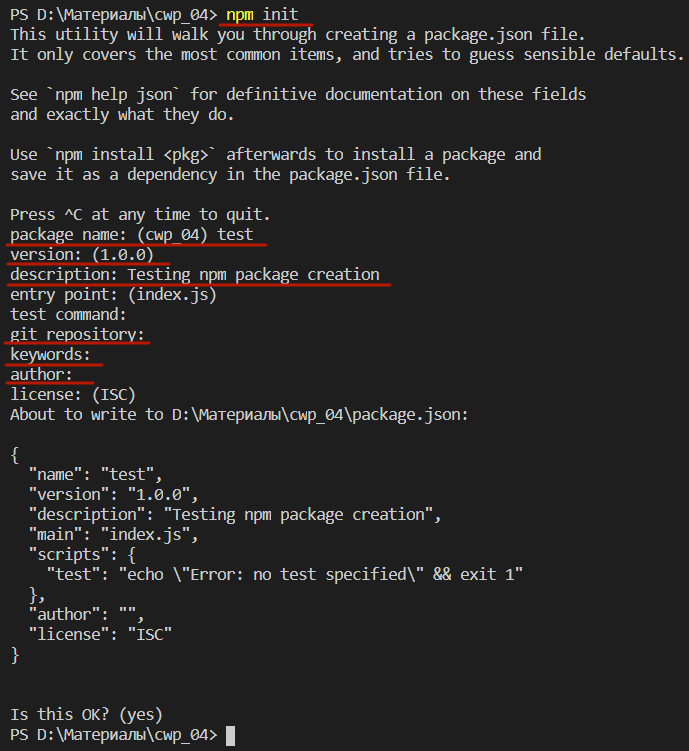
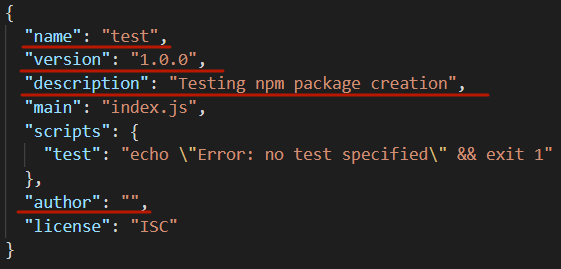
Лекция 05а

ПСКП, ПОИТ-3

**ФАЙЛ PACKAGE.JSON**

1. **package.json**: файл конфигурации приложения Node.js. Любая директория, содержащая данный файл, интерпретируется как Node.js-пакет.
2. **package.json**: содержит метаданные проекта (название, версия, описание проекта, …), список зависимостей вашего пакета, которые будут установлены при вызове команды **npm install**, скрипты, вызывающие другие команды консоли.
3. **package.json**: создание вручную или команда **npm init**. Секции name и version – минимальный набор для публикации.

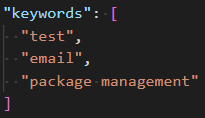




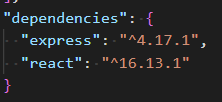
1. **package.json**: **name** – имя пакета. Должно быть короче 214 символов, состоять только из букв в нижнем регистре, цифр, символов подчеркивания (\_) и дефисов (-), без использования пробелов.
2. **package.json**: **version** – текущая версия пакета. Должна соответствовать правилам **семантического версионирования (X.Y.Z)**.
3. **package.json**: **description** – краткое описание пакета. Если планируется публикация пакета, данное свойство поможет пользователям сайта npm понять назначение пакета.
4. **package.json**: **main** – точка входа в проект.
5. **package.json**: **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”).
6. **package.json**: **author** – информация об авторе пакета (строка/json-объект). Может содержать имя автора, email и адрес сайта.



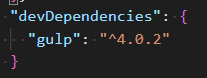
1. **package.json**: **license** – информация о лицензии пакета.
2. **package.json**: **keywords** –массив ключевых слов, относящихся к функционалу проекта. Правильно подобранные ключевые слова могут помочь людям быстрее найти нужный пакет при просмотре сайта npm.



1. **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.



1. **package.json**: **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.



1. **package.json**: при удалении пакета с использованием команды **npm uninstall** **<package\_name>** запись о нем также будет автоматически удалена из списка зависимостей.

**СЕМАНТИЧЕСКОЕ ВЕРСИОНИРОВАНИЕ**

1. **Semver**: семантическое версионирование – спецификация, которая описывает правила присвоения версии релизам программного обеспечения. Сайт - <https://semver.org>.
2. **Semver**: 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.
3. **Semver**: номер версии задается в формате **MAJOR.MINOR.PATCH** или **X.Y.Z**, где X, Y, Z – целые, неотрицательные числа.
4. **Semver**: нельзя изменять уже опубликованные версии. Любые изменения в проекте должны сопровождаться новой версией.
5. **Semver**: **MAJOR** – изменяется при внесении обратно-несовместимых изменений. Начальная разработка – значение 0 (все может измениться в дальнейшем). Может включать изменения, характерные для MINOR и PATCH. Когда увеличивается, значения MINOR и PATCH должны быть обнулены.
6. **Semver**: **MINOR** – изменяется при добавлении нового и изменении старого функционала, не нарушая обратную совместимость. Должно быть изменено, если какой-либо функционал помечен как устаревший. Может быть увеличено при реализации нового функционала или внесении изменений в приватный код. При увеличении значения, значение PATCH должно быть обнулено.
7. **Semver**: **PATCH** – изменяется при внесении обратно-совместимых, небольших исправлений.
8. **Semver**: предрелиз может быть обозначен добавлением дефиса и серией разделённых точкой идентификаторов, следующих сразу за PATCH. Должна содержать лишь буквенно-цифровые символы и дефис. Не должна быть пустой. 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

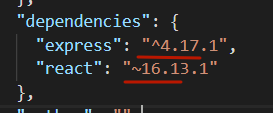
1. **Semver**: сборочные метаданные могут быть обозначены добавлением знака плюс и ряда разделённых точкой идентификаторов, следующих сразу за PATCH или предрелизной версией. Должны содержать лишь буквенно-цифровые символы и дефис. Не должны быть пустыми.

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

1. **Semver**: 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

1. **package.json:** **cпецификаторы версий** — специальные символы, которые определяют диапазон версий пакетов, подходящих для использования в нашем пакете.



1. **package.json:**

* **~:** подходят новые PATCH версии

Пример: ~0.13.0: 0.13.1 — подходят, 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
* **=:** 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 некоего пакета.

1. **package.json:** калькулятор версий - [semver.npmjs.com/](https://semver.npmjs.com/)

