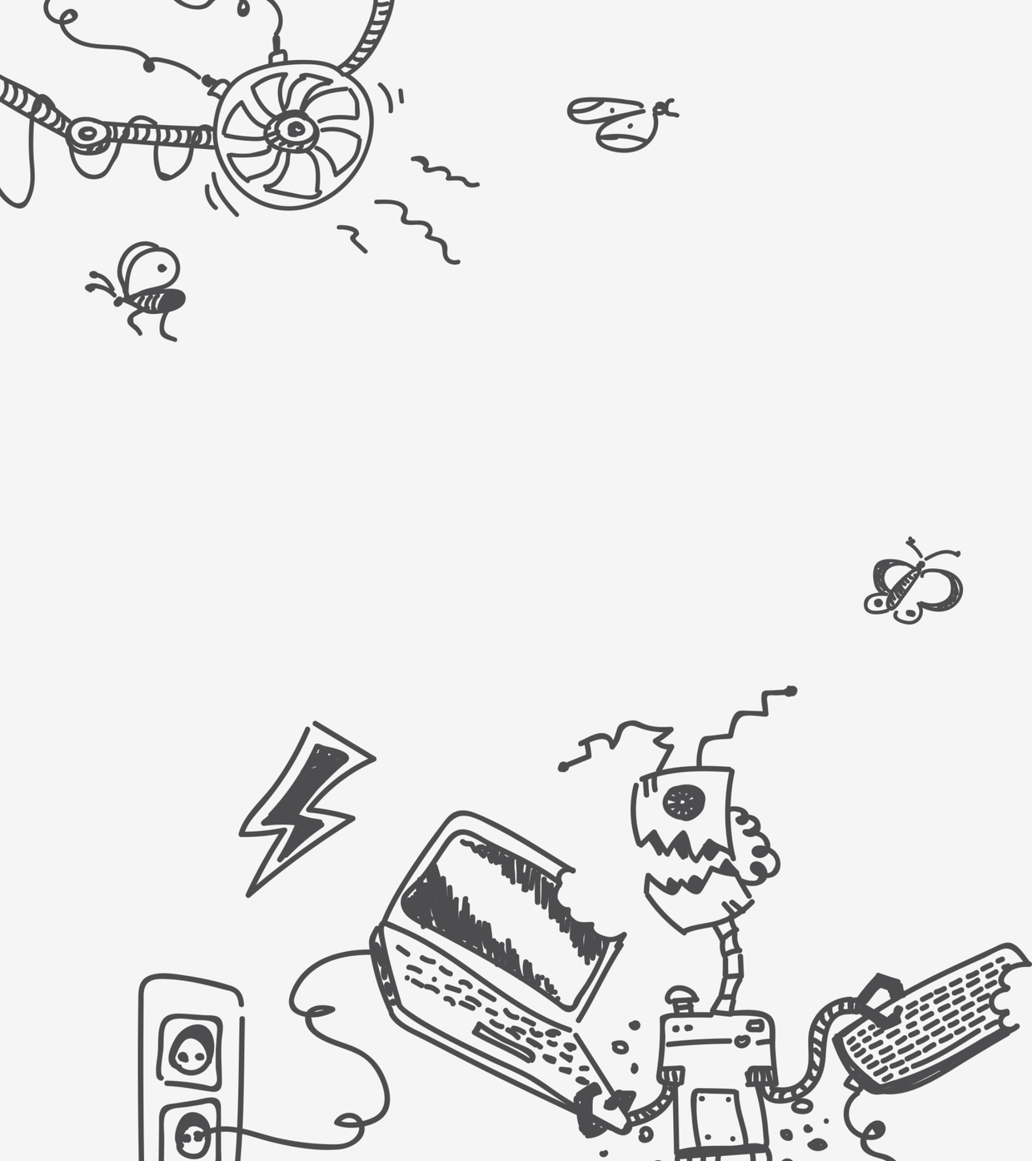


Korzystanie z zależności

Korzystanie z zainstalowanych paczek

Jak pamiętacie paczki **npm** można instalować na dwa sposoby.
Jest to silnie powiązane z tym, jak będziemy chcieli z nich korzystać:

Instalacja lokalna

Paczka ta będzie dostępna tylko i wyłącznie dla projektu, w którym została zainstalowana.

Zazwyczaj stosujemy taką instalacje dla modułów, które będziemy załączać w naszym kodzie za pomocą funkcji `require()`.

Instalacja globalna

Instalacja globalna polega na instalacji danej paczki w ścieżce systemowej. Zazwyczaj stosujemy instalację globalną, jeżeli paczka udostępnia nam narzędzie konsolowe (CLI). Dzięki instalacji globalnej będziemy mogli z npm korzystać w konsoli.

Korzystanie z globalnych paczek

Paczki globalne instalujemy po to, żeby korzystać z ich narzędzi konsolowych **(CLI – Command Line Interface)**. Są to programy napisane w JavaScript, które możemy uruchomić z linii komend.

Jednym z przykładów takiego programu jest `gulp`. Jest to narzędzie do zarządzania zadaniami podczas prac deweloperskich.

Użycie gulpa zostanie pokazane podczas kolejnego modułu.

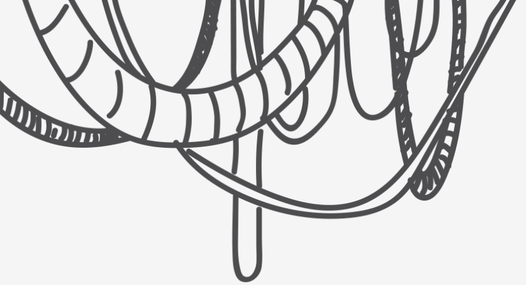
Korzystanie z lokalnych paczek

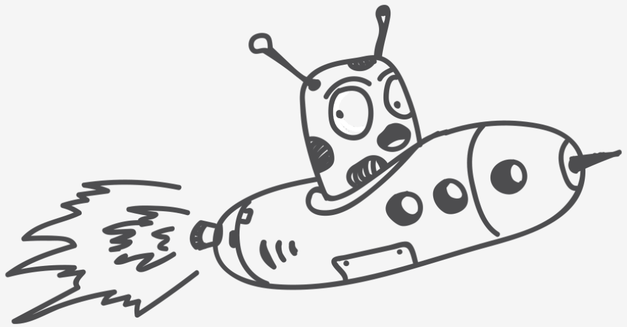
Paczki lokalne instalujemy po to, żeby korzystać z nich w naszym kodzie jak z bibliotek. Dzięki temu możemy w łatwy sposób rozszerzać nasz projekt o dodatkowe funkcje, napisane już przez kogoś innego.

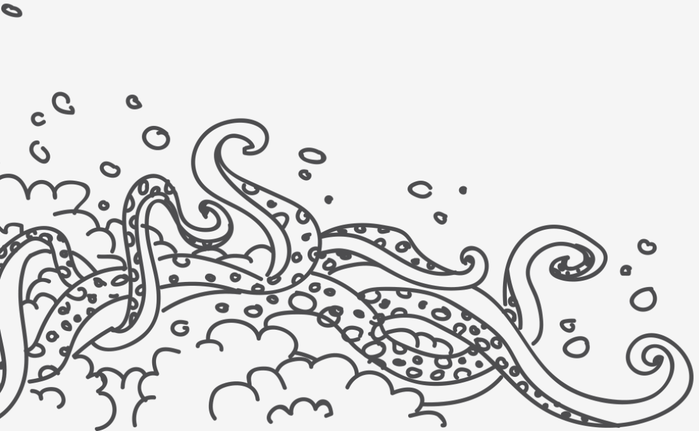
Aby dołączyć kod jakiegoś modułu, musimy użyć funkcji `require()`, która jest częścią npm i środowiska Node.js.

Node.js jest technologią serwerową (służy do pisania aplikacji serwerowych w JS zamiast w PHP, Ruby albo innych językach), funkcja `require()` nie jest dostępna z poziomu przeglądarki.

Aby skorzystać z doinstalowanych bibliotek, będziemy musieli poznać jedno z narzędzi do budowania kodu (**build tool**). Więcej o nich dowiecie się podczas wprowadzenia do Webpacka.







Przykład

Łączenie plików lokalnych

Stwórzmy dwa pliki:

- `app.js`,
- `date.js`.

W pliku `date.js` wpiszemy:

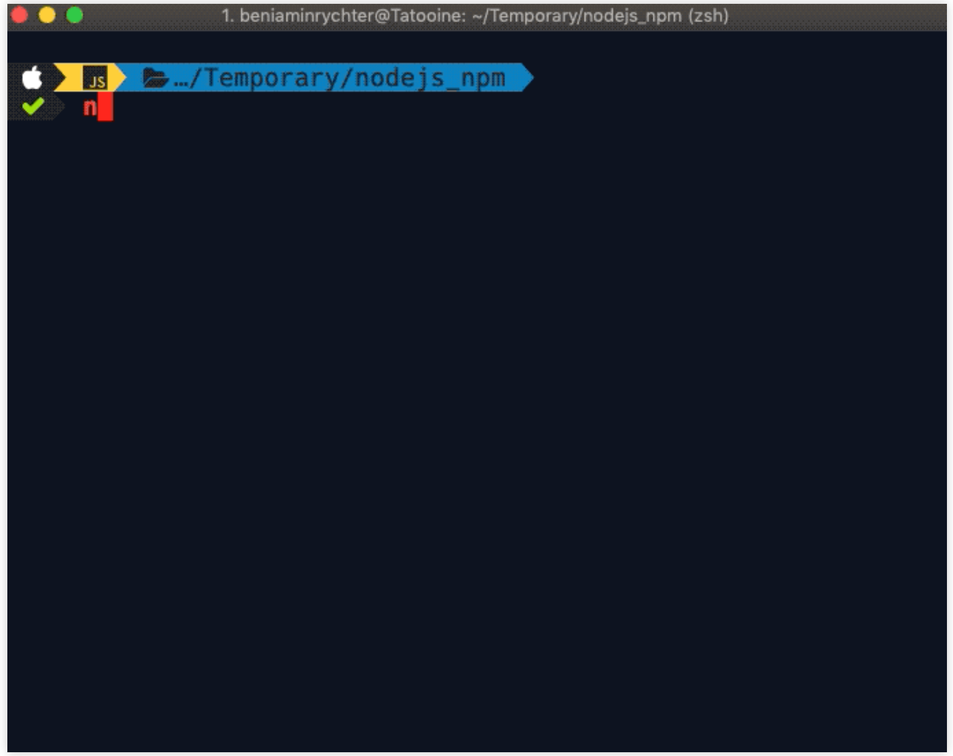
```
console.log("Aktualna data to: 10-10-2010");
```

W pliku `app.js` wpiszemy:

```
require("./date.js");
console.log("Główny plik aplikacji!");
```

Teraz uruchomimy konsolę i wpiszemy:

```
node app.js
```



Jak widzisz uruchomiliśmy plik `app.js`, do którego został załączony plik `date.js`.

Dodawanie paczek npm do projektu

Dodajmy do naszego pliku `date.js` paczkę `Moment.js` (paczka przydatna w operacjach na datach i czasie).

Użyjemy z niej funkcji `format`.

W pliku `date.js` dopiszemy następujący kod:

```
const moment = require("moment");
const currentDate = moment().format('MMMM D YYYY, H:mm:ss');
console.log("Aktualna data to: " + currentDate);
```

Dodawanie paczek npm do projektu

W konsoli wpiszemy:

```
node app.js
```

Powinien pokazać się błąd:

```
internal/modules/cjs/loader.js:583
    throw err;
    ^

Error: Cannot find module 'moment'
    at Function.Module._resolveFilename (internal/modules/cjs/loader.js:581:15)

  /.../
```

Dodawanie paczek npm do projektu

Dlaczego wystąpił błąd? To dlatego, że nie mamy zainstalowanej paczki **`moment`**.

Wpiszemy zatem w konsolę:

```
npm i moment
```

Po chwili powinniśmy dostać informację, że paczka została dodana i zainstalowana.

Następnie uruchamiamy plik **`app.js`** za pomocą **`node`**:

```
node app.js
```

Tym razem wszystko działa! Bilbioteka **`Moment.js`** została zaimportowana prawidłowo i możemy z niej skorzystać.