**WHAT IS PHP?**

PHP or in recursive acronym *Hypertext Preprocessor* is an open source scripting language that is for web development and can be embedded in HTML.

**PHP SAMPLE CODE**

<!DOCTYPE HTML>   
 <html>  
<head>  
  <title>PHP</title>   
   </head>  
  <body>  
  < ?phpecho'PHP SCRIPT';?>   
  </body>  
</html>

**HISTORY OF PHP**

PHP was created by Rasmus Lerdorf in the year 1994 and in the present time PHP is now produced by The PHP Development Team. Before, PHP stands for **Personal Home Page** but in present time it is already known for its recursive acronym Hypertext Preprocessor.

**PHP IS USED IN:**

Server-side Scripting: Server-side scripting is the target field for PHP. PHP parser, web server, and a web browser are the three things to make it work. The users need to run the web server as long as it has a PHP that is already installed.

Command line Scripting: Running php does not require to use any server or browser. The only thing to use PHP without a browser and a server is the PHP Parser. It is a type of usage that is ideal for scripts.

**WHAT IS NODE JS?**

Node.js is a Javascript free and open source cross-platform for server-side programming that allows users to build network applications quickly. Node.JS is a system that uses event-driven programming to build scalable applications and network programs. Node.js basically allows the developer to run JavaScript in the back end, away from the browser.

**HISTORY OF NODE JS**

Node.js was originally written in 2009 by Ryan Dahl by making use of Google's V8 VM, the same runtime environment for JavaScript that Google Chrome uses. The initial release supported only Linux and Mac OSX. Its development and maintenance was led by Dahl and later sponsored by Joyent

Node.js has become wildly popular, with coders everywhere using it to create APIs and build a new matrix of interoperability across the Internet. It is a runtime system for creating (mostly) server-side applications. It's best known as a popular means for JavaScript coders to build real-time Web APIs.

**NODE.JS’s Backend Process**

Node.js allows you to run JavaScript code in the backend, outside a browser. In order to execute the JavaScript you intend to run in the backend, it needs to be interpreted and, well, executed. This is what Node.js does, by making use of Google's V8 VM, the same runtime environment for JavaScript that Google Chrome uses.

Node.js ships with a lot of useful modules, so you don't have to write everything from scratch, like for example something that outputs a string on the console. Node.js operates on a single-thread, using non-blocking I/O calls, allowing it to support tens of thousands of concurrent connections

In the database side, data gets queued through some kind of cache or message queuing infrastructure and digested by a separate database batch-write process, or computation intensive processing backend services, written in a better performing platform for such tasks. Similar behavior can be implemented with other languages/frameworks, but not on the same hardware, with the same high, maintained throughput.

### NODE.JS example

### Once you install Node.JS into the computer, it can automatically be run. Here is an example of a simple NodeJS code:

// Call the console.log function.  
console.log("Webtek is Awesome!");

Save the file, and execute it through Node.js:  
node webtek.js

### Here is an example of a more complicated Node.JS code:

// Load the http module to create an http server.  
var http = require('http');  
// Configure our HTTP server to respond to all requests.  
var server = http.createServer(function (request, response) {  
response.writeHead(200, {"Content-Type": "text/plain"});  
response.end("Web Technology is Awesome!\n");  
});  
// Listen on port 8000, IP defaults to 127.0.0.1  
server.listen(8000);  
// Put a friendly message on the terminal  
console.log("Server running at http://127.0.0.1:8000/");

You can run this by saving it in a file called "example-console.js" and running it with node example-console.js

**Express.JS and Koa.JS framework**

Node.js builds server apps with a lightweight, efficient JavaScript framework called Express.js. There are other frameworks it’s compatible with, but Express.js is best known as another quarter of the MEAN software stack. Newer to the scene and following in Express’ footsteps is the Koa.js framework, a callback-less lightweight framework written by the same author as Express. Get a rundown of more Node frameworks in this 15 Frameworks to Know for Next-Level Node Development article.