**BAB VI**

**UDT SETTER GETTER**

* 1. Tujuan

1. Praktikan mampu memahami penggunaan UDT, *setter* dan *getter* di masing masing bahasa pemrograman C#, JAVA, dan PHP.
2. Praktikan mampu membedakan UDT, *setter,* dan *getter.*
3. Praktikan mampu membuat membuat UDT, *setter,* dan *getter* di masing masing bahasa pemrograman C#, Java dan PHP.
4. Praktikan mampu mengerti dan menghafal dari *source code* UDT, *setter* dan *getter* di masing masing bahasa pemrograman C#, Java dan PHP.
5. Praktikan mampu membuat program sederhana yang menggunakan UDT *setter* dan *getter* pada masing masing bahasa pemrograman C#, Java dan PHP.
   1. Dasar Teori
      1. **Enkapsulasi**

Enkapsulasi (*encapsulation*) adalah metode untuk mengatur struktur pada *class*. Dengan tujuan untuk menentukan hak akses pada properti atau *method.*

Enkapsulasi adalah salah satu yang paling penting di pemrograman berorientasi objek (OOP). Karena tahap enkapsulasi lah yang menentukan pemberian hak akses pada setiap properti atau *method*. Enkapsulasi atau *encapsulation* pada OOP terdiri dari 3 yaitu:

1. ***Modifier public***  
   *Public* artinya kelas, *method*, maupun variabel yang memiliki *modifier* ini bisa diakses oleh semua *class*, baik yang ada di *package* itu maupun yang bukan. *Modifier* ini juga memberikan akses kepada objek yang diinstans (dibentuk) di kelas tersebut.
2. ***Modifier private***  
   *Private* artinya kelas, *method*, ataupun variabel yang memiliki *modifier* ini cuma bisa diakses oleh kelas itu sendiri. Kelas lain maupun objek yang dibentuk oleh kelas lain sama sekali tidak punya akses *method*, kelas, maupun variabel dengan *modifier private*. Nah, sifat tertutup total dari *private* ini tidak memandang apakah si kelas lain yang ingin memanggil *method*, kelas, atau variabel ini dari *package* yang sama atau bukan, tetep tidak diberikan akses. Karena sifatnya inilah, maka *method* atau variabel *private* tidak bisa diwariskan ke subkelas. Diantara *method* dan variabel yang paling sering dideklarasikan sebagai *private* adalah variabel.

Karena:   
1. Biasanya masing-masing kelas itu punya variabel sendiri-sendiri jadi tidak perlu memanggil *variable* dari kelas lain.  
2. Melindungi supaya variabel tertentu dari kemungkinan berubah nilainya jika diakses oleh *method* dari kelas yang berbeda.

1. ***Modifier protected***  
   Sama halnya seperti *private, protected* juga membatasi akses terhadap *method*/variabel/kelas, namun tidak bersifat total. *Modifier* ini biasanya digunakan untuk membatasi akses terhadap:
2. Subkelas dari suatu kelas.
3. Kelas lain dari *package* yang sama.

*(sumber: https://www.malasngoding.com/php-oop-part-4-pengertian-enkapsulasi-public-private-protected/)*

* + 1. **UDT (*User Data Type* )**
    2. **Setter**

*Setter* adalah sebuah *method* yang digunakan untuk memberikan nilai pada suatu *attribute*, *object, list, entity*, dllnya. *Method setter* tidak memiliki pengembalian nilai, cirinya *method* ini berisi kata *void* pada awal penulisan *method.*

*Method setter* kegunaannya sebagai pemberi nilai pada *attribute* atau *object* tertentu, kalian akan melihat banyak model dan penamaan *method* ini. Kita akan tahu *method* ini apakah *setter* atau bukan tergantung dari isi *method* yang bersangkutan.

Contoh dan penerapan dari *method setter.*

|  |
| --- |
| public class GetterSetter {  private String name;    public void setName(String name) {  this.name = name;  }  } |

Pada kode diatas, variabel boleh ditulis secara global tidak harus didalam *method*, tergantung kebutuhan. Disamping itu *method setter* memiliki parameter dalam konstruktornya, sebagai acuan *value* yang boleh di masukan. Kebetulan dalam contoh diatas, menggunakan *Type Data String* sebagai acuannya.

*(sumber :* [*https://putuguna.com/2017/07/memahami-fungsi-method-getter-dan-setter-pada-java.html*](https://putuguna.com/2017/07/memahami-fungsi-method-getter-dan-setter-pada-java.html)*)*

* + 1. **Getter**

*Method getter* adalah kebalikan dari *setter* yaitu pengambilan nilai dari suatu *object* atau *attribute* yang sudah berisi nilai. Ada dua macam tipe *method* ini, yaitu:

1. Menggunakan pengembalian nilai

Maksud dari pengembalian nilai ini adalah *method* yang bersangkatan menjadikan dirinya sebagai nilai dari sebuah *object/type* *data*. Untuk menjadikan dirinya sebagai nilai, maka *type method* ini tidak boleh *void* tapi harus ber-*type* data tertentu (bisa *object, string, int, boolean*, dll).

Contoh:

|  |
| --- |
| package education;    public class GetterSetter {  private String name;    public String getName() {  return name;  }  } |

Pada *method* diatas, *method getter* ini tipe *String*, dia akan mengambil nilai dari suatu *attribute/object* untuk dijadikan nilai dirinya sendiri. Pada kasus ini, *method* getName() mengambil nilai dari *attribute global name* sebagai nilai *method* itu sendiri. Maka otomatis, ketika di*printout*, *method* getName() akan berisikan nilai di *attribute name*.

1. Tanpa pengembalian nilai

Jika tanpa pengembalian nilai, berarti datanya diolah didalam *method*, bukan *method* tersebut yang bernilai tertentu, melainkan didalam *method* tersebut ada nilai tertentu.

Contoh:

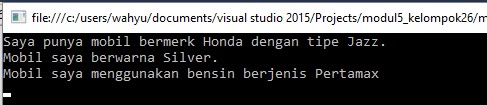
|  |
| --- |
| package education;    public class GetterSetter {  private final String name = "Anton";    public void getName(){  System.out.println("Nama : " + name);  }  } |

Jika dipanggil *method* getName() maka dia akan menampilkan *value* dari *attribute name*, yaitu “Anton”.

*(sumber :* [*https://putuguna.com/2017/07/memahami-fungsi-method-getter-dan-setter-pada-java.html*](https://putuguna.com/2017/07/memahami-fungsi-method-getter-dan-setter-pada-java.html)*)*

* 1. Hasil Percobaan dan Analisis
     1. **Bahasa Pemrograman C#**

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace modul5\_kelompok26  {  class Program  {  class Mobil //kelas objek atau setter getter  {  private string Merk; //deklarasi atribut  private string Tipe;  private string Warna;  private string jenisBBM;  public void setMerk(string val) //method set atribut  {  this.Merk = val; //menentukan nilai atribut  }  public string getMerk() //method get atribut  {  return Merk; //mendapatkan nilai atribut dan mengembalikan ke fungsi yang memanggilnya  }  public void setTipe(string val)  {  this.Tipe = val;  }  public string getTipe()  {  return Tipe;  }  public void setWarna(string val)  {  this.Warna = val;  }  public string getWarna()  {  return Warna;  }  public void setJenisBBM(string val)  {  this.jenisBBM = val;  }  public string getJenisBBM()  {  return jenisBBM;  }  }  static void Main(string[] args)  {  Mobil mobilSaya = new Mobil(); //membuat objek mobil baru  mobilSaya.setMerk("Honda"); //set attribut untuk objek mobil  mobilSaya.setTipe("Jazz");  mobilSaya.setWarna("Silver");  mobilSaya.setJenisBBM("Pertamax");  Console.WriteLine(  "Saya punya mobil bermerk " + mobilSaya.getMerk() +  " dengan tipe " + mobilSaya.getTipe() +  ".\nMobil saya berwarna " + mobilSaya.getWarna() +  ".\nMobil saya menggunakan bensin berjenis " + mobilSaya.getJenisBBM()  ); //cetak nilai atribut objek ke console  Console.ReadLine();  }  }  } |



Gambar 6.1 *output method setter getter* pada C#

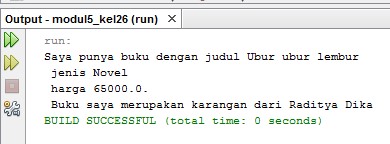
Pertama membuat kelas untuk *method setter getter* dengan nama Mobil, lalu mendeklarasikan variabel yang digunakan. Setelah itu membuat *method setter* diantaranya setMerek(string val), setTipe(string val), setWarna(string val), setJenisBBM(string val) dimana setiap *method setter* ini akan memasukkan nilai argumen dari parameter (string val) kedalam masing-masing atribut. Kemudian membuat *method getter* diantaranya getMerek(), getTipe(), getWarna(), getJenisBBM() dimana setiap *method getter* ini akan mendapat nilai atribut dan mengembalikan nilai ke fungsi yang memanggilnya.

Pada fungsi *main* membuat objek dari kelas mobil dengan nama mobilSaya kemudian menetapkan atribut merk, tipe, warna, dan jenis bbm untuk objek mobil. Setelah itu akan dieksekusi perintah untuk menampilkan *output* seperti pada gambar 6.1.

* + 1. **Bahasa Pemrograman Java**

|  |
| --- |
| package modul5\_kel26;  public class modul5Class {  //deklarasi atribut  private String judul;  private String jenis;  private Double harga;  private String penulis;    //membuat setter  public void setJudul(String judul) {  this.judul = judul;  }  public void setJenis(String jenis) {  this.jenis = jenis;  }  public void setHarga(Double harga) {  this.harga = harga;  }  public void setPenulis(String penulis) {  this.penulis = penulis;  }    //membuat getter  public String getJudul(){  return judul;  }  public String getJenis(){  return jenis;  }  public Double getHarga(){  return harga;  }  public String getPenulis() {  return penulis;  }  } |

|  |
| --- |
| package modul5\_kel26;  public class Modul5\_kel26 {  public static void main(String[] args) {  modul5Class buku\_baru = new modul5Class();  //mengatur nilai atribut  buku\_baru.setJudul("Ubur ubur lembur");  buku\_baru.setJenis("Novel");  buku\_baru.setHarga (65000.0);  buku\_baru.setPenulis("Raditya Dika");  //mencetak nilai  System.out.println("Saya punya buku dengan judul " + buku\_baru.getJudul());  System.out.println(" jenis "+ buku\_baru.getJenis());  System.out.println(" harga "+ buku\_baru.getHarga()+".");  System.out.println(" Buku saya merupakan karangan dari "+ buku\_baru.getPenulis());  }  } |



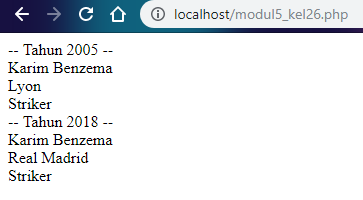
Gambar 6.2 output method setter getter pada Java

Pertama membuat kelas untuk *method setter getter* dengan nama modul5Class, lalu mendeklarasikan variabel yang digunakan. Setelah itu membuat *method setter* diantaranya setJudul(String judul), setJenis(String jenis), setHarga(Double harga), setPenulis(String penulis) dimana setiap *method setter* ini akan memasukkan nilai argumen dari parameter kedalam masing-masing atribut. Kemudian membuat *method getter* diantaranya getJudul(), getJenis(), getHarga(), getPenulis() dimana setiap *method getter* ini akan mendapat nilai atribut dan mengembalikan nilai ke fungsi yang memanggilnya.

Pada fungsi *main* membuat objek dari kelas dengan nama buku\_baru kemudian menetapkan atribut judul, jenis, harga, dan penulis untuk objek tersebut. Setelah itu akan dieksekusi perintah untuk menampilkan *output* seperti pada gambar 6.2.

* + 1. **Bahasa Pemrograman PHP**

|  |
| --- |
| <?php  class PemainBola {  private $nama;  private $club;  private $posisi;  public function \_\_construct($nama, $club, $posisi){  $this->nama = $nama;  $this->club = $club;  $this->posisi = $posisi;  }  public function setClub($clubBaru){  $this->club = $clubBaru;  }  public function setPosisi($posisiBaru){  $this->posisi = $posisiBaru;  }  public function getNama(){  return $this->nama;  }  public function getClub(){  return $this->club;  }  public function getPosisi(){  return $this->posisi;  }  }  $benzema = new PemainBola("Karim Benzema", "Lyon", "Striker");  echo "-- Tahun 2005 -- <br>";  echo $benzema->getNama() . "<br>";  echo $benzema->getClub() . "<br>";  echo $benzema->getPosisi() . "<br>";  $benzema->setClub("Real Madrid");  echo "-- Tahun 2018 -- <br>";  echo $benzema->getNama() . "<br>";  echo $benzema->getClub() . "<br>";  echo $benzema->getPosisi() . "<br>";  ?> |

****

Gambar 6.3 output setter getter pada PHP

Pada program ini, diawali dengan pendeklarasian variabel yang menggunakan enkapsulasi *private* yang variabelnya terdiri atas variabel nama, *club*, dan juga posisi. Selanjutnya membuat fungsi bernama *construct* dengan menggunakan enkapsulasi *public*, yang berparameter berisikan variabel nama, *club* dan posisi yang telah di deklarasikan sebelumnya.

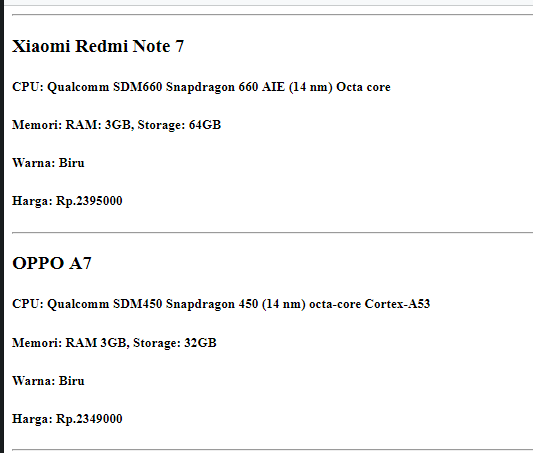
Lalu masuk ke pembuatan *method setter-*nya dan juga *getter-*nya yang nantinya akan dipanggil di *main program.*

Setelah itu membuat objek baru yaitu “benzema” yang parameternya berisikan “Karim Benzema, Lyon, Striker” yang nantinya akan digunakan untuk mengisi variabel nama, *club* dan posisi.

Kemudian membuat hasil *output*nya dan juga menggunakan *method getter* untuk memanggil setiap setiap variabel yang telah di deklarasikan sebelumnya, sehingga nantinya akan menghasilkan tampilan seperti yang tertera pada Gambar 6.3.

* 1. Tugas

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>Daftar Harga Smartphone 2019</title>  </head>  <body>  <?php  class Smartphone {  private $nama;  private $cpu;  private $memori;  private $warna;  private $harga;  public function \_\_construct(){  echo "<hr>";  }  //setter  public function setNama($nama){  $this->nama = $nama;  }  public function setCPU($cpu){  $this->cpu = $cpu;  }  public function setMemori($memori){  $this->memori = $memori;  }  public function setWarna($warna){  $this->warna = $warna;  }  public function setHarga($harga){  $this->harga = $harga;  }  //getter  public function getNama(){  return $this->nama;  }  public function getCPU(){  return $this->cpu;  }  public function getMemori(){  return $this->memori;  }  public function getWarna(){  return $this->warna;  }  public function getHarga(){  return $this->harga;  }  }  //objek 1  $smartphone1 = new Smartphone();  $smartphone1->setNama("Xiaomi Redmi Note 7");  $smartphone1->setCPU("Qualcomm SDM660 Snapdragon 660 AIE (14 nm) Octa core");  $smartphone1->setMemori("RAM: 3GB, Storage: 64GB");  $smartphone1->setWarna("Biru");  $smartphone1->setHarga(2395000);  echo "<h3>".$smartphone1->getNama()."</h3>";  echo "<h5>CPU: ".$smartphone1->getCPU() . "</h5>";  echo "<h5>Memori: ".$smartphone1->getMemori() . "</h5>";  echo "<h5>Warna: ".$smartphone1->getWarna() . "</h5>";  echo "<h5>Harga: Rp.".$smartphone1->getHarga() . "</h5>";  //objek 2  $smartphone2 = new Smartphone();  $smartphone2->setNama("OPPO A7");  $smartphone2->setCPU("Qualcomm SDM450 Snapdragon 450 (14 nm) octa-core Cortex-A53");  $smartphone2->setMemori("RAM 3GB, Storage: 32GB");  $smartphone2->setWarna("Biru");  $smartphone2->setHarga(2349000);  echo "<h3>".$smartphone2->getNama()."</h3>";  echo "<h5>CPU: ".$smartphone2->getCPU() . "</h5>";  echo "<h5>Memori: ".$smartphone2->getMemori() . "</h5>";  echo "<h5>Warna: ".$smartphone2->getWarna() . "</h5>";  echo "<h5>Harga: Rp.".$smartphone2->getHarga() . "</h5>";  //objek 3  $smartphone3 = new Smartphone();  $smartphone3->setNama("Vivo V15");  $smartphone3->setCPU("MediaTek Helio P70 Octa-core");  $smartphone3->setMemori("RAM 6GB, Storage: 64GB");  $smartphone3->setWarna("Biru");  $smartphone3->setHarga(3699000);  echo "<h3>".$smartphone3->getNama()."</h3>";  echo "<h5>CPU: ".$smartphone3->getCPU() . "</h5>";  echo "<h5>Memori: ".$smartphone3->getMemori() . "</h5>";  echo "<h5>Warna: ".$smartphone3->getWarna() . "</h5>";  echo "<h5>Harga: Rp.".$smartphone3->getHarga() . "</h5>";  //objek 4  $smartphone4 = new Smartphone();  $smartphone4->setNama("Realme 3");  $smartphone4->setCPU("MediaTek Helio P60 (12 nm, Indonesia), MediaTek Helio P70 (12nm, Global varian),  octa-core 4x ARM Cortex-A73, 4x ARM Cortex-A53");  $smartphone4->setMemori("RAM 3GB, Storage: 32GB");  $smartphone4->setWarna("Hitam");  $smartphone4->setHarga(1825000);  echo "<h3>".$smartphone4->getNama()."</h3>";  echo "<h5>CPU: ".$smartphone4->getCPU() . "</h5>";  echo "<h5>Memori: ".$smartphone4->getMemori() . "</h5>";  echo "<h5>Warna: ".$smartphone4->getWarna() . "</h5>";  echo "<h5>Harga: Rp.".$smartphone4->getHarga() . "</h5>";  ?>  </body>  </html> |



Gambar 6.4 Tampilan *output* program

Pertama mendeklarasikan kelas dengan nama *Smartphone* lalu menentukan atribut dengan pendeklarasian variabel yang menggunakan enkapsulasi *private* yang digunakan untuk UDT *Smartphone*. Setelah itu mendeklarasikan fungsi konstruktor yang akan membuat garis lurus dengan *tag* html.

Lalu masuk ke pembuatan *method setter-*nya yang akan menetapkan nilai ke atribut berdasarkan parameter dan juga *getter-*nya yang akan mengembalikan nilai dari atribut dan dipanggil di *main program.*

Setelah itu membuat objek baru yaitu $smartphone1 lalu mengisi masing-masing atribut dengan *method setter*. Kemudian menampilkan nilai atribut yang telah diisi nilainya dengan *method getter*. Membuat lagi tiga objek, dengan mengisi argumen yang berbeda pada *method setter* untuk tiap objeknya dan ditampilkan nilainya dengan *method getter* untuk tiap objek sehingga akan menghasilkan tampilan seperti yang tertera pada Gambar 6.4.

* 1. Kesimpulan

1. Untuk membuat program dapat mengirimkan beberapa variabel digunakan *User Data Type.*
2. *User Data Type* dapat digunakan untuk mengirimkan berbagai tipe data.
3. *Method setter* tidak memiliki pengembalian nilai karena berfungsi untuk menetapkan nilai saja.
4. Sebaliknya *Method getter* hanya mengembalikan nilai yang sudah ada pada atribut.
5. Enkapsulasi padaatribut menggunakan *private* supaya tidak bisa dimasukkan nilai selain dari *method setter.*
6. *Method setter* menggunakan enkapsulasi *public* agar bisa diakses dari kelas lain yang ingin menetapkan nilai pada atribut yang menggunakan enkapsulasi *private.*
7. *Method getter* menggunakan enkapsulasi *public* supaya bisa dipanggil dari kelas lain.