JAVASCRIPT (ES6)

PACKAGE MANAGER & HTTP REQUEST

OLEH: AHMAD FAISOL, ST., MT.

PACKAGE MANAGER

PACKAGE MANAGER MERUPAKAN SEBUAH TOOLS YANG DAPAT MEMBANTU PENGELOLAAN MODULE/PACKAGE PADA PROYEK MENJADI LEBIH MUDAH

DENGAN ***PACKAGE MANAGER***, KITA DAPAT DENGAN MUDAH MEMASANG DAN MENGHAPUS *DEPENDENCIES LIBRARY*, ATAU MEMASANG *MODULE/PACKAGE* YANG DIBUAT OLEH ORANG LAIN PADA PROYEK KITA. SELAIN ITU, KITA JUGA DAPAT MENULIS *MODULE/PACKAGE* YANG KITA BUAT UNTUK DIKONSUMSI SECARA PUBLIK

PACKAGE MANAGER DI BEBERAPA BAHASA PEMROGRAMAN:

- JAVA : MAVEN DAN GRADLE
- PHP : COMPOSER
- PYTHON PIP
- NPM DAN YARN

PACKAGE MANAGER (NPM)

NPM MERUPAKAN PENGELOLA PACKAGE UNTUK JAVASCRIPT YANG DAPAT MEMUDAHKAN KITA DALAM MENGELOLA PACKAGE YANG TERSEDIA PADA NPMJS.COM

NPM ADALAH STANDARD PACKAGE MANAGER YANG DISEDIAKAN OLEH NODE.JS DAN SUDAH OTOMATIS TERPASANG SAAT KITA MENGINSTALL NODE.JS

NPM DAPAT KITA MANFAATKAN UNTUK RANAH FRONT-END DEVELOPMENT SEPERTI MEMASANG BOOTSTRAP, JQUERY, ATAU JAVASCRIPT PACKAGE/FRAMEWORK LAIN YANG MEMBANTU PENGEMBANGAN FRONT-END DEVELOPMENT

NPM COMMAND

Command	Description	Common Options
init	Membuat berkas ***package.json*** pada proyek	[--force|-f|--yes|-y|--scope]
Install *<package-name>*	Memasang dan mendaftarkan package pada berkas ***package.json***	[-P|--save-prod|-D|--save-dev|-O|--save-optional] [-E|--save-exact] [-B|--save-bundle] [--no-save] [--dry-run]
run *<command>*	Menjalankan perintah yang terdapat pada objek scripts yang terdapat di berkas ***package.json***.	[--silent] [-- <args>...]
uninstall *<package-name>*	Menghapus dan mengeluarkan package dari berkas ***package.json***.	[-S|--save|-D|--save-dev|-O|--save-optional|--no-save]
version	Untuk melihat versi package yang tersedia secara global atau lokal	[<newversion> | major | minor | patch | premajor | preminor | prepatch | prerelease [--preid=<prerelease-id>] | from-git]
<command> ***-h***	Melihat helper pada sebuah perintah seperti install, run, uninstall	Misal: npm install -h

BERKAS PACKAGE.JSON

BERKAS **package.json** MERUPAKAN MERUPAKAN MANIFEST YANG MENAMPUNG SELURUH INFORMASI DARI APLIKASI YANG KITA BUAT, SEPERTI NAMA, DESKRIPSI, VERSI, *AUTHOR*, *REMOTE REPOSITORY*, DAN LAINNYA

NAMUN HAL YANG PALING PENTING ADALAH INFORMASI YANG MENAMPUNG DAFTAR DEPENDENCIES MODULES YANG DIGUNAKAN PADA APLIKASI KITA

PADA *DEPENDENCIES MODULES* KITA DAPAT MELIHAT *MODULES* APA SAJA YANG DIGUNAKAN DALAM PENGEMBANGAN APLIKASI. CONTOHNYA JIKA KITA MEMASANG MODULE/LIBRARY JQUERY, MAKA KITA DAPAT MELIHAT JQUERY BESERTA VERSI YANG DIGUNAKAN TERDAFTAR PADA OBJEK DEPENDENCIES DI DALAM BERKAS **package.json**

UNTUK MEMASANG PACKAGE YANG TERSEDIA PADA NPM KITA GUNAKAN PERINTAH install KEMUDIAN NAMA PACKAGE-NYA

DEPENDENCIES VS DEVDEPENDENCIES

TERDAPAT DUA TIPE OBJEK PACKAGE DEPENDENCIES DALAM BERKAS **package.json**

YAITU OBJEK dependencies DAN OBJEK devDepedencies

OBJEK dependencies MERUPAKAN OBJEK YANG MENAMPUNG PACKAGE YANG KITA GUNAKAN UNTUK MEMBUAT APLIKASI. BIASANYA SEBUAH FRAMEWORK / LIBRARY SEPERTI REACT, ANGULAR, VUE, JQUERY ATAU FRAMEWORK LAINNYA. UNTUK MEMASANG PACKAGE PADA DEPENDENCIES KITA CUKUP GUNAKAN PERINTAH:

```
npm install <package-name>
```

SEDANGKAN OBJEK devDependecies DIGUNAKAN UNTUK MENDAFTARKAN PACKAGE YANG DIGUNAKAN HANYA SELAMA PENGEMBANGAN SAJA. CONTOHNYA PACKAGE YANG BERFUNGSI SEBAGAI WEB SERVER LOKAL SEPERTI http-server, ATAU *PACKAGE* YANG BERFUNGSI UNTUK MEMBUNDEL JAVASCRIPT SEPERTI *WEBPACK*.

UNTUK MEMASANG PACKAGE SEBAGAI devDependencies KITA GUNAKAN PERINTAH:

```
npm install <package-name> --save-dev
```

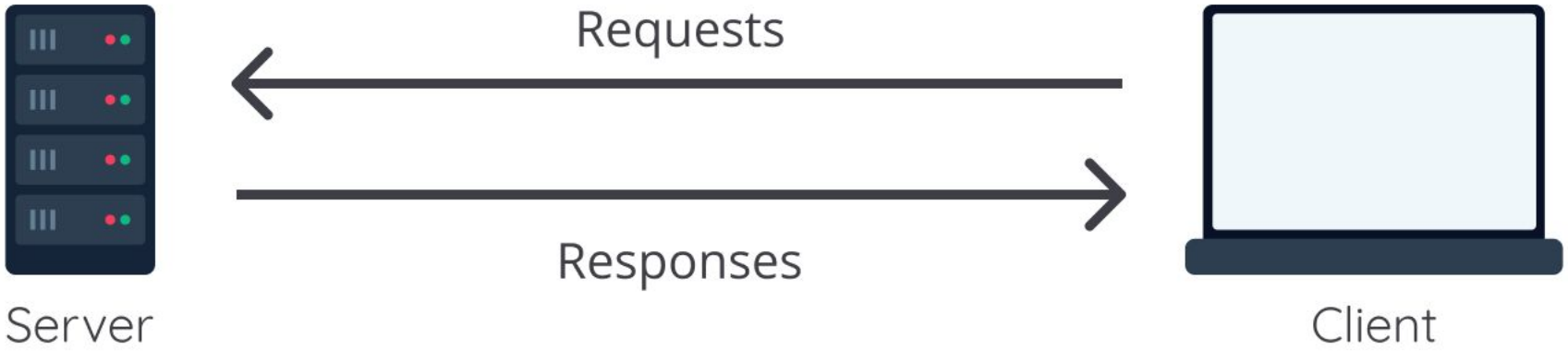
CAKUPAN PACKAGE

DALAM PEMASANGAN PACKAGE, TERDAPAT DUA CAKUPAN (SCOPE), YAITU LOCAL DAN GLOBAL

LOCAL BERARTI MENYIMPAN PACKAGE HANYA DI LINGKUP PROYEK, SEDANGKAN GLOBAL DIGUNAKAN PADA CAKUPAN GLOBAL ATAU DAPAT DIAKSES PADA PROYEK MANAPUN.

UNTUK MEMASANG PACKAGE SECARA GLOBAL DAPAT MENAMBAHKAN FLAG **-g** SETELAH PENULISAN **npm install <nama_package>**

HTTP REQUEST



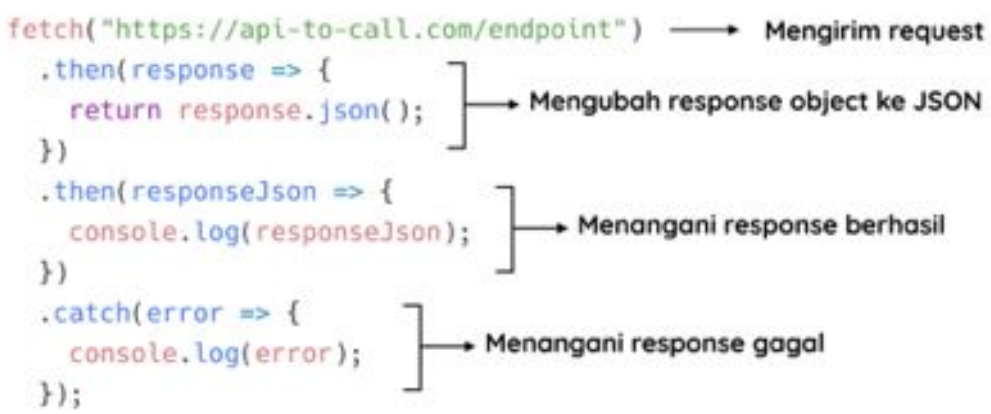
FETCH

FETCH MERUPAKAN CARA BARU DALAM MEMBUAT NETWORK REQUEST

DENGAN *FETCH* KITA DAPAT MEMBUAT *HTTP REQUEST* LEBIH MUDAH TANPA HARUS MEMBUAT *INSTANCE* DAN MEMANGGIL BANYAK METHOD

FETCH MEMANFAATKAN PROMISE SEHINGGA KITA DAPAT MENGURANGI PENERAPAN CALLBACK, DAN DAPAT DITULISKAN DENGAN GAYA *SYNCHRONOUS* MENGGUNAKAN *ASYNC* DAN *AWAIT*

DASAR PENGGUNAAN FETCH



JIKA REQUEST BERHASIL DIPROSES OLEH SERVER, FUNGSI *fetch*() AKAN MENGEMBALIKAN *PROMISE RESOLVE* DAN MEMBAWA *RESPONSE* OBJECT DI DALAMNYA

NAMUN NILAI *RESPONSE* YANG DIBAWA *RESOLVE* BELUM SEBAGAI DATA JSON YANG KITA BUTUHKAN, MELAINKAN INFORMASI MENGENAI *RESPONSE* ITU SENDIRI, SEPERTI *STATUS CODE*, *TARGET URL*, *HEADERS*, DSB

MAKA DARI ITU, UNTUK MENDAPATKAN DATA JSON YANG DIBUTUHKAN, KITA PERLU MENGUBAH RESPONSE OBJECT KE DALAM BENTUK JSON DENGAN MEMANGGIL METHOD *json*()

FETCH OPTIONS

FUNGSI *fetch()* DAPAT MENERIMA DUA BUAH PARAMETER

SELAIN MENETAPKAN TARGET URL, KITA JUGA DAPAT MEMBERIKAN *OPTIONS* UNTUK MENETAPKAN *METHOD*, *HEADER*, *BODY*, DSB PADA REQUEST YANG AKAN DIJALANKAN

PENERAPAN OPTIONS INI BERSIFAT PILIHAN ALIAS TIDAK WAJIB

FETCH METHOD

NILAI DARI PROPERTI *METHOD* DITULISKAN DALAM BENTUK STRING, CONTOHNYA *“POST”*, *“PUT”* ,“DELETE” , DSB. NILAI *DEFAULT* DARI PROPERTI INI ADALAH “GET”, SEHINGGA JIKA KITA MEMBUAT GET REQUEST, KITA TIDAK PERLU MENETAPKAN NILAI METHOD SECARA EKSPLISIT.

```
fetch('https://api-to-call.com/endpoint', {
    method: 'POST'
})
```

FETCH HEADER

UNTUK MENAMBAHKAN REQUEST HEADER DENGAN FETCH KITA GUNAKAN PROPERTI *HEADERS* PADA *OPTIONS*. CONTOHNYA, UNTUK MENAMBAHKAN PROPERTI Content-Type DENGAN NILAI application/json ATAU BISA JUGA UNTUK PROPERTI API TOKEN

```
fetch('https://api-to-call.com/endpoint', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
        'X-Auth-Token': 'my-4p1-t0k3n'
    }
})
```

DATA PADA BODY REQUEST

UNTUK MENGIRIMKAN DATA PADA BODY REQUEST KITA GUNAKAN PROPERTI BODY PADA OPTIONS, DAN HARUS DIRUBAH MENJADI JSON

```
fetch('https://api-to-call.com/endpoint', {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json',
        'X-Auth-Token': 'my-4p1-t0k3n'
    },
    body: JSON.stringify({
        id: 10,
        title: 'title example',
        description: 'Lorem ipsum dolor, sit amet consectetur adipisicing elit...'
    })
})
```