







# Yarn

**LoftSchool**  
ОТ МЫСЛИТЕЛЯ К СОЗДАТЕЛЮ

## Пояснения к методическим указаниям

### Источники:

-  [Документация](#)
-  [CLI документации](#)
-  [Переход с npm на Yarn](#)
-  [5 вещей, которые вы можете сделать с Yarn](#)

### Замечания:

- Данные методические указания (далее “методичка”) не могут заменить официальную документацию! Они представляют из себя облегченную версию документации, переведенную на русский язык и разбавленную опытом преподавателей LoftSchool.
- Многие термины являются сложно переводимыми на русский язык, ввиду их специфичности. Поэтому не удивляйтесь, если русский язык будет соседствовать с английскими терминами. Среди них:
  - “package” будем переводить как “npm-модуль”, “модуль”, или “пакет”.
  - “dependency” — модули, от которых “зависит” ваш модуль. Будем переводить “зависимость”, “пакет” или “модуль”.

## Почему yarn?

### Определение

Yarn — это современная альтернатива npm. Yarn работает с **тем же файлом package.json** и так же скачивает необходимые модули **в папку node-modules**. Однако он делает это **быстрее**, чем npm. Пожалуй, для большинства разработчиков это его главное и единственное действительно ощутимое преимущество.

### Какие у Yarn преимущества перед npm?

1. Yarn **кеширует все модули, которые загружает**. Вам никогда не придется загружать один и тот же модуль дважды.
2. Это также означает, что вы можете иметь доступ к ранее загруженным модулям удобно и **без подключения к сети интернет**.
3. Использует чексуммы ("checksums"), чтобы верифицировать каждый устанавливаемый модуль. Это даёт **новый уровень безопасности**.
4. Использует специальный локфайл ("lockfile") в связке с детерминированным алгоритмом установки модулей, что позволяет Yarn гарантировать, что **ваше приложение будет одинаково устанавливаться и работать на разных системах** (у npm это одно из слабых мест).



## *Зачем мне изучать и применять Yarn?*

В октябре 2016 года команда Facebook (компания, подарившая нам, в числе прочего, пожалуй самый популярный в 2016 году frontend фреймворк — React) анонсировала новый пакетный менеджер под названием Yarn. В 2017 году yarn начал набирать популярность и продолжает делать это очень активно.

В официальных документациях многих популярных npm-пакетов (например, для webpack), мы видим в инструкциях по установке — **Yarn как рекомендуемый пакетный менеджер**.

Нравится нам это или нет — Yarn постепенно становится стандартом. А это значит, что работодатели (а также тимлиды и айти-директоры) **будут требовать от соискателей знания Yarn**.

Выпускники Лофтскула будут знать и уметь пользоваться Yarn. И это хорошо: ученики добавят ещё одну технологию в копилку своих знаний, а значит станут (хоть и немного, но) более конкурентоспособны на рынке труда.

## *Yarn и npm могут работать вместе?*

Могут, без проблем. В рамках одного проекта разные разработчики могут работать с yarn и npm - по своему вкусу. Вы также всегда можете легко перейти от yarn обратно к npm.

## Установка

### MacOS

При помощи [brew](#)

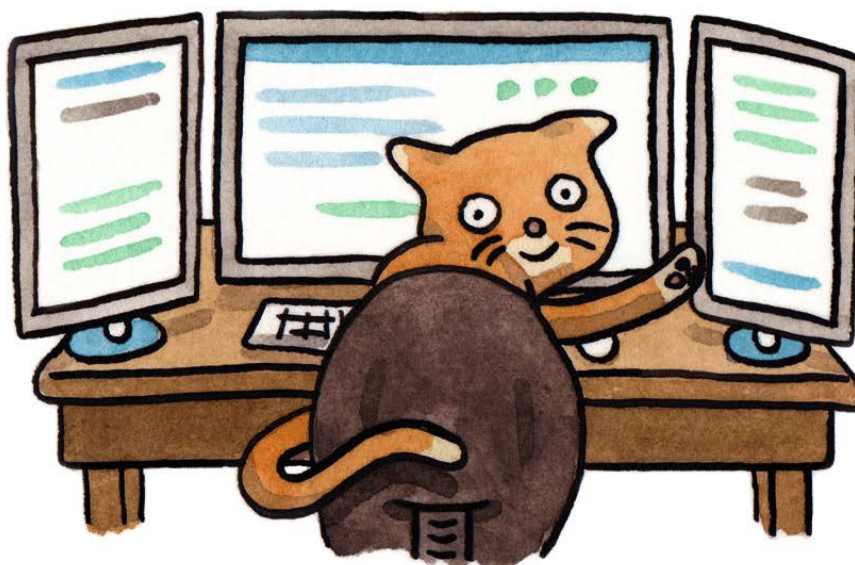
```
brew update  
brew install yarn
```

После установки Yarn нужно добавить в PATH

Добавьте **`export PATH="$PATH: `yarn global bin``** в ваш профайл (**`.profile`**, **`.bashrc`**, **`.zshrc`** )

После чего проверьте, правильно ли вы всё сделали:

```
yarn --version
```



## Windows

Возможны 2 варианта:

1. Скачать [установщик](#).
2. Или установить при помощи [chocolatey](#).

```
choco install yarn
```

**Внимание!** Если вы используете антивирус - добавьте директорию **%LocalAppData%\Yarn** в исключения.

## Linux

1. Debian or Ubuntu Linux

```
curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo  
apt-key add -  
echo "deb https://dl.yarnpkg.com/debian/ stable main" |  
sudo tee /etc/apt/sources.list.d/yarn.list
```

2. Ubuntu 14.04 and Debian Stable

Сначала [обновите версию node.js](#)

```
sudo apt-get update && sudo apt-get install yarn
```

Для других Linux систем читайте [здесь](#)

## Как пользоваться

### 1. Инициализация нового проекта

```
yarn init -y
```

Создастся файл package.json

### 2. Добавить пакет (зависимость)

```
yarn add [package]  
yarn add [package]@[version]
```

После этого Yarn создаст локфайл **yarn.lock** в корне вашего проекта. В этот файл будут автоматически записываться чексуммы всех устанавливаемых вами зависимостей. Как говорят сами разработчики “You don’t need to read or understand this file — just check it into source control” (т.е. не трогайте локфайл и не пытайтесь его понять).

Не забудьте добавить его в git (git add yarn.lock).

### 3. Обновление

```
yarn upgrade [package]  
yarn upgrade [package]@[version]
```

### 4. Удаление

```
yarn remove [package]
```

## 5. Установить все зависимости проекта

```
yarn
```

или

```
yarn install
```

### Миграция с npm на Yarn

#### npm

#### Yarn

```
npm install
```

```
yarn install
```

```
npm install [package]
```

```
-
```

```
npm install --save [package]
```

```
yarn add [package]
```

```
npm install --save-dev [package]
```

```
yarn add [package] [--dev/-D]
```

```
npm install --global [package]
```

```
yarn global add [package]
```

```
npm uninstall [package]
```

```
-
```

```
npm uninstall --save [package]
```

```
yarn remove [package]
```

```
npm uninstall --save-dev [package]
```

```
yarn remove [package]
```

```
npm cache clean
```

```
yarn cache clean
```

```
rm -rf node_modules && npm install
```

```
yarn upgrade
```



## Offline установка пакетов

Любая зависимость, установленная хотя бы 1 раз попадает в кэш Yarn. Затем эту же зависимость можно установить для другого проекта, не скачивая её ещё раз из сети интернет. Т.е. в режиме “оффлайн”.

Рассмотрим пример:

1. Создадим структуру проекта

```
mkdir yarn  
cd yarn
```

2. Создадим package.json и yarn.lock файлы

```
yarn
```

3. Установим jquery

```
yarn add jquery
```

4. Установим browser-sync

```
yarn add browser-sync -D
```

5. В результате мы получили package.json файл

```
cat package.json  
{  
  "dependencies": {  
    "jquery": "^3.1.1"  
  },  
  "devDependencies": {  
    "browser-sync": "^2.18.7"  }  
}
```

6. Теперь выйдем из проекта и создадим новый

```
cd ..  
mkdir yarn-offline  
cd yarn-offline
```

7. Выключим интернет

8. Инициализация

```
yarn
```

9. Попробуем установить jquery

```
yarn add jquery  
  
// warning You don't appear to have an internet  
connection. Try the --offline flag to use the cache for  
registry queries.
```

10. Следуем рекомендации

```
yarn add jquery --offline  
  
// error Couldn't find any versions for "jquery" that  
matches "latest" in our cache. Possible versions: "3.1.1"
```

12. Видим, что yarn не знает точно, какая версия jquery последняя. Предполагает, что версия 3.1.1. Нас это вполне устраивает.

```
yarn add jquery@3.1.1 --offline
```

Получилось!

13. Убедимся, что всё установлено корректно.

```
cat package.json  
ls node_modules/
```

14. Аналогичным образом самостоятельно установите browser-sync.

