



Custom Env

#Node JS Notes

What are Environment Variables?

- From programming we know that variables are stored values that can be changed. They're mutable and can vary - hence the name *variables*.
- *Environment variables* are variables external to our application which reside in the OS or the container of the app is running in. An environment variable is simply a name mapped to a value.
- By convention, the name is capitalized e.g. `SENDER_EMAIL_ADDRESS=me@thisemail.com`. The values are strings.



Why do we need environment variables?

- **Security:** Some things like API keys should not be put in plain text in the code and thereby directly visible.
- **Flexibility:** you can reduce conditional statements à la *“If production server then do X else if test server then do Y else if localhost then do Z ...”*.
- **Adoption:** Popular services like GitLab or Heroku support the usage of environment variables.



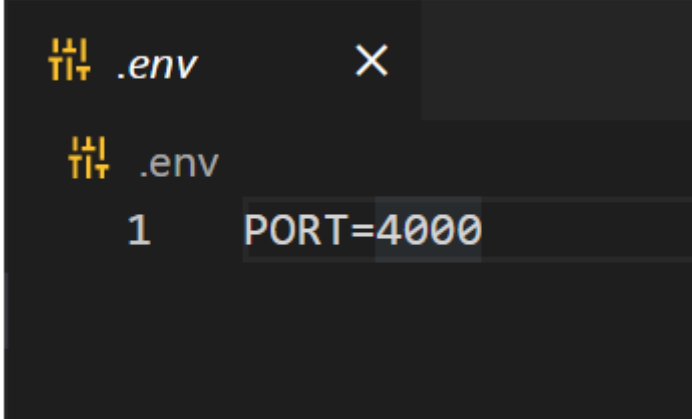
Environment Variables

- HTTP Port and Address
- Database, cache, and other storage connection information
- Location of static files/folders
- Email Id and Password
- Token



How does a .env file look like?

- A **.env file** is more or less a plain text document. It should be placed in the root of your project. You specify key value pairs and you can write single line comments using #. Here is an example how a .env file looks like:
- environment variables are available through the process.env global variable



```
.env
PORT=4000
```

Custom-Env

- Custom env is a library built to make development more feasible by allowing multiple .env configurations for different environments. This is done by loading environment variables from a .env.envname file, into the node's process.env object.
- Package
 - <https://www.npmjs.com/package/custom-env>
- Install
 - `npm i custom-env --save`



Quick Steps

app.js

```
require('custom-env').env(process.env.NODE_ENV)
```

package.json

```
"scripts": {  
  "test": "echo \"Error: no test specified\" && exit 1",  
  "dev": "SET NODE_ENV=development&& node app.js",  
  "prod": "SET NODE_ENV=production&& node app.js"  
},
```

.env

.env.production

.env.development

PORT=9000



C:\Windows\system32\cmd.exe

```
D:\demo>npm init -y
```

C:\Windows\system32\cmd.exe

```
D:\demo>npm init -y
Wrote to D:\demo\package.json:

{
  "name": "demo",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.17.2"
  },
  "devDependencies": {},
  "description": ""
}

D:\demo>
```



C:\Windows\system32\cmd.exe

```
D:\demo>npm i express -g
```

```
changed 50 packages, and audited 51 packages in 3s
```

```
2 packages are looking for funding  
  run `npm fund` for details
```

```
found 0 vulnerabilities
```

```
D:\demo>
```

C:\Windows\system32\cmd.exe

```
D:\demo>npm i express --save
```

```
up to date, audited 51 packages in 1s
```

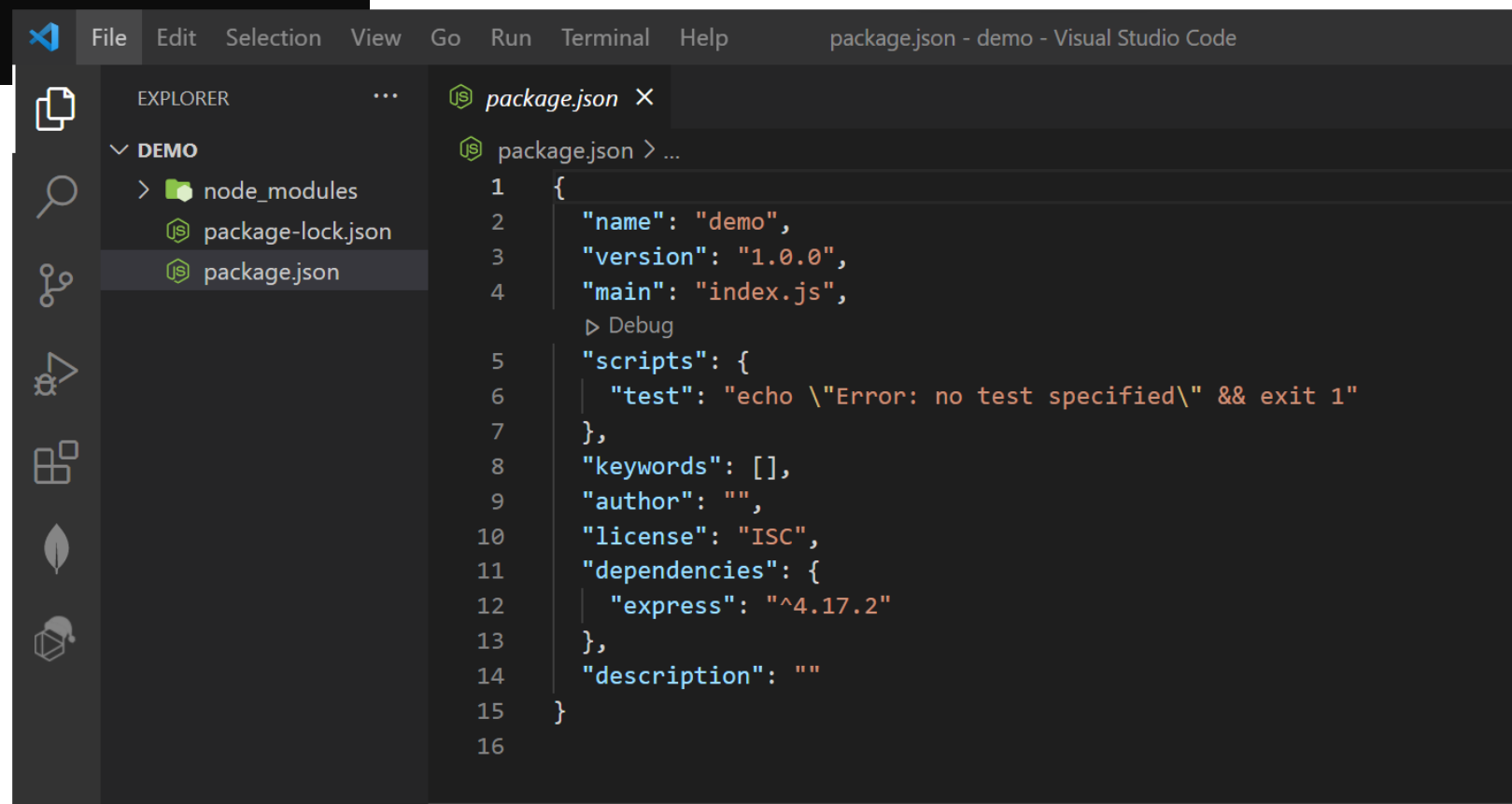
```
2 packages are looking for funding  
  run `npm fund` for details
```


```
found 0 vulnerabilities
```

```
D:\demo>
```



```
C:\Windows\system32\cmd.exe  
  
D:\demo>code .
```






```
File Edit Selection View Go Run Terminal Help app.js - demo - Visual Studio Code

EXPLORER
  DEMO
    > node_modules
    app.js
    package-lock.json
    package.json

app.js
1  const express = require('express')
2  const app = express()
3  const port = 3000
4
5  app.get('/', (req, res) => {
6    res.send('Hello World!')
7  })
8
9  app.listen(port, () => {
10    console.log(`Example app listening at http://localhost:${port}`)
11  })
```



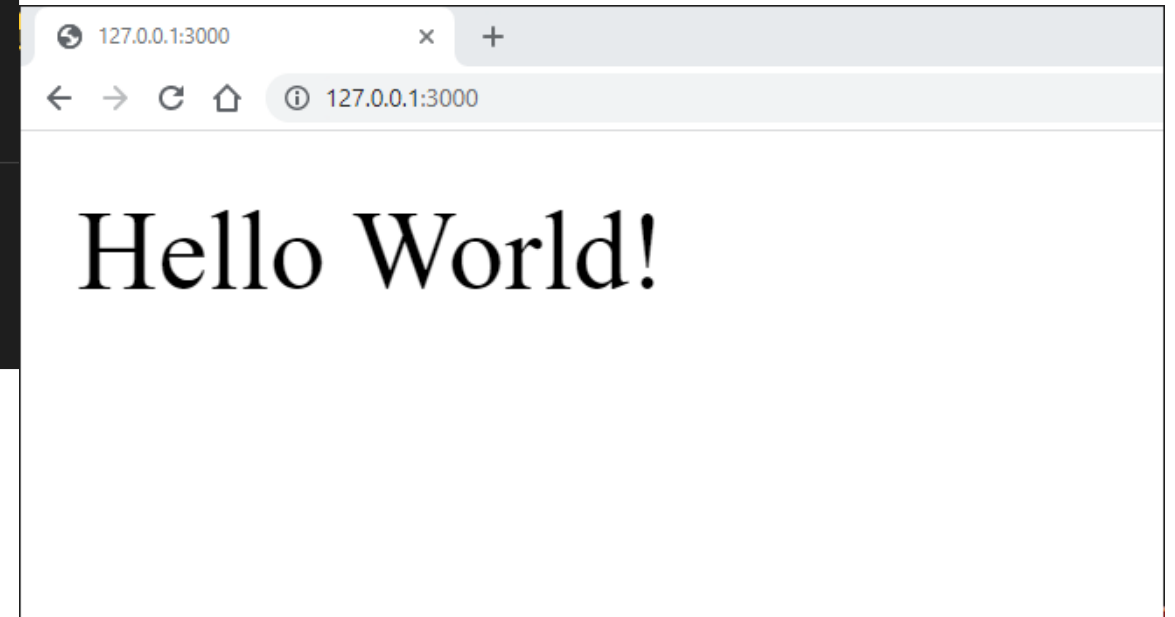
```
File Edit Selection View Go Run Terminal Help appjs - demo - Visual Studio Code

EXPLORER
  DEMO
    node_modules
    app.js
    package-lock.json
    package.json

app.js
1  const express = require('express')
2  const app = express()
3  const port = 3000
4
5  app.get('/', (req, res) => {
6    res.send('Hello World!')
7  })
8
9  app.listen(port, () => {
10    console.log(`Example app listening at http://localhost:${port}`)
11  })

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

D:\demo>node app.js
Example app listening at http://localhost:3000
█
```



npmjs.com/package/custom-env

npm

Search packages

Search

Sign Up

Sign In

custom-env

2.0.1 • Public • Published 2 years ago

Readme

Explore BETA

2 Dependencies

42 Dependents

5 Versions

Custom-Env

Custom env is a library built to make development more feasible by allowing multiple .env configurations for different environments. This is done by loading environment variables from a .env.envname file, into the node's process.env object.

Installation

```
npm install custom-env
```

Usage

Place this at the top of your application

```
require('custom-env').env()
```

Create a .env file in your app's root directory and add the environment variables each on new line:

```
APP_ENV=dev
DB_HOST=localhost
DB_USER=root
DB_PASS=root
```

Simple! The process.env is now loaded with the environment variables above

Install

```
> npm i custom-env
```

Repository

github.com/erisanolasheni/custom-env

Homepage

github.com/erisanolasheni/custom-env...

Weekly Downloads

18,480

Version	License
2.0.1	ISC

Unpacked Size	Total Files
11.2 kB	5

Issues	Pull Requests
3	0

Last publish

2 years ago



```
C:\Windows\system32\cmd.exe

D:\demo>npm install custom-env --save

C:\Windows\system32\cmd.exe

D:\demo>npm install custom-env --save

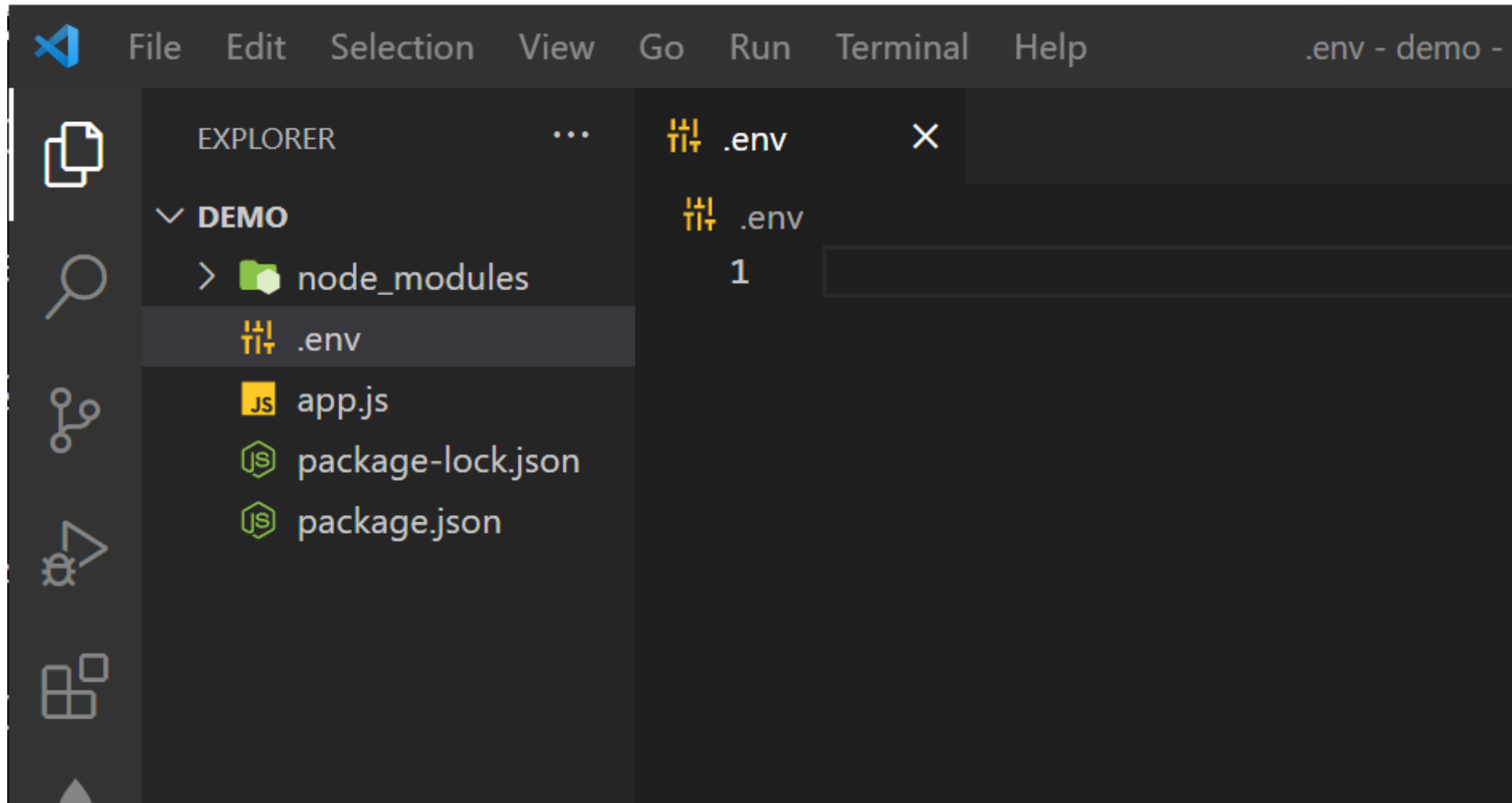
added 3 packages, and audited 54 packages in 2s

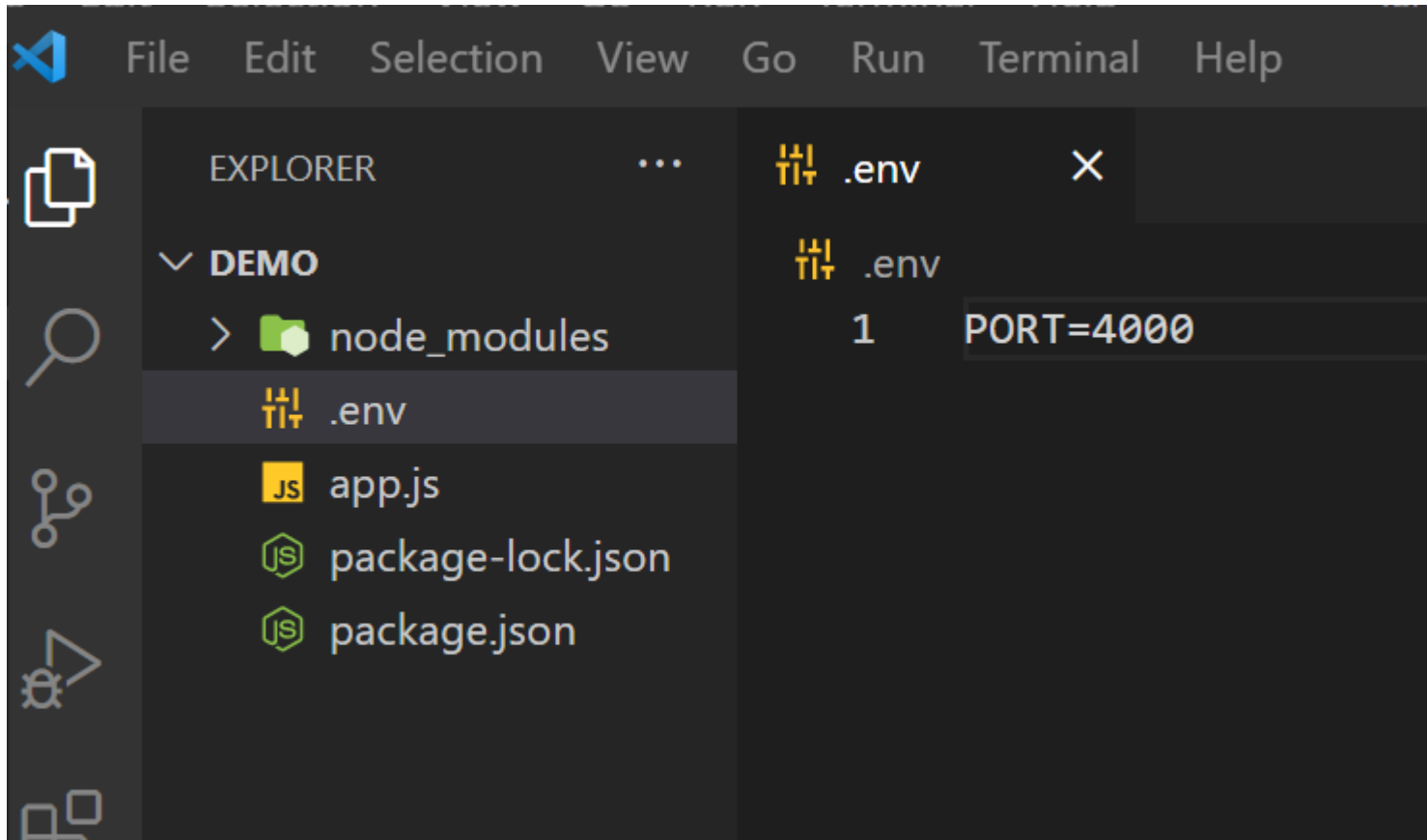
2 packages are looking for funding
  run `npm fund` for details

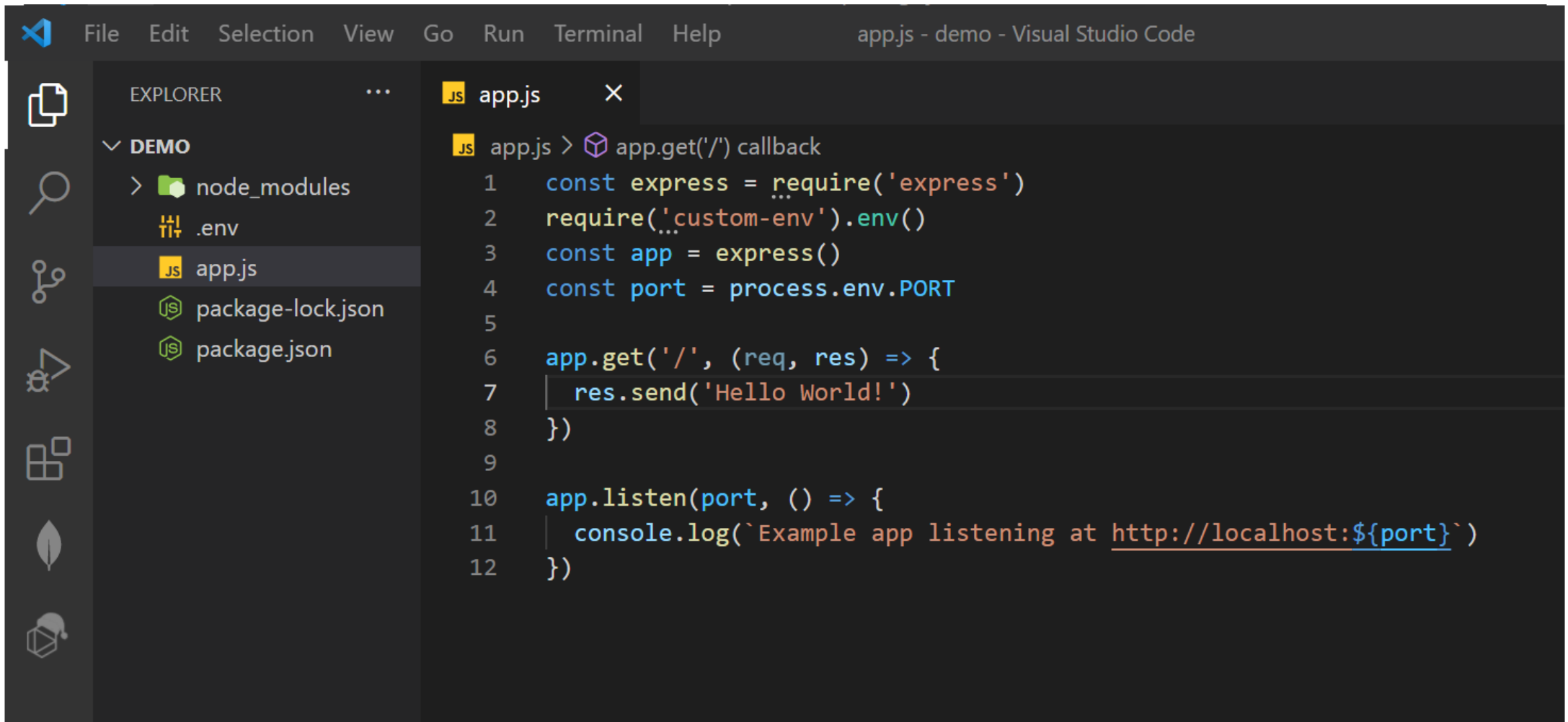

found 0 vulnerabilities

D:\demo>
```









```
File Edit Selection View Go Run Terminal Help app.js - demo - Visual Studio Code

EXPLORER
DEMO
  > node_modules
  .env
  app.js
  package-lock.json
  package.json

app.js
app.js > app.get('/') callback
1  const express = require('express')
2  require('custom-env').env()
3  const app = express()
4  const port = process.env.PORT
5
6  app.get('/', (req, res) => {
7    res.send('Hello World!')
8  })
9
10 app.listen(port, () => {
11   console.log(`Example app listening at http://localhost:${port}`)
12 })
```

C:\Windows\system32\cmd.exe - node app.js

```
D:\demo>node app.js
```

```
No env file present for the current environment:  development
```

```
Falling back to .env config
```

```
Example app listening at http://localhost:4000
```



EXPLORER

DEMO

node_modules

.env

.env.development

.env.production

app.js

package-lock.json

package.json

app.js

1 const express = require('express')

2 require('..custom-env').env('development')

3

4 const app = express()

5 const port = process.env.PORT

6

7 app.get('/', (req, res) => {

8 | res.send('Hello World!')

9 | })

10

11 app.listen(port, () => {

12 | console.log(`Example app listening at http://localhost:\${port}`)

13 | })

.env.development

1 PORT=5000

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

cmd

+

▼

D:\demo>node app.js

Example app listening at http://localhost:5000

Akash Technolabs

www.akashsir.com

19

EXPLORER

▼ DEMO

- > node_modules
- .env
- .env.development
- .env.production
- app.js
- package-lock.json
- package.json

JS app.js

JS app.js > ...

```
1  const express = require('express')
2  require('..custom-env').env('production')
3
4  const app = express()
5  const port = process.env.PORT
6
7  app.get('/', (req, res) => {
8    res.send('Hello World!')
9  })
10
11 app.listen(port, () => {
12   console.log(`Example app listening at http://localhost:${port}`)
13 })
```

.env.production

.env.production

1 PORT=9000

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

cmd +

```
D:\demo>node app.js
Example app listening at http://localhost:9000
█
```



JS app.js

JS app.js > ...

```
1  const express = require('express')
2  require('custom-env').env(process.env.NODE_ENV)
3
4  const app = express()
5  const port = process.env.PORT
6
7  app.get('/', (req, res) => {
8    res.send('Hello World!')
9  })
10
11 app.listen(port, () => {
12   console.log(`Example app listening at http://localhost:5000`)
13 })
```

package.json

package.json > ...

```
1  {
2    "name": "demo",
3    "version": "1.0.0",
4    "main": "index.js",
5    "scripts": {
6      "test": "echo \"Error: no test specified\" && exit 1",
7      "dev": "SET NODE_ENV=development&& node app.js",
8      "prod": "SET NODE_ENV=production&& node app.js"
9    },
10   "keywords": [],
11   "author": "",
12   "license": "ISC",
13   "dependencies": {
14     "custom-env": "^2.0.1",
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

cmd + v

D:\demo>npm run dev

> demo@1.0.0 dev

> SET NODE_ENV=development&& node app.js

Example app listening at http://localhost:5000

| UUID



UUIDs (Universally Unique IDentifiers)

- The “**Universally unique identifier**”, or **UUID**, was designed to provide a consistent format for any unique ID we use for our data. UUIDs address the problem of generating a unique ID - either randomly, or using some data as a seed.
- <https://www.uuidgenerator.net/>
- <https://www.uuidtools.com/>



Download

- Download

- npm install uuid

- Calling

- `var uuid = require('uuid');`

- Print

- `console.log(uuid.v4());`

- Multiple

```
for(let x = 0; x<10; x++){  
  console.log(uuid.v4());  
}
```




```
ca. C:\Windows\system32\cmd.exe

D:\demo>npm i uuid --save

added 1 package, and audited 55 packages in 1s

2 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

D:\demo>
```



npmjs.com/package/uuid#uuidparsestr

Nonsense Poetry Manager

ProductsPricingDocumentationCommunity

npm

Search packages

Search

Sign Up

Sign In

uuid

DT

8.3.2 • Public • Published a year ago

Readme

Explore

BETA

0 Dependencies

41,790 Dependents

34 Versions

uuid

CI passing

Browser passing

For the creation of RFC4122 UUIDs

- **Complete** - Support for RFC4122 version 1, 3, 4, and 5 UUIDs
- **Cross-platform** - Support for ...
 - CommonJS, ECMAScript Modules and CDN builds
 - Node 8, 10, 12, 14
 - Chrome, Safari, Firefox, Edge, IE 11 browsers
 - Webpack and rollup.js module bundlers
 - React Native / Expo
- **Secure** - Cryptographically-strong random values
- **Small** - Zero-dependency, small footprint, plays nice with "tree shaking" packagers
- **CLI** - Includes the `uuid` command line utility

Upgrading from `uuid@3.x` ? Your code is probably okay, but check out [Upgrading From uuid@3.x](#) for details.

Quickstart

To create a random UUID...

1. Install

Install

> npm i uuid

Repository

github.com/uuidjs/uuid

Homepage

github.com/uuidjs/uuid#readme

Weekly Downloads

66,055,458

Version	License
8.3.2	MIT
Unpacked Size	Total Files
116 kB	66
Issues	Pull Requests
17	2

Akash Technolabs

www.akashsir.com

26



Method	Work
<code>uuid.NIL</code>	The nil UUID string (all zeros)
<code>uuid.parse()</code>	Convert UUID string to array of bytes
<code>uuid.validate()</code>	Test a string to see if it is a valid UUID
<code>uuid.v1()</code>	Create a version 1 (timestamp) UUID
<code>uuid.v3()</code>	Create a version 3 (namespace w/ MD5) UUID
<code>uuid.v4()</code>	Create a version 4 (random) UUID
<code>uuid.v5()</code>	Create a version 5 (namespace w/ SHA-1) UUID
<code>uuid.stringify()</code>	Convert array of bytes to UUID string



```
var uuid = require('uuid');
//Print UUID
console.log("v1: " + uuid.v1());
console.log("v3: " + uuid.v3('Test v3', 'cdb3c65c-84ee-11eb-8dcd-0242ac130003'));
console.log("v4: " + uuid.v4());
console.log("v5: " + uuid.v5('Test v5', '5dd74eb6-84ef-11eb-8dcd-0242ac130003'));
//Validate
console.log(uuid.validate('a7010bb2-7d9d-4947-a4d5-13fa012b8c0f'));
//Get Version
console.log(uuid.version('83dc308b-ac4e-4d1e-9e0b-9e49041308c4'));

for(let x = 0; x<10; x++){
  console.log(uuid.v4());
}
```



```
v1: 30a82410-6289-11ec-b6fa-2b3aa3491731  
v3: 0c3269a6-7b8d-3aaa-9e6e-377bb7bd3a22  
v4: add8b0fb-477a-4c47-b50d-2adcdebda297  
v5: 5011256a-edf5-546f-a510-3ac19150f61f
```

true

4

```
c49ce5c5-ba95-4d1e-ba9d-fefb883ba167  
c0b5da71-fe27-40ce-b34d-d5de888d7378  
7531687b-2022-44d6-a7fd-0acab08c5b71  
78fa4b17-930e-4639-94fb-f2a72c3b1e94  
cc68f02b-6bba-4899-b932-2bb23d63de24  
109a754b-8825-42ac-9390-a010f35e143a  
817d317c-8654-4352-882b-beb5b911336f  
d71ea42c-a182-4608-9819-85914b4e65d9  
8df43b81-f6a4-4c0f-bbc2-9bd43f67900b  
c0c379c4-4f87-45b0-b3c4-789be7a64df9
```



Different UUID versions

- **v1** and **v4** can be called without passing any parameters but, for **v3** and **v5**, we need to pass the name (a string) and the namespace identifier (a UUID), in this order.
- `console.log("v1: " + uuid.v1());`
- `console.log("v3: " + uuid.v3('Test v3', 'cdb3c65c-84ee-11eb-8dcd-0242ac130003'));`
- `console.log("v4: " + uuid.v4());`
- `console.log("v5: " + uuid.v5('Test v5', '5dd74eb6-84ef-11eb-8dcd-0242ac130003'));`



Validating UUIDs

- To validate if a given string is a UUID or not, we simply need to call the [validate](#) function. This function returns a Boolean indicating if the string is a valid UUID (true) or not (false). We will test it both against an arbitrary string and to a v4 UUID.
- `console.log(uuid.validate('a7010bb2-7d9d-4947-a4d5-13fa012b8c0f'));`



Version Check

- To obtain the version of a given valid UUID, we simply need to call the version function, passing as input the UUID as a string.
- As output, it will return a number with the RFC version of the UUID. If the parameter is not a valid UUID, then this function call will throw a TypeError exception.



Quick

```
var uuid = require('uuid');  
//Print UUID  
console.log("v1: " + uuid.v1());  
console.log("v3: " + uuid.v3('Test v3', 'cdb3c65c-84ee-11eb-8dcd-0242ac130003'));  
console.log("v4: " + uuid.v4());  
console.log("v5: " + uuid.v5('Test v5', '5dd74eb6-84ef-11eb-8dcd-0242ac130003'));  
//Validate  
console.log(uuid.validate('a7010bb2-7d9d-4947-a4d5-13fa012b8c0f'));  
//Get Version  
console.log(uuid.version('83dc308b-ac4e-4d1e-9e0b-9e49041308c4'));  
  
for(let x = 0; x<10; x++){  
  console.log(uuid.v4());  
}
```



Cron



What is a cron job?

- A cron job is the event scheduler that runs a job as per the given time interval.
- So it executes a file linked which is linked to that particular job, it runs the code snippet of it and this configuration is set on the server.
- But in NodeJS we can actually set cron job within the application code using the [node-cron](#) module.



Different intervals for scheduling tasks

* * * * *

| | | | |

| | | | | day of week

| | | | month

| | | day of month

| | hour

| minute

second (optional)

Seconds (optional): 0 – 59

Minute: 0 – 59

Hour: 0 – 23

Day of the Month: 1 – 31

Month: 1 – 12

Day of the week: 0 – 7 (0 and 7 both represent Sunday)

Examples:

(*/10 * * * *) – Runs on every 10 minutes

(* * 21 * *) – Runs 21th of every month

(0 8 * * 1) – Runs 8 AM on every Monday

<https://github.com/kelektiv/node-cron/tree/master/examples>



```
JS app.js X
JS app.js > ...
6
7   var CronJob = require('cron').CronJob;
8
9   // Creating a cron job which runs on every 10 second
10  var job = new CronJob('* * * * *', function() {
11    console.log('You will see this message every second');
12  }, null, true, 'America/Los_Angeles');
13
14  job.start();
15
16
17
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
D:\demo>node app.js
Example app listening at http://localhost:3000
You will see this message every second
You will see this message every second
You will see this message every second
You will see this message every second
You will see this message every second
█
```



```
var CronJob = require('cron').CronJob;

// Creating a cron job which runs on every 10 second
var job = new CronJob('* * * * *', function() {
  console.log('You will see this message every second');
}, null, true, 'America/Los_Angeles');

job.start();
```



Get Exclusive Video Tutorials



www.apptutorials.com

<https://www.youtube.com/user/Akashtips>





Get More Details

www.akashsir.com



If You Liked It !

Rating Us Now



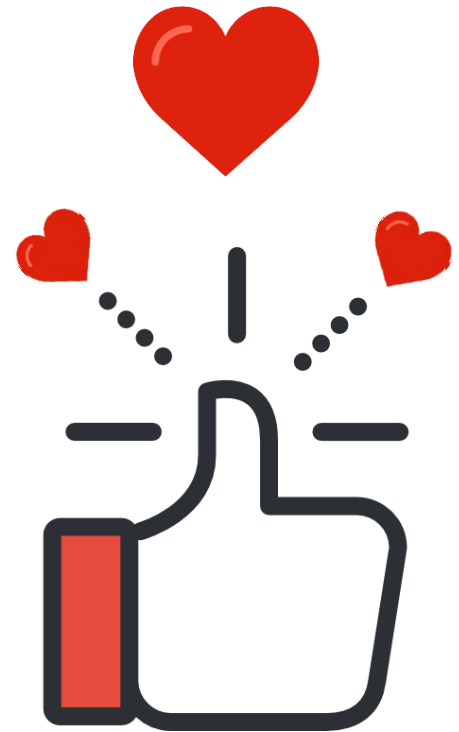
Just Dial

https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/O79PXX79-XX79-170615221520-S5C4_BZDET



Sulekha

<https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad>



Connect With Me



Akash Padhiyar
#AkashSir

www.akashsir.com
www.akashtechlabs.com
www.akashpadhiyar.com
www.apptutorials.com

Social Info



Akash.padhiyar



Akashpadhiyar



Akash_padhiyar



+91 99786-21654



#Akashpadhiyar
#apptutorials