C# Examples - [6] DirectoryInfo class tutorial

Here’s a **perfect, detailed prompt** for learning the **DirectoryInfo class** in C# in the same style as your previous prompts:

Prompt:

You are a **C# tutor**. Explain the **DirectoryInfo class in C#** for a developer with around **3 years of experience**. Cover the following in detail:

1. **Concepts**:

   * What is the DirectoryInfo class, its purpose, and when to use it.
   * Explain how it differs from the **Directory class** and when to prefer DirectoryInfo over Directory.
   * Real-world scenarios where DirectoryInfo is useful (like accessing directory properties, managing directories, iterating files).

2. **Must-know properties and methods**:

   * Properties: `Name`, `FullName`, `Parent`, `Exists`, `CreationTime`, `Attributes`.
   * Methods: `Create()`, `Delete()`, `GetFiles()`, `GetDirectories()`, `MoveTo()`, `Refresh()`.
   * For **each property/method**, provide a **small, simple C# code example** with output.

3. **Differences**:

   * Compare **DirectoryInfo vs Directory class**, and also mention **FileInfo vs File class** briefly in terms of:

     * Ease of use
     * Performance

* Accessing directory metadata
* Real-world usage scenarios

4. **Practical example**:

* Provide a **complete, commented C# program** demonstrating:

* Creating a DirectoryInfo object
* Accessing directory properties (`Name`, `FullName`, `CreationTime`)
* Performing operations like `Create()`, `Delete()`, `GetFiles()`, `GetDirectories()`
* Include **expected outputs** for clarity.

5. **Best practices and interview insights**:

* Show **proper exception handling**, **efficient usage**, and **key points a 3-year experienced .NET developer should know for interviews or real-world projects**.

Important: Include **small, simple, working C# code examples** and **outputs** for every property, method, and concept to make understanding easy and practical.

If you want, I can now generate a **full tutorial with examples, outputs, and a detailed comparison** of DirectoryInfo vs Directory, FileInfo vs File, and FileStream/StreamReader/StreamWriter. Do you want me to do that next?

Absolutely! Let's turn your detailed prompt into a **full, practical tutorial on the `DirectoryInfo` class in C#**, keeping it in the same style as your previous lessons. I'll include **concepts, properties, methods, code examples with outputs, comparisons, and best practices**.

C# Tutorial: DirectoryInfo Class

1. Concepts

The `DirectoryInfo` class is part of the **System.IO namespace** and provides **instance-based methods and properties** to work with directories in a file system.

- **Purpose:**
  `DirectoryInfo` is used to **access metadata** (name, creation time, attributes) and **perform operations** (create, delete, move, list contents) on directories.
- **When to use:**
  Use `DirectoryInfo` when you need to **work with a specific directory instance** and repeatedly access its properties or perform multiple operations.
- **Difference from Directory class:**
  - `Directory` is a **static class** with static methods.
  - `DirectoryInfo` is **instance-based** and caches directory information, making it more efficient for repeated operations on the same directory.
- **Real-world scenarios:**
  - Managing project folders dynamically.
  - Accessing directory metadata for logs or reports.
  - Iterating files and subdirectories for backup or cleanup tasks.

---

2. Must-know Properties and Methods

Properties

a) Name

```csharp
DirectoryInfo dir = new DirectoryInfo(@"C:\Temp");
Console.WriteLine("Directory Name: " + dir.Name);
```

Output:

```pgsql
Directory Name: Temp
```

b) FullName

```csharp
Console.WriteLine("Full Path: " + dir.FullName);
```

Output:

```mathematica
Full Path: C:\Temp
```

c) Parent

```csharp
Console.WriteLine("Parent Directory: " + dir.Parent);
```

Output:

```mathematica
Parent Directory: C:\
```

d) Exists

```csharp
Console.WriteLine("Does directory exist? " + dir.Exists);
```

Output (if exists):

```graphql
Does directory exist? True
```

e) CreationTime

```csharp
Console.WriteLine("Created on: " + dir.CreationTime);
```

Output:

```nginx
```

```
Created on: 9/29/2025 9:30:15 PM
```

f) Attributes

```csharp
Console.WriteLine("Attributes: " + dir.Attributes);
```

Output:

```mathematica
Attributes: Directory
```

Methods

a) Create()

```csharp
DirectoryInfo newDir = new DirectoryInfo(@"C:\Temp\NewFolder");
newDir.Create();
Console.WriteLine("Directory created: " + newDir.Exists);
```

Output:

```yaml
Directory created: True
```

b) Delete()

```csharp
newDir.Delete();
Console.WriteLine("Directory exists after delete? " + newDir.Exists);
```

Output:

```pgsql
```

```
Directory exists after delete? False
```

c) GetFiles()

```csharp
FileInfo[] files = dir.GetFiles();
foreach (var file in files)
    Console.WriteLine(file.Name);
```

Output (example):

```
example1.txt
example2.docx
```

d) GetDirectories()

```csharp
DirectoryInfo[] subDirs = dir.GetDirectories();
foreach (var d in subDirs)
    Console.WriteLine(d.Name);
```

Output (example):

```nginx
Logs
Backups
```

e) MoveTo()

```csharp
DirectoryInfo moveDir = new DirectoryInfo(@"C:\Temp\MoveFolder");
moveDir.Create();
moveDir.MoveTo(@"C:\Temp\MovedFolder");
Console.WriteLine("Moved to: " + moveDir.FullName);
```

Output:

```
css

Moved to: C:\Temp\MovedFolder
```

f) Refresh()

```
csharp

dir.Refresh(); // Updates directory info
```

- Useful if **directory content changed externally**. No direct output, but ensures the DirectoryInfo object is up-to-date.

---

3. Differences

Feature	DirectoryInfo	Directory
Type	Instance-based	Static class
Ease of Use	Good for repeated operations on same directory	Good for single operations
Performance	Caches directory info	Always accesses filesystem
Metadata	Easy to access properties like CreationTime, Attributes	Must use separate static methods
Real-world Use	Managing project folders, iterating directories	Quick checks, one-time folder operations

Similar comparison for FileInfo vs File:

- `FileInfo` = instance-based, caches metadata, repeated operations.
- `File` = static, quick one-time operations.

---

4. Practical Example

```
csharp
```

```
using System;
using System.IO;

class Program
{
    static void Main()
    {
        try
        {
            // 1. Create DirectoryInfo object
            DirectoryInfo dir = new DirectoryInfo(@"C:\Temp\Demo");

            // 2. Create directory if it doesn't exist
            if (!dir.Exists)
            {
                dir.Create();
                Console.WriteLine("Directory Created: " + dir.FullName);
            }

            // 3. Access directory properties
            Console.WriteLine("Name: " + dir.Name);
            Console.WriteLine("Full Path: " + dir.FullName);
            Console.WriteLine("Parent: " + dir.Parent);
            Console.WriteLine("Created On: " + dir.CreationTime);
            Console.WriteLine("Attributes: " + dir.Attributes);

            // 4. Create a subdirectory
            DirectoryInfo subDir = dir.CreateSubdirectory("SubFolder");
            Console.WriteLine("Subdirectory Created: " + subDir.FullName);

            // 5. List all directories
            Console.WriteLine("Subdirectories:");
            foreach (var d in dir.GetDirectories())
                Console.WriteLine(d.Name);

            // 6. List all files
            Console.WriteLine("Files:");
            foreach (var f in dir.GetFiles())
                Console.WriteLine(f.Name);

            // 7. Move directory
            DirectoryInfo moveDir = new DirectoryInfo(@"C:\Temp\DemoMove");
```

```
            dir.MoveTo(moveDir.FullName);
            Console.WriteLine("Directory moved to: " + moveDir.FullName);

            // 8. Delete moved directory
            moveDir.Delete(true);
            Console.WriteLine("Directory deleted successfully.");
        }
        catch (Exception ex)
        {
            Console.WriteLine("Error: " + ex.Message);
        }
    }
}
```

Expected Output (example):

```
mathematica

Directory Created: C:\Temp\Demo
Name: Demo
Full Path: C:\Temp\Demo
Parent: C:\Temp
Created On: 9/29/2025 9:45:00 PM
Attributes: Directory
Subdirectory Created: C:\Temp\Demo\SubFolder
Subdirectories:
SubFolder
Files:
Directory moved to: C:\Temp\DemoMove
Directory deleted successfully.
```

---

5. Best Practices & Interview Insights

- **Exception Handling:** Always wrap file/directory operations in `try-catch` to handle `IOException`, `UnauthorizedAccessException`.
- **Efficient Usage:** Use `DirectoryInfo` for repeated access to the same directory.
- **Refreshing Data:** Call `Refresh()` if directory content changes outside the program.
- **Interview Tips:** Know the difference between static (`Directory`) and instance-based (`DirectoryInfo`) classes. Understand `FileInfo vs File` similarly.

- **Real-world Advice:** Combine `DirectoryInfo` with `FileInfo` to manipulate both directories and files efficiently.

✅ This tutorial covers **concepts, properties, methods, examples with outputs, comparisons, and best practices** for a 3-year experienced .NET developer.

If you want, I can next create a **complete "FileSystem Mastery" guide** that compares **FileStream, StreamReader/Writer, File/FileInfo, Directory/DirectoryInfo** all in one **cohesive practical reference**.

Do you want me to create that next?