Chapter 26: Working With Files and Directories Without Using FileSystemObject

Section 26.1: Determining If Folders and Files Exist

Files:

To determine if a file exists, simply pass the filename to the `Dir$` function and test to see if it returns a result. Note that `Dir$` supports wild-cards, so to test for a *specific* file, the passed `pathName` should to be tested to ensure that it does not contain them. The sample below raises an error - if this isn't the desired behavior, the function can be changed to simply return `False`.

```
Public Function FileExists(pathName As String) As Boolean
    If InStr(1, pathName, "*") Or InStr(1, pathName, "?") Then
        'Exit Function   'Return False on wild-cards.
        Err.Raise 52    'Raise error on wild-cards.
    End If
    FileExists = Dir$(pathName) <> vbNullString
End Function
```

Folders (Dir$ method):

The `Dir$()` function can also be used to determine if a folder exists by specifying passing `vbDirectory` for the optional `attributes` parameter. In this case, the passed `pathName` value must end with a path separator (`\`), as matching *filenames* will cause false positives. Keep in mind that wild-cards are only allowed after the last path separator, so the example function below will throw a run-time error 52 - "Bad file name or number" if the input contains a wild-card. If this isn't the desired behavior, uncomment `On Error Resume Next` at the top of the function. Also remember that `Dir$` supports relative file paths (i.e. `..\Foo\Bar`), so results are only guaranteed to be valid as long as the current working directory is not changed.

```
Public Function FolderExists(ByVal pathName As String) As Boolean
    'Uncomment the "On Error" line if paths with wild-cards should return False
    'instead of raising an error.
    'On Error Resume Next
    If pathName = vbNullString Or Right$(pathName, 1) <> "\" Then
        Exit Function
    End If
    FolderExists = Dir$(pathName, vbDirectory) <> vbNullString
End Function
```

Folders (ChDir method):

The `ChDir` statement can also be used to test if a folder exists. Note that this method will temporarily change the environment that VBA is running in, so if that is a consideration, the `Dir$` method should be used instead. It does have the advantage of being much less forgiving with its parameter. This method also supports relative file paths, so has the same caveat as the `Dir$` method.

```
Public Function FolderExists(ByVal pathName As String) As Boolean
    'Cache the current working directory
    Dim cached As String
    cached = CurDir$

    On Error Resume Next
```

```
    ChDir pathName
    FolderExists = Err.Number = 0
    On Error GoTo 0
    'Change back to the cached working directory.
    ChDir cached
End Function
```

Section 26.2: Creating and Deleting File Folders

NOTE: For brevity, the examples below use the FolderExists function from the **Determining If Folders and Files Exist** example in this topic.

The `MkDir` statement can be used to create a new folder. It accepts paths containing drive letters (`C:\Foo`), UNC names (`\\Server\Foo`), relative paths (`..\Foo`), or the current working directory (`Foo`).

If the drive or UNC name is omitted (i.e. `\Foo`), the folder is created on the current drive. This may or may not be the same drive as the current working directory.

```
Public Sub MakeNewDirectory(ByVal pathName As String)
    'MkDir will fail if the directory already exists.
    If FolderExists(pathName) Then Exit Sub
    'This may still fail due to permissions, etc.
    MkDir pathName
End Sub
```

The `RmDir` statement can be used to delete existing folders. It accepts paths in the same forms as `MkDir` and uses the same relationship to the current working directory and drive. Note that the statement is similar to the Windows `rd` shell command, so will throw a run-time error 75: "Path/File access error" if the target directory is not empty.

```
Public Sub DeleteDirectory(ByVal pathName As String)
    If Right$(pathName, 1) <> "\" Then
        pathName = pathName & "\"
    End If
    'Rmdir will fail if the directory doesn't exist.
    If Not FolderExists(pathName) Then Exit Sub
    'Rmdir will fail if the directory contains files.
    If Dir$(pathName & "*") <> vbNullString Then Exit Sub

    'Rmdir will fail if the directory contains directories.
    Dim subDir As String
    subDir = Dir$(pathName & "*", vbDirectory)
    Do
        If subDir <> "." And subDir <> ".." Then Exit Sub
        subDir = Dir$(, vbDirectory)
    Loop While subDir <> vbNullString

    'This may still fail due to permissions, etc.
    RmDir pathName
End Sub
```