngularJS is build by google , we are sure that we are dealing with efficient and reliable code that will scale with your project.

**AngularJS is equipped with a lot of features**

AngularJS similar to backbone or JavaScript MVC is complete solution of rapid front-end development. No other plugins or frameworks are necessary to build a data driven web application.

**Rest Easy**: RestFul actions are very fast and becoming the standard for communicating from the server side to client side. AngularJS turns into a simple JavaScript object , as Models, following the MVVM( Model View View-Model) pattern.

**MVVM** – Model talks to viewModel objects (through something called the $scope object) which listen the changes in the Models. These can be delivered and rendered by the views, Which is the HTML that express the code. AngularJS is also provide us stateless controllers, Which initialize and control the $scope object.

**Data Binding And Dependency Injection:** In MVVM pattern everything communicate to automatic across the UI whenever anything changes. This eliminate getters and setters or class declaration. In our case AngularJS will handle all those dependency injections if or we need we can display data with simple JavaScript primitive types like arrays. In this case everything happens automatic we can ask dependency parameter in AngularJS service function, rather that making one giant main() method call to execute our code .