NPM & NodeJS Modules

Objective

- Know how to create your own modules in NodeJS
- Know how to install and use others modules via NPM.

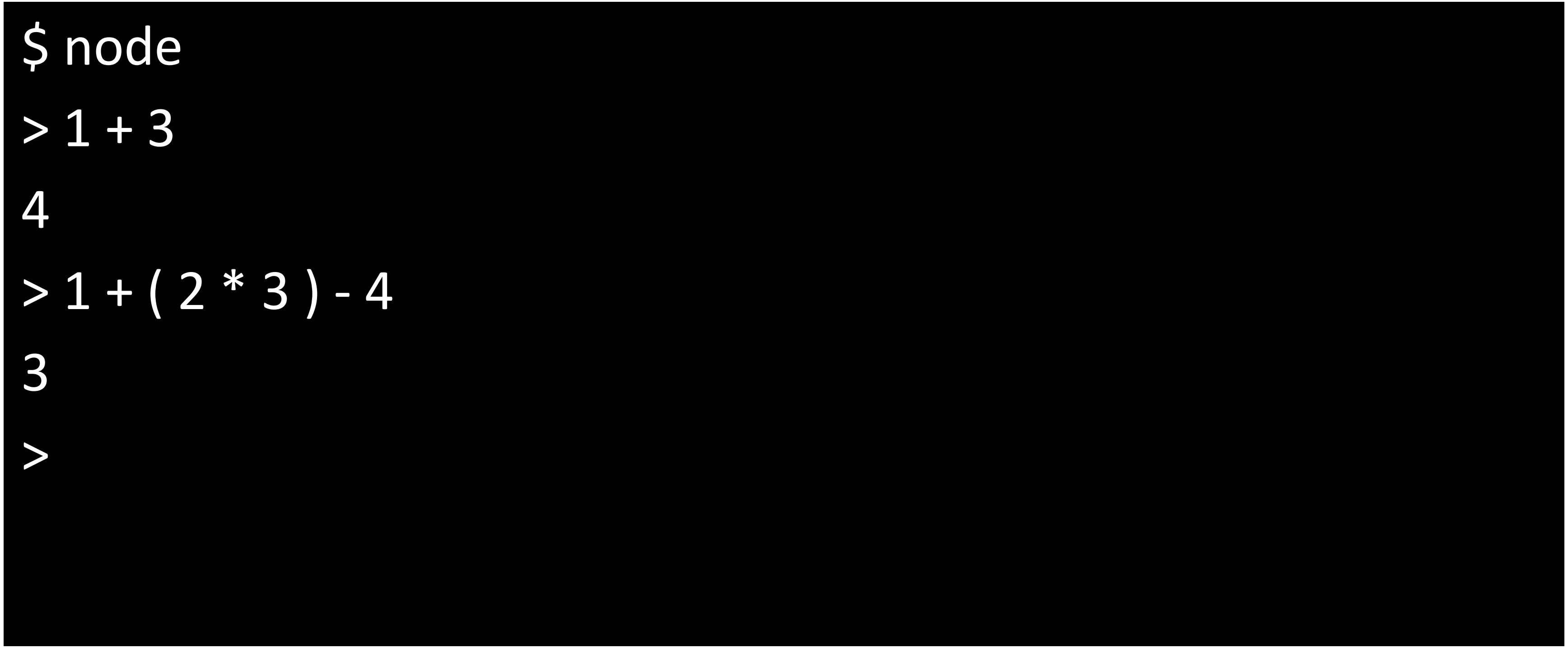
Agenda

- Module
- REPL Terminal
- Node Package Manager

REPL Terminal

- REPL stands for Read Eval Print Loop and it represents a computer environment
  - –**Read** – Reads user's input, parses the input into JavaScript data-structure, and stores in memory.
  - –**Eval** – Takes and evaluates the data structure.
  - –**Print** – Prints the result.
  - –**Loop** – Loops the above command until the user presses **ctrl-c** twice.

Simple Expression

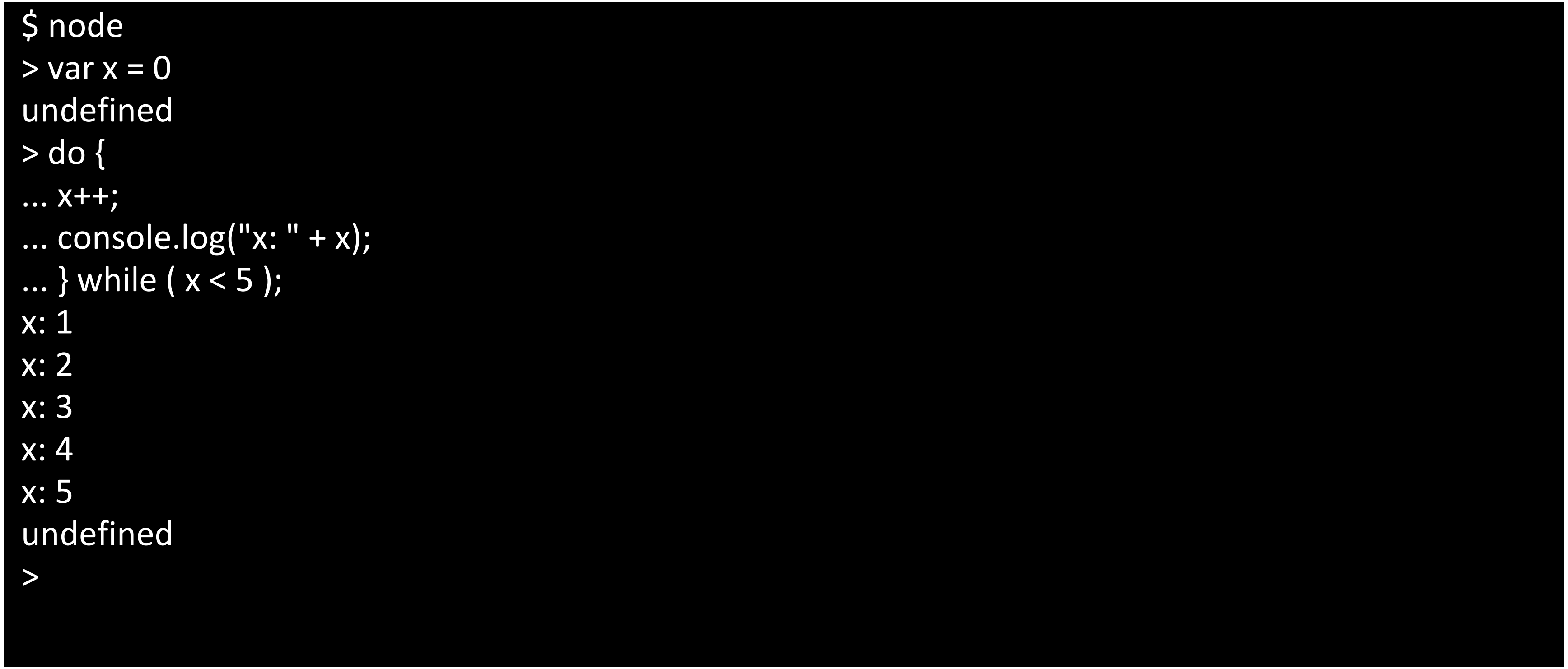


```
$ node
> 1 + 3
4
> 1 + ( 2 * 3 ) - 4
3
>
```

Variables

- If **var** keyword is not used, then the value is stored in the variable and printed
- Whereas if **var** keyword is used, then the value is stored but not printed. You can print variables using **console.log()**.

Multiline Expression



```
$ node
> var x = 0
undefined
> do {
... x++;
... console.log("x: " + x);
... } while ( x < 5 );
x: 1
x: 2
x: 3
x: 4
x: 5
undefined
>
```

Underscore Variable

```
$ node
> var x = 10
undefined
> var y = 20
undefined
> x + y
30
> var sum = _
undefined
> console.log(sum)
30
undefined
>
```

Module

- A set of functions you want to include in your application
- Node.js has a set of built-in modules which you can use without any further installation.

Create Your Own Modules

- You can create your own modules, and easily include them in your applications

Example

Create a module that returns the current date and time:

```
exports.myDateTime = function () {
    return Date();
};
```

Create Your Own Modules

- Use the *exports* keyword to make properties and methods available outside the module file.

```
var http = require('http');
var dt = require('./myfirstmodule');

http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/html'});
    res.write("The date and time are currently:
" + dt.myDateTime());
    res.end();
}).listen(8080);
```

Node Package Manager

Node Package Manager (NPM) provides two main functionalities

- Online repositories for node.js packages/modules which are searchable on search.nodejs.org
- Command line utility to install Node.js packages, do version management and dependency management of Node.js packages.

Install

- Installing Modules using NPM
  - $ npm install <Module Name>
- Now you can use this module in your js file as following
  - var express = require('express');

Global vs Local Installation

- By default, NPM installs any dependency in the local mode
- For global: *npm install express –g*
- Attributes of Package.json
  - **name** − name of the package
  - **version** − version of the package
  - **description** − description of the package
  - **homepage** − homepage of the package
  - **author** − author of the package
  - **contributors** − name of the contributors to the package
  - **dependencies** − list of dependencies. NPM automatically installs all the dependencies mentioned here in the node_module folder of the package.
  - **repository** − repository type and URL of the package
  - **main** − entry point of the package
  - **keywords** − keywords

Global vs Local Installation

- You will need to provide all the required information about your module. You can take help from the above-mentioned package.json file to understand the meanings of various information demanded
  - *npm adduser*
- It is time now to publish your module
  - *$ npm publish*

Thank you