**Constructor**

Ketika suatu objek dibuat, semua properti pada kelas tersebut harus memiliki nilai. Kita dapat langsung menginisialisasi pada properti tertentu atau menginisialisasinya melalui ***constructor***. *Constructor* adalah fungsi spesial dari sebuah kelas yang digunakan untuk membuat objek.

Sesuai namanya, *constructor* digunakan untuk mengonstruksi objek baru.

Jadi kenapa *constructor* disebut sebagai fungsi yang spesial? Apa bedanya dengan fungsi lain pada *class*? Beberapa perbedaan antara *constructor* dan fungsi biasa adalah:

1. *Constructor* memiliki nama yang sama dengan nama kelas.
2. *Constructor* tidak memiliki nilai kembalian (*return type*).
3. *Constructor* akan secara otomatis dipanggil ketika sebuah objek dibuat.
4. Jika kita tidak mendefinisikan *constructor*, *default constructor* tanpa argumen akan dibuat.

Secara *default* sebuah kelas memiliki *constructor* yang tidak menerima argumen.

1. var dicodingCat = Animal();

Karena kita tidak memasukkan nilai ketika membuat objek, maka nilai *default* dari properti atau variabel akan digunakan. Anda perlu berhati-hati jika tidak memberikan nilai pada properti, karena akan membuat properti bernilai *null* sehingga bisa menyebabkan eror.

Untuk memberikan nilai pada properti, silakan akses properti yang ada di dalam sebuah kelas.

1. var dicodingCat = Animal();
2. dicodingCat.name = 'Gray';
3. dicodingCat.age = 2;
4. dicodingCat.weight = 4.2;

Dengan membuat *constructor*, kita tidak hanya bisa menginisialisasikan nilai namun juga menjalankan instruksi tertentu ketika objek dibuat.

1. Animal(String name, int age, double weight) {
2. this.name = name;
3. this.age = age;
4. this.weight = weight;
5. // other instructions
6. }

Keyword this di atas menunjuk pada objek yang ada di kelas tersebut. *Keyword* *this* ini umumnya digunakan untuk menghindari ambiguitas antara atribut dari *class* dan parameter yang memiliki nama yang sama.

Jika *constructor* hanya digunakan untuk menginisialisasi nilai properti, maka kode konstruktor dapat diringkas menjadi seperti berikut:

1. Animal(this.name, this.age, this.weight);

**Named Constructor**

Pada beberapa kasus kita mungkin akan membutuhkan beberapa *constructor* untuk skenario yang berbeda-beda. Pada situasi ini kita bisa memanfaatkan ***named constructor***.

Dengan menggunakan *named constructor*, kita dapat membuat beberapa *constructor* pada kelas yang sama. Setiap *constructor* akan memiliki nama yang unik.

1. class\_name.constructor\_name (arguments){
2. // Statements
3. }

Contoh pada *class* Animal adalah seperti berikut:

1. Class Animal {
2. ...
3. Animal.Name(this.\_name);
4. Animal.Age(this.\_age);
5. Animal.Weight(this.\_weight);
6. ...
7. }