Chapter 28: Creating a procedure

Section 28.1: Introduction to procedures

A `Sub` is a procedure that performs a specific task but does not return a specific value.

```
Sub ProcedureName ([argument_list])
    [statements]
End Sub
```

If no access modifier is specified, a procedure is `Public` by default.

A `Function` is a procedure that is given data and returns a value, ideally without global or module-scope side-effects.

```
Function ProcedureName ([argument_list]) [As ReturnType]
    [statements]
End Function
```

A `Property` is a procedure that *encapsulates* module data. A property can have up to 3 accessors: `Get` to return a value or object reference, `Let` to assign a value, and/or `Set` to assign an object reference.

```
Property Get|Let|Set PropertyName([argument_list]) [As ReturnType]
    [statements]
End Property
```

Properties are usually used in class modules (although they are allowed in standard modules as well), exposing accessor to data that is otherwise inaccessible to the calling code. A property that only exposes a `Get` accessor is "read-only"; a property that would only expose a `Let` and/or `Set` accessor is "write-only". Write-only properties are not considered a good programming practice - if the client code can *write* a value, it should be able to *read* it back. Consider implementing a `Sub` procedure instead of making a write-only property.

Returning a value

A `Function` or `Property Get` procedure can (and should!) return a value to its caller. This is done by assigning the identifier of the procedure:

```
Property Get Foo() As Integer
    Foo = 42
End Property
```

Section 28.2: Function With Examples

As stated above Functions are smaller procedures that contain small pieces of code which may be repetitive inside a Procedure.

Functions are used to reduce redundancy in code.

Similar to a Procedure, A function can be declared with or without an arguments list.

Function is declared as a return type, as all functions return a value. The Name and the Return Variable of a function are the Same.

1. Function With Parameter:

```
Function check_even(i as integer) as boolean
if (i mod  2) = 0 then
check_even = True
else
check_even=False
end if
end Function
```

2. Function Without Parameter:

```
Function greet() as String
greet= "Hello Coder!"
end Function
```

The Function can be called in various ways inside a function. Since a Function declared with a return type is basically a variable. it is used similar to a variable.

Functional Calls:

```
call greet() 'Similar to a Procedural call just allows the Procedure to use the
             'variable greet
string_1=greet() 'The Return value of the function is used for variable
                 'assignment
```

Further the function can also be used as conditions for if and other conditional statements.

```
for i = 1 to 10
if check_even(i) then
msgbox i & " is Even"
else
msgbox i & " is Odd"
end if
next i
```

Further more Functions can have modifiers such as By ref and By val for their arguments.