1. Kompetensi

Setelah kegiatan perkuliahan selesai, mahasiswa diharapkan dapat:

1. Memahami konstruktor   
2. Menerapkan access modifier   
3. Menggunakan atribut/method class   
4. Melakukan instansiasi atribut/method   
5. Memahami notasi UML

2. Pendahuluan

Pemrograman berorientasi objek memodelkan dunia nyata kedalam dunia pemrograman. Dengan demikian sebuah program dapat dikorelasikan atau dipahami dengan mudah seperti halnya di dunia nyata. Enkapsulasi adalah salah satu konsep penting dalam pemodelan pemrograman berorientasi objek ini.

Enkapsulasi adalah mekanisme untuk membungkus atribut dan method menjadi satu unit/kesatuan. Didalam enkapsulasi, atribut yang ada didalam class disembunyikan dari class lain. Atribut tersebut dapat diakses dari class lain melalui method yang disediakan. Enkapsulasi disebut juga dengan data hiding.

Method yang digunakan untuk mengakses data dari variabel / method yang telah dienkapsulasi biasa disebut dengan **method getter**. Sedangkan method yang digunakan untuk memasukkan atau mengubah nilai dari variabel yang telah dienkapsulasi biasa disebut dengan **method setter**.

2.1 Menganalisa Objek di Dunia Nyata

Seperti namanya, enkapsulasi cara kerja sama seperti dengan kapsul obat. Bagian luar dari kapsul atau cangkangnya berfungsi untuk menyembunyikan dan membuat agar isi dari obat tersebut tetap rapi dan terhindar dari penambahan zat-zat yang tidak diperlukan. Begitu juga dengan enkapsulasi, fungsinya adalah menyembunyikan isi data agar tidak bisa dimodifikasi di luar class kecuali dengan method tertentu yang diizinkan oleh class tempat data itu berada.

Mari kita amati behaviour dari objek di dunia nyata. Sebagai contoh, sebuah mobil. Bagaimana perilaku dari sebuah mobil? Nyalakan mesin. Tambah kecepatan. Tambah kecepatan lagi. Kurangi kecepatan. Berhenti. Mundur pelan. Berhenti. Behaviour tersebut dapat dilaksanakan melalui beberapa mekanisme. Kunci kontak untuk menyalakan dan mematikan mesin. Tongkat persneling, pedal gas dan pedal rem untuk menambah atau mengurangi kecepatan. Dalam hal ini, kita menggunakan pedal gas dan pedal rem untuk mengatur kecepatan mobil tersebut.

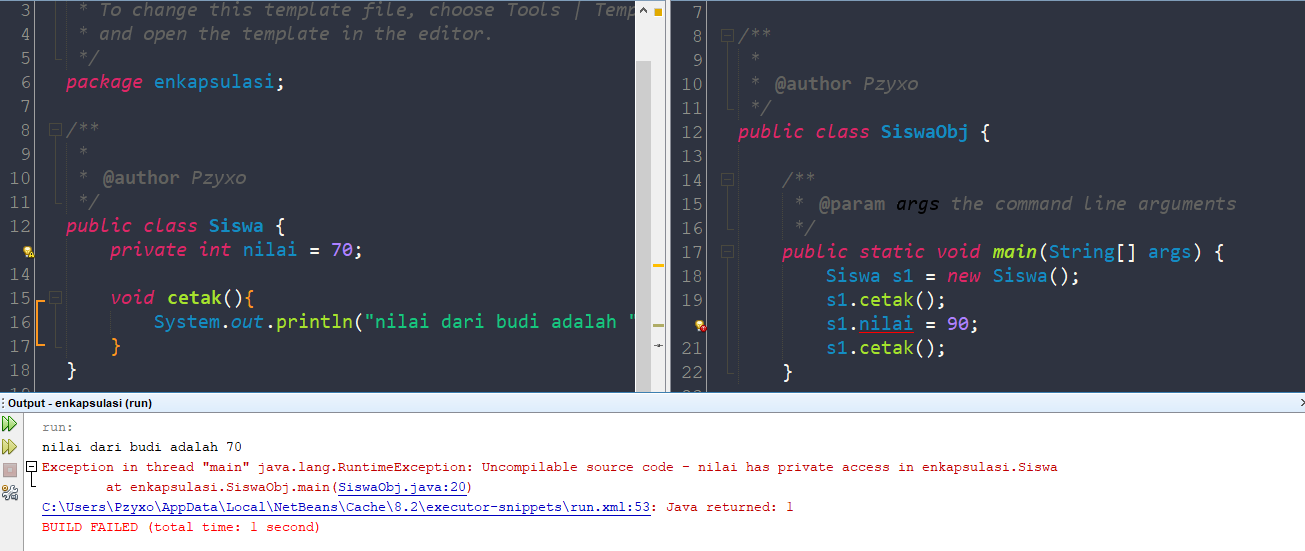
2.2 Implementasi

Implementasi dari enkapsulasi java ini adalah dengan menambahkan access modifier private pada attribut yang akan disembunyikan informasinya. Dan juga minimarket access modifier public pada method yang berkaitan dengan attribut (method yang digunakan untuk mengeset value atau set method dan method yang digunakan untuk mendapatkan value atau get method).

Secara default, ketika kita mendeklarasikan variabel pada suatu class dengan access modifier public, variabel tersebut akan dapat dimodifikasi nilainya di class yang memanggil class tersebut. Sebagai contoh, terdapat variabel int usia pada class Siswa. Pada class SiswaObj, dipanggillah class Siswa tersebut. Setelah dipanggil / diinisiasi objeknya, maka pada class SiswaObj dapat merubah nilai dari variabel int usia tadi secara manual. Hal ini terlihat tidak ada masalah, akan tetapi dapat menjadi masalah ketika nilai atau value yang dimasukkan tidak sesuai dengan persyaratan. Untuk itulah, variabel int usia diberi access modifier private agar tidak dapat diubah secara manual di dalam class SiswaObj.



Pada contoh gambar di atas, terlihat bahwa variabel nilai memiliki access modifier public dan nilainya dapat diubah manual di class lain.

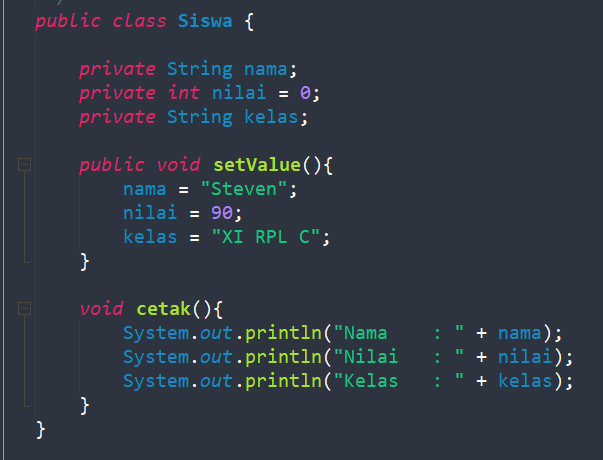


Ketika access modifier dari variabel nilai diubah, maka variabel tersebut tidak bisa diakses lagi di class lain kecuali melalui method yang ada diclass tempat variabel nilai itu berada.

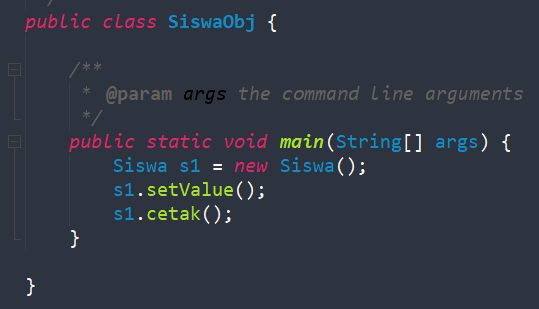
2.3 Percobaan Modifier Private

Untuk lebih jelasnya dalam penggunaannya di program Java, cobalah untuk membuat script sesuai dengan petunjuk berikut!

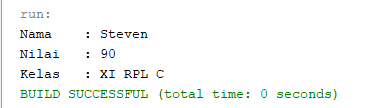
Buatlah program seperti pada contoh di bawah!



Buatlah class Siswa dengan kode program seperti di atas!



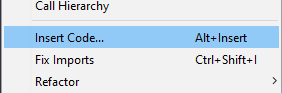
Lalu pada package yang sama, buatlah class SiswaObj dengan isi kode seperti di atas!



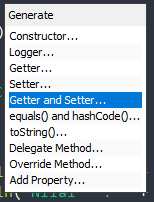
Hasil Output akan terlihat seperti di atas.

2.4 Percobaan Method Getter dan Setter

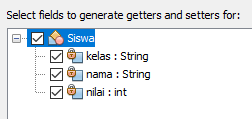
Masih menggunakan kode program yang sama seperti di atas, lalu pada class Siswa, klik kanan pada kode program dan pilih insert code



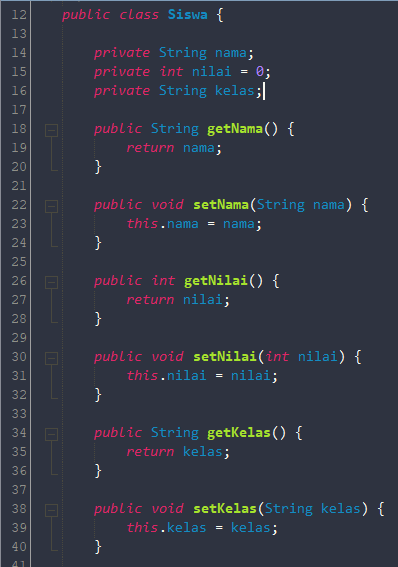
Lalu pilihlah Getter and Setter



Lalu centang semua variabel yang akan diberikan method getter dan setter, kemudian klik generate



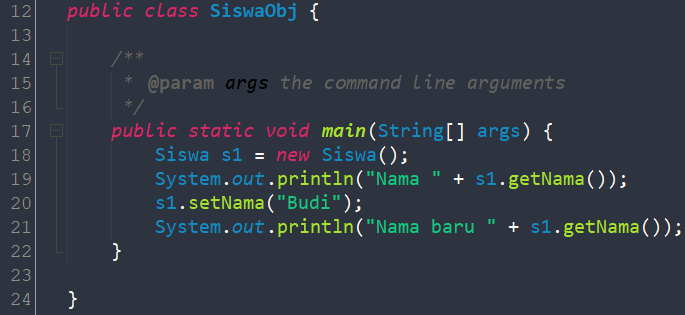
Lalu akan muncul sejumlah baris kode program seperti di bawah



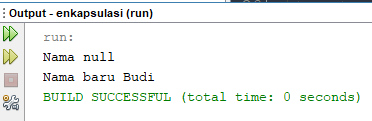
Method getter ketika dibuat melalui generate code di netbeans, akan berawalan dengan get. Sedangkan untuk setter akan berawalan dengan set.

Lalu bagaimana cara mengaksesnya melalui class lain?

Kita bisa mengaksesnya sama halnya ketika kita mengakses method biasa.



Lalu ketika dijalankan akan menjadi seperti berikut



Pada baris pertama output terdapat null karena kita belum memberikan nilai pada variabel nama. Setelah kita berikan nilai menggunakan setter pada baris 20, maka output juga akan menyesuaikan dengan nilai yang kita berikan.