ARBEITEN MIT DATEIEN

Vincent Uhlmann
IT-Akademie Dr. Heuer GmbH

DATEIOPERATIONEN IN C#

- In C# bietet der System.IO Namespace Klassen und Methoden für den Umgang mit Dateien und Ordnern
- Enthält Typen, die das Lesen und Schreiben in Dateien ermöglichen, sowie Typen, die grundlegende Datei- und Verzeichnisunterstützung bieten

- Klassen, die derzeit für uns von Bedeutung sind
- Path
- File
- Directory

PATH KLASSE

- Die Path-Klasse ist eine Klasse im System.IO-Namespace, die Methoden zur Manipulation von Dateipfaden bietet
- Path.Combine(string, string): Kombiniert mehrere Zeichenfolgen zu einem vollständigen Dateipfad
- Path.GetDirectoryName(string): Gibt den Verzeichnispfad eines Dateipfads zurück
- Path.GetFileName(string): Gibt den Dateinamen (mit Erweiterung) aus einem Pfad zurück
- Path.GetExtension(string): Gibt die Dateierweiterung (z.B. .txt, .jpg) zurück
- ...

PATH KLASSE

```
string path = "C:\\Users\\Vince\\Desktop";
path = Path.Combine(path, "test1.txt");

string directory = Path.GetDirectoryName(path);
Console.WriteLine(directory);    // C:\Users\Vince\Desktop

string filename = Path.GetFileName(path);
Console.WriteLine(filename);     // test1.txt

string extension = Path.GetExtension(path);
Console.WriteLine(extension);    // .txt
```

FILE KLASSE

- Die File-Klasse bietet Methoden, um grundlegende Dateioperationen wie Lesen, Schreiben, Kopieren, Löschen und Überprüfen der Existenz von Dateien auszuführen
- File.WriteAllText(path, content): Schreibt Text in eine Datei
- File.WriteAllLines(path, string[] lines): Schreibt mehrere Zeilen in eine Datei
- File.AppendAllText(path, content): Fügt Text am Ende einer Datei hinzu
- File.AppendAllLines(path, string[] lines): Fügt mehrere Zeilen am Ende einer Datei hinzu
- File.ReadAllText(path): Liest den gesamten Textinhalt einer Datei als String
- File.ReadAllLines(path): Liest alle Zeilen einer Datei und gibt sie als String-Array zurück

FILE KLASSE

- File.Exists(path): Überprüft, ob eine Datei existiert
- File.Copy(sourcePath, destinationPath): Kopiert eine Datei von einem Ort zum anderen
- File.Move(sourcePath, destinationPath): Verschiebt eine Datei
- File.Delete(path): Löscht eine Datei

FILE KLASSE

```
string basePath = "C:\\Users\\Vince\\Desktop";
string path1 = Path.Combine(basePath, "test1.txt");
string path2 = Path.Combine(basePath, "test2.txt");

File.WriteAllText(path1, "Hello, World!");
File.WriteAllLines(path2, new string[] { "Hello, World!" });
File.AppendAllText(path1, "Another line!");
File.AppendAllLines(path2, new string[] { "Another line!" });
```

FILE KLASSE

```
string text = File.ReadAllText(path1);
Console.WriteLine(text);      // Hello, World!Another line!

string[] lines = File.ReadAllLines(path2);
Console.WriteLine(lines[0]); // Hello, World!

bool exists = File.Exists(path1);
Console.WriteLine(exists);   // True

File.Copy(path1, Path.Combine(basePath, "test1-copy.txt"));
File.Move(path1, Path.Combine(basePath, "test1-moved.txt"));
File.Delete(path1);
```

DIRECTORY KLASSE

- Die Directory-Klasse bietet Methoden, um mit Verzeichnissen zu arbeiten, z.B. Verzeichnisse erstellen, löschen und überprüfen, ob ein Verzeichnis existiert
- Directory.CreateDirectory(path): Erstellt ein Verzeichnis (einschließlich untergeordneter Verzeichnisse, falls nötig)
- Directory.Exists(path): Überprüft, ob ein Verzeichnis existiert
- Directory.Delete(path, recursive): Löscht ein Verzeichnis und optional alle darin enthaltenen Dateien und Unterverzeichnisse
- Directory.GetFiles(path): Gibt ein Array von Dateinamen (Pfaden) im Verzeichnis zurück
- Directory.GetDirectories(path): Gibt ein Array von Verzeichnispfaden im Verzeichnis zurück

DIRECTORY KLASSE

```
string basePath = "C:\\Users\\Vince\\Desktop";
string dirPath = Path.Combine(basePath, "TestDirectory");

Directory.CreateDirectory(dirPath);

bool dirExists = Directory.Exists(dirPath);
Console.WriteLine(dirExists); // True

string[] files = Directory.GetFiles(dirPath);
foreach (string file in files)
{
    Console.WriteLine(file);
}
```

DIRECTORY KLASSE

```
string[] dirs = Directory.GetDirectories(dirPath);
foreach (string dir in dirs)
{
    Console.WriteLine(dir);
}

Directory.Delete(dirPath, true); // true = rekursives Löschen
```

ENVIRONMENT KLASSE

- Die Klasse Environment bietet Informationen über das aktuelle System und die Umgebung des Programms
- Sie enthält statische Methoden und Eigenschaften, um auf Umgebungsvariablen, Betriebssysteminformationen und Standardverzeichnisse zuzugreifen
- Die Methode GetFolderPath() ermöglicht den Zugriff auf Standardverzeichnisse, die von Windows definiert sind (z.B. Desktop, Dokumente, AppData)

DIRECTORY KLASSE

```
string desktopPath = Environment.GetFolderPath(Environment.SpecialFolder.Desktop);
Console.WriteLine(desktopPath);          // C:\Users\Vince\Desktop

string documentsPath = Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments);
Console.WriteLine(documentsPath);        // C:\Users\Vince\Documents

string appDataPath = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData);
Console.WriteLine(appDataPath);          // C:\Users\Vince\AppData\Roaming

string programFilesPath = Environment.GetFolderPath(Environment.SpecialFolder.ProgramFiles);
Console.WriteLine(programFilesPath);     // C:\Program Files
```