S3 - QAP 1 William Stamp



**Semester 3 - QAP 1**

**Understanding core global objects**

1. **Events:**

Node.js has an event-driven architecture that has a built-in module called “Events”, that allows you to create, fire, and listen for events.

In the event-driven model, an “EventEmitter” object raises an event that causes the previously attached listeners of the event executed.

An EventEmitter object has two main functions:

* Emit a named event.
* Attach and detach one or more event listeners to the named event.

Event emitters and listeners are crucial to NodeJS development, and many other programming languages. They are very useful when you have some function that needs to execute “whenever this other thing happens”, without requiring that function to finish or even work for that matter.

2. **Stream:**

A node stream is a method of transferring large amounts of data on mobile devices or websites by breaking the file or data down into manageable chunks.

There are four fundamental stream types within Node.js:

* **Writable:** streams to which data can be written (for example, fs.createWriteStream()).
* **Readable:** streams from which data can be read (for example, fs.createReadStream()).
* **Duplex:** streams that are both Readable and Writable (for example, net.Socket).
* **Transform:** Duplex streams that can modify or transform the data as it is written and read (for example, zlib.createDeflate()).

Additionally, this module includes the utility functions stream.pipeline(), stream.finished(), stream.Readable.from() and stream.addAbortSignal().

3. **Process:**

The “process” object is an instance of an EventEmitter. It provides information about, and control over, the current Node.js process.

The process object is an instance of EventEmitter and emits the following events:

**Exit-**

Emitted when the process is about to exit. There is no way to prevent the exiting of the event loop at this point, and once all exit listeners have finished running, the process will exit.

**BeforeExit-**

This event is emitted when node empties its event loop and has nothing else to schedule. Normally, the node exits when there is no work scheduled, but a listener for 'beforeExit' can make asynchronous calls, and cause the node to continue.

**UncaughtException-**

Emitted when an exception bubbles all the way back to the event loop. If a listener is added for this exception, the default action (which is to print a stack trace and exit) will not occur.

**signalEvents-**

Emitted when the processes receives a signal such as SIGINT, SIGHUP(process termination), etc..



**npm** is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the default package manager for the JavaScript runtime environment Node.js. It consists of a command line client, also called npm, and an online database ([https://www.npmjs.com](https://www.npmjs.com/)) of public and paid-for private packages, called the npm registry.

Use npm to:

* Adapt packages of code for your apps, or incorporate packages as they are.
* Download standalone tools you can use right away.
* Run packages without downloading using [npx](https://www.npmjs.com/package/npx).
* Share code with any npm user, anywhere.
* Restrict code to specific developers.
* Create organizations to coordinate package maintenance, coding, and developers.
* Form virtual teams by using organizations.
* Manage multiple versions of code and code dependencies.
* Update applications easily when underlying code is updated.
* Discover multiple ways to solve the same puzzle.
* Find other developers who are working on similar problems and projects.