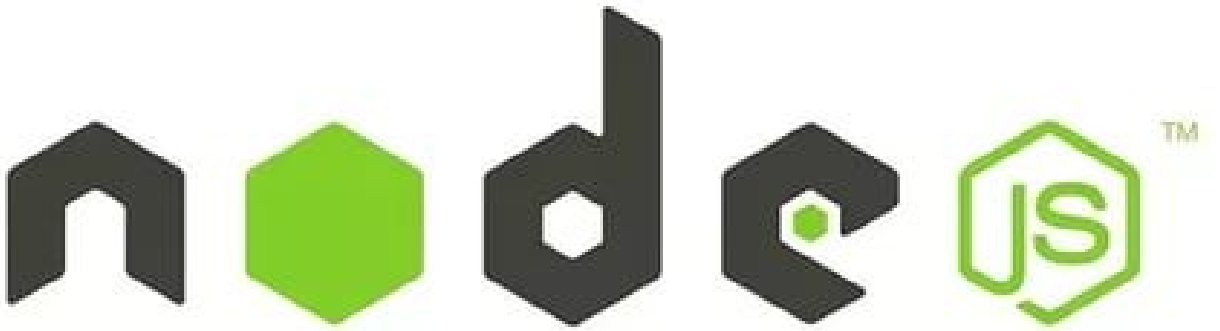

node
JS
TM

Node.js - пакеты

- Пакетный менеджер npm
- Создание манифест файла
- Установка Express

1 | Пакетный менеджер npm

С помощью npm можно устанавливать пакеты локально или глобально. В локальном режиме пакеты устанавливаются в каталог **node_modules** родительского каталога. Владельцем каталога является текущий пользователь. Глобальные пакеты устанавливаются в каталог {prefix}/lib/node_modules/, владельцем которого является root (префиксом в данном случае обычно является каталог /usr/ или /usr/local).

1 | Пакетный менеджер npm | установка

Глобальная установка пакетов. Для глобальной установки используется флаг --global, его можно записывать сокращенно -g.

```
$ npm install express  --global
$ npm install express  -g
```

Локальная установка используется в npm по умолчанию.

достаточно не использовать флаг --global

```
$ npm install express
```

При установке определенной версии пакета используется @

```
$ npm install underscore@1.8.2
```

1 | Пакетный менеджер npm | проверка

Куда будет установлен пакет ?

```
$ npm config get prefix
/usr
```

Полученный префикс нужно подставить в

```
{prefix}/lib/node_modules/
```

Вывод списка глобально установленных пакетов

```
$ npm list --global
$ npm list -g --depth=0
```

Вывод списка локально установленных пакетов

```
$ npm list
$ npm list --depth=0
```

1 | Пакетный менеджер npm | удаление

Удаление пакетов

```
$ npm uninstall underscore
```

Обновление пакетов при выходе новой заплатки

```
$ npm update underscore
```

Поиск пакетов в репозитарии по названию

```
$ npm search mkdir
```

1 | Пакетный менеджер npm | подключение

Пакет в большинстве случаев содержит модуль который мы можем подключить в нашем приложении. Установим пакет и подключим модуль в коде.

```
$ npm install mkdir
```

mkdir.js:

```
var mkdirp = require('mkdirp');
mkdirp('foo', function (err) {
    if (err) console.error(err)
    else console.log('Directory created!')
});
```

```
$ node. mkdir.js
Directory created!
```

1 | Пакетный менеджер npm | манифест

Создадим манифест файл package.json для управления зависимостями. Он генерируется с помощью команды npm init, которая выведет в консоль несколько вопросов для создания файла:

```
$ npm init
$ npm init --yes //без вопросов
```

Результат:

```
{
  "name": "tmp",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

1 | Пакетный менеджер npm | поддержка

Поддержка актуальности манифест файла при установке нового пакета осуществляется с помощью флага --save. При этом в манифест в dependencies добавляется строка с названием установленного пакета.

```
$ npm install mkdir --save
```

1 | Пакетный менеджер npm | package.json

Для установки модулей из манифест файла package.json нужно использовать команду.

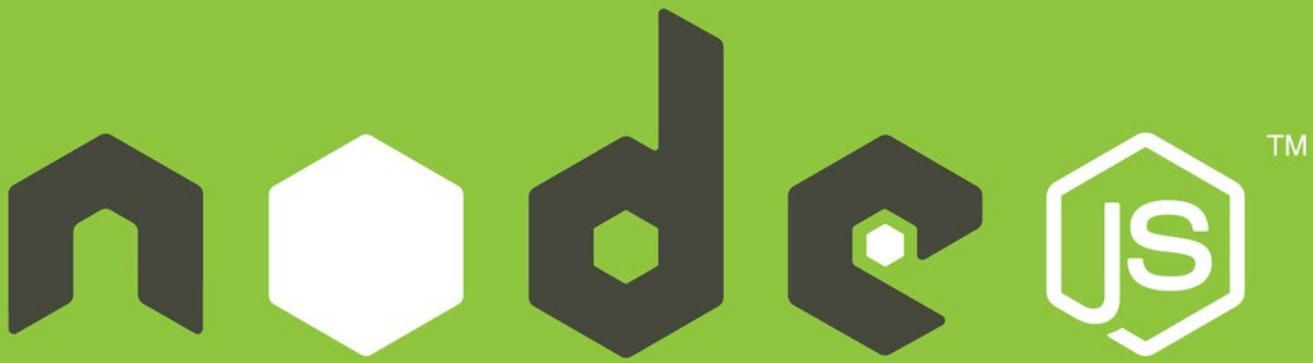
```
$ npm install
```

1 | Пакетный менеджер npm | заключение

!

В каталоге npm тысячи пакетов и их число будет расти. Используйте уже готовые решения для экономии времени и во избежании кривых костыльных решений. Пользуйтесь поиском.

Express.js



2 | Установка express

`Express –` это минималистичный и гибкий веб-фреймворк для приложений Node.js, предоставляющий обширный набор функций для мобильных и веб-приложений.

2 | Установка express

Устанавливаем модуль express глобально:

```
$ npm install -g express
```

```
$ npm install -g express-generator
```

В папке проектов сгенерируем каркас выполнив консольную команду:

```
$ express chat
```

2 | Установка express

В результате выполнения предыдущей команды в консоле увидим следующее:

```
create : chat
create : chat/package.json
create : chat/app.js
create : chat/public
create : chat/public/javascripts
create : chat/public/images
create : chat/public/stylesheets
create : chat/public/stylesheets/style.css
create : chat/routes
create : chat/routes/index.js
create : chat/routes/users.js
create : chat/views
create : chat/views/index.jade
create : chat/views/layout.jade
create : chat/views/error.jade
create : chat/bin
create : chat/bin/www
```

2 | Установка express

В сгенерированных файлах есть файл манифест package.json который содержит в себе описание характеристик и зависимостей проекта от других модулей. Добавим сюда еще одну зависимость:

```
{
  "name": "chat",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "body-parser": "~1.13.2",
    "cookie-parser": "~1.3.5",
    "debug": "~2.2.0",
    "express": "~4.13.1",
    "jade": "~1.11.0",
    "mustache": "*",  // шаблонизатор
    "morgan": "~1.6.1",
    "serve-favicon": "~2.3.0"
  }
```

2 | Установка express

После чего находясь в папке проекта нам необходимо запустить команду:

```
$ npm install
```

В процессе выполнения мы увидим установку модулей которые были прописаны в манифест файле в `dependencies`. В папке проекта появится дополнительная папка `node_modules` которая будет содержать инсталлированные модули.

Для просмотра установленных локальных модулей введите команду:

```
$ npm list --dept=0
```

2 | Установка express

Для того чтобы мы могли увидеть работу http сервера в файл app.js нужно добавить несколько строк кода:

```
var port = process.env.PORT || 3000;
...
app.listen(port);
```

Чтобы проверить работу сервера express запустим его командой:

```
$ node app.js
```

И перейдем в браузере по адресу: `localhost:3000`

Полезный ссылки

http://expressjs.com/ru/ - express минималистичный веб фреймворк