

Pertemuan - 3

Visual Basic Variables

Variables

- Sebuah storage yang berlokasi di memori (RAM)
 - Menyimpan data.informasi sepanjang jalannya program
 - Lokasi penyimpanan ini dapat dikenali dengan namanya
- Setiap variable mempunyai tiga properties:
 - **Name** – merujuk pada lokasi – **tidak dapat diubah**
 - **Value** – informasi yang disimpan – **dapat berubah** selama program dieksekusi, oleh karena itu dinamakan “variable”
 - **Data Type** – tipe informasi yang dapat disimpan - **tidak dapat diubah**

Bagaimana berpikir tentang Variables

- Anda programmer membuat sebuah nama untuk variabel
- Visual Basic mengasosiasikan nama tersebut dengan sebuah lokasi yang berada di RAM komputer
- Nilai saat ini yang terkait dengan variable tersebut disimpan dalam lokasi memori tersebut
- Anda tinggal dengan mudah menggunakan nama yang anda tentukan ketika anda perlu mengakses nilai tersebut.

Kegunaan Variables

- Menyalin dan menyimpan nilai yang dimasukkan oleh user
- Melaksanakan manipulasi aritmatik pada nilai.
- Menguji nilai untuk melihat jika nilai tersebut memenuhi kriteria
- Secara temporal menyimpan dan memanipulasi nilai dari property control
- Menyimpan data/informasi sehingga bisa dipanggil ulang untuk digunakan kemudian dalam program

Memilih tipe data

Data type – Menentukan tipe data dari variable yang dapat disimpan

- Integer variables: Long, Integer, Short, Byte
- Floating-point variables: Single, Double
- Fixed decimal point variable: Decimal
- Boolean variables: True, False
- Character variable: Char
- Text variable: String
- Object variable
 - Tipe data Default yang ditentukan oleh Visual Basic
 - Dapat menyimpan banyak tipe data yang berbeda
 - Kurang efisien dibanding tipe data lainnya

Type data Visual Basic

Data type	Prefix	Size	Values
Byte	byt	1 byte	positive integer value from 0 to 255
Short	shr	2 byte	integer from -32,768 to +32,767
Integer	int	4 byte	integer from +/- 2,147,483,647
Long	lng	8 byte	integer from +/- 9,223,372,036,854,775,807
Single	sng	4 byte	single-precision, floating-point number
Double	dbl	8 byte	double-precision, floating-point number
Decimal	dec	16 byte	number with up to 28 significant digits
Char	chr	2 byte	Any single character
Boolean	bln	2 byte	True or False
String	str	(4 byte)	Text - Any number/combination of characters
Date	dtm	8 byte	8 character date: #dd/mm/yyyy#
Object	obj	(4 byte)	An address that refers to an object

Nama Variable

- Karakter pertama harus huruf atau underscore
- Harus mengandung hanya huruf, bilangan dan underscores (tidak ada spasi, karakter khusus dll)
- Dapat memiliki hanya sampai 255 karakter
- Tidak berupa keyword Bahasa VB
- Konversi Penamaan
 - Harus mempunyai arti
 - Diikuti awalan tiga karakter – 3 huruf pertama dalam huruf kecil untuk menunjukkan tipe data
 - Setelah itu, huruf kapital untuk tiap katanya
 - Contoh: `intTestScore`

Mendeklarasikan Variable

- Deklarasi variable adalah perintah untuk membuat sebuah variable dalam memori
- Syntax: **Dim *VariableName* As *DataType***
 - Dim (singkatan dari Dimension) - keyword
 - VariableName – nama variabelnya
 - As - keyword
 - DataType – satu dari banyak keyword yang mengindikasikan tipe nilai yang akan dimuat dalam variabel
- Contoh: **Dim intLength as Integer**

Deklarasi dan Inisiasi sebuah Variable

- Nilai awal atau inisiasi mungkin ditentukan bersamaan dengan perintah Dim

- Syntax:

Dim VariableName As DataType = Value

- Tinggal tambahkan " = value" pada perintah Dim
- = 5 → memberikan nilai awal pada variable
- Contoh: **Dim intLength as Integer = 5**

Type Data Pada Variabel

- Setiap variabel harus dideklarasikan type datanya sehingga VB mengalokasikan sumber daya yang lebih efisien untuk variabel tersebut.
- Kita harus memahami dengan baik tiap jenis type data dan kapan type data tertentu digunakan.
- Tabel berikut merupakan daftar type data yang dapat Kita gunakan lengkap dengan ukuran dan scopenya:

Aturan Deklarasi Variable

- Variable HARUS dideklarasikan sebelum kode yang menggunakannya
- Variable harus dideklarasikan pertama dalam prosedur (konvensi)
- Mendeklarasikan nilai awai pada variable dalam perintah deklarasi adalah sebuah opsional

Tujuan Deklarasi Variable

- Tujuan pendeklarasian variabel adalah agar komputer mengetahui dengan pasti type data yang akan digunakan dalam variabel tersebut serta scopenya.
- Sebuah variabel harus memiliki nama, type data, scope, dan value.
- Berikut adalah contoh deklarasi variabel :

Dim sNama As String

Dim dGajiPokok As Decimal

Dim dTunjangan As Decimal

Dim dGajiTotal As Decimal

Dim dPajak As Decimal

sNama = "Anto"

dGajiPokok = 600000

dTunjangan = 150000

Dim dGajiPokok As Decimal = 600000

Dim sUmur, sTinggi, sGaji As Single

Nilai Default untuk Tipe Data

Data type	Default (Initial) value
All numeric types	Zero (0)
Boolean	False
Char	Binary 0
String or Object	Empty
Date	12:00 a.m. on January 1, 0001

Penamaan Konstanta

- Program sering membutuhkan untuk menggunakan nilai yang ditentukan
 - Seperti contoh: **decTotal *= 1.06**
 - Tambahkan 6% pajak penjualan pada total order

Penamaan Konstanta (lanjutan)

- Syntax: **Const *CONST_NAME* As *DataType* = *Value***
Terlihat seperti deklarasi variable, kecuali :
 - Menggunakan Const bukan Dim
 - Nilai awal diperlukan
 - Berdasarkan konvensi, semua ditulis dalam huruf besar dengan karakterk underscore untuk memisahkan kata

Penamaan Konstanta (lanjutan)

- Contoh
 - `Const sngSALES_TAX_RATE As Single = 1.06`
 - `decTotal *= sngSALES_TAX_RATE`

Aksesibilitas variabel

- Aksesibilitas variable adalah suatu keadaan dimana variable tersebut dapat diakses.
- Beberapa kata kunci yang digunakan untuk mendeklarasikan variable (ataupun elemen-elemen lain seperti kelas, fungsi, atau prosedur) berdasarkan aksesibilitas adalah sebagai berikut:
- **Public**
Variabel yang dideklarasikan dengan kata kunci Public memiliki ruang lingkup global yang juga berarti dapat diakses dibagian manapun dalam suatu project, Bahkan project lain yang memiliki referensi terhadap project tersebut.

Aksesibilitas variable (lanjutan)

- **Protected**

Kata kunci Protected mendeklarasikan elemen yang hanya dapat diakses dari sebuah kelas atau kelas turunannya. Kata kunci ini hanya digunakan pada level kelas dan hanya untuk mendeklarasikan anggota dari sebuah kelas.

- **Friend**

- Kata kunci Friend mirip dengan Public, hanya saja ruang lingkupnya hanya dalam sebuah project. Project yang lain tidak dapat mengaksesnya.

- **Protected Friend**

Kata kunci Protected Friend merupakan gabungan dari Protected dan Friend, yang berarti mendeklarasikan sebuah elemen untuk sebuah kelas atau turunannya dan hanya dapat diakses dari dalam sebuah project saja.

- **Private**

Kata kunci Private digunakan untuk mendeklarasikan elemen yang hanya dapat diakses dari dalam modul, kelas, atau struktur dimana elemen tersebut dideklarasikan. Kata kunci ini tidak dapat digunakan dari dalam sebuah prosedur.

Ruang Lingkup dan Type Variabel

- **Procedure level / local scope**
- Apabila suatu variabel dideklarasikan dalam suatu prosedur tertentu, maka variabel tersebut hanya dapat diakses dan berlaku untuk prosedur tersebut saja.

```
Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
```

```
Handles Button3.Click
```

```
Dim sPesan As String
```

```
sPesan = "tes Variabel local"
```

```
MessageBox.Show (sPesan, "Variabel local", MessageBoxButtons.OK,
```

```
_messageBoxIcon.Information)
```

```
End Sub
```

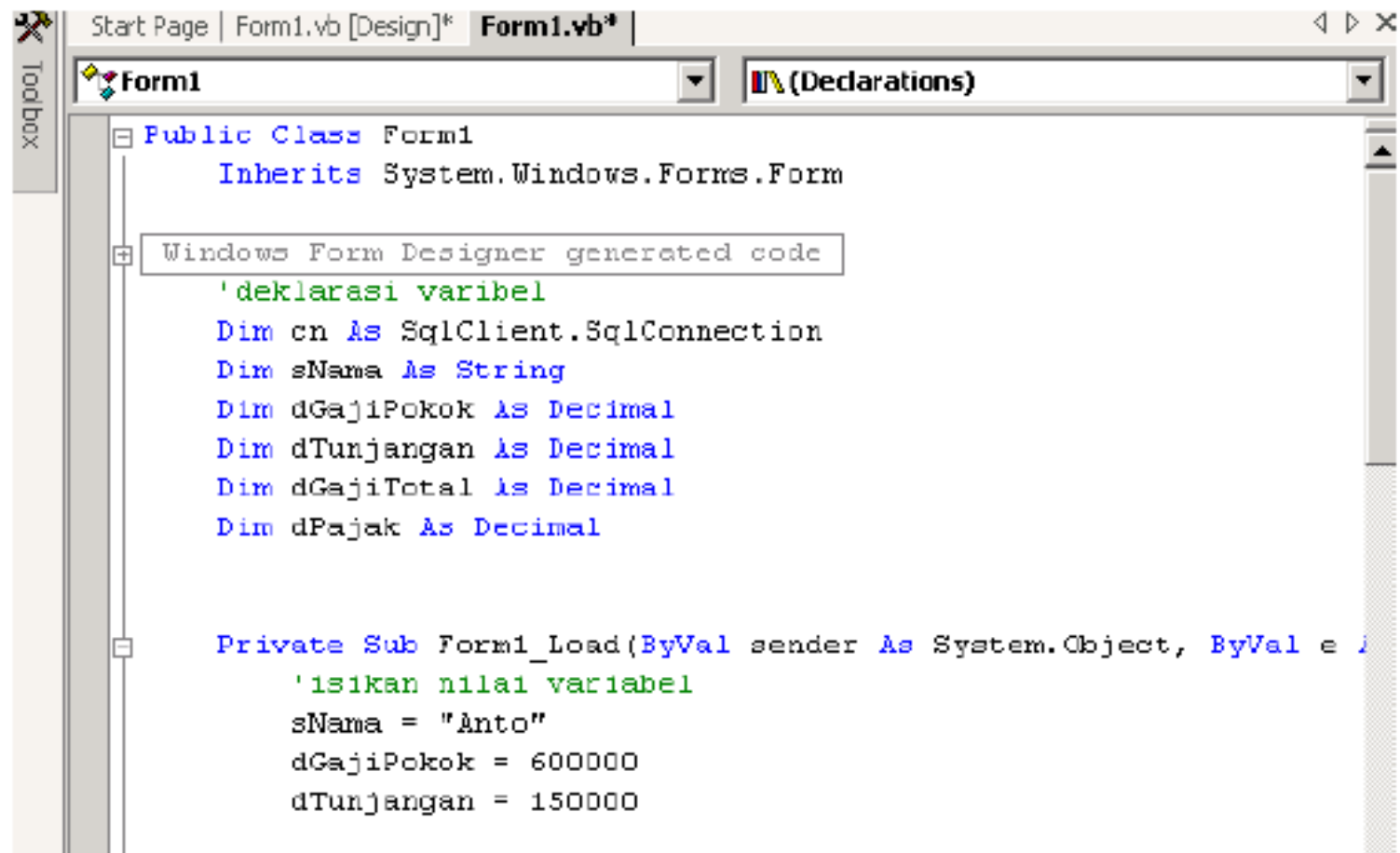
Ruang Lingkup dan Type Variabel

- Variabel sPesan dideklarasikan di dalam prosedur Button3_Click sehingga hanya berlaku di dalam prosedur tersebut saja. sPesan tidak dapat diakses dari luar Button3_Click.
- Karena variabel ini scope nya lokal untuk prosedur tertentu saja maka nama variabel tersebut hanya berlaku di dalam prosedur dimana variabel tersebut dideklarasikan.

Ruang Lingkup dan Type Variabel

- **Module Level**
- Kita dapat membuat suatu variabel yang dapat diakses dari prosedur manapun dalam suatu file.
- Misalkan Kita ingin mendeklarasikan suatu string koneksi yang akan dipakai terus menerus di dalam suatu module. Maka Kita harus mendeklarasikan variabel tersebut di bagian deklarasi module sebagaimana contoh berikut :

Gambar Code deklarasi pada VB.Net



- **Variabel Public/Global**
- Cara pendeklarasian variabel jenis ini hampir sama dengan jenis module level dengan menambahkan kata Public sebagai pengganti keyword Dim.
- Variabel ini dapat diakses dari module lain, dan bahkan dapat diakses oleh Project lain selama Project tersebut membuat reference ke Project dimana variabel tersebut dideklarasikan.
- Misalkan Kita memiliki Class yang digunakan untuk membuka dan menutup koneksi database. Maka Kita dapat mendefinisikan variabel untuk koneksi sebagai Public sehingga form, Class, maupun Project lain dapat mengakses variabel tersebut dan melakukan koneksi ke database dengan memanfaatkan Class tersebut.

Public cn **As** SqlConnection

Perintah Penugasan

- Syntax: ***variablename = expression***
- Memberikan nilai dari ekspresi pada sebuah variable. (Variabel harus disebelah kiri dan ekspresi disebelah kanan.)
- Contoh:
 - `intNumber1 = 4`
 - `intNumber2 = 3 * (2 + 2)`
 - `intNumber3 = intNumber1`
 - `IntNumber1 = intNumber1 + 6`

Val Function

- *Val function* adalah metode untuk melakukan konversi dari string ke numerik
- Menggunakan rumus Val(string)
- Jika karakter mengandung nilai numerik diawalnya, maka fungsi Val akan mengembalikan nilainya
- Jika tidak, akan menghasilkan nilai nol

Val Function

Val Function	Value Returned
■ Val("34.90")	34.9
■ Val("86abc")	86
■ Val("\$24.95")	0
■ Val("3,789")	3
■ Val("")	0
■ Val("x29")	0
■ Val("47%")	47
■ Val("Geraldine")	0

ToString Method

- Menghasilkan representasi string dari nilai dalam variable yang memanggil method tersebut
- Setiap tipe data VB memiliki *ToString* method
- Gunakan bentuk `VariableName.ToString`
- Sebagai contoh

```
Dim number as Integer = 123
lblNumber.text = number.ToString
```

 - Memberikan string "123" pada property text dari control `lblNumber`

Melakukan Perhitungan dengan Variabel

■ Operator Aritmatika

$\wedge$	Exponential
$*$	Multiplication
$/$	Floating Point Division
$\backslash$	Integer Division
MOD	Modulus (remainder from division)
$+$	Addition
$-$	Subtraction
$\&$	String Concatenation (putting them together)

Operator Aritmatika

■ Contoh :

- `decTotal = decPrice + decTax`
- `decNetPrice = decPrice - decDiscount`
- `dblArea = dblLength * dblWidth`
- `sngAverage = sngTotal / intItems`
- `dblCube = dblSide ^ 3`

Concatenating Strings

- Concatenate: menggabungkan string
- Concatenation operator: ampersand (&)
- Termasuk spasi sebelum dan sesudah operator &
- Bilangan setelah operator & dikonversi jadi string
- Berikut contohnya :
 - `strFName = "Bob"`
 - `strLName = "Smith"`
 - `strName = strFName & ""` → "Bob "
 - `strName = strName & strLName` → "Bob Smith"
 - `intX = 1      intY = 2`
 - `intResult = intX + intY`
 - `strOutput = intX & " + " & intY & " = " & intResult` → "1 + 2 = 3"

Kombinasi Operator Penugasan

- Sering digunakan untuk mengubah nilai variabel dan hasilnya dimasukkan ke variable yang sama
- Contoh : `var = var - 5`
- Mengurangi 5 dari nilai yang disimpan ddalam var

Operator	Usage	Equivalent	Effect
<code>+=</code>	<code>x += 2</code>	<code>x = x + 2</code>	tambahkan pada
<code>-=</code>	<code>x -= 5</code>	<code>x = x - 5</code>	kurangani dari
<code>*=</code>	<code>x *= 10</code>	<code>x = x * 10</code>	kalikan dengan
<code>/=</code>	<code>x /= y</code>	<code>x = x / y</code>	bagi dengan
<code>\=</code>	<code>x \= y</code>	<code>x = x \ y</code>	pembagian int dng
<code>&=</code>	<code>x &= "."</code>	<code>x = x & "."</code>	Concatenate

Urutan Operator Arithmetic

- Operator *precedence* menjelaskan urutan dari pelaksanaan operator
- Dari urutan tinggi ke rendah :
 - Exponentiation (^)
 - Multiplicative (* and /)
 - Integer Division (\)
 - Modulus (MOD)
 - Additive (+ and -)
- Tanda kurung digunakan untuk mendahulukan pengerjaan

Semua urutan operator

- Parenthesis
- Exponential
- Multiplication / Division
- Integer Division
- MOD
- Addition / Subtraction
- String Concatenation
- Relational Operators (< , > , >= , <= , <>)
- Logical Operators (AND, OR, NOT)

Contoh

- $6 * 2 ^ 3 + 4 / 2 = 50$
 - $7 * 4 / 2 - 6 = 8$
 - $5 * (4 + 3) - 15 \text{ Mod } 2 = 34$
-
- $\text{intX} = 10$
 - $\text{intY} = 5$
 - $\text{intResultA} = \text{intX} + \text{intY} * 5$ 'iResultA is 35
 - $\text{iResultB} = (\text{intX} + \text{intY}) * 5$ 'iResultB is 75
 - $\text{dResultA} = \text{intX} - \text{intY} * 5$ 'dResultA is -15
 - $\text{dResultB} = (\text{intX} - \text{intY}) * 5$ 'dResultB is 25