

Macoratti.net C# - Monitorando a mudança em arquivos no .NET Core



Hoje veremos como monitorar a alteração em arquivos no sistema local usando a classe [FileSystemWatcher](#) do namespace [System.IO](#).

A classe [FileSystemWatcher](#) monitora as notificações de alteração do sistema de arquivos e gera eventos quando um diretório, ou um arquivo em um diretório, é alterado.

Curso C# Vídeo Aulas
Do básico ao intermediário

Por um preço justo

Podemos usar este recurso em cenários onde precisamos reagir a alterações feitas em arquivos, como por exemplo, quando um arquivo é carregado no servidor, ou quando um arquivo é armazenado em cache na memória e existe a possibilidade do cache ser invalidado se o arquivo for alterado.

O componente pode ser usado para monitorar arquivos em uma máquina local, remota ou ambiente de rede local. Para usá-lo em sua aplicação você deve declarar o namespace [System.IO](#).

Etapas para Configurar o componente da Classe [FileSystemWatcher](#):

1. Criar uma instância do componente
2. Configurar as propriedades e métodos necessários
3. Criar um manipulador para os eventos do sistema de arquivos

As principais propriedades da classe [FileSystemWatcher](#) são:

EnableRaisingEvents	O diretório só será monitorado se esta propriedade estiver definida como true.
Path	Indica a pasta que será monitorada pelo componente.
IncludeSubDirectories	Valor booleano, se for true, irá monitorar os subdiretórios
Filter	Extensão do arquivo que nosso componente irá filtrar (.txt por exemplo). Se usar valor padrão (*.*), ele irá filtrar todos os arquivos.
NotifyFilter	Atributos (ou gatilhos) que, ao serem alterados, farão com que nosso componente dispare um determinado evento que irá notificar o usuário que determinado arquivo ou diretório foi alterado.

Há vários tipos de alterações que você pode monitorar em um arquivo ou diretório. Por exemplo, você pode observar alterações em [Attributes](#), o [LastWrite](#) data e hora, ou o [Size](#) de arquivos ou diretórios. Isso é feito definindo a propriedade [NotifyFilter](#) para um dos valores de [NotifyFilters](#). Conforme abaixo:

A seguir temos os atributos para os filtros de notificação ([NotifyFilter](#)) :

Attributes	O atributo do arquivo ou diretório.
CreationTime	A hora que o arquivo ou diretório foi criado.
DirectoryName	O nome do diretório.
FileName	O nome do arquivo.
LastAccess	A data da última abertura do arquivo ou diretório.
LastWrite	Aa data da última escrita no arquivo ou diretório.
Security	As configurações de segurança do arquivo ou diretório.
Size	O tamanho do arquivo ou diretório.

Vejamos isso funcionando na prática.

Criando um projeto no VS 2017

Abra o [VS 2017 Community](#) e crie um projeto do tipo Console usando o [.NET Core](#) com o nome [NetCore_Monitor](#).

A seguir inclua o código abaixo no arquivo [Program.cs](#):

```
using System;
using System.IO;

namespace NetCore_Monitor
{
    class Program
    {
        private static FileSystemWatcher _monitorar;
```

```

public static void MonitorarArquivos(string path, string filtro)
{
    _monitorar = new FileSystemWatcher(path, filtro)
    {
        IncludeSubdirectories = true
    };

    _monitorar.Created += OnFileChanged;
    _monitorar.Changed += OnFileChanged;
    _monitorar.Deleted += OnFileChanged;
    _monitorar.Renamed += OnFileRenamed;

    _monitorar.EnableRaisingEvents = true;

    Console.WriteLine($"Monitorando arquivos e: {filtro}");
}
private static void OnFileChanged(object sender, FileSystemEventArgs e)
{
    Console.WriteLine($"O Arquivo {e.Name} {e.ChangeType}");
}
private static void OnFileRenamed(object sender, RenamedEventArgs e)
{
    Console.WriteLine($"O Arquivo {e.OldName} {e.ChangeType} para {e.Name}");
}

static void Main(string[] args)
{
    Console.WriteLine("Monitorando o sistema com : FileSystemWatcher");
    string path = @"c:\dados";
    string filtro = "*.txt";
    MonitorarArquivos(path, filtro );
    Console.ReadLine();
}
}
}

```

Neste código temos que :

Definimos uma variável do tipo `FileSystemWatcher` e a seguir criamos o método `MonitorarArquivos(path,filtro)`.

Usando o construtor de `FileSystemWatcher`, você pode fornecer o diretório que deve ser observado e também pode fornecer um critério para filtrar apenas arquivos específicos que correspondam à expressão do filtro.

Nota: Para monitorar as alterações em todos os arquivos, defina a propriedade `Filter` para uma cadeia de caracteres vazia (""), ou use caracteres curinga ("*."). E para um arquivo específico informe o nome do arquivo.

Ao definir a propriedade `IncludeSubdirectories`, definimos se vamos incluir subdiretórios no monitoramento.

A seguir definimos os eventos que desejamos monitorar e todos esses eventos são do tipo `FileSystemEventHandler` com a exceção do evento `Renamed` que é do tipo `RenomeadoEventHandler` que deriva de `FileSystemEventHandler` e oferece informações adicionais sobre o evento.

As informações recebidas com uma alteração de arquivo são do tipo `FileSystemEventArgs`. Ele contém o nome do arquivo que mudou, bem como o tipo de alteração que é uma enumeração do tipo `WatcherChangeTypes`:

```

private static void OnFileChanged (sender do objeto, FileSystemEventArgs e)
{
    Console.WriteLine ($ "Arquivo {e.Name} {e.ChangeType}");
}

```

Ao renomear o arquivo, informações adicionais são recebidas com o parâmetro `RenamedEventArgs`. este tipo deriva de `FileSystemEventArgs` e define informações adicionais sobre o nome original do arquivo:

```

private static void OnFileRenamed (sender do objeto, RenamedEventArgs e)
{
    Console.WriteLine ($ "Arquivo {e.OldName} {e.ChangeType} para {e.Name}");
}

```

Estamos definindo a pasta `c:\dados` e o filtro `*.txt` veremos a seguinte saída após realizar alterações em arquivos `.txt` desta pasta:

```

C:\Program Files\dotnet\dotnet.exe
Monitorando o sistema com : FileSystemWatcher
Monitorando arquivos *.txt em c:\dados
O Arquivo txt\curso ado net.txt Changed
O Arquivo txt\Clientes.txt Renamed para txt\Clientes_2.txt
O Arquivo txt\DIP.txt Deleted

```


Observe que fizemos uma alteração no arquivo [ado net.txt](#) , a seguir renomeamos o arquivo [Clientes.txt](#) para [Clientes_2.txt](#) e deletamos o arquivo [DIP.txt](#)

Observe o seguinte ao usar a classe [FileSystemWatcher](#) :

1 - Arquivos ocultos não são ignorados.

2 - Em alguns sistemas, [FileSystemWatcher](#) informa as alterações nos arquivos usando o formato de nome curto. Por exemplo, uma alteração "LongFileName.LongExtension" poderia ser relatada como "LongFil~.LON".

3 - Esta classe contém uma demanda de link e uma demanda de - O tamanho máximo que você pode definir para a propriedade InternalBufferSize para o monitoramento de um diretório pela rede é de 64 KB.

Pegue o projeto completo aqui:  [NetCore_Monitor.zip](#)

*Bem-aventurados os limpos de coração, porque eles verão a Deus;
Bem-aventurados os pacificadores, porque eles serão chamados filhos de Deus;
[Mateus 5:8,9](#)*

[Veja os Destaques e novidades do SUPER DVD Visual Basic \(sempre atualizado\) : clique e confira !](#)

Quer migrar para o VB .NET ?

- Veja mais sistemas completos para a plataforma .NET no [Super DVD .NET](#) , confira...
- [Curso Básico VB .NET - Vídeo Aulas](#)

Quer aprender C# ??

- Chegou o [Super DVD C#](#) com exclusivo material de suporte e vídeo aulas com curso básico sobre C#.
- [Curso C# Basico - Vídeo Aulas](#)

Quer aprender os conceitos da Programação Orientada a objetos ?

- [Curso Fundamentos da Programação Orientada a Objetos com VB .NET](#) 

Quer aprender o gerar relatórios com o ReportViewer no VS 2013 ?

- [Curso - Gerando Relatórios com o ReportViewer no VS 2013 - Vídeo Aulas](#)

Quer aprender a criar aplicações Web Dinâmicas usando a ASP .NET MVC 5 ?

- [Curso ASP .NET MVC 5 - Vídeo Aulas](#)

Gostou ?  [Compartilhe no Facebook](#)  [Compartilhe no Twitter](#)

Referências:

- [Seção VB .NET do Site Macoratti.net](#)
- [Super DVD .NET - A sua porta de entrada na plataforma .NET](#)
- [Super DVD Vídeo Aulas - Vídeo Aula sobre VB .NET, ASP .NET e C#](#)
- [Seção C# do site Macoratti.net](#)
- [Super DVD C#](#)
- [Super DVD Visual Basic](#)
- [Curso Básico VB .NET - Vídeo Aulas](#)

- [Curso C# Básico - Vídeo Aulas](#)
- [A classe FileSystemWatch - Macoratti.net](#)
- [Windows Forms : Perguntas e Respostas - Macoratti](#)

[José Carlos Macoratti](#)