**List**

List sesuai namanya dapat menampung banyak data ke dalam satu objek. Dalam kehidupan sehari-hari kita menggunakan list untuk menyimpan daftar belanja, nomor telepon, dsb. Begitu pula dengan Dart kita bisa menyimpan bermacam-macam tipe data seperti *string*, *number*, dan *boolean*. Cara penulisannya pun sangat mudah. Perhatikan saja contoh berikut:

1. List<int> numberList = [1, 2, 3, 4, 5];

Kode di atas adalah contoh dari satu objek List yang berisi kumpulan data dengan tipe *integer*. Karena kompiler bisa mengetahui tipe data yang ada dalam sebuah objek List, maka tak perlu kita menuliskannya secara eksplisit.

1. var numberList = [1, 2, 3, 4, 5];
2. var stringList = ['Hello', 'Dicoding', 'Dart'];

Sesuai contoh di atas, kita mendefinisikan tipe data yang bisa dimasukkan ke dalam List di dalam tanda kurung sudut (<>). Sama seperti variabel, jika kita tidak mendefinisikan nilai secara eksplisit ke dalam List, maka List akan menyimpan tipe *dynamic* atau bisa menyimpan semua tipe data.

1. List dynamicList = [1, 'Dicoding', true]; // List<dynamic>

Ketika bermain dengan sebuah List, tentunya ada saat di mana kita ingin mengakses posisi tertentu dari List tersebut. Untuk melakukannya, kita bisa menggunakan fungsi *indexing* seperti berikut:

1. print(dynamicList[1]);

Perhatikan kode di atas. Fungsi indexing ditandai dengan tanda [ ]. Jika Anda mengira bahwa konsol akan menampilkan angka 1, maka tebakan Anda kurang tepat. Karena dalam sebuah List, indeks dimulai dari 0. Maka ketika kita akan mengakses data pada dynamicList yang berada pada indeks ke-1, artinya data tersebut merupakan data pada posisi ke-2. Jadi data yang akan ditampilkan pada konsol adalah ***Dicoding***.

Lalu apa yang akan terjadi jika kita berusaha menampilkan item dari List yang berada di luar dari ukuran List tersebut? Sebagai contoh, Anda ingin mengakses indeks ke-3 dari dynamicList:

1. print(dynamicList[3]);

Hasilnya adalah eror! Kompiler akan memberitahukan bahwa perintah itu tidak bisa dijalankan. Berikut pesan eror yang akan muncul:

Unhandled exception:  
RangeError (index): Invalid value: Not in range 0..2, inclusive: 3

Pesan di atas memberitahu kita bahwa List telah diakses dengan indeks ilegal. Ini akan terjadi jika indeks yang kita inginkan negatif atau lebih besar dari atau sama dengan ukuran List tersebut.

Masih ingat *looping*? Untuk menampilkan seluruh item dari *list* kita bisa memanfaatkan *looping*. Contohnya perhatikan kode berikut:

1. for(int i = 0; i < stringList.length; i++) {
2. print(stringList[i]);
3. }

Pada kode di atas kita memanfaatkan perulangan sebanyak jumlah data di dalam *list* untuk mencetak data yang ada di dalam *list*. Banyaknya data di dalam *list* bisa kita panggil melalui properti .length.

Selain itu kita juga bisa menggunakan fungsi foreach untuk menampilkan data di dalam *list*.

1. stringList.forEach((s) => print(s));

Mekanisme di atas dikenal sebagai ***lambda*** atau ***anonymous function***. Kita akan mempelajarinya lebih dalam pada modul yang akan datang.

Sejauh ini kita baru belajar menginisialisasikan dan mengakses data dari sebuah List. Pastinya Anda bertanya, “Bagaimana kita memanipulasi data pada List tersebut?” Nah, untuk menambahkan data ke dalam *list*, kita bisa menggunakan fungsi add().

1. stringList.add('Flutter');

Fungsi *add* ini akan menambahkan data di akhir *list*. Sehingga ketika dicetak, konsol akan menampilkan data berikut:

1. [Hello, Dicoding, Dart, Flutter]

Lalu bagaimana jika kita ingin menambahkan data namun tidak di akhir List? Jawabannya adalah dengan fungsi ***insert***. Di dalam fungsi *insert* kita perlu memasukkan 2 parameter, yaitu indeks *list* dan data yang akan dimasukkan.

1. stringList.insert(0, 'Programming');
2. // stringList = [Programming, Hello, Dicoding, Dart, Flutter]

Untuk mengubah nilai di dalam *list*, kita bisa langsung menginisialisasikan nilai baru sesuai indeks yang diinginkan.

1. stringList[1] = 'Application';

Sedangkan untuk menghapus data terdapat beberapa fungsi ***remove*** yang bisa kita gunakan, antara lain:

1. stringList.remove('Programming'); // Menghapus list dengan nilai Programming
2. stringList.removeAt(1);           // Menghapus list pada index ke-1
3. stringList.removeLast();          // Menghapus data list terakhir
4. stringList.removeRange(0, 2);     // Menghapus list mulai index ke-0 sampai ke-1 (indeks 2 masih dipertahankan)

**Spread Operator**

Dart memiliki fitur menarik untuk menambahkan banyak nilai ke dalam List dengan cara yang singkat, yaitu spread operator.

Sesuai namanya “spread”, fitur ini digunakan untuk menyebarkan nilai di dalam collections menjadi beberapa elemen. Spread operator dituliskan dengan tiga titik (...). Mari kita lihat contoh kode berikut:

1. var favorites = ['Seafood', 'Salad', 'Nugget', 'Soup'];
2. var others = ['Cake', 'Pie', 'Donut'];
3. var allFavorites = [favorites, others];
4. print(allFavorites);
6. /\* output
7. [[Seafood, Salad, Nugget, Soup], [Cake, Pie, Donut]]
8. \*/

Nilai List tidak akan tergabung, Alih-alih menggabungkan nilainya, variabel allFavorites menjadi List yang menampung dua List di dalamnya. Nah, dengan spread operator kita dapat menggabungkan kedua List di atas:

1. var favorites = ['Seafood', 'Salad', 'Nugget', 'Soup'];
2. var others = ['Cake', 'Pie', 'Donut'];
3. var allFavorites = [...favorites, ...others];
4. print(allFavorites);
6. /\* output
7. [Seafood, Salad, Nugget, Soup, Cake, Pie, Donut]
8. \*/

Yup, dengan spread operator kita dapat memasukkan masing-masing elemen di dalam List ke dalam List lainnya.

Untuk mengatasi List yang bisa bernilai null, kita dapat menggunakan null-aware spread operator (...?) seperti berikut:

1. var list;
2. var list2 = [0, ...?list];
3. print(list2);
5. /\* output
6. [0]
7. \*/