Run Node.js scripts from the command line

The usual way to run a Node.js program is to run the globally available `node` command (once you install Node.js) and pass the name of the file you want to execute.

If your main Node.js application file is `app.js`, you can call it by typing:

```
node app.js
```

Above, you are explicitly telling the shell to run your script with `node`. You can also embed this information into your JavaScript file with a "shebang" line. The "shebang" is the first line in the file, and tells the OS which interpreter to use for running the script. Below is the first line of JavaScript:

```
#!/usr/bin/node
```

Above, we are explicitly giving the absolute path of interpreter. Not all operating systems have `node` in the bin folder, but all should have `env`. You can tell the OS to run `env` with node as parameter:

```
#!/usr/bin/env node
```

```
// your javascript code
```

To use a shebang, your file should have executable permission. You can give `app.js` the executable permission by running:

```
chmod u+x app.js
```

While running the command, make sure you are in the same directory which contains the `app.js` file.

Pass string as argument to `node` instead of file path

To execute a string as argument you can use `-e`, `--eval "script"`. Evaluate the following argument as JavaScript. The modules which are predefined in the REPL can also be used in script.

On Windows, using cmd.exe a single quote will not work correctly because it only recognizes double `"` for quoting. In Powershell or Git bash, both `'` and `"` are usable.

```
node -e "console.log(123)"
```

Restart the application automatically

As of nodejs V16, there is a built-in option to automatically restart the application when a file changes. This is useful for development purposes.

To use this feature, you need to pass the `--watch' flag to nodejs.

```
node --watch app.js
```

So when you change the file, the application will restart automatically. Read the `--watch` flag documentation.