

# Klassen

Verschiedene vordefinierte Klassen in PowerShell und .NET im Namespace `System`

- [Guid](#)
- [Math](#)
- [IO](#)
  - [DirectoryInfo](#)
- [CIM/WIM-Klasse](#)
  - [Win32\\_CDRomDrive](#)

# Guid

Die Klasse `Guid` dient zum Erstellen und Verwalten von **Global Unique Identifiers (GUIDs)**.

Eine GUID ist eine 128-Bit lange Kennung, die mit extrem hoher Wahrscheinlichkeit weltweit eindeutig ist. Sie wird häufig verwendet, um Objekte, Dateien, Datenbankeinträge oder Sitzungen eindeutig zu identifizieren.

## Grundlagen

Im Gegensatz zu Klassen wie `Button` oder `Form` besitzt `Guid` keine grafische Darstellung.

Sie dient ausschließlich dazu, eindeutige Kennungen zu erzeugen und zu verarbeiten.

- `Guid` → erzeugt eindeutige Kennungen
- `Math` → führt mathematische Berechnungen aus
- `DateTime` → verarbeitet Datum und Uhrzeit
- `Random` → erzeugt Zufallszahlen

## GUID erzeugen

```
# Neue GUID erzeugen
$guid = [Guid]::NewGuid()

$guid
```

Ausgabe:

```
0f8fad5b-d9cb-469f-a165-70867728950e
```

## GUID aus einem String erstellen

```
$guid = [Guid]::Parse(
    "0f8fad5b-d9cb-469f-a165-70867728950e"
```

)

# Leere GUID

```
[Guid]::Empty
```

Ausgabe:

```
00000000-0000-0000-0000-000000000000
```

Diese GUID besitzt ausschließlich Nullen und dient häufig als Standard- oder Platzhalterwert.

## Eigenschaften

| Eigenschaft        | Beschreibung                                                             |
|--------------------|--------------------------------------------------------------------------|
| <code>Empty</code> | Enthält eine GUID, deren sämtliche Bits auf <code>0</code> gesetzt sind. |

### Empty

## Empty

**Typ** = `System.Guid`

`Empty` stellt eine leere GUID dar.

Sie wird häufig verwendet, um zu prüfen, ob einer Variablen bereits eine gültige GUID zugewiesen wurde.

```
if ($id -eq [Guid]::Empty) {  
    "Keine GUID vorhanden."  
}
```

# Methoden

| Methode                  | Beschreibung                                    |
|--------------------------|-------------------------------------------------|
| <code>NewGuid()</code>   | Erstellt eine neue eindeutige GUID.             |
| <code>Parse()</code>     | Erstellt eine GUID aus einem String.            |
| <code>TryParse()</code>  | Versucht einen String in eine GUID umzuwandeln. |
| <code>ToString()</code>  | Gibt die GUID als Zeichenkette zurück.          |
| <code>Equals()</code>    | Vergleicht zwei GUIDs.                          |
| <code>CompareTo()</code> | Vergleicht zwei GUIDs für Sortierungen.         |

## NewGuid()

# NewGuid()

```
[Guid]::NewGuid()
```

## Beschreibung

Erzeugt eine neue zufällige GUID.

Jeder Aufruf liefert eine neue Kennung, die mit extrem hoher Wahrscheinlichkeit eindeutig ist.

## Rückgabe

**Rückgabety**`p` `System.Guid`

## Beispiel

```
$id = [Guid]::NewGuid()
```

```
$id # fd5d1b64-ac86-46b5-a511-92c68ce09d2b
```

## Parse()

# Parse()

```
[Guid]::Parse($string)
```

## Beschreibung

Konvertiert einen String in ein `Guid`-Objekt.

Ist der String ungültig, wird eine Ausnahme ausgelöst.

## Rückgabe

Rückgabetyp `System.Guid`

## Beispiel

```
$id = [Guid]::Parse(  
    "0f8fad5b-d9cb-469f-a165-70867728950e"  
)
```

## TryParse()

# TryParse()

```
[Guid]::TryParse($string, [ref]$guid)
```

## Beschreibung

Versucht einen String in eine GUID umzuwandeln.

Im Gegensatz zu `Parse()` wird **keine Ausnahme** ausgelöst, wenn der String ungültig ist.

## Rückgabe

Rückgabetyp `System.Boolean`

## Beispiel

```
$guid = [Guid]::Empty

if ([Guid]::TryParse($text, [ref]$guid)) {
    "Gültige GUID"
}
else {
    "Ungültige GUID"
}
```

## ToString()

# ToString()

```
$guid.ToString()
```

## Beschreibung

Gibt die GUID als Zeichenkette zurück.

## Rückgabe

Rückgabetyp `System.String`

## Beispiel

```
$guid = [Guid]::NewGuid()

$guid.ToString()
```

## Equals()

# Equals()

```
$guid1.Equals($guid2)
```

## Beschreibung

Vergleicht zwei GUIDs auf Gleichheit.

## Rückgabe

Rückgabetyt `System.Boolean`

## Beispiel

```
$id1 = [Guid]::NewGuid()  
$id2 = $id1  
  
$id1.Equals($id2)
```

### CompareTo()

## CompareTo()

```
$guid1.CompareTo($guid2)
```

## Beschreibung

Vergleicht zwei GUIDs.

Die Methode wird hauptsächlich zum Sortieren verwendet.

## Rückgabe

Rückgabetyt `System.Int32`

- `< 0` → kleiner
- `0` → gleich
- `> 0` → größer

# Tipps & Tricks

## Neue eindeutige Dateinamen erzeugen

```
$file = "{0}.txt" -f ([Guid]::NewGuid())
```

Beispiel:

```
92d4bb75-0d95-4d9d-9d88-d7548cb9cbf4.txt
```

---

## Prüfen, ob ein String eine gültige GUID ist

```
[Guid]::TryParse($text, [ref]([Guid]::Empty))
```

# Math

Die Klasse **Math** stellt eine Sammlung mathematischer Funktionen und Konstanten bereit. Sie enthält ausschließlich **statische Methoden und Felder**, weshalb kein Objekt erstellt werden muss.

Mit `Math` lassen sich unter anderem:

- Zahlen runden
- Beträge berechnen
- Potenzen und Wurzeln bestimmen
- Minimal- und Maximalwerte ermitteln
- trigonometrische Funktionen verwenden
- mathematische Konstanten wie  $\pi$  oder  $e$  nutzen

Die Klasse gehört zum Namespace **System** und steht in PowerShell standardmäßig zur Verfügung.

## Grundlagen

| Eigenschaft   | Wert                                   |
|---------------|----------------------------------------|
| Namespace     | <code>System</code>                    |
| Assembly      | <code>System.Runtime</code>            |
| Instanzierbar | <input type="checkbox"/> Nein          |
| Statisch      | <input checked="" type="checkbox"/> Ja |
| Vererbt von   | <code>Object</code>                    |

## Klasse verwenden

Da alle Mitglieder statisch sind, erfolgt der Aufruf direkt über den Klassennamen.

```
[Math]::Sqrt(25)
```

Ausgabe

```
5
```

---

# Keine Instanziierung erforderlich

Im Gegensatz zu vielen anderen .NET-Klassen wird kein Objekt erzeugt.

## Nicht erforderlich

```
New-Object Math
```

## Stattdessen

```
[Math]::Pow(2,8)
```

---

# Rückgabewerte

Die meisten Methoden geben einen numerischen Wert zurück.

Je nach Methode erfolgt die Rückgabe beispielsweise als:

| Datentyp | Beschreibung                    |
|----------|---------------------------------|
| Double   | Gleitkommazahl                  |
| Int32    | Ganze Zahl                      |
| Decimal  | Dezimalzahl (je nach Parameter) |

---

# Konstanten

| Name | Typ    | Beschreibung    |
|------|--------|-----------------|
| PI   | Double | Kreiszahl $\pi$ |
| E    | Double | Eulersche Zahl  |

# Beispiel

```
[Math]::PI  
[Math]::E
```

3,14159265358979

2,71828182845905

# Methoden

## Übersicht

| Methode    | Beschreibung                             |
|------------|------------------------------------------|
| Abs()      | Berechnet den Betrag einer Zahl          |
| Ceiling()  | Rundet auf die nächstgrößere ganze Zahl  |
| Clamp()    | Begrenzt einen Wert auf einen Bereich    |
| Floor()    | Rundet auf die nächstkleinere ganze Zahl |
| Max()      | Gibt den größeren Wert zurück            |
| Min()      | Gibt den kleineren Wert zurück           |
| Pow()      | Berechnet Potenzen                       |
| Round()    | Rundet Zahlen                            |
| Sign()     | Bestimmt das Vorzeichen                  |
| Sqrt()     | Berechnet Quadratwurzeln                 |
| Truncate() | Entfernt Nachkommastellen                |

### Abs()

## Abs()

Berechnet den Betrag einer Zahl

[Math]::Abs(-15)

Ausgabe

15

# Clamp()

Die Methode **Clamp()** begrenzt einen Wert auf einen angegebenen Minimal- und Maximalwert. Liegt der Wert innerhalb des Bereichs, wird er unverändert zurückgegeben.

Diese Methode ist ab **.NET Core 2.0** bzw. **.NET Standard 2.1** verfügbar. Sie steht daher **nicht** in **Windows PowerShell 5.1 (.NET Framework)** zur Verfügung, sondern erst ab **PowerShell Core 6.0**.

## Syntax

```
[Math]::Clamp(<Wert>, <Minimum>, <Maximum>)
```

## Parameter

| Parameter | Typ       | Beschreibung             |
|-----------|-----------|--------------------------|
| Wert      | numerisch | Zu prüfender Wert        |
| Minimum   | numerisch | Kleinster erlaubter Wert |
| Maximum   | numerisch | Größter erlaubter Wert   |

## Rückgabewert

Gleicher Datentyp wie die Eingabewerte.

## Beispiel

```
[Math]::Clamp(120,0,100)
[Math]::Clamp(-5,0,100)
[Math]::Clamp(50,0,100)
```

## Ausgabe

```
100
0
50
```

## Ceiling()

# Ceiling()

Die Methode **Ceiling()** rundet eine Zahl **immer auf die nächstgrößere ganze Zahl auf**. Besitzt die Zahl bereits keinen Nachkommateil, bleibt sie unverändert.

## Syntax

```
[Math]::Ceiling(<Zahl>)
```

## Parameter

| Parameter | Typ              | Beschreibung            |
|-----------|------------------|-------------------------|
| Zahl      | Double / Decimal | Die aufzurundende Zahl. |

## Rückgabewert

**Double** bzw. **Decimal**

Die aufgerundete ganze Zahl.

## Beispiel

```
[Math]::Ceiling(4.2)
[Math]::Ceiling(-4.2)
```

Ausgabe

```
5
-4
```

## Floor()

# Floor()

Die Methode **Floor()** rundet eine Zahl **immer auf die nächstkleinere ganze Zahl ab**.

## Syntax

```
[Math]::Floor(<Zahl>)
```

## Parameter

| Parameter | Typ              | Beschreibung           |
|-----------|------------------|------------------------|
| Zahl      | Double / Decimal | Die abzurundende Zahl. |

## Rückgabewert

**Double** bzw. **Decimal**

Die abgerundete ganze Zahl.

## Beispiel

```
[Math]::Floor(4.9)
[Math]::Floor(-4.2)
```

Ausgabe

```
4
-5
```

## Min()

# Min()

Die Methode **Min()** gibt den kleineren von zwei Werten zurück.

## Syntax

```
[Math]::Min(<Wert1>, <Wert2>)
```

## Parameter

| Parameter | Typ | Beschreibung |
|-----------|-----|--------------|
|-----------|-----|--------------|

|       |           |                        |
|-------|-----------|------------------------|
| Wert1 | numerisch | Erster Vergleichswert  |
| Wert2 | numerisch | Zweiter Vergleichswert |

## Rückgabewert

Gleicher Datentyp wie die Eingabewerte.

## Beispiel

```
[Math]::Min(12,8)
```

Ausgabe

```
8
```

### Max()

## Max()

Den größeren Wert bestimmen

```
[Math]::Max(15,42)
```

Ausgabe

```
42
```

### Pow()

## Pow()

Berechnet Potenzen

```
[Math]::Pow(5,3)
```

Ausgabe

125

## Round()

# Round()

Rundet Zahlen

```
[Math]::Round(3.14159,2)
```

Ausgabe

```
3.14
```

## Sign()

# Sign()

## Beschreibung

Die Methode **Sign()** bestimmt das Vorzeichen einer Zahl.

## Syntax

```
[Math]::Sign(<Zahl>)
```

## Parameter

| Parameter | Typ       | Beschreibung     |
|-----------|-----------|------------------|
| Zahl      | numerisch | Zu prüfende Zahl |

## Rückgabewert

**Rückgabotyp:** `System.Int32`

| Rückgabe | Bedeutung |
|----------|-----------|
|----------|-----------|

|    |                  |
|----|------------------|
| -1 | Zahl ist negativ |
| 0  | Zahl ist 0       |
| 1  | Zahl ist positiv |

## Beispiel

```
[Math]::Sign(-20)
```

```
[Math]::Sign(0)
```

```
[Math]::Sign(20)
```

### Ausgabe

```
-1
```

```
0
```

```
1
```

## Sqrt()

# Sqrt()

Berechnet Quadratwurzeln

```
[Math]::Sqrt(64)
```

### Ausgabe

```
8
```

## Truncate()

# Truncate()

Die Methode **Truncate()** entfernt den Nachkommateil einer Zahl, ohne zu runden.

## Syntax

```
[Math]::Truncate(<Zahl>)
```

## Parameter

| Parameter | Typ              | Beschreibung     |
|-----------|------------------|------------------|
| Zahl      | Double / Decimal | Zu kürzende Zahl |

## Rückgabewert

**Double** bzw. **Decimal**

Die Zahl ohne Nachkommastellen.

## Beispiel

```
[Math]::Truncate(4.9)  
[Math]::Truncate(-4.9)
```

Ausgabe

```
4  
-4
```

# Weitere Details

## Häufige Einsatzgebiete

Die Klasse `Math` wird häufig verwendet für:

- mathematische Berechnungen
- Koordinaten- und Geometrieberechnungen
- Diagramme und Visualisierungen
- Spieleentwicklung
- Größen- und Positionsberechnungen
- statistische Auswertungen
- wissenschaftliche Berechnungen

# Hinweise

- Alle Methoden sind **statisch**.
  - Die Klasse kann nicht instanziiert werden.
  - Für viele Methoden existieren mehrere Überladungen, die unterschiedliche numerische Datentypen unterstützen (`Int32`, `Double`, `Decimal` usw.).
  - Die meisten Berechnungen arbeiten mit dem Datentyp `Double`, wodurch Rundungsungenauigkeiten bei Gleitkommazahlen auftreten können.
- 

## Siehe auch

- `System.Random`
- `System.Convert`
- `System.Decimal`
- `System.Numerics`

10

IO

# DirectoryInfo

Beinhaltet Informationen über einen Ordner

# CIM/WIM-Klasse

# Win32\_CDROMDrive

`Win32_CDROMDrive` ist eine WMI/CIM-Klasse, die Windows nach vorhandenen optischen Laufwerken abfragt.

## Syntax

```
Get-CimInstance Win32_CDROMDrive
```

### Beispiel

| Caption            | Drive | Manufacturer                | VolumeName |
|--------------------|-------|-----------------------------|------------|
| -----              | ----- | -----                       | -----      |
| ASUS DRW-24D5MT D: |       | (Standard-CD-ROM-Laufwerke) |            |

| Caption                | Drive              | Manufacturer | VolumeName                        |
|------------------------|--------------------|--------------|-----------------------------------|
| Bezeichnung des Geräts | Laufwerksbuchstabe | Hersteller   | Name des eingelegten Datenträgers |

## Eigenschaften

### CapabilityDescriptions

Optionen des Laufwerks

### Caption

Modellbezeichnung

### Drive

Laufwerksbuchstabe

# Manufacturer

Name des Herstellers, wenn keins vorhanden ist, wird in Klammern der generische Windows-Treiber genannt

# Media Loaded

Angabe ob ein Medium eingelegt ist oder nicht

# MediaType

Der Typ vom Laufwerk

| MediaType  | Bedeutung                                     |
|------------|-----------------------------------------------|
| Unknown    | Windows kann den Laufwerkstyp nicht bestimmen |
| CD-ROM     | CD-Laufwerk, nur Lesen                        |
| CD-R       | CD-R-Laufwerk                                 |
| CD-RW      | CD-RW-fähiges Laufwerk                        |
| DVD-ROM    | DVD-Laufwerk, nur Lesen                       |
| DVD-R      | DVD-R-fähiges Laufwerk                        |
| DVD-RAM    | DVD-RAM-fähiges Laufwerk                      |
| DVD-RW     | DVD-RW-fähiges Laufwerk                       |
| DVD+R      | DVD+R-fähiges Laufwerk                        |
| DVD+RW     | DVD+RW-fähiges Laufwerk                       |
| DVD Writer | DVD-Brenner                                   |
| HD-DVD-ROM | HD-DVD-Laufwerk                               |
| HD-DVD-R   | HD-DVD-Brenner                                |
| HD-DVD-RW  | HD-DVD-RW-fähig                               |
| BD-ROM     | Blu-ray-Laufwerk, nur Lesen                   |
| BD-R       | Blu-ray-Brenner für BD-R                      |
| BD-RE      | Blu-ray-Brenner für BD-RE                     |

# Name

Modell-/Gerätename

# PNPDeviceID

Hardwareerkennung

## VolumeName

Name des eingelegten Datenträgers