

ПРОГРАММИРОВАНИЕ В INTERNET

МОДУЛИ. NPM. PACKAGE.JSON

Модуль

=

текстовый файл, содержащий код на языке JS, специальным образом оформленный и размещенный. Может использоваться приложением. Является фундаментальной единицей структурирования кода Node.js-приложений.

Форматы модулей

- IIFE (Immediately Invoked Function Expression)
- CommonJS
- AMD (Asynchronous Module Definition)
- UMD (Universal Module Definition)
- ES6 Module

IIFE =

(Immediately Invoked Function Expression)

это конструкция, позволяющая
вызывать функцию
непосредственно после ее
определения.

```
(function() {  
    //Your code goes here  
})();
```

```
(( ) => {  
    //Your code goes here  
})();
```

CommonJS

=

группа, которая проектирует,
прототипирует и
стандартизирует различные
JavaScript API.

CommonJS

- поддержка **require** для импорта модуля;
- **имя модуля – строка**, может включать символы идентификации путей;
- модуль должен явно экспортировать всю свою функциональность, поддержка объекта **export**;
- **Переменные**, объявленные внутри модуля, **не видны за его пределами**.

require

- **синхронно** загружает модуль;
- кэширует модуль;
- удалить из кэша можно с помощью **delete require.cache[...];**
- если модуль удален, то для его использования нужен **новый require.**

```
const http = require('http'); // предоставить доступ к базовому модулю http
```

```
const myModule = require('./module'); // предоставить доступ к небазовому модулю
```

Алгоритм поиска импортируемого модуля

- 1) Проверяется, есть ли **встроенный модуль с указанным именем**. Если есть, импортирует его.
- 2) Если в функцию require передали **путь** до модуля, то импортируется модуль по указанному пути.
- 3) Модуль или пакет ищется в специальной директории **node_modules** по восходящему принципу. После осуществляется поиск среди глобальных пакетов.

```
console.log('http -->', require.resolve('http'));           // модуль базовый
console.log('../Lec04/m04-02 -->', require.resolve('../Lec04/m04-02')); // модуль установлен вручную
console.log('async-->', require.resolve('async'));           // модуль установлен с помощью npm
```

```
// set NODE_PATH=%AppData%\npm\node_modules - для windows 10 (глобально-установленные модули)
```

```
// npm install async      установка локально   %AppData%\npm\node_modules
// npm install -g async   установка глобально
// npm uninstall async    удалить локальный пакет async
// npm uninstall -g async  удалить глобальный пакет async
```

```
D:\PSCA\Lec05>node 05-03
http --> http
../Lec04/m04-02 --> D:\PSCA\Lec04\m04-02.js
async--> C:\Users\Win10_ISiT_Server\AppData\Roaming\npm\node_modules\async\dist\async.js
D:\PSCA\Lec05>
D:\PSCA\Lec05>
```


Если в качестве имени указана папка, то дополнительная информация в файле package.json

```
    "homepage": "https://caolan.github.io/async/",  
    "keywords": [  
      "async",  
      "callback",  
      "module",  
      "utility"  
    ],  
    "license": "MIT",  
    "main": "dist/async.js",  
    "module": "dist/async.mjs",  
    "name": "async",  
    "nyc": {  
      "exclude": [  
        "test"  
      ]  
    },  
    "repository": {  
      "type": "git",  
      "url": "git+https://github.com/caolan/async.git"  
    },  
  },  
}
```

Содержимое функции require

```
D:\PSCA\Lec05>node 05-04
{ [Function: require]
  resolve: { [Function: resolve] paths: [Function: paths] },
  main:
    Module {
      id: '.',
      exports: {},
      parent: null,
      filename: 'D:\\PSCA\\Lec05\\05-04.js',
      loaded: false,
      children: [],
      paths:
        [ 'D:\\PSCA\\Lec05\\node_modules',
          'D:\\PSCA\\node_modules',
          'D:\\node_modules' ] },
  extensions:
    [Object: null prototype] { '.js': [Function], '.json': [Function], '.node': [Function] },
  cache:
    [Object: null prototype] {
      'D:\\PSCA\\Lec05\\05-04.js':
        Module {
          id: '.',
          exports: {},
          parent: null,
          filename: 'D:\\PSCA\\Lec05\\05-04.js',
          loaded: false,
          children: [],
          paths: [Array] } } }
```

Кэширование модуля

```
JS module.js > ...
1  var num = 10;
2
3  function multiply(a) {
4      return num * a;
5  }
6
7  module.exports = {
8      num: num,
9      multiply: multiply
10 }
```

```
JS index.js  X
JS index.js > [?] myModule
1  var myModule = require('./module');
2  console.log(myModule.multiply(2));
3
4  var myModule = require('./module');
5  console.log(myModule.multiply(2));
6
7
```

```
JS module.js  X
JS module.js > [?] num
1  //var num = 10;
2  var num = 5;
3
4  function multiply(a) {
5      return num * a;
6  }
7
8  module.exports = {
9      num: num,
10     multiply: multiply
11 }
```

```
C:\Program Files\nodejs\node.exe --inspect-brk=23066 index.js
Debugger listening on ws://127.0.0.1:23066/542e9a87-1275-4bec-b24f-2f98576525f6
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.
20
20
```

Удаление из кэша

VS Code Variables Panel:

```
Local
> this: Object
  __dirname: "d:\Материалы\свр_04"
  __filename: "d:\Материалы\свр_04\index.js"
> exports: Object {}
> module: Module {id: ".", path: "d:\Матери..."
> myModule: Object {num: 10, multiply: }
> require: function require(path) { ... }
  [[FunctionLocation]]: internal#location
  [[Scopes]]: Scopes[3]
  arguments: TypeError: 'caller', 'callee', and 'a...'
  cache: Object {d:\Материалы\свр_04\index.js: Mod...}
    > d:\Материалы\свр_04\index.js: Module {i...}
      caller: TypeError: 'caller', 'callee', and 'a...'
      ...
```

VS Code Editor (index.js):

```
1 var myModule = require('./module');
2 console.log(myModule.multiply(2));
3
4 //delete require('./module'); // не работает
5 //delete require.cache['./module']; // не работает
6 console.log(require.resolve('./module'));
7 delete require.cache[require.resolve('./module')]; // ок
8 //delete require(require.resolve('./module')); // не работает
9
10 var myModule = require('./module');
11 console.log(myModule.multiply(2));
12
13
```

VS Code Editor (module.js):

```
1 //var num = 10;
2 var num = 5;
3
4 function multiply(a) {
5   return num * a;
6 }
7
```

VS Code Editor (require function):

```
require: function require(path) { ... }
  [[FunctionLocation]]: internal#location
  > [[Scopes]]: Scopes[3]
    arguments: TypeError: 'caller', 'callee', and 'a...'
  > cache: Object {d:\Материалы\свр_04\index.js: Mod...}
    > d:\Материалы\свр_04\index.js: Module {id: ".", ...}
    > d:\Материалы\свр_04\module.js: Module {id: "d:\Материалы\свр_04\module.js", path: "d:\Материалы\свр_04\module.js", caller: TypeError: 'caller', 'callee', and 'arguments' ...}
    > extensions: Object {'.js': , '.json': , '.node': }
    length: 1
  > main: Module {id: ".", path: "d:\Материалы\свр_04\module.js", caller: ...}
  name: "require"
  > prototype: Object {constructor: }
  > resolve: function resolve(request, options) { ... }
  > __proto__: function () { ... }
```

Terminal:

```
C:\Program Files\nodejs\node.exe --inspect-brk=7876 index.js
Debugger listening on ws://127.0.0.1:7876/6f9f6c24-a6e2-4112-b91e-bda46e604a03
For help, see: https://nodejs.org/en/docs/inspector
Debugger attached.
20
d:\Материалы\свр_04\module.js
10
```

Пример

```
//----- m05-09.js -----  
var c = 0;  
module.exports = ()=>{return ++c;}
```

```
//----- 05-09.js -----  
var count1 = require('./m05-09');  
var count2 = require('./m05-09');  
  
console.log('count1 = ', count1());  
console.log('count2 = ', count2());  
console.log('count1 = ', count1());  
console.log('count2 = ', count2());
```

Результат выполнения еще раз доказывает, что модули **кэшируются после первого раза, когда они загружаются.**

```
D:\PSCA\Lec05>node 05-09  
count1 = 1  
count2 = 2  
count1 = 3  
count2 = 4  
  
D:\PSCA\Lec05>
```


exports

```
var x = 1;
global.y = 2;
process.z = 3;

console.log('----- m05-05 -----');
console.log('m05-05.x = ', x); // 1
console.log('m05-05.y = ', y); // 2
// console.log('m05-05.z = ', z); // error
console.log('-----');
console.log('m05-05.global.x = ', global.x); // undefined
console.log('m05-05.global.y = ', global.y); // 2
console.log('m05-05.global.z = ', global.z); // undefined
console.log('-----');
console.log('m05-05.process.x = ', process.x); // undefined
console.log('m05-05.process.y = ', process.y); // undefined
console.log('m05-05.process.z = ', process.z); // 3
console.log('-----');

function Op(op1, op2){
  this.op1 = op1;
  this.op2 = op2;
  this.add = ()=>{return this.op1+this.op2;}
  this.sub = ()=>{return this.op1-this.op2;};
}

exports.X = x;
exports.Y = y;
exports.OP = Op;
```

Каждый модуль имеет свою собственную область видимости. Другими словами, переменные и функции, объявленные в модуле, не видны в других файлах.

```
var m05 = require('./m05-05');

console.log('-----05-05-----');
console.log('05-05.global.x = ', global.x); // undefined
console.log('05-05.global.y = ', global.y); // 2
console.log('05-05.process.z = ', process.z); // 3

console.log('-----');
console.log(' (new m05.OP(1,2)).add() =', (new m05.OP(1,2)).add()); // 3
console.log(' (new m05.OP(1,2)).sub() =', (new m05.OP(1,2)).sub()); // -1
console.log(' m05.X = ', m05.X); // 1
console.log(' m05.Y = ', m05.Y); // 2
console.log('-----');
global.y = 22;
console.log('05-05.global.y = ', global.y); // 22
console.log(' m05.Y = ', m05.Y); // 2
```

module.exports vs exports

```
var num = 5;

// var exports = module.exports = {};
exports.multiply = function (a) {
  return num * a;
}

module.exports = num;
```

exports – это просто **ссылка** на module.exports.

```
var myModule = require('./module');

console.log(myModule.multiply(2)); // Error: myModule.multiply is not a function
console.log(myModule);             // 5
```

ES6 Module

- официальный **стандартный формат** упаковки кода Javascript;
- операторы **import** и **export**;
- по умолчанию в Node.js используется формат CommonJS, поэтому, **чтобы разрешить использование ES6**, нужно:
 - 1) указать расширение файла **.mjs**;
 - или
 - 2) указать поле **"type": "module"** в package.json.

Enabling

#

Node.js treats JavaScript code as CommonJS modules by default. Authors can tell Node.js to treat JavaScript code as ECMAScript modules via the .mjs file extension, the `package.json` "type" field, or the --input-type flag. See [Modules: Packages](#) for more details.

Экспорт по умолчанию до объявления

```
var num = 10;  
  
export default function multiply(a) {  
  return num * a;  
}
```

Экспорт по умолчанию после объявления

```
var num = 10;  
  
function multiply(a) {  
  return num * a;  
}  
  
export {multiply as default};
```

Модули могут иметь **только один** экспорт по умолчанию.

Импорт экспорта по умолчанию

```
import myModule from './module.js';  
console.log(myModule(2));
```

При импорте экспорта по умолчанию ему можно дать **любое имя** и **не нужно использовать фигурные скобки**.

Именованный экспорт до объявления

```
// export class  
// export const  
// export let  
export function multiply(a) {  
    return 10 * a;  
}
```

Именованный экспорт после объявления

```
var num = 5;  
  
function multiply(a) {  
    return num * a;  
}  
  
export {  
    multiply, num  
};
```

Именованные экспорты
позволяют выполнять несколько
экспортов в одном файле.

Импорт именованного экспорта

```
import { multiply as mul } from './module.js';  
console.log(mul(2));
```

Можно переименовать экспорт **псевдонимом**, если есть коллизии в файле.

```
import {multiply, num} from './module.js';  
  
console.log(multiply(2));  
console.log(num);
```

Когда нужно импортировать именованный компонент, **нужно использовать то же имя, с которым он был экспортирован**. Имена должны быть импортированы **внутри фигурных скобок**.

Можно включить в импорт **несколько** компонентов.

```
import * as myModule from './module.js';  
  
console.log(myModule.multiply(2));  
console.log(myModule.num);
```

Можно **импортировать все** именованные экспорты в объект.

Динамический import

```
var num = 5;

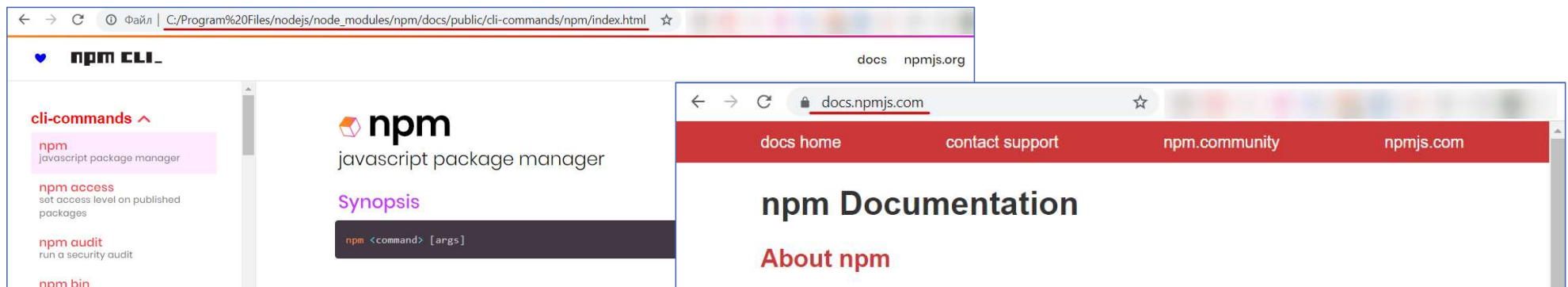
function multiply(a) {
  return num * a;
}

export {
  multiply, num
};
```

```
import('./module3.js')
  .then((module) => {
    console.log(module.multiply(2));
    console.log(module.num);
  });
```

NPM (Node Package Manager)

- устанавливается вместе с Node.js;
- предназначен для скачивания/публикации пакетов;
- инструмент командной строки;
- глобальное хранилище <https://registry.npmjs.org/>
- <https://www.npmjs.com/>



Пакет

=

один или несколько js-файлов
и файл-манифест `package.json`.

Просмотр локальных пакетов

ls, list, la, ll

```
D:\PSCA\Lec05>npm list  
D:\PSCA\Lec05  
`-- (empty)
```

```
D:\PSCA\Lec05>npm ls  
D:\PSCA\Lec05  
`-- (empty)
```

```
D:\PSCA\Lec05>npm la  
| D:\PSCA\Lec05  
|  
`-- (empty)
```

```
D:\PSCA\Lec05>npm ll  
| D:\PSCA\Lec05  
|  
`-- (empty)
```


Скачивание install, i, add

```
D:\PSCA\Lec05>npm install nodemailer
npm WARN saveError ENOENT: no such file or directory, open 'D:\PSCA\Lec05\package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN enoent ENOENT: no such file or directory, open 'D:\PSCA\Lec05\package.json'
npm WARN Lec05 No description
npm WARN Lec05 No repository field.
npm WARN Lec05 No README data
npm WARN Lec05 No license field.

+ nodemailer@6.3.0
added 1 package from 1 contributor and audited 1 package
found 0 vulnerabilities
```



Имя	Дата изменения	Тип	Размер
lib	19.08.2019 23:05	Папка с файлами	
.DS_Store	26.10.1985 11:15	Файл "DS_STORE"	9 КБ
.prettierrc.js	26.10.1985 11:15	JetBrains WebStorm	1 КБ
CHANGELOG.md	26.10.1985 11:15	Файл "MD"	21 КБ
CONTRIBUTING.md	26.10.1985 11:15	Файл "MD"	5 КБ
LICENSE	26.10.1985 11:15	Файл	1 КБ
<u>package.json</u>	19.08.2019 23:05	JSON File	2 КБ
README.md	26.10.1985 11:15	Файл "MD"	5 КБ

Удаление

uninstall, remove, rm, r, un, unlink

```
D:\PSCA\Lec05>npm uninstall nodemailer
npm WARN saveError ENOENT: no such file or directory, open 'D:\PSCA\Lec05\package.json'
npm WARN enoent ENOENT: no such file or directory, open 'D:\PSCA\Lec05\package.json'
npm WARN Lec05 No description
npm WARN Lec05 No repository field.
npm WARN Lec05 No README data
npm WARN Lec05 No license field.

removed 1 package in 1.473s
found 0 vulnerabilities
```

```
D:\PSCA\Lec05>npm list
D:\PSCA\Lec05
`-- (empty)
```

Глобальные пакеты

-g

```
D:\PSCA\Lec05>npm list -g
C:\Users\Win10_ISiT_Server\AppData\Roaming\npm
+-- async@3.1.0
`-- bower@1.8.8

D:\PSCA\Lec05>npm ls -g
C:\Users\Win10_ISiT_Server\AppData\Roaming\npm
+-- async@3.1.0
`-- bower@1.8.8

D:\PSCA\Lec05>npm la -g
| C:\Users\Win10_ISiT_Server\AppData\Roaming\npm
|
+-- async@3.1.0
|   Higher-order functions and common patterns for asynchronous code
|   git+https://github.com/caolan/async.git
|   https://caolan.github.io/async/
+-- bower@1.8.8
|   The browser package manager
|   git+https://github.com/bower/bower.git
|   http://bower.io
|
D:\PSCA\Lec05>npm ll -g
| C:\Users\Win10_ISiT_Server\AppData\Roaming\npm
|
+-- async@3.1.0
|   Higher-order functions and common patterns for asynchronous code
|   git+https://github.com/caolan/async.git
|   https://caolan.github.io/async/
+-- bower@1.8.8
|   The browser package manager
|   git+https://github.com/bower/bower.git
|   http://bower.io
```

```
D:\PSCA\Lec05>npm install nodemailer -g
+ nodemailer@6.3.0
added 1 package from 1 contributor in 0.495s
```

```
D:\PSCA\Lec05>npm list
D:\PSCA\Lec05
`-- (empty)
```

```
D:\PSCA\Lec05>npm list -g
C:\Users\Win10_ISiT_Server\AppData\Roaming\npm
+-- async@3.1.0
+-- bower@1.8.8
+-- nodemailer@6.3.0
```

Информация о пакете

view, info, show, v

```
D:\PSCA\Lec05>npm view nodemailer

nodemailer@6.3.0 | MIT | deps: none | versions: 219
Easy as cake e-mail sending from your Node.js applications
https://nodemailer.com/

keywords: Nodemailer

dist
.tarball: https://registry.npmjs.org/nodemailer/-/nodemailer-6.3.0.tgz
.shasum: a89b0c62d3937bdcdeecbf55687bd7911b627e12
.integrity: sha512-TEHBNBPHv7Ie/0o3HXnb7xrPSSQmH1dXwQKRuMKDBGt/2
.unpackedSize: 478.3 kB

maintainers:
- andris <andris@node.ee>

dist-tags:
beta: 2.4.0-beta.0  latest: 6.3.0

published a month ago by andris <andris@kreat.ee>
```

```
D:\PSCA\Lec05>npm view request

request@2.88.0 | Apache-2.0 | deps: 20 | versions: 125
Simplified HTTP request client.
https://github.com/request/request#readme

keywords: http, simple, util, utility

dist
.tarball: https://registry.npmjs.org/request/-/request-2.88.0.tgz
.shasum: 9c2fca4f7d35b592efe57c7f0a55e81052124fef
.integrity: sha512-NAqBSrijGLZdMOWZNsInLJpkJokL72XYjUpnB0iwsRgxt
.unpackedSize: 206.9 kB

dependencies:
aws-sign2: ~0.7.0          extend: ~3.0.2          http-signature: ~1.2.0
aws4: ^1.8.0              forever-agent: ~0.6.1  is-typedarray: ~1.0.0
caseless: ~0.12.0         form-data: ~2.3.2      isstream: ~0.1.0
combined-stream: ~1.0.6   har-validator: ~5.1.0  json-stringify: ~1.0.0

maintainers:
- fredkschott <fkschott@gmail.com>
- mikeal <mikeal.rogers@gmail.com>
- nylen <jnylen@gmail.com>
- simov <simeonvelichkov@gmail.com>

dist-tags:
latest: 2.88.0

published a year ago by mikeal <mikeal.rogers@gmail.com>
```

Поиск пакета

search, s, se, find

```
D:\PSCA\Lec05>npm search request
```

NAME	DESCRIPTION	AUTHOR	DATE	VERSION	KEYWORDS
request	Simplified HTTP...	=fredkschott...	2018-08-10	2.88.0	http simple util ut
request-promise-native	The simplified HTTP...	=analog-nico...	2019-02-15	1.0.7	xhr http https prom
got	Simplified HTTP...	=sindresorhus...	2019-01-17	9.6.0	http https get got
request-promise	The simplified HTTP...	=analog-nico...	2019-02-15	4.2.4	xhr http https prom
request-promise-core	Core Promise...	=analog-nico	2019-02-14	1.1.2	xhr http https prom
request-progress	Tracks the download...	=satazor	2016-12-01	3.0.0	progress request mi
morgan	HTTP request logger...	=dougwilson	2018-09-11	1.9.1	express http logger
cacheable-request	Wrap native HTTP...	=lukechilds	2019-06-07	6.1.0	HTTP HTTPS cache ca
superagent	elegant & feature...	=defunctzombie...	2019-06-15	5.1.0	agent ajax ajax api
teeny-request	Like request, but...	=fhinkel...	2019-08-14	5.2.1	request node-fetch
request-ip	A small node.js...	=pbojinov	2018-10-29	2.1.3	request ip ip addre
@octokit/request	Send parameterized...	=bkeepers...	2019-07-25	5.0.2	octokit github api
cookie-parser	Parse HTTP request...	=defunctzombie...	2019-02-13	1.4.4	cookie middleware
simple-get	Simplest way to...	=feross	2018-08-09	3.0.3	request http GET ge
type-is	Infer the...	=dougwilson...	2019-04-26	1.6.18	content type checki
raf	requestAnimationFrame...	=chrisdickinson...	2018-11-02	3.4.1	requestAnimationFrame
proxy-addr	Determine address...	=dougwilson	2019-04-16	2.0.5	ip proxy x-forwarded
timed-out	Emit 'ETIMEDOUT' or...	=floatdrop...	2017-01-16	4.0.1	http https get got
retry-request	Retry a request.	=stephenpluspl...	2019-06-18	4.1.1	request retry strea
co-body	request body...	=dead_horse...	2018-05-21	6.0.0	request parse parse

Nodemailer

```
PS D:\NodeJS\samples\cwp_05\mail_nodemailer> npm install nodemailer  
added 1 package, and audited 10 packages in 828ms
```

```
PS D:\NodeJS\samples\cwp_05\mail_nodemailer> npm install nodemailer-smtp-transport  
added 8 packages, and audited 10 packages in 2s
```

Nodemailer

```
const nodemailer = require('nodemailer');
const smtpTransport = require('nodemailer-smtp-transport');

const sender = ' ';
const receiver = ' ';
const pass = ' ';

function send(message) {
  let transporter = nodemailer.createTransport(smtpTransport({
    host: 'smtp.gmail.com',
    port: 587,
    secure: false,
    auth: {
      user: sender,
      pass: pass
    }
  }));

  var mailOptions = {
    from: sender,
    to: receiver,
    subject: 'Lab6',
    text: message,
    html: `<i>${message}</i>`,
    attachments: [{ filename: 'cat.jpg', path: __dirname + '/cat.jpg', cid: 'img' }]
  };

  transporter.sendMail(mailOptions, function (error, info) {
    error ? console.log(error) : console.log('Email sent: ' + info.response);
  });

  send('Test message');
}
```

```
PS D:\NodeJS\samples\cwp_05\mail_nodemailer> node 05-02
Email sent: 250 2.0.0 OK 1665175261 k13-20020adff28d000000b0022ac672654dsm2780957wro.58 - gsmt
PS D:\NodeJS\samples\cwp_05\mail_nodemailer>
```

Lab6 Входящие x

dubovik.m14@gmail.com

кому: мне ▼

Test message



Обработка параметров с формы

```
var http = require('http');
var fs = require('fs');
var url = require('url');
const { parse } = require('qs');

let http_handler = (req, resp) => {

  resp.writeHead(200, {'Content-Type': 'text/html; charset=utf-8'});
  if (url.parse(req.url).pathname == '/' && req.method == 'GET' ) {
    resp.end(fs.readFileSync('./05-07.html'));
  } else if (url.parse(req.url).pathname == '/' && req.method == 'POST' ) {
    let body = '';
    req.on('data', chunk => {body += chunk.toString();});
    req.on('end', () => {
      let parm = parse(body);
      resp.end(`<h1>OK: ${parm.reciver}, ${parm.sender}, ${parm.message} </h1>`);
    })
  } else resp.end('<h1>Not support</h1>');
}

let server = http.createServer(http_handler);
server.listen(3000);
console.log('Server running at http://localhost:3000/');
```


Обработка параметров с формы

05-07 x +

localhost:3000

Lec 05

Отправитель	box@mailserver
Получатель	box@mailserver
Сообщение	Выполнил
OK	

05-07 x +

localhost:3000

Lec 05

Отправитель	smw60@mail.ru
Получатель	smw@belstu.by
Сообщение	Выполнил Смелов В.В.
OK	

localhost:3000 x +

localhost:3000

OK: smw60@mail.ru, smw@belstu.by, Выполнил Смелов В.В.

Порядок публикации пакета

1. Регистрация на <https://www.npmjs.com/>
2. Добавление учетной записи пользователя (adduser, login). Потребуется указать name, password, email

```
D:\PSCA\Lec05>npm adduser  
Username: smw60
```

3. Проверка (отобразить имя пользователя)

```
D:\PSCA\Lec05>npm whoami  
smw60
```

Порядок публикации пакета

4. Инициализация проекта (init)

```
D:\PSCA\Lec05\node_modules\p0508>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.
```

```
Press ^C at any time to quit.
package name: (p0508) p0508
version: (1.0.0) 1.0.1
```

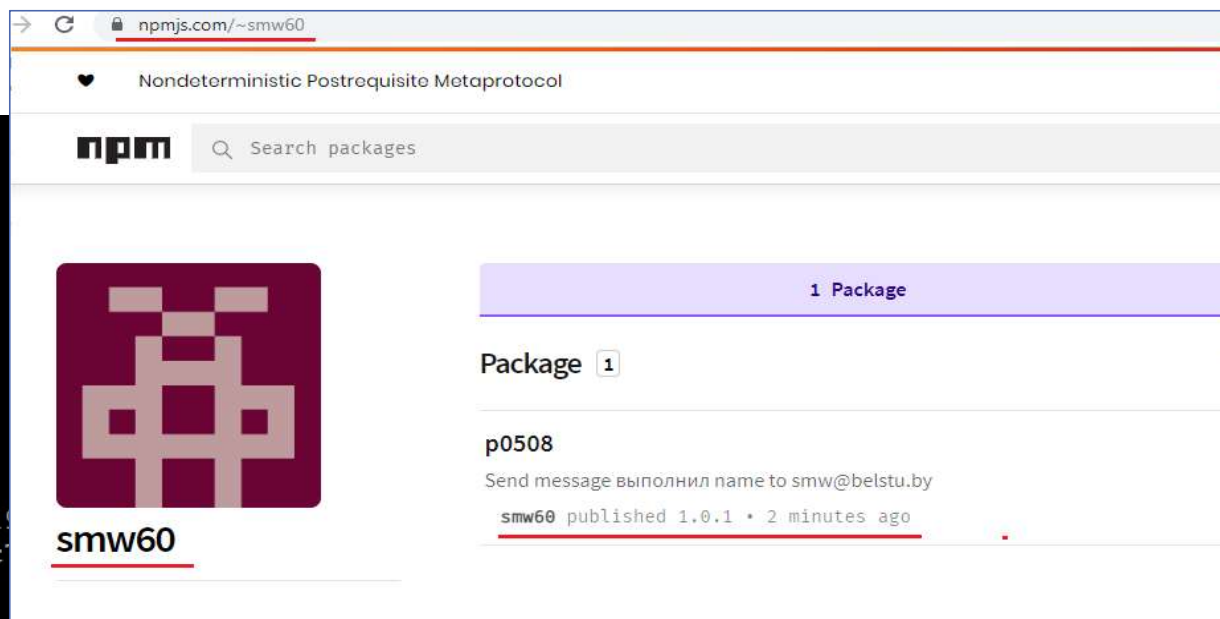
WORK (D:) > PSCA > Lec05 > node_modules > p0508				
Имя	Дата изменения	Тип	Размер	
node_modules	20.08.2019 15:06	Папка с файлами		
index.js	20.08.2019 16:41	JetBrains WebStorm	1 КБ	
package.json	20.08.2019 16:40	JSON File	1 КБ	

```
{
  "dependencies": {"sendmail": "1.6.1"},
  "description": "send message выполнил name to smw@belstu.by",
  "keyword": "send name in email-message to smw@belstu.by without smtp",
  "main": "index.js",
  "maintainers": [ {"name": "smw60", "email": "smw60@mail.ru" }],
  "name": "p0508",
  "publishConfig": { "access": "public"},
  "version": "1.0.1"
}
```

Порядок публикации пакета

5. Публикация пакета: publish (--access=public, если есть scope)

```
D:\PSCA\Lec05\node_modules\p0508>npm publish
npm notice
npm notice package: p0508@1.0.1
npm notice === Tarball Contents ===
npm notice 419B package.json
npm notice 751B index.js
npm notice === Tarball Details ===
npm notice name: p0508
npm notice version: 1.0.1
npm notice package size: 672 B
npm notice unpacked size: 1.2 kB
npm notice shasum: d5c364d4b7c759d004714fa9219
npm notice integrity: sha512-tVkyYT+H5urHG[...]c
npm notice total files: 2
npm notice
+ p0508@1.0.1
```



Файл package.json =

файл конфигурации приложения Node.js. Любая директория, содержащая данный файл, интерпретируется как Node.js-пакет.

Содержит метаданные проекта (название, версия, описание проекта, ...), список зависимостей вашего проекта, которые будут установлены при вызове команды `npm install`, скрипты, вызывающие другие команды консоли.

Не забывайте всегда добавлять папку `node_modules` в `.gitignore`, т.к. папка может весить гигабайты.



Создание package.json

1. Создание вручную
2. Создание с помощью команды npm init

```
PS D:\Материалы\cwp_04> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
```

```
See `npm help json` for definitive documentation on these fields
and exactly what they do.
```

```
Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.
```

```
Press ^C at any time to quit.
package name: (cwp_04) test
version: (1.0.0)
description: Testing npm package creation
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
```



```
{
  "name": "test",
  "version": "1.0.0",
  "description": "Testing npm package creation",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```

Содержимое package.json

- **name** – имя пакета. Должно быть короче 214 символов, состоять только из букв в нижнем регистре, цифр, символов подчеркивания (_) и дефисов (-), без использования пробелов.
- **version** – текущая версия пакета. Должна соответствовать правилам семантического версионирования (X.Y.Z).
- **description** – краткое описание пакета. Если планируется публикация пакета, данное свойство поможет пользователям сайта npm понять назначение пакета.

Содержимое package.json

- **main** – точка входа в проект.
- **scripts** – набор скриптов, которые можно запускать средствами npm. Запуск производится через консоль командами вида `npm run XXX` или `yarn XXX`, где XXX – имя скрипта (в примере – “test”).
- **author** – информация об авторе пакета (строка/json-объект). Может содержать имя автора, email и адрес сайта.

```
"author": "John Wick <jw@mail.com> (https://revengeforthedog.com)",  
"author": {  
  "name": "John Wick",  
  "email": "jw@mail.com",  
  "url": "https://revengeforthedog.com"  
},
```

Содержимое package.json

- **license** – информация о лицензии пакета.
- **keywords** – массив ключевых слов, относящихся к функционалу проекта. Правильно подобранные ключевые слова могут помочь людям быстрее найти нужный пакет при просмотре сайта npm.

```
"keywords": [
  "test",
  "email",
  "package management"
]
```

Содержимое package.json

- **dependencies** – список всех пакетов, которые используются при работе приложения. Команда `npm install <package_name>` добавит пакет в список. Команда `npm install` загрузит все зависимости, перечисленные в данной секции.
- **devDependencies** – информация о пакетах, которые используются в процессе разработки, но не используются при работе самого приложения. Команда `npm install <package_name> --save-dev` для добавления нового пакета. Команда `npm install` не устанавливает эти пакеты.

```
"dependencies": {  
  "express": "^4.17.1",  
  "react": "^16.13.1"  
}
```

```
"devDependencies": {  
  "gulp": "^4.0.2"  
}
```

при удалении пакета с использованием команды `npm uninstall <package_name>` запись о нем также будет автоматически удалена из списка зависимостей.

Пример package.json

```
"name": "nodemailer",
"version": "6.9.1",
"description": "Easy as cake e-mail sending from your Node.js applications",
"main": "lib/nodemailer.js",
"scripts": {
  "test": "grunt --trace-warnings"
},
"repository": {
  "type": "git",
  "url": "https://github.com/nodemailer/nodemailer.git"
},
"keywords": [
  "Nodemailer"
],
"author": "Andris Reinman",
"license": "MIT",
"bugs": {
  "url": "https://github.com/nodemailer/nodemailer/issues"
},
"homepage": "https://nodemailer.com/",
```

```
"devDependencies": {
  "@aws-sdk/client-ses": "3.259.0",
  "aws-sdk": "2.1303.0",
  "bunyan": "1.8.15",
  "chai": "4.3.7",
  "eslint-config-nodemailer": "1.2.0",
  "eslint-config-prettier": "8.6.0",
  "grunt": "1.5.3",
  "grunt-cli": "1.4.3",
  "grunt-eslint": "24.0.1",
  "grunt-mocha-test": "0.13.3",
  "libbase64": "1.2.1",
  "libmime": "5.2.0",
  "libqp": "2.0.1",
  "mocha": "10.2.0",
  "nodemailer-ntlm-auth": "1.0.3",
  "proxy": "1.0.2",
  "proxy-test-server": "1.0.0",
  "sinon": "15.0.1",
  "smtp-server": "3.11.0"
},
"engines": {
  "node": ">=6.0.0"
}
```

Семантическое версионирование

- **спецификация**, которая описывает правила присвоения версии релизам программного обеспечения. Сайт - <https://semver.org>.
- призвано **решить проблему “ада зависимостей” (dependency hell)**. Данное состояние может наступить, когда проект имеет слишком много зависимостей (невозможность обновить пакет без необходимости выпуска новой версии каждой зависимой библиотеки). Решение – нумерация версий.

MAJOR.MINOR.PATCH

* целые, неотрицательные числа

- **MAJOR** – новые возможности без сохранения обратной совместимости. Начальная разработка – значение 0. Когда увеличивается, значения MINOR и PATCH должны быть обнулены.
- **MINOR** – новые возможности или измененные старые с сохранением совместимости. Должно быть изменено, если какой-либо функционал помечен как устаревший. При увеличении значения, значение PATCH должно быть обнулено.
- **PATCH** – исправление ошибок, рефакторинг.

- **Нельзя изменять уже опубликованные версии.** Любые изменения в проекте должны сопровождаться новой версией.
- **Предрелиз** может быть обозначен **добавлением дефиса** и серией разделённых точкой идентификаторов, следующих сразу за RATCH. Должна содержать лишь буквенно-цифровые символы и дефис. Не должна начинаться с нуля. Указывает на то, что эта версия не стабильна и может не удовлетворять требованиям совместимости.

Пример: 1.0.0-alpha, 1.0.0-alpha.1, 1.0.0-beta

- **Сборочные метаданные** могут быть обозначены **добавлением знака плюс** и ряда разделённых точкой идентификаторов, следующих сразу за RATCHN или предрелизной версией. Должны содержать лишь буквенно-цифровые символы и дефис.

Пример: 1.0.0+20130313144700, 1.0.0-beta+exp.sha.5114f85.

- **Приоритет версии** определяется по первому отличию при сравнении идентификаторов **слева направо**. Сборочные метаданные при определении приоритета не учитываются.

Пример: 1.0.0 < 2.0.0 < 2.1.0 < 2.1.1

Спецификаторы версий

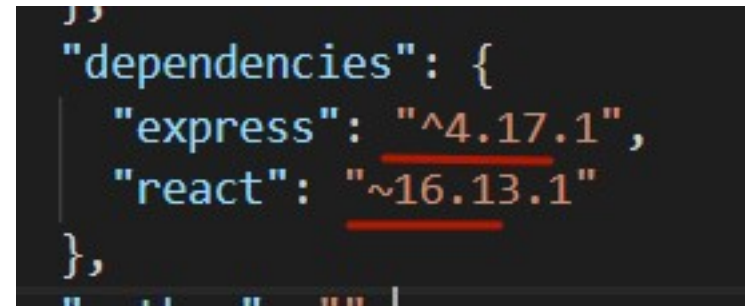
- ~: подходят **новые PATCH** версии

Пример: ~0.13.0: 0.13.1 — подходит, 0.14.0 — нет

- ^: подходят **новые PATCH и MINOR** версии

Пример: ^0.13.0: 0.13.1, 0.14.0 и т.д. — подходят

- *: подходят **новые MAJOR** версии
- >: подходят любые версии пакета, которые **больше заданной**
- >=: подходят любые версии пакета, которые **равны или больше заданной**
- <=: подходят пакеты, версии которых **равны заданной или меньше её**
- <: подходят пакеты, версии которых **меньше заданной**



```
    "dependencies": {  
      "express": "^4.17.1",  
      "react": "~16.13.1"  
    },  
    "devDependencies": {
```

Спецификаторы версий

- =: подходит **только заданная версия** пакета
- -: подходит все из **диапазона версий**

Пример: 2.1.0 - 2.6.2

- ||: позволяет **комбинировать** наборы условий, касающихся пакетов

Пример: < 2.1 || > 2.6

- отсутствие дополнительных символов: подходит **только заданная версия** пакета-зависимости и никакая другая
- latest: указывает на то, что вам требуется **самая свежая версия** пакета.

Калькулятор версий

npm semantic version calculator

semver.npmjs.com

npm semver calculator

New to semantic versioning? [Learn the basics.](#)

pick a package

express

enter a range

^4.17.1

• 0.1.0	• 0.8.2	• 2.2.1	• 3.9.2	• 4.9.0	• 4.17.0
• 0.2.0	• 0.9.0	• 2.3.0	• 3.9.3	• 4.10.0	• 4.17.1
• 0.2.1	• 0.9.1	• 2.4.0	• 3.10.0	• 4.11.0	• 4.17.2
• 0.2.2	• 0.9.2	• 2.4.1	• 3.10.1	• 4.11.1	• 4.17.3
• 0.3.0	• 0.10.0	• 3.0.0	• 4.0.0	• 4.11.2	• 4.17.4
• 0.3.1	• 1.0.0-rc.1	• 3.0.1	• 4.0.1	• 4.12.0	• 4.17.5
• 0.3.2	• 1.0.0-rc.2	• 3.1.0	• 4.1.0	• 4.13.0	• 4.17.9
• 0.4.0	• 1.0.0-rc.3	• 3.2.0	• 4.2.0	• 4.13.1	• 4.17.10

<https://semver.npmjs.com/>