Module 2
Working with modules

zeolearn™

Outline

- Understanding modules
- Usage of require, exports and module.exports
- Types of modules
- Knowing npm (node package manager)
- Overview of package.json file
- Publishing modules to npm

Modules

- Javascript style of organizing code
- A few different module systems available for JavaScript, but they all work in a similar way
- Code in single file is a module
- Expose functionality to outside world with exports or module.exports object
- Import the modules using require

require

- Node.js follows the CommonJS module system
  - require - builtin function
- All require do is;
  - reads a javascript file
  - executes the file
  - return the exports object

module.exports or exports

- module.exports is initialized to an empty object
- exports is just a reference to module.exports
- Whatever is contained in the module.exports variable at the end of your script is the exported value of your module
- Whatever is contained in the module.exports variable is imported when used via require

Types of Modules

- Builtin

```
const os= require(os);
```

- Local or user define

```
const logger= require('./logger');
```

- Third party

```
const request= require('request');
```

npm

- Package management tool for node js applications

npm help

```
MOKSHAs-MacBook-Pro:Zeolearn moksha$ npm help

Usage: npm <command>

where <command> is one of:
    access, adduser, bin, bugs, c, cache, completion, config,
    ddp, dedupe, deprecate, dist-tag, docs, edit, explore, get,
    help, help-search, i, init, install, install-test, it, link,
    list, ln, logout, ls, outdated, owner, pack, ping, prefix,
    prune, publish, rb, rebuild, repo, restart, root, run,
    run-script, s, se, search, set, shrinkwrap, star, stars,
    start, stop, t, tag, team, test, tst, un, uninstall,
    unpublish, unstar, up, update, v, version, view, whoami
```

package.json

- Configuration file for npm
- Records metadata about current project, its dependencies etc.

semver

MAJOR.MINOR.PATCH

Version ranges

- Tilde Ranges ~1.2.3 ~1.2 ~1
  - Allows patch-level changes if a minor version is specified on the comparator. Allows minor-level changes if not
  - ~1.2.3 := >=1.2.3 <1.(2+1).0 := >=1.2.3 <1.3.0
  - ~1.2 := >=1.2.0 <1.(2+1).0 := >=1.2.0 <1.3.0 (Same as 1.2.x)
  - ~1 := >=1.0.0 <(1+1).0.0 := >=1.0.0 <2.0.0 (Same as 1.x)

Version ranges

- Caret Ranges ^1.2.3 ^0.2.5 ^0.0.4
  - Allows changes that do not modify the left-most non-zero digit in the [major, minor, patch] tuple
  - ^1.2.3 := >=1.2.3 <2.0.0
  - ^0.2.3 := >=0.2.3 <0.3.0
  - ^0.0.3 := >=0.0.3 <0.0.4

zeolearn™

Summary

- Techniques for modularizing JavaScript code
- Using require() to import
- Using exports or module.exports to export
- Types of modules
- Overview and usage of npm
- Understanding versioning and semver

zeolearn™

Check your knowledge

- Which function is used to import modules?
- Name 3 different module types?
- How do you export a code from a module?
- Code in how many files is a module?
- What does npm stands for?
- What does leftmost digit stands for in semver?
- What is npm configuration file named as?