

Forum:[Hack Forums](#)

Thread: [Adding images to array from folder](#)

SIM-1517404 - March 5, 2017, 7:38 am

Hi,

I was wondering if it's possible to add all images in a folder to an array in my project? The amount of images in the folder will change.

SIM-1517404 - March 5, 2017, 7:53 am

(Mar 5, 2017 08:46 AM)Orka Wrote: [\[CITE\]](#) Yes it is possible, but I recommend storing their absolute paths instead of array of graphic/bitmaps.

So have a string array with their paths ?

Having the path stored in arrays would work, but I don't want to manually add the path to every image. I want to loop through the folder, adding all available images.

I might also in the future want to read files from a database.

+Haze - March 5, 2017, 8:52 am

What is your goal, what are you trying to do after you've got an array of the images?

SIM-1517404 - March 5, 2017, 8:55 am

(Mar 5, 2017 09:52 AM)+Haze Wrote: [\[CITE\]](#) What is your goal, what are you trying to do after you've got an array of the images?

They are panorama images, so I'm loading them in a panorama viewer engine.

Lucasheer - March 5, 2017, 9:06 am

I wrote a few examples along with how you can possibly get the image paths you want. If Bitmaps are what you want, and you don't have too many images, then you might want to use Example 1, and dispose them afterwards. OR, if you have many images, and you can dispose as you use, then use Example 2. The panorama engine might even have some way to load images via string paths instead of actual images in memory, if that makes sense.

Code Private Function GetImagePaths() As String() 'You could use .Where, with EnumerateFiles. But this builds a list of files with ending of .jpg or .png (might want to modify the strings IDK)

```
Dim files As String() = IO.Directory.GetFiles(Application.StartupPath)
```

```
Dim fileList As New List(Of String)
```

```
If Not (IsNothing(files)) Then
```

```
    If (files.Length > 0) Then
```

```
        For I As Integer = 0 To files.Length - 1
```

```
            If (files(I).ToLower.EndsWith(".jpg") Or files(I).ToLower.EndsWith(".png")) Then
```

```
                fileList.Add(files(I))
```

```
            End If
```

```
        Next
```

```
    End If
```

```
End If
```

```
Return fileList.ToArray()
```

```
End Function
```

Private Function AccessingImageExample1(ByVal imagePaths As String()) As Bitmap() 'uses a ton of memory, but returns every single image in the folder.

```
Dim bitmapList As New List(Of Bitmap)
If Not (IsNothing(imagePaths)) Then
    If (imagePaths.Length > 0) Then
        For I As Integer = 0 To imagePaths.Length - 1
            If (My.Computer.FileSystem.FileExists(imagePaths(I))) Then
                bitmapList.Add(New Bitmap(imagePaths(I)))
            End If
        Next
    End If
End If
Return bitmapList.ToArray()
End Function
```

Private Sub AccessingImageExample2(ByVal imagePaths As String()) 'example demonstrates using the bitmaps then disposing (better for lots of images, saves memory)

```
If Not (IsNothing(imagePaths)) Then
    If (imagePaths.Length > 0) Then
        For I As Integer = 0 To imagePaths.Length - 1
            If (My.Computer.FileSystem.FileExists(imagePaths(I))) Then
                Using iBitmap = New Bitmap(imagePaths(I))
                    'use iBitmap for anything
                    iBitmap.Dispose()
                End Using
            End If
        Next
    End If
End If
End Sub
```

Private Sub DisposeBitmaps(ByVal bitmaps As Bitmap()) 'To be used in example 1, to mass dispose, but its better to dispose as you use.

```
If Not (IsNothing(bitmaps)) Then
    If (bitmaps.Length > 0) Then
        For I As Integer = 0 To bitmaps.Length - 1
            bitmaps(I).Dispose()
        Next
    End If
End If
End Sub
```

Ninja - March 5, 2017, 9:13 am

Loading in all images at once can be heavy on your machine depending on how many you load in at once.

SIM-1517404 - March 5, 2017, 9:16 am

(Mar 5, 2017 10:06 AM)Lucasheer Wrote: [\[CITE\]](#) I wrote a few examples along with how you can possibly get the image paths you want. If Bitmaps are what you want, and you don't have too many images, then you might want to use Example 1, and dispose them afterwards. OR, if you have many images, and you can dispose as you use, then use Example 2. The panorama engine might even have some way to load images via string paths instead of actual images in memory, if that makes sense.

Code Private Function GetImagePaths() As String() 'You could use .Where, with EnumerateFiles.
But this builds a list of files with ending of .jpg or .png (might want to modify the strings IDK)

```
Dim files As String() = IO.Directory.GetFiles(Application.StartupPath)
Dim fileList As New List(Of String)
If Not (IsNothing(files)) Then
    If (files.Length > 0) Then
        For I As Integer = 0 To files.Length - 1
            If (files(I).ToLower.EndsWith(".jpg") Or files(I).ToLower.EndsWith(".png")) Then
                fileList.Add(files(I))
            End If
        Next
    End If
End If
Return fileList.ToArray()
End Function
```

Private Function AccessingImageExample1(ByVal imagePaths As String()) As Bitmap() 'uses a ton of memory, but returns every single image in the folder.

```
Dim bitmapList As New List(Of Bitmap)
If Not (IsNothing(imagePaths)) Then
    If (imagePaths.Length > 0) Then
        For I As Integer = 0 To imagePaths.Length - 1
            If (My.Computer.FileSystem.FileExists(imagePaths(I))) Then
                bitmapList.Add(New Bitmap(imagePaths(I)))
            End If
        Next
    End If
End If
Return bitmapList.ToArray()
End Function
```

Private Sub AccessingImageExample2(ByVal imagePaths As String()) 'example demonstrates using the bitmaps then disposing (better for lots of images, saves memory)

```
If Not (IsNothing(imagePaths)) Then
    If (imagePaths.Length > 0) Then
        For I As Integer = 0 To imagePaths.Length - 1
            If (My.Computer.FileSystem.FileExists(imagePaths(I))) Then
                Using iBitmap = New Bitmap(imagePaths(I))
                    'use iBitmap for anything
                    iBitmap.Dispose()
                End Using
            End If
        Next
    End If
End If
End Sub
```

Private Sub DisposeBitmaps(ByVal bitmaps As Bitmap()) 'To be used in example 1, to mass dispose, but its better to dispose as you use.

```
If Not (IsNothing(bitmaps)) Then
    If (bitmaps.Length > 0) Then
        For I As Integer = 0 To bitmaps.Length - 1
```

```

        bitmaps(l).Dispose()
    Next
End If
End If
End Sub

```

I don't want to dispose of the images as they are used, because they might be used again. If I can store the file paths in an array, that would probably be better as the engine loads images from a path, but this can of course be modified. The amount of images will range anywhere from 10 to maybe up to 50.

Visual Freedom - March 5, 2017, 9:30 am

I suggest you to load only a few images into the ram and do not keep all in RAM due performance and resource use.

Owl - March 5, 2017, 12:14 pm

(Mar 5, 2017 09:55 AM)Swatchi Wrote: [\[CITE\]](#) They are panorama images, so I'm loading them in a panorama viewer engine.

Here is my suggestion

```

Code var imgs = Directory.GetFiles(@"path", "*.png");

using (var first = new Bitmap(imgs[0]))
using (var final = new Bitmap(first.Width * imgs.Length, first.Height))
{
    using (var g = Graphics.FromImage(final))
        for (var i = 0; i < imgs.Length; i++)
            g.DrawImage(Image.FromFile(imgs[i]), new Point(first.Width * i, 0));

    final.Save(@"path", ImageFormat.Png);
}

```

As you can see it loads only 1 image into memory at a time (image fromfile)

SIM-1517404 - March 5, 2017, 12:22 pm

(Mar 5, 2017 01:14 PM)Orka Wrote: [\[CITE\]](#) Here is my suggestion

```

Code var imgs = Directory.GetFiles(@"path", "*.png");

using (var first = new Bitmap(imgs[0]))
using (var final = new Bitmap(first.Width * imgs.Length, first.Height))
{
    using (var g = Graphics.FromImage(final))
        for (var i = 0; i < imgs.Length; i++)
            g.DrawImage(Image.FromFile(imgs[i]), new Point(first.Width * i, 0));

    final.Save(@"path", ImageFormat.Png);
}

```

As you can see it loads only 1 image into memory at a time (image fromfile)

I don't need to draw the image, the engine does that for me, I just need to store the paths in an array so that for different situations I can change the filepath for the engine. :)

I've come to the conclusion that the best way to do this is to store the paths and not the images, although storing the paths take away the opportunity to directly access the filename, but it'll have to do :D

Do you know how to do this? I guess it's really just to get the file name of every element in a folder at this point.

SIM-1517404 - March 5, 2017, 12:50 pm

(Mar 5, 2017 01:43 PM)Orka Wrote: [\[CITE\]](#) Oh.. Anyway yes also possible
Code `Directory.GetFiles(@"path", "*.png").Select(f => Path.GetFileNameWithoutExtension(f));`
Does this grab the name of all the files ? Or one at a time ? So that I can loop through an array and assign a filename to each spot ?

SIM-1517404 - March 5, 2017, 1:03 pm

(Mar 5, 2017 02:00 PM)Orka Wrote: [\[CITE\]](#) My code will return a collection of file names. I don't understand what you mean on how to use it.

`"Directory.GetFiles(@"path", "*.png").Select(f => Path.GetFileNameWithoutExtension(f));"`

Where are the file names stored ? In an array or ?

SIM-1517404 - March 5, 2017, 1:39 pm

(Mar 5, 2017 02:35 PM)Orka Wrote: [\[CITE\]](#)

Please try for yourself..

Ah ok, thank you ! :)