MATERI

Hari-7-Object Oriented 1

Secara singkat class dapat diartikan blueprint dari suatu objek, maksudnya adalah kita mendesain suatu objek berdasarkan class yang kita buat, sebagai contoh dalam dunia nyata, jika kita ingin membuat sebuah objek meja maka kita perlu menggambar rancangan meja tersebut, dengan demikian kita akan mulai membuat meja berdasarkan rancangan tersebut, jika rancangannya baik maka hasilnya baik, begitu juga sebaliknya, jadi class dapat kita analogikan sebagai rancangannya dan hasil dari rancangan tersebut adalah objeknya.

Mendefenisikan Class

Untuk membuat kelas kita gunakan keyword class yang telah disediakan oleh bahasa Dart. Perhatikan contoh berikut:

```
class Dadu {

}
```

Pertama kita mulai dengan keyword class, diikuti dengan nama kelasnya yaitu Dadu, kemudian tubuh kelas diawali dengan buka kurung kurawal "{" dan diakhiri dengan tutup kurung kurawal } Jika sebuah kelas telah dibuat, maka kita langsung dapat menciptakan objek dari kelas tersebut, tentu jika kelas kita seperti contoh kelas Dadu, maka objek tersebut tidak dapat melakukan apapun, karena kita belum menambahkan atribut ataupun metode kedalamnya. untuk menciptakan objek pada pemrograman dart kita gunakan keyword new. Perhatikan contoh program berikut ini:

```

Dadu dd = new Dadu();
```

Diawali dengan nama kelas yaitu Dadu kemudian nama objeknya yaitu dd (kita bisa menentukan nama objek yang lain tidak harus dd) kemudian new diakhiri dengan nama kelas.

```
nama_kelas nama_objek = new nama_kelas();
```

contoh functional

```
void main(){
  double panjang, lebar;

  panjang = 10;
  lebar = 5;

  var luasPersegiPanjang = panjang * lebar;
  print(luasPersegiPanjang);
}
```

akan di ubah ke dalam class dibawah ini

pada main.dart

```
  PersegiPanjang kotak; //inisialisasi nama object nya
  double luasKotak; //inisialisasi tipe data

  kotak = new PersegiPanjang(); //inisialisasi/ buat alias nama
  kotak.panjang = 2; //menambahkan value panjang
  kotak.lebar = 3; //menambahkan value lebar
  luasKotak = kotak.hitungLuas(); //membuat inisialisasi hitung luas
diambil dari object kotak yang di panggil dari class persegi panjang
  print(luasKotak); // mencetak luas kotak
}

class PersegiPanjang {
  double panjang; // inisialisai tipe data panjang
  double lebar; // initialisasi tipe data lebar

  double hitungLuas() {
    return this.panjang * lebar; // menghitung luas
  }
}
```

2. Enkapsulasi (Pembungkusan)

Enkapsulasi (***encapsulation***) adalah *sebuah metode untuk mengatur struktur class dengan cara menyembunyikan alur kerja dari class tersebut*.

Struktur class yang dimaksud adalah **property** dan **method**. Dengan *enkapsulasi*, kita bisa membuat pembatasan akses kepada *property* dan *method*, sehingga hanya *property* dan *method* tertentu saja yang bisa diakses dari luar class. *Enkapsulasi* juga dikenal dengan istilah '***information hiding'***. Dengan *enkapsulasi*, kita bisa memilih *property* dan *method* apa saja yang boleh diakses, dan mana yang tidak boleh diakses. Dengan menghalangi kode program lain untuk mengubah *property* tertentu, *class* menjadi lebih terintegrasi, dan menghindari kesalahan ketika seseorang '*mencoba*' mengubahnya. Programmer yang merancang *class* bisa menyediakan *property* dan *method* khusus yang memang ditujukan untuk diakses dari luar.Pada pemrograman dart tidak ada keyword untuk membuat attribut atau metode menjadi private atau pun public, untuk membuat attribut ataupun fungsi menjadi private, cukup tambahkan underscore "_" sebelum nama attribut ataupun metode.

Getter dan setter

main.dart

```
void main(List<String> args) {
  PersegiPanjang kotak; // inisialisasi persegi panjang
  double luasKotak; // inisialisasi tipe data luas kotak

  kotak = new PersegiPanjang(); //menginisialisasi atau mengaliskan kotak sebagai persegi panjang/ pointer menunjuk object persegipanjang
  kotak.setPanjang(4.0); // set nilai panjang(pembeda dengan method getter dan setter adalah adanya setPanjang)
  kotak.setLebar(6.0); //set nilai lebar (pembeda dengan method getter dan setter adalah adanya setLebar)

  luasKotak = kotak.hitungLuas(); // allias luaskotak
  print(luasKotak); // mencetak luas kotak

}
```

`pada persegi_panjang.dart`

```
class PersegiPanjang {
  double _panjang; //initialisasi tipe data panjang
  double _lebar; //initialisasi tipe data lebar
  void setPanjang(double value){
     if(value < 0){ // validasi jika nilai di bawah 0
       value *= -1; // akan di kalikan -1, misal -1 * -2 hasilnya 2
     }
     _panjang= value; //alias
  }
  double getPanjang(){ //get panjang
    return _panjang; // mengembalikan nilai get panjang
  }
  void setLebar (double value){
    if(value < 0 ){ // validasi jika nilai di bawah 0
      value *= -1; // akan di kalikan -1, misal -1 * -2 hasilnya 2
    }
    _lebar = value; //alias
  }
  double getLebar(){ //get panjang
    return _lebar; // mengembalikan nilai get panjang
  }
  double hitungLuas(){
      return this. _panjang * _lebar; //mereturn hasil
  }
}
```

.

Method Getter dan Setter

main.dart

```
void main(List<String> args) {
  PersegiPanjang2 persegi; //inisialisasi
  double luasPersegi; // inisialisasi tipe data
  persegi = new PersegiPanjang2(); //memanggil object persegipanjang2
  persegi.panjang=3.0; //set nilai
  persegi.lebar=-2.0; //set nilai

  luasPersegi = persegi.luas; //alias
  print(luasPersegi); //cetak

}
```

pada persegi_panjang2.dart

```
class PersegiPanjang2 {
  double _panjang; //inisialisasi tipe data
  double _lebar; // inisialisasi tipe data lebar
  void set lebar(double value){
    if(value < 0){ //validasi
      value *= -1; //return -1
    }
    _lebar = value; //alias
  }
  double get lebar{ //getter lebar
    return _lebar; //mengembalikan lebar
  }
  void set panjang(double value){
    if(value < 0){ //validasi
      value *= -1; // return -1
    }
    _panjang = value; //alias
  }
  double get panjang{
    return _panjang; //return panjang
  }
  double get luas => _panjang * _lebar; //hitung luas
}
```

dari contoh diatas ada yang menggunakan getter, setter dan juga method getter, setter kedua nya bisa di gunakan bersamaan dan juga dapat di gunakan/ dipilih salah satu karena itu meupakan optional dan juga beda cara penulisan saja.

Rating - Feedback

Berikan Rating pada posting ini:

★★★★★ 0

Berikan kritik dan saran..

Submit