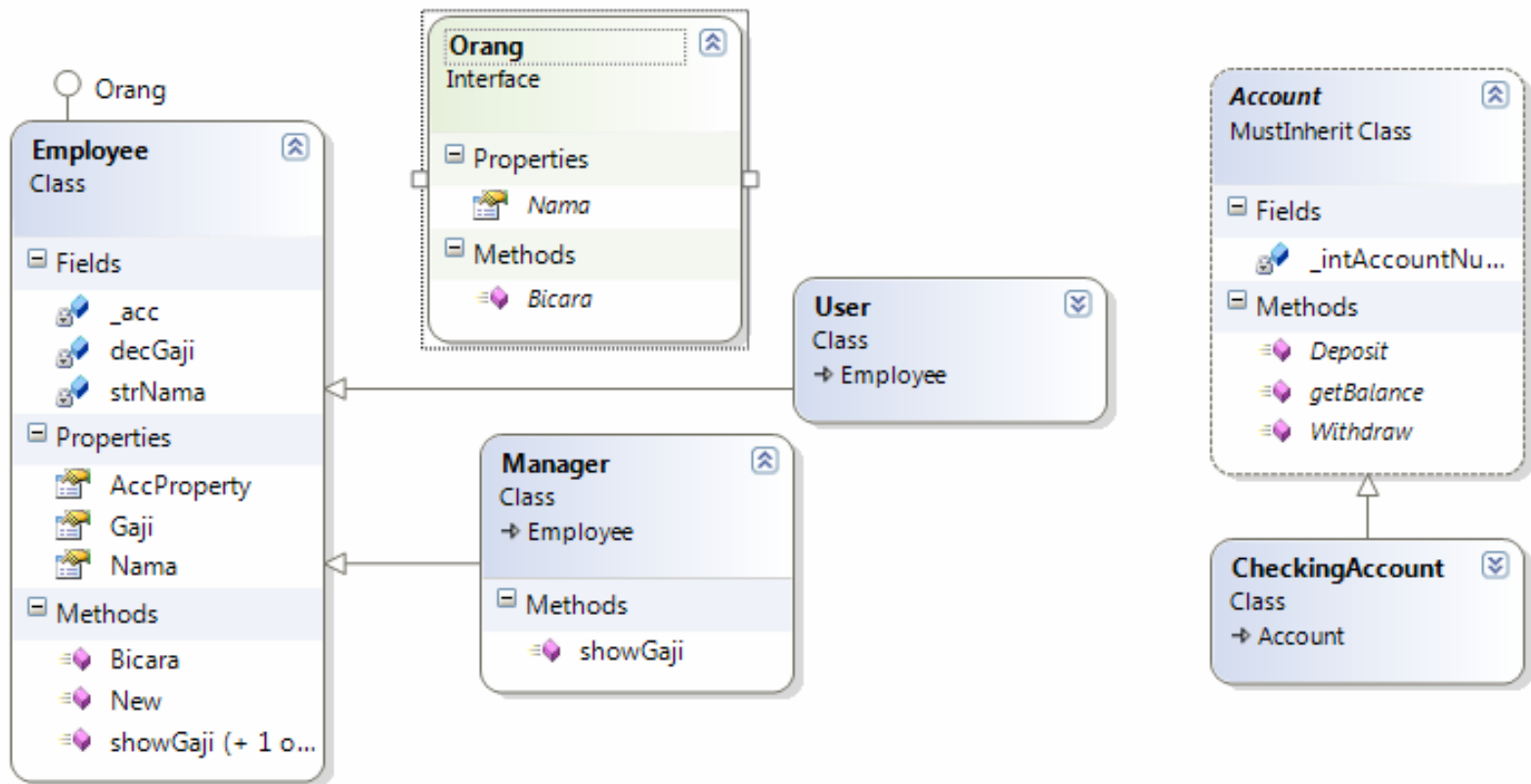


AASE #13

OOP, Database, Tipe Data

OOP di .NET

■ Class, Generalisasi, Interface



Implements Interface

```
Public Class Employee  
    Implements Orang
```

```
    Private strNama As String  
    Private decGaji As Decimal
```

```
    Public Sub New()  
        Me.Gaji = 20  
    End Sub
```

```
    Property Gaji() As Decimal  
        Get  
            Gaji = decGaji  
        End Get  
        Set(ByVal value As Decimal)  
            decGaji = value  
        End Set  
    End Property
```

```
    Public Overridable Function showGaji() As Decimal  
        Return Me.Gaji  
    End Function
```

```
    Public Overridable Function showGaji(byVal matauang) As Decimal  
        Return Me.Gaji  
    End Function
```

```
    Public Function Bicara() As String Implements Orang.Bicara  
        Return "Nama saya: " & Me>Nama  
    End Function
```

```
    Public Property Nama() As String Implements Orang>Nama  
        Get  
            Nama = strNama  
        End Get  
        Set(ByVal value As String)  
            strNama = value  
        End Set  
    End Property  
End Class
```

Konstruktor

Property Gaji (get & set)

Agar bisa dioverride oleh subclass
Overloading juga!

Implementasi dari Interface

Property Nama (get & set)

```
Public Class Manager
    Inherits Employee
    Public Overrides Function showGaji() As Decimal
        Return MyBase.Gaji + 30
    End Function
End Class
```

The diagram illustrates the code structure with annotations. A box labeled "Sub class" has an arrow pointing to the "Public Class Manager" line. Another box labeled "Override" has an arrow pointing to the "Return MyBase.Gaji + 30" line.

```
Public Interface Orang
    Property Nama() As String
    Function Bicara() As String
End Interface
```

The diagram illustrates the interface definition with an annotation. A box labeled "Interface" has an arrow pointing to the "Public Interface Orang" line.

```
Anton bicara=> Nama saya: Antonius RC
Budi Rahardjo bicara=> Nama saya: Budi Rahadjo
Budi Raradjo gaji=> 40
Chandra K bicara=> Nama saya: Chandra Kurniawan
Chandra K adalah manajer
Chandra K gajinya=> 50
```

Module Module1

Sub **Main()**

```
Dim anton As Orang = New Employee
```

```
anton>Nama = "Antonius RC"
```

```
Console.WriteLine("Anton bicara=> " & anton.Bicara())
```

```
Dim budi As Employee = New Employee
```

```
budi>Nama = "Budi Rahadjo"
```

```
budi.Gaji = 40
```

```
Console.WriteLine("Budi Rahardjo bicara=> " & budi.Bicara())
```

```
Console.WriteLine("Budi Raradjo gaji=> " & budi.showGaji())
```

```
Dim chandra As Employee = New Manager
```

```
chandra>Nama = "Chandra Kurniawan"
```

```
Console.WriteLine("Chandra K bicara=> " & chandra.Bicara())
```

```
Console.WriteLine("Chandra K adalah manajer")
```

```
Console.WriteLine("Chandra K gajinya=> " & chandra.showGaji())
```

```
Console.ReadLine()
```

End Sub

End Module

Hak Akes Property

Misal Property Password dan Username

```
Public Class User
    Inherits Employee

    Private _uname As String
    Private _pass As String

    Property Password() As String
        Get
            Return _pass
        End Get
        Set(ByVal value As String)
            If Len(value) >= 6 Then
                _pass = value
            Else
                Throw New Exception("Password must be at least
                6 characters.")
            End If
        End Set
    End Property

    Property Username() As String
        Get
            Return _uname
        End Get
        Set(ByVal value As String)
            _uname = value
        End Set
    End Property
End Class
```

ReadOnly dan WriteOnly

```
Public Class User
    Inherits Employee

    Private _uname As String
    Private _pass As String

    Public ReadOnly Property Username() As
    String
        Get
            Return _uname
        End Get
    End Property

    Public WriteOnly Property Password() As
    String
        Set(ByVal value As String)
            _pass = value
        End Set
    End Property
End Class
```

Instilah-istilah OOP di VB.NET

- Current Class : MyClass
 - Parent Class : MyBase
 - Abstract Class : MustInherit
 - Method Abstract : MustOverride
 - Extend : Inherits
 - Agar method bisa dioverride : Overridable
 - Ketika mengoverride method : Overrides
 - Static : Shared
 - Interface = Implements
-

Employee memiliki Account

```
Private _acc As Account

Public Property AccProperty() As Account
    Get
        Return _acc
    End Get
    Set(ByVal value As Account)
        _acc = value
    End Set
End Property
```

```
Public MustInherit Class Account ← Abstract Class
    Private _intAccountNumber As Integer
    Public MustOverride Sub Deposit(ByVal Amount As Double)
    Public MustOverride Sub Withdraw(ByVal Amount As Double)
    Public MustOverride Function getBalance() As Double
End Class
```

CheckingAccount

Public Class **CheckingAccount**

Inherits Account

Private _dblBalance As Double = 2000

Public **ReadOnly** Property Balance()

Get

Return _dblBalance

End Get

End Property

Public **Overridable** Function GetMinBalance() As Double

Return 200

End Function

Public **Overrides** Sub Withdraw(ByVal Amount As Double)

Dim dblMinBalance As Double = GetMinBalance()

If dblMinBalance < (Balance - Amount) Then

_dblBalance -= Amount

Else

Throw New Exception("Minimum balance error.")

End If

End Sub

Public **Overrides** Sub Deposit(ByVal Amount As Double)

_dblBalance += Amount

End Sub

Public **Overrides** Function getBalance() As Double

Return Balance

End Function

End Class

Property ReadOnly

Method biasa

Implementasi Abstract

Implementasi Abstract

Implementasi Abstract

Main Class

```
Dim oCheckingAccount As CheckingAccount = New CheckingAccount()
```

```
oCheckingAccount.Deposit(100.0)
```

```
oCheckingAccount.Withdraw(20.0)
```

```
chandra.AccProperty = oCheckingAccount
```

```
Console.WriteLine("Chandra K tabungannya => " & chandra.AccProperty.getBalance)
```

```
Anton bicara=> Nama saya: Antonius RC  
Budi Rahardjo bicara=> Nama saya: Budi Rahadjo  
Budi Raradjo gaji=> 40  
Chandra K bicara=> Nama saya: Chandra Kurniawan  
Chandra K adalah manajer  
Chandra K gajinya=> 50  
Chandra K tabungannya => 2080
```

Web Service dengan .NET

- **Procedure** : adalah suatu kumpulan perintah-perintah yang digunakan untuk suatu tujuan tertentu dan diberi nama tertentu.
 - Procedure tidak mengembalikan nilai
 - Di dalam VB : keywordnya **sub ... end sub**
 - Tidak ada keyword **return**
 - **Function** : adalah suatu kumpulan perintah-perintah yang digunakan untuk suatu tujuan tertentu dan diberi nama tertentu serta mengembalikan nilai tertentu keluar kepada fungsi yang memanggilnya.
 - Function mengembalikan nilai
 - Di dalam VB : keywordnya **function end function**
 - Ada keyword **return**
-

Contoh Procedure

- **Dalam VB:**

```
Private Sub LuasPersegiPanjang(ByVal panjang as Integer, ByVal lebar  
as Integer)
```

```
    Dim luas as Integer
```

```
    luas = panjang * lebar
```

```
    Console.WriteLine("Luas = " & Str(luas))
```

```
End Sub
```

- **Dalam C#:**

```
private void LuasPersegiPanjang(int panj, int lebar){
```

```
    int luas;
```

```
    luas = panj * lebar;
```

```
    Console.WriteLine("Luas = " + Convert.ToString(luas));
```

```
}
```

Contoh Function

- **Dalam VB:**

```
Private Function LuasPersegiPanjang(ByVal panjang as Integer, ByVal lebar as Integer) as Integer
```

```
    Return panjang*lebar;
```

```
End Sub
```

```
Console.WriteLine("Luas = " & LuasPersegiPanjang(5,3));
```

- **Dalam C#:**

```
private int LuasPersegiPanjang(int panj, int lebar){
```

```
    return panj*lebar;
```

```
}
```

```
Console.WriteLine("Luas = " + LuasPersegiPanjang(5,3));
```

Prinsip-prinsip Method

- Jika kita mengharapkan nilai kembalian dari suatu method gunakan function
 - Sedangkan jika kita tidak mengharapkan suatu nilai kembalian dari suatu method, kita gunakan sub/procedure/function-void
 - Pakailah semua method yang telah dibuat pada webservice untuk digunakan pada program desktop atau web application yang kita buat
 - Jika kita deklarasikan method yang bersifat private, maka method tersebut tidak akan tampak dalam list method yang ada di daftar service dalam Web Service.
 - Jika kita membuat method yang bersifat public, maka akan tampak di dalam list method yang ada di daftar service dalam Web Service.
-

Database Webservice

Imports System.Web.Services karena kita menggunakan Web Service

Imports System.Web.Services

Imports System.Data.OleDb jika kita menggunakan OleDb database, misalnya Access atau Oracle atau MySQL

Imports System.Data.OleDb

Imports System.Data.SqlClient jika kita menggunakan SQLServer database

Imports System.Data.SqlClient

Database Webservice

- Buatlah variable private beripe string, misal bernama strConnection yang berisi cara koneksi database

Buatlah variabel private bertipe string, misal bernama strSQL yang berisi SQL query yang bergantung pada query yang ingin kita lakukan

```
Private strSQL As String
```

Buatlah variabel private bertipe class SqlConnection, misal bernama oSqlConnection untuk obyek koneksi database yang akan dibangun berdasarkan strConnection

```
Private oSqlConnection As SqlConnection = New  
SqlConnection(strConn) //untuk SQLServer
```

```
- Private oOleDbConnection As OleDbConnection = New  
OleDbConnection(strConn) //untuk OleDb
```

Database Webservice

Buatlah variabel private beripe class SqlCommand, misalnya bernama oSqlCommand untuk obyek perintah SQL Query yang akan dilakukan melalui SqlConnection yang telah kita buat

```
Private oSqlCommand As SqlCommand //untuk SQL Server  
Private oOleDbCommand As OleDbCommand //untuk OleDb
```

Berikan perintah pada SqlCommand bertipe CommandType.Text jika perintah SQL biasa

```
Me.oSqlCommand = New SqlCommand(Me.strSQL, Me.oSqlConnection)  
//untuk SQL Server  
Me.oOleDbCommand = New OleDbCommand(Me.strSQL,  
Me.oOleDbConnection) //untuk OleDb  
//Jika ingin mengganti SQL Query  
Me.oSqlCommand.CommandType = CommandType.Text  
Me.oSqlCommand.CommandText = strSQL
```

Database Webservice

Kemudian panggil method `ExecuteReader` dari obyek `SqlCommand` yang akan membaca database per-record dengan menggunakan looping (method `read`) yang terlebih dahulu memanggil method `open()` dari obyek `SqlConnection`

```
Me.oSqlConnection.open()  
Me.oSqlDataReader = Me.oSqlCommand.ExecuteReader()  
Dim Nama As String  
If oSqlDataReader.HasRows Then  
    While oSqlDataReader.Read() Do  
        Nama = oSqlDataReader("Nama")  
    End While  
End If
```

Jika ingin mengeksekusi query selain `select`, silahkan buat variabel integer, misalnya hasil untuk menampung hasil eksekusi perintah selain `select`. Jika hasilnya 1 maka proses tersebut berhasil, jika tidak maka proses tersebut gagal!

Database Webservice

Untuk mengeksekusi perintah selain query select panggil method `ExecuteNonQuery` dari obyek `SqlCommand` yang terlebih dahulu memanggil method `open()` dari obyek `SqlConnection`

```
Me.oSqlConnection.open()  
Int hasil = Me.oSqlCommand.ExecuteNonQuery  
If hasil = 1 Then  
    Response.Write("Sukses!")  
Else  
    Response.Write("Gagal!")  
End If
```

Untuk menampilkan hasil select query di dalam `DataGrid` atau `DataList`, kita harus menggunakan class `DataSet`, `SqlDataAdapter` yang akan memanggil method `fill`-nya yang akan diisi oleh `DataSet` yang kita miliki

```
Me.strSQL = "select * from user"  
Me.objDataAdapter = New OleDbDataAdapter(Me.strSQL, Me.strConn)  
Dim ds As DataSet = New DataSet  
Me.objDataAdapter.Fill(ds)
```

Database Webservice

- Semua obyek SqlConnection, SqlDataReader, SqlCommand, SqlDataAdapter bisa diganti OleDbConnection, OleDbDataReader, OleDbCommand, dan OleDbDataAdapter jika kita ingin menggunakan database seperti Access/Postgres/MySQL/Oracle
 - Untuk Web Service yang menggunakan database, jika ingin mengembalikan data berupa sejumlah record dari sebuah query/tabel, misalnya select * from tabel, maka hanya ada satu tipe kembalian, yaitu obyek **DataSet** yang akan diterima di client dengan obyek yang sama juga dan kemudian bisa digunakan langsung untuk ditampilkan pada DataList/DataGrid
-

Type Data dan Parameter WS.NET

- Primitive Type
 - String, Char, Byte, Boolean, Int16, Int32, Single, Double, DateTime
 - Enum Type
 - Public Enum Warna
 - Merah
 - Kuning
 - Hijau
 - End Enum
 - Dim warnaku as Warna = Warna.Merah
 - Class
 - DataSet
 - Array
-

Contoh-contoh

Enum:

```
Public Enum Warna
    Merah
    Kuning
    Hijau
End Enum
```

```
<?xml version="1.0" encoding="utf-8" ?>
<Warna xmlns="http://tempuri.org/">Kuning</Warna>
```

Idx diisi 2



```
<WebMethod()> _
    Public Function CobaEnum(ByVal idx As Integer) As Warna
        Select Case idx
            Case 1
                Return Warna.Merah
            Case 2
                Return Warna.Kuning
            Case Else
                Return Warna.Hijau
        End Select
    End Function
```

■ Tipe Data Primitif : String dan Integer

```
<?xml version="1.0" encoding="utf-8" ?>  
<string xmlns="http://tempuri.org/">Hello World</string>
```

<WebMethod()> _

Public Function HelloWorld() As String

Return "Hello World"

End Function

```
<?xml version="1.0" encoding="utf-8" ?>  
<int xmlns="http://tempuri.org/">9</int>
```

<WebMethod()> _

Public Function Jumlahkan(ByVal a As Integer, ByVal b As Integer) As Integer

Return a + b

End Function

Array

```
- <ArrayOfArrayOfInt xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://tempuri.org/">
  - <ArrayOfInt>
    <int>1</int>
    <int>2</int>
    <int>3</int>
  </ArrayOfInt>
  - <ArrayOfInt>
    <int>4</int>
    <int>5</int>
  </ArrayOfInt>
</ArrayOfArrayOfInt>
```

<WebMethod()> _

Public Function JmlMatrik() As Integer()

Dim a As Integer() = {1, 2, 3}

Dim b As Integer() = {4, 5}

Dim hasil() As Integer = {a, b}

Return hasil

End Function

Array multi dimensi tidak support, harus jagged array

Class

Public Class Mahasiswa

```
Private _nim As String  
Private _nama As String  
Private _ipk As Double
```

Property Nim() As String

```
Get  
    Return _nim  
End Get  
Set(ByVal value As String)  
    _nim = value  
End Set  
End Property
```

Property Nama() As String

```
Get  
    Return _nama  
End Get  
Set(ByVal value As String)  
    _nama = value  
End Set  
End Property
```

Property IPK() As Double

```
Get  
    Return _ipk  
End Get  
Set(ByVal value As Double)  
    _ipk = value  
End Set  
End Property
```

Public Function Bicara() As String

```
Return "Saya bernama " & Nama() & ", NIM saya " & Nim() & " dan IPK adalah " & IPK()  
End Function
```

End Class

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
- <Mahasiswa xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
    xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://tempuri.org/">  
    <Nim>22002529</Nim>  
    <Nama>anton</Nama>  
    <IPK>3.68</IPK>  
</Mahasiswa>
```

Class

```
<WebMethod(> _
```

```
Public Function getMahasiswa(ByVal mynim As String) As Mahasiswa
```

```
Dim m As Mahasiswa = New Mahasiswa()
```

```
If mynim = "22002529" Then
```

```
    m.Nim = "22002529"
```

```
    m>Nama = "anton"
```

```
    m.IPK = 3.68
```

```
Else
```

```
    m.Nim = "22002521"
```

```
    m>Nama = "mahas"
```

```
    m.IPK = 3.54
```

```
End If
```

```
Return m
```

```
End Function
```

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
- <Mahasiswa xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://tempuri.org/">
```

```
<Nim>22002529</Nim>
```

```
<Nama>anton</Nama>
```

```
<IPK>3.68</IPK>
```

```
</Mahasiswa>
```

Buatlah Clientnya!
