PROJECT MILESTONE 2 PRM281

Belgium Campus Kempton Park

PLANNING TO MONITOR FILES

Monique Scheurwater

Contents

Pseudocode 2
Topics 6
What topics I'll need? 6
Why and how will I need them? 6
Windows Form Design 7

Pseudocode

```
START
(MAINFORM CODE)
VOID (button) ChooseDirectory(object sender, EventArgs e)
{
        ChooseDirectory()
}
FUNCTION ChooseDirectory()
{
        SET path to " " (empty)
        IF the folderbrowserdialog.showdialog equals to dialogresult.ok THEN
        {
                SET the filesystemwatcher path to the selectedpath of the folderbrowserdialog
                SET path equal to the filesystemwatcher path
                RETURN path
        }
        ELSE THEN
        {
                RETURN "No directory chosen"
        }
}
VOID (button) Monitor(object sender, EventsArgs e)
{
        CHANGE button stop text colour to black
        CHANGE button monitor text colour green
        Watcher()
}
FUNCTION Watcher()
{
        SET filesystemwatcher notifyfilter to filter the lastaccess, lastwrite, filename and
        directoryname
        IF filesystemwatcher is changed THEN
        {
                OnChanged
        }
        IF filesystemwatcher is created THEN
        {
                OnChanged
        }
        IF filesystemwatcher is deleted THEN
        {
                OnChanged
        }
IF filesystemwatcher is renamed THEN
        {
                OnRenamed
        }
```

```
ENABLE the filesystemwatcher to start monitoring
}
VOID OnChanged(object source, FileSystemEventArgs e)
{
        SET wct to e.changetype
        SET fullpath to e.fullpath
        SET m to fullpath + wct
        PRINT m
        ADD m to the changes list
        SET listbox datasource to null
        SET listbox datasource to changes list
        SET update label visibility to true
        SET update label text to m
        SET change to change + 1
        INSTANTIATE Change class
        SET mydel to a new Delegate to write the changes to a textfile
        SET c.full to fullpath
        SET c.changetype to wct
        USE mydel to write list c into a textfile
}
VOID OnRenamed(object source, RenamedEventArgs e)
{
        SET wct to e.changetype
        SET fullpath to e.fullpath
        SET oldpath to e.oldpath
        SET m to oldpath + fullpath + wct
        PRINT m
        ADD m to the changes list
        SET listbox datasource to null
        SET listbox datasource to changes list
        SET update label visibility to true
        SET update label text to m
        SET change to change + 1
        SET mydel to a new Delegate to write the changes to a textfile
        SET c.full to fullpath
        SET c.changetype to wct
        USE mydel to write list c into a textfile
}
VOID (button) Stop(object sender, EventArgs e)
{
        CHANGE monitor button text colour to black
        CHANGE stop button text colour to red
        DISABLE filesystemwatcher to stop monitoring
        CHANGE label howmany visibility to true
        CHANGE label howmany text to how many changes happend
```

```
}
VOID (button) AllChanges(object sender, EventArgs e)
{
        CHANGE datagridview visibility to true
        CHANGE listbox visibility to false
        INSTANTIATE list of type changes class
        Filehandler read list
        FOREACH item in the list
        {
                ADD the list to the bindingsource
        }
        CHANGE datagridview datasource to the bindingsource
}
VOID Writechanges(changes c)
{
        Filehandler.writerrename c
}
VOID (button) Exit(object sender, EventArgs e)
{
        SET dialogresult to a messagebox
        IF dialogresult equals yes THEN
        {
                Exit the application
        }
}

(FILEHANDLER CLASS)
SET filename to info.txt
FUNCTION List of type changes class
{
        INSTANTIATE streamreader with the name reader
        USE reader TO
        {
                SET line to reader. Read the next line
                WHILE line is not null DO
                {
                        INSTANTIATE array value
                        SET value to where the line splits
                        ADD value to list change
                        SET line to the next line in textfile
                }
        }
        RETURN the change list
}
```

```
VOID WriterRename(Changes c)
{
        INSTANTIATE streamwriter named writer
        USE writer TO
        {
                Write list c into the textfile
        }
}

(CANGES CLASS)

CONSTRUCTOR Changes(string full, string changetype)
{
        SET full equall to full
        SET changetype to changetype
}
CONSTRUCTOR Changes()
{

}
FUNCTION ToString()
{
        RETURN full + "#" + changetype
}
```

Topics

What topics I'll need?

1. FileSystemWatcher
2. Events
3. Threading
4. Classes
5. Collections

Why and how will I need them?

1. FileSystemWatcher – It is used to monitor files. It watches a file directory in the system for changes and activates events when changes happen.
2. Events – You cannot use FileSystemWatcher without events. The events is basically what shows you when a file is created, renamed or deleted.
3. Threading – It is a must have to ensure that your program's responsiveness is always good. To ensure that the program is not slow.
4. Classes – You would basically do everything inside classes. Classes makes your program neater and more secure. You'd use classes to choose a directory, for the file system watcher, etc.
5. Collections – You'd use a list to store the paths in when it updates and then the same list or a different list to display the information.

Windows Form Design