

# NodeJs

REPL



مدرس: مهرداد دادخواه



# REPL

**REPL** also known as **Read Evaluate Print Loop** is a programming language environment (Basically a console window) that takes single expression as user input and returns the result back to the console after execution.

# REPL

```
Welcome to Node.js v13.12.0.
Type ".help" for more information.
> .help
.breakps Sometimes you get stuck, this gets you out
.clear Alias for .break
.editor Enter editor mode
.exit Exit the repl
.help Print this help message
.load Load JS from a file into the REPL session
.save Save all evaluated commands in this REPL session to a file
Press ^C to abort current expression, ^D to exit the repl
> 
```



REPL

Use the tab to autocomplete

# Exploring JavaScript objects

```
node-handbook: node
> Number.
Number.__defineGetter__      Number.__defineSetter__    Number.__lookupGetter__
Number.__lookupSetter__    Number.__proto__           Number.constructor
Number.hasOwnProperty       Number.isPrototypeOf       Number.propertyIsEnumerable
Number.toLocaleString      Number.toString            Number.valueOf

Number.apply                Number.arguments            Number.bind
Number.call                 Number.caller               Number.length
Number.name

Number.EPSILON              Number.MAX_SAFE_INTEGER    Number.MAX_VALUE
Number.MIN_SAFE_INTEGER    Number.MIN_VALUE           Number.NEGATIVE_INFINITY
Number.NaN                  Number.POSITIVE_INFINITY   Number.isFinite
Number.isInteger            Number.isNaN                Number.isSafeInteger
Number.parseFloat           Number.parseInt             Number.prototype

> |
```

# Explore global objects

```
node-handbook: node
> global.
global.__defineGetter__      global.__defineSetter__
global.__lookupGetter__    global.__lookupSetter__
global.__proto__            global.constructor
global.hasOwnProperty       global.isPrototypeOf
global.propertyIsEnumerable global.toLocaleString
global.toString             global.valueOf

global.Array                 global.ArrayBuffer
global.Boolean               global.Buffer
global.DTRACE_HTTP_CLIENT_REQUEST global.DTRACE_HTTP_CLIENT_RESPONSE
global.DTRACE_HTTP_SERVER_REQUEST global.DTRACE_HTTP_SERVER_RESPONSE
global.DTRACE_NET_SERVER_CONNECTION global.DTRACE_NET_STREAM_END
global.DataView              global.Date
global.Error                  global.EvalError
global.Float32Array          global.Float64Array
global.Function               global.GLOBAL
global.Infinity               global.Int16Array
global.Int32Array             global.Int8Array
global.Intl                    global.JSON
global.Map                     global.Math
global.NaN                     global.Number
global.Object                 global.Promise
global.Proxy                   global.RangeError
global.ReferenceError         global.Reflect
global.RegExp                  global.Set
global.String                  global.Symbol
global.SyntaxError            global.TypeError
global.URIError               global.Uint16Array
global.Uint32Array            global.Uint8Array
global.Uint8ClampedArray      global.WeakMap
global.WeakSet                 global.WebAssembly
```





## Dot commands

- **.help**: shows the dot commands help
- **.editor**: enables editor mode, to write multiline JavaScript code with ease. Once you are in this mode, enter ctrl-D to run the code you wrote.
- **.break**: when inputting a multi-line expression, entering the .break command will abort further input. Same as pressing ctrl-C.
- **.clear**: resets the REPL context to an empty object and clears any multi-line expression currently being input.
- **.load**: loads a JavaScript file, relative to the current working directory
- **.save**: saves all you entered in the REPL session to a file (specify the filename)
- **.exit**: exits the repl (same as pressing ctrl-C two times)

## accept arguments from the command line

```
node app.js
```

```
node app.js joe
```

```
node app.js name=joe
```





## process object & argv property

```
process.argv.forEach((val, index) => {  
  console.log(`${index}: ${val}`)  
})
```

```
const args = process.argv.slice(2)
```



# Minimist - parse argument options

<https://www.npmjs.com/package/minimist>

```
const args = require('minimist')(process.argv.slice(2))
args['name'] //joe
```