Todo(s) App

Learning Competencies

- Mampu membuat class yang terdefinisi dengan baik dan memiliki satu tujuan saja
- Mampu menerapkan hubungan antar class dan object yang fleksibel dan rapi
- Identifikasi dan implementasi class berdasarkan kebutuhan di dunia nyata
- Menggunakan design pattern MVC untuk memodelkan masalah

Summary

Tantangan berikut adalah awal dari pembuatan aplikasi TODO list via command-line. Secara general, aplikasi TODO ini memiliki fitur sebagai berikut:

Berikut command-command yang ada pada Todo(s) App:

```
$ node todo.js # menampilkan command apa saja yang tersedia
$ node todo.js help # menampilkan command apa saja yang tersedia
$ node todo.js list # Melihat daftar TODO
$ node todo.js add <task_content> # Menambahkan TODO ke dalam list
$ node todo.js findById <task_id> # Melihat detail TODO sesuai `task_id` nya
$ node todo.js delete <task_id> # Menghapus TODO sesuai `task_id` nya
$ node todo.js complete <task_id> # Menandai status TODO selesai
$ node todo.js uncomplete <task_id> # Menandai status TODO belum selesai
```

`task_id` di sini berfungsi sebagai ID unik untuk membedakan tiap TODO.

Karena kita akan mengambil informasi input melalui command line seperti itu, maka kita memerlukan process.argv untuk mengevaluasi input dari argument.

Data kamu (seperti yang dicontohkan pada repo) akan sangat sederhana. Hanya berisi deskripsi tugas perbaris. Nantinya akan ada tambahan informasi lainnya, oleh karena itu kita menggunakan format array of object dalam file JSON. ***Contoh***:

```
[
  {"task": "Bikin aplikasi"},
  {"task":                                   "Main                                   Node.js"}
]
```

Release 0: Implement the `help` command

Buat code untuk menampilkan semua command yang dapat dijalankan. Ketika menjalankan perintah :

```
$ node todo.js

//atau

$ node todo.js help
```

Maka akan menampilkan:

```
$ node todo.js
$ node todo.js help
$ node todo.js list
$ node todo.js add <task_content>
$ node todo.js findById <task_id>
$ node todo.js delete <task_id>
$ node todo.js complete <task_id>
$ node todo.js uncomplete <task_id>
```

Release 1: CRUD

1. **Implement the `list` command**

   Buatlah code untuk menampilkan list todos sesuai dengan isi file data.json, sehingga ketika kita menjalankan perintah ini:

   ```
   $ node todo.js list
   ```

   Aplikasi akan mengeluarkan output daftar tugas/todo yang telah didaftarkan. Contohnya:

   ```
   // hasilnya (ada nomor ID dan list todo nya) :
   1. Bikin Aplikasi
   2. Main node js
   ```

   Kita juga harus membuat "command" sederhana dengan menambahkan kondisi tertentu. Sebagai contoh, bagaimana aplikasi kita tahu user ingin menambahkan tugas ke dalam daftar TODO?

2. **Implement the `add` command**

Requirements:

- User dapat menambahkan tugas ke dalam TODO list-nya
- Isi file data.json terupdate! Pastikan todo baru telah tersimpan pada file json

```
$ node todo.js add "Shut down my computer"

//outputnya :
Added "Shut down my computer" to your TODO list...
```

3. Implement the `findById` command

Requirements:

- User dapat melihat tugas dalam TODO list-nya berdasarkan ID nya

```
$ node todo.js findById 2

// outputnya :
2. Main node js
```

4. Implement the `delete` command

Requirements:

- User dapat menghapus tugas dari daftar TODO mereka
- Isi file data.json terupdate ! Pastikan tugas baru di delete sudah hilang dari file

```
$ node todo.js list

//outputnya:
1. Bikin aplikasi
2. Main node js
3. Shut down my computer

$ node todo.js delete 3
//outputnya:
Deleted "Shut down my computer" from your TODO list...

$ node todo.js list
//outputnya:
1. Bikin aplikasi
2. Main node js
```

5. Implement `complete` and `uncomplete` command

Requirements:

- User dapat menandai tugas apabila sudah selesai
- Todo yang sudah selesai akan terlihat berbeda dengan tugas yang belum selesai

Note: Untuk mengimplementasikan fitur ini kamu butuh mengubah struktur di file `data.json` dan code yang melakukan parsing file terkait. Pastikan ketika kamu menjalankan add, delete, dll file json kamu ikut ter-update.

Berikut salah satu cara menampilkan data yang sudah complete & uncomplete agar lebih mudah dibaca:

```
$ node todo.js list

//outputnya:
1. [ ] Bake a delicious peanut butter cake
2. [x] Shut down my computer
3. [ ] Change the world


$ node todo.js complete 3

//outputnya:
1. [ ] Bake a delicious peanut butter cake
2. [x] Shut down my computer
3. [x] Change the world


$ node todo.js uncomplete 2

//outputnya:
1. [ ] Bake a delicious peanut butter cake
2. [ ] Shut down my computer
3. [x] Change the world
```

Contoh kurung siku dengan tanda `x` atau kosong, mengindikasikan tugas sudah selesai atau belum.
Faktor-faktor apa saja yang harus kita perhatikan sehingga kita dapat menyimpan data ini secara tepat? Update aplikasi kamu untuk mengakomodasi fitur-fitur ini.

Release 2: Make It More Sophisticated

Untuk melanjutkan aplikasi TODO, kita menambahkan beberapa spesifikasi fitur baru:

1. Melihat daftar TODO sesuai tanggal dibuat dengan `list:created`
2. Melihat daftar TODO sesuai daftar yang sudah selesai dengan `list:completed`
3. Menambahkan beberapa tag pada TODO dengan `tag <id> <name_1> <name_2>`
4. Mencari atau melakukan filter TODO yang memiliki tag tertentu dengan `filter:<tag_name>`

Maka dari itu kita akan tambahkan tiga perintah baru ke dalam aplikasi TODO kita.

```
$ node todo.js list:created asc|desc
$ node todo.js list:completed asc|desc
$ node todo.js tag <task_id> <tag_name_1> <tag_name_2> ... <tag_name_N>
$ node todo.js filter:<tag_name>
```

Berikut user stories-nya:

1. Sebagai user, saya ingin melihat daftar tugas yang disortir berdasarkan tanggal pembuatan atau `created_date`, secara ascending maupun descending.
2. Sebagai user, saya ingin melihat daftar tugas yang sudah saya selesaikan disortir berdasarkan tanggal selesainya atau `completed_date`, secara ascending maupun descending.
3. Sebagai user saya ingin menandai tugas dengan beberapa tag. Misalnya: home, work, fun, dll.
4. Sebagai user, saya ingin melihat daftar tugas berdasarkan tag dan disortir berdasarkan tanggal pembuatan atau `created_date`.

Setiap fitur sepertinya butuh perubahan file format dan cara mem-parsing data akan berbeda.

1. **Implement the `list:created` command**

   Implementasi perintah/command sehingga dapat berjalan seperti di bawah:

   ```
   $ node todo.js list:created
   ```

   Perintah di atas menghasilkan daftar tugas dan disortir berdasarkan tanggal pembuatan.

   User Experience Alert: Ada yang bilang sesuatu yang default itu penting. Menurut kalian, sortir yang default apakah akan menampilkan tugas terbaru atau tugas yang pertama diinput? Dan kenapa alasannya? Bayangkan jika kamu telah punya daftar yang banyak dan panjang.

   Setelah kamu sudah punya sortir default, namun user tetap harus dapat memilih jenis sortir apa yang diinginkan, sehingga implementasi perintah dapat berjalan dengan beberapa cara seperti ini :

```
$ node todo.js list:outstanding
$ node todo.js list:outstanding asc
$ node todo.js list:outstanding desc
```

2. **Implement the `list:completed` command**

   Implementasi perintah seperti berikut:

```
$ node todo.js list:completed
```

   Perintah di atas akan menampilkan daftar yang sudah selesai dikerjakan dan di sort berdasarkan tanggal selesainya. Pertanyaan **user experience** yang sama juga bisa diimplementasikan di sini, yang tentang default sortirnya. Implementasinya pun harus sama dengan sortir, memiliki default namun user tetap dapat memilih jenis sortirnya juga.

3. **Implement the `tag` command**

   Implementasi perintah seperti berikut:

```
$ node todo.js list
1. Eat some cookies
2. Play with cat

$ node todo.js tag 2 hobby pet
Tagged task "Play with cat" with tags: hobby pet
```

   Setiap tugas dapat memiliki tag lebih dari satu, jadi kita harus mengganti format file kita untuk mengakomodasi hal ini dan tag suatu task tidak boleh ada yang duplicate.

   Cek kembali apakah tag yang ditambahkan ke tugas sudah ter-update juga datanya di file json !

4. **Implement the `filter` command**

   Sekarang mari kita implementasi perintah `filter` seperti berikut:

```
$ node todo.js filter:hobby

//output
2. Play with cat [hobby, pet]
```

Perintah di atas akan menampilkan semua tugas yang terdapat tag hobby dan disortir berdasarkan tanggal pembuatan. Jika terdapat kesulitan, sesuaikan dengan aturan yang kamu bisa.

Release 3: Optimize Your Learning

Ingat dan perhatikan prinsip single responsibility serta separation of concerns.

Saat bekerja, perhatikan bagaimana perubahan struktur berdampak pada aplikasi kita secara keseluruhan. Ketika fitur baru ditambahkan, berapa banyak baris kode yang harus kita ubah? Seberapa sulitnya melakukan perubahan itu? Bagaimana tingkat coupling dan cohesion-nya? Ingat kembali prinsip orthogonality dan law of demeter.