HTTP

Node as a Web Server

- Node started as a Web server and evolved into a much more generalized framework.
- Node `http` module is designed with streaming and low latency in mind.
- Node is very popular today to create and run Web servers.

Web Server Example

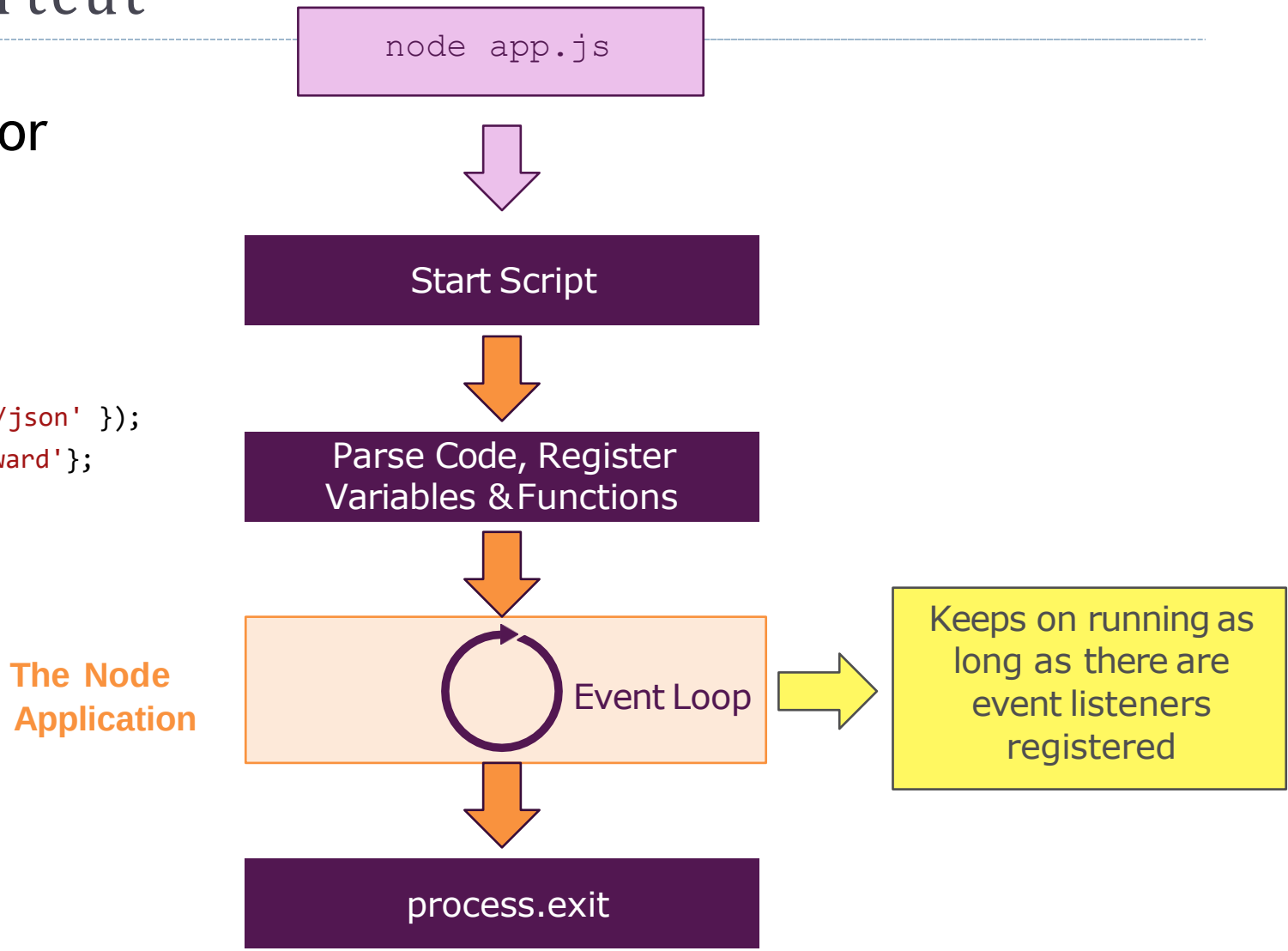
```
const http = require('http');
const server = http.createServer();

server.on('request', function(req, res) {
                        res.writeHead(200, {'Content-Type': 'text/plain'});
                        res.write('Hello World!');
                        res.end();
                    });
server.listen(3000);
```

After we run this code. The node program doesn’t stop. it keeps waiting for request

Web Server Example Shortcut

Passing a callback function to `createServer()` is a shortcut for listening to "request" event.

```
const http = require('http');
http.createServer(function (req, res) {
      res.writeHead(200, { 'Content-Type': 'application/json' });
      const person = { firstname: 'Josh', lastname: 'Edward'};
      res.end(JSON.stringify(person));
}).listen(3000, '127.0.0.1');
```



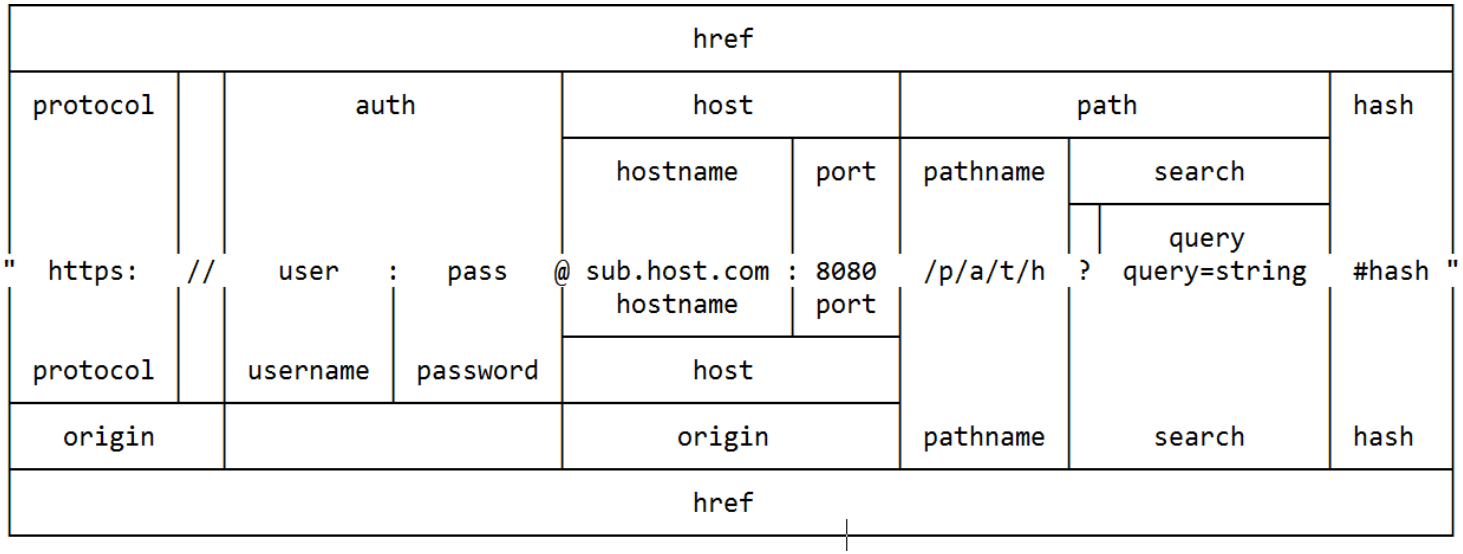
Understanding Request & Response

- A request message from a client to a server includes, within the first line of that message, the method to be applied to the resource, the identifier of the resource, and the protocol version in use.
- After receiving and interpreting a request message, a server responds with an HTTP response message.

```
const http = require('http');
http.createServer((req, res) => {
    console.log(req.url, req.method, req.headers);
    res.setHeader('Content-Type', 'text/html');
    res.write('My First Page');
    res.write('Hello From Node.js');
    res.end();
}).listen(3000);
```

HTTP Request: Reading Get and Post Data

- Handling basic GET & POST requests is relatively simple with Node.js.
- We use the `url` module to parse and read information from the URL.
- The `url` module uses the WHATWG URL Standard (https://url.spec.whatwg.org/)



Using URL Module

- Parsing the URL string using the WHATWG API:

```
const url = require('url');
const myURL =
    new URL('https://user:pass@sub.host.com:8080/p/a/t/h?course1=nodejs&course2=angular#hash');
console.log(myURL);
```

```
URL {
  href: 'https://user:pass@sub.host.com:8080/p/a/t/h?course1=nodejs&course2=angular#hash',
  origin: 'https://sub.host.com:8080',
  protocol: 'https:',
  username: 'user',
  password: 'pass',
  host: 'sub.host.com:8080',
  hostname: 'sub.host.com',
  port: '8080',
  pathname: '/p/a/t/h',
  search: '?course1=nodejs&course2=angular',
  searchParams: URLSearchParams { 'course1' => 'nodejs', 'course2' => 'angular' },
  hash: '#hash'
}
```

Parsing the Query String

```
const url = require('url');
const myURL =
      new URL('https://user:pass@sub.host.com:8080/p/a/t/h?course1=nodejs&course2=angular#hash');
let params = myURL.searchParams;
console.log(params);
console.log(params.get('course1'),
params.get('course2'));
```

```
URLSearchParams { 'course1' => 'nodejs', 'course2' => 'angular' }
nodejs angular
```