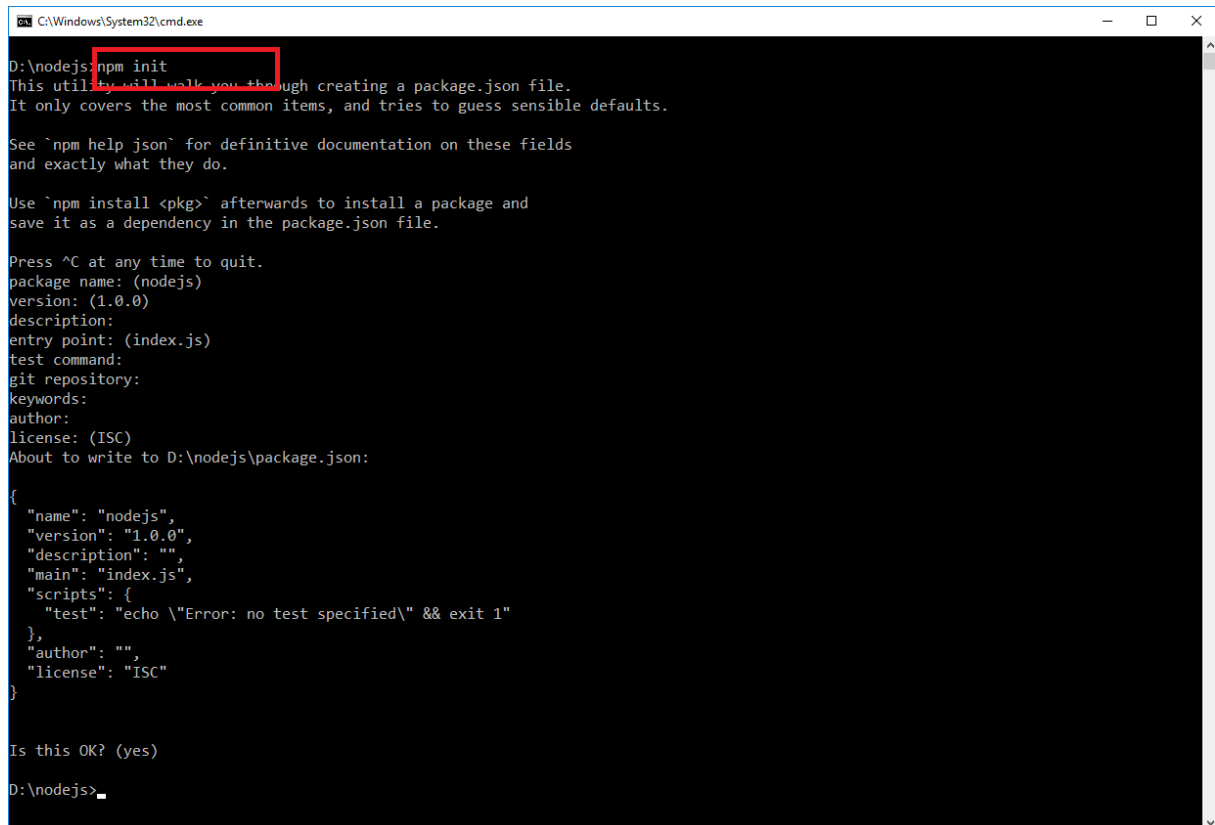


Wstęp do laboratorium 2

Aby rozpocząć przygodę z menadżerem pakietów do każdego projektu musimy stworzyć plik `package.json` poprzez wywołanie komendy: `npm init`



```
C:\Windows\System32\cmd.exe
D:\nodejs>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.

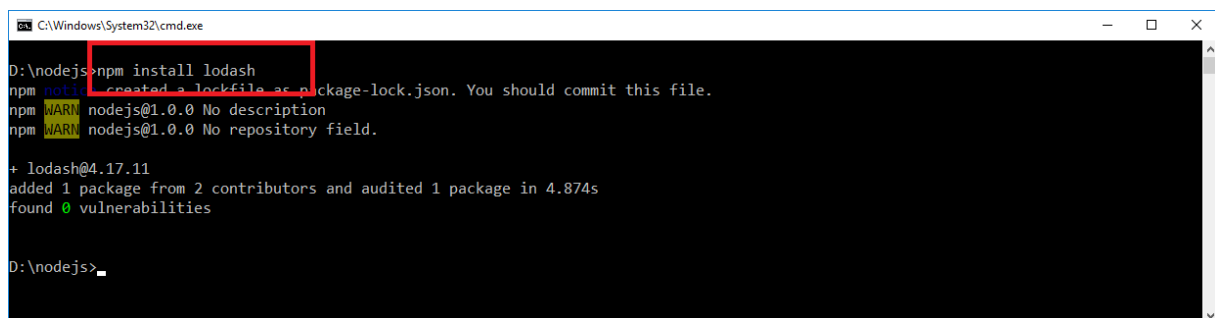
See `npm help json` for definitive documentation on these fields
and exactly what they do.

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: (nodejs)
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author:
license: (ISC)
About to write to D:\nodejs\package.json:
{
  "name": "nodejs",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}

Is this OK? (yes)
D:\nodejs>_
```

W tak przygotowanym katalogu możemy już instalować zewnętrzne pakiety, aby to uczynić musimy wpisać polecenie np: `npm install lodash`

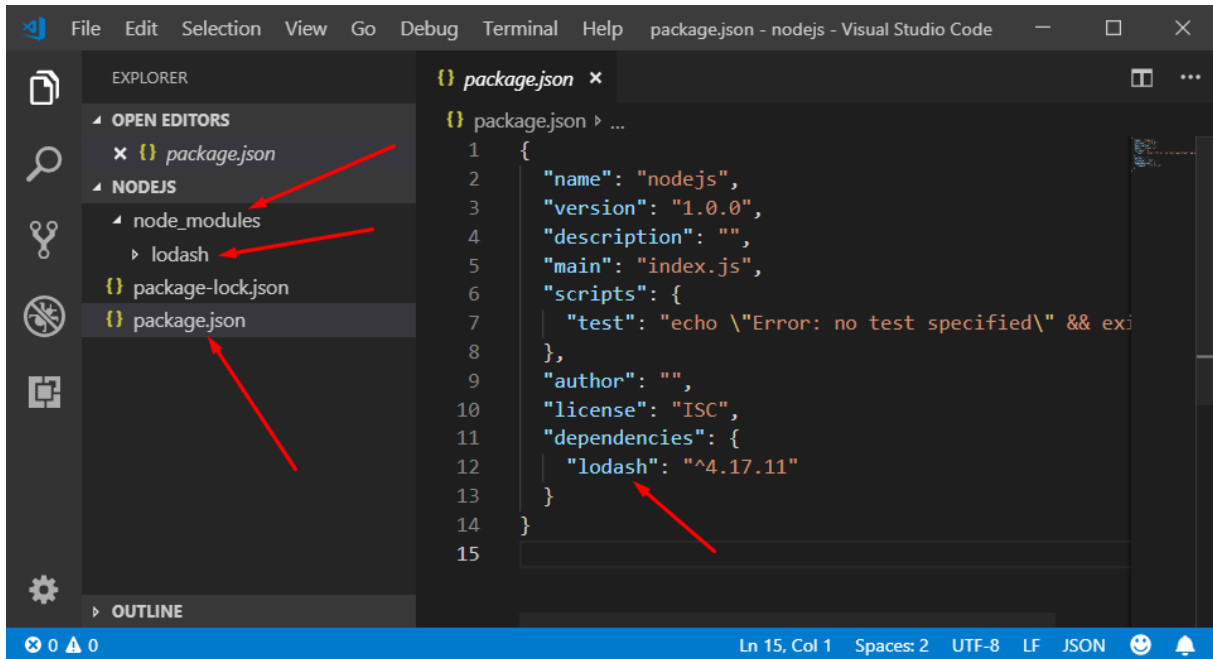


```
C:\Windows\System32\cmd.exe
D:\nodejs>npm install lodash
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN nodejs@1.0.0 No description
npm WARN nodejs@1.0.0 No repository field.

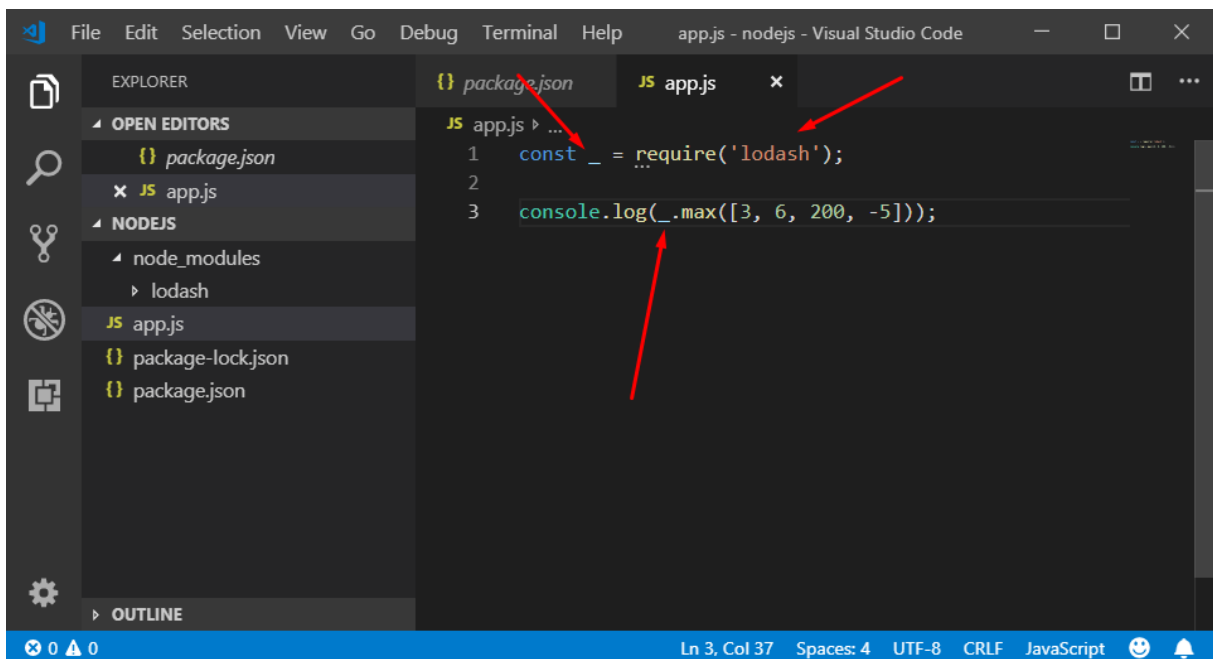
+ lodash@4.17.11
added 1 package from 2 contributors and audited 1 package in 4.874s
found 0 vulnerabilities

D:\nodejs>_
```

W tym momencie NPM pobierze dany pakiet i umieści go w naszym katalogu 'node_modules' oraz dopisze do pliku 'package.json' informację o zainstalowanym pakiecie:



Aby wykorzystać zewnętrzny pakiet w naszej aplikacji musimy go po prostu wczytać:

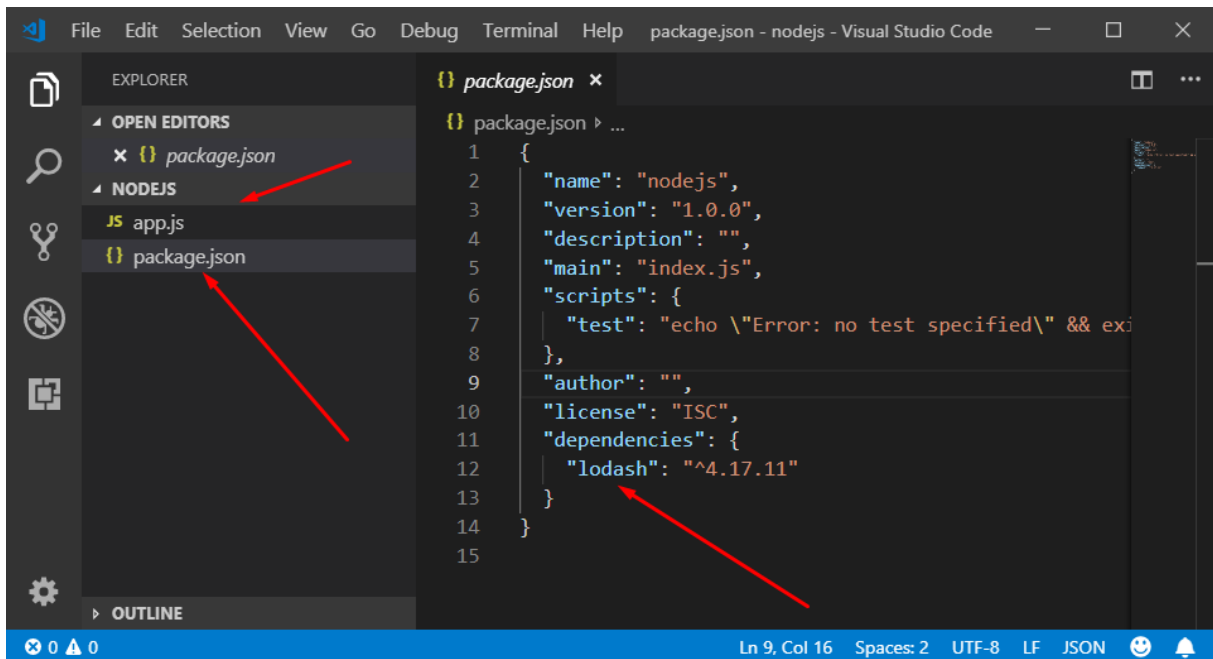


UWAGA!!!!

Jeżeli mamy już plik package.json pobrany z jakiegoś repozytorium nie tworzymy nowego pliku a jedynie instalujemy niezbędne pakiety które znajdują się w pliku package.json:

Przykład:

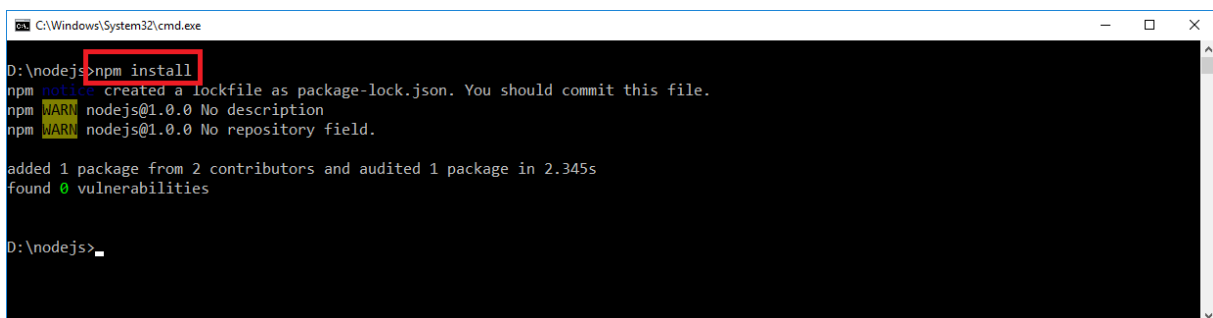
W takim przypadku jeżeli mamy plik package.json ale nie mamy pobranych zewnętrznych modułów powinniśmy uruchomić ich instalację poleceniem: `npm install`



This screenshot shows the Visual Studio Code interface. The Explorer sidebar on the left has the 'NODEJS' folder expanded, showing 'app.js' and 'package.json'. A red arrow points to 'package.json'. The main editor displays the content of 'package.json':

```
1 {
2   "name": "nodejs",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \\\"Error: no test specified\\\" && exit 1"
8   },
9   "author": "",
10  "license": "ISC",
11  "dependencies": {
12    "lodash": "^4.17.11"
13  }
14 }
```

A red arrow points to the 'lodash' dependency in the 'dependencies' object. The status bar at the bottom indicates 'Ln 9, Col 16'.



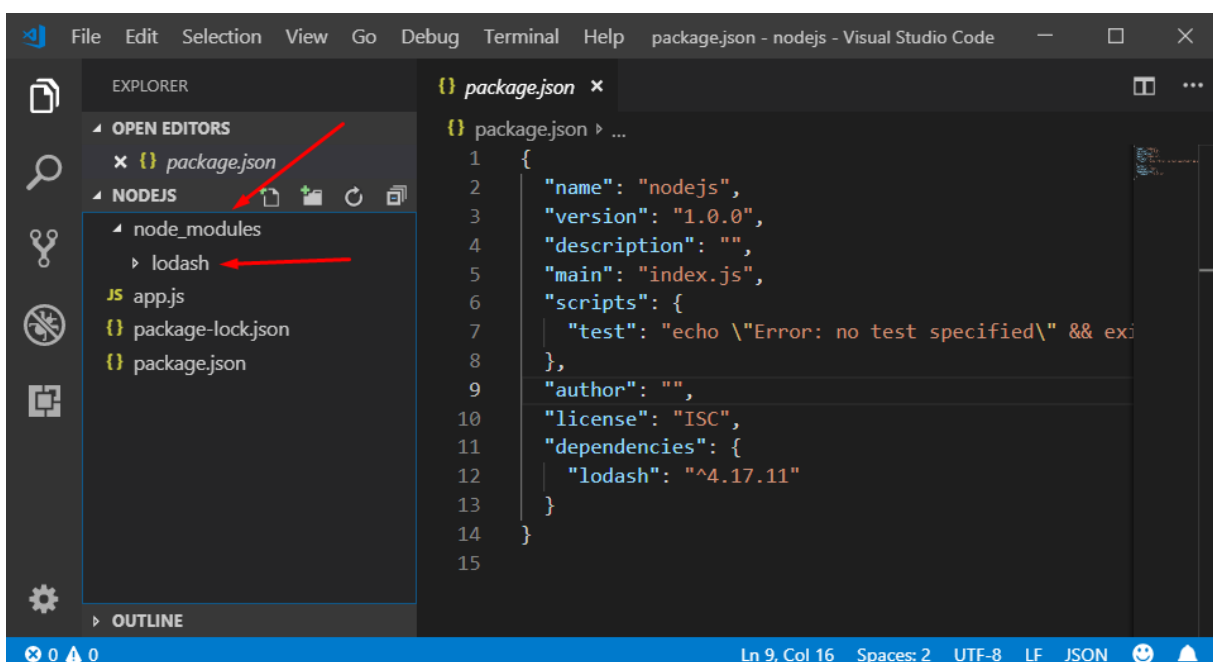
This screenshot shows a terminal window with the command `npm install` executed. The output is as follows:

```
D:\nodejs>npm install
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN nodejs@1.0.0 No description
npm WARN nodejs@1.0.0 No repository field.

added 1 package from 2 contributors and audited 1 package in 2.345s
found 0 vulnerabilities

D:\nodejs>
```

The command `npm install` is highlighted with a red box.



This screenshot shows the Visual Studio Code interface after the installation. The Explorer sidebar shows the 'node_modules' folder expanded, with 'lodash' listed as a subfolder. A red arrow points to 'lodash'. The main editor still shows the 'package.json' file with the 'lodash' dependency. The status bar at the bottom indicates 'Ln 9, Col 16'.