

TableLayoutPanel

Ein `TableLayoutPanel` ist ein Layout-Container, der seine enthaltenen Controls in einem **Raster aus Zeilen und Spalten** anordnet.

Im Gegensatz zu einem normalen `Panel` werden Controls nicht über ihre `Location` positioniert, sondern einer bestimmten Zelle innerhalb des Rasters zugewiesen. Das `TableLayoutPanel` übernimmt anschließend automatisch die Positionierung und Größenanpassung aller enthaltenen Controls.

Grundlagen

Ein `TableLayoutPanel` organisiert Controls in einem **Tabellenlayout**.

- `Panel` → freie Positionierung über `Location`
- `FlowLayoutPanel` → automatische Anordnung hintereinander
- `TableLayoutPanel` → Anordnung in Zeilen und Spalten

TableLayoutPanel erstellen

```
# Klassisch
$table = New-Object System.Windows.Forms.TableLayoutPanel

# .NET-Style
$table = [System.Windows.Forms.TableLayoutPanel]::new()
```

Controls hinzufügen

```
$table.Controls.Add($button)
```

oder direkt in eine bestimmte Zelle

```
$table.Controls.Add($button, 1, 0) # Spalte 1, Zeile 0
```

Zeilen und Spalten festlegen

```
$table.ColumnCount = 2
$table.RowCount = 3
```

Eigenschaften

Eigenschaft	Beschreibung
ColumnCount	Anzahl der Spalten.
RowCount	Anzahl der Zeilen.
ColumnStyles	Definiert Breite jeder Spalte.
RowStyles	Definiert Höhe jeder Zeile.
GrowStyle	Legt fest, wie neue Zeilen oder Spalten entstehen.
CellBorderStyle	Zeichnet Rahmen zwischen den Zellen.
Dock	Dockt das Panel an den Parent an.
Anchor	Verankert das Panel am Parent.
AutoSize	Passt die Größe automatisch an.
AutoScroll	Aktiviert Scrollleisten.
BackColor	Hintergrundfarbe.
Padding	Innenabstand.
Margin	Außenabstand.
Name	Interner Name.
Location	Position des Panels.
Size	Größe des Panels.
Visible	Sichtbarkeit.
Enabled	Aktiviert bzw. deaktiviert das Panel.

AutoScroll

AutoScroll

Typ	System.Boolean
Standardwert	False

Die Eigenschaft `AutoScroll` legt fest, ob das `TableLayoutPanel` automatisch Scrollleisten anzeigt, wenn der Inhalt größer ist als der sichtbare Bereich.

Ist `AutoScroll` auf `True` gesetzt, erscheinen horizontale und/oder vertikale Scrollleisten nur dann, wenn sie tatsächlich benötigt werden. Dadurch bleiben auch Controls erreichbar, die sich außerhalb des aktuell sichtbaren Bereichs befinden.

Beispiel:

```
$table.AutoScroll = $true
```

Sind mehr Controls vorhanden, als im verfügbaren Bereich angezeigt werden können, fügt das `TableLayoutPanel` automatisch die erforderlichen Scrollleisten hinzu.

Hinweis:

- `AutoScroll` ist standardmäßig deaktiviert (`False`).
- Scrollleisten werden nur angezeigt, wenn der Inhalt die sichtbare Größe des `TableLayoutPanel` überschreitet.
- Besonders nützlich bei dynamischen Benutzeroberflächen, deren Anzahl an Controls zur Laufzeit variieren kann.

AutoSize

AutoSize

Typ	<code>System.Boolean</code>
Standardwert	<code>False</code>

Passt die Größe automatisch an den Inhalt an.

Beispiel:

```
$table.AutoSize = $true
```

CellBorderStyle

CellBorderStyle

Typ	System.Windows.Forms.TableLayoutPanelCellBorderStyle
Standardwert	None

Legt fest, ob zwischen den Zellen Rahmen gezeichnet werden.

Mögliche Werte

- None
- Single
- Inset
- Outset
- InsetDouble
- OutsetDouble

Beispiel:

```
$table.CellBorderStyle = "Single"
```

ColumnCount

ColumnCount

Typ	System.Int32
Standardwert	0

Legt fest, aus wie vielen Spalten das Layout besteht.

Beispiel:

```
$table.ColumnCount = 3
```

ColumnStyles

ColumnStyles

Typ	System.Windows.Forms.TableLayoutPanelColumnStyleCollection
Standardwert	leere Sammlung

Bestimmt die Breite jeder einzelnen Spalte.

Es gibt drei verschiedene Größenarten:

Größe	Beschreibung
<code>Absolute</code>	Feste Pixelgröße
<code>Percent</code>	Prozentuale Verteilung
<code>AutoSize</code>	Größe richtet sich nach dem Inhalt

Beispiel:

```
$table.ColumnStyles.Add(  
    [System.Windows.Forms.ColumnStyle]::new("Percent",50)  
)  
  
$table.ColumnStyles.Add(  
    [System.Windows.Forms.ColumnStyle]::new("Percent",50)  
)
```

Dock

Dock

Typ	<code>System.Windows.Forms.DockStyle</code>
Standardwert	<code>None</code>

Legt fest, wie das TableLayoutPanel innerhalb seines Parent-Containers andockt wird.

Beispiel:

```
$table.Dock = "Fill"
```

Dies ist die häufigste Einstellung.

GrowStyle

GrowStyle

Typ	<code>System.Windows.Forms.TableLayoutPanelGrowStyle</code>
Standardwert	<code>AddRows</code>

Legt fest, wie das Panel reagiert, wenn mehr Controls hinzugefügt werden als Zellen vorhanden sind.

Mögliche Werte

- `AddRows`
- `AddColumns`
- `FixedSize`

Beispiel:

```
$table.GrowStyle = "AddRows"
```

Bei `FixedSize` wird eine Ausnahme ausgelöst, wenn kein Platz mehr vorhanden ist.

RowCount

RowCount

Typ	<code>System.Int32</code>
Standardwert	<code>0</code>

Legt die Anzahl der Zeilen fest, die das `TableLayoutPanel` enthält.

Zusammen mit `ColumnCount` bestimmt diese Eigenschaft die Größe des Tabellenrasters. Jede Zeile kann anschließend über die Eigenschaft `RowStyles` individuell konfiguriert werden.

Beispiel:

```
$table.RowCount = 3
```

Das `TableLayoutPanel` besitzt nun drei Zeilen.

Hinweis:

- `RowCount` legt lediglich die Anzahl der Zeilen fest. Die Höhe der einzelnen Zeilen wird über `RowStyles` bestimmt.
- Zusammen mit `ColumnCount` ergibt sich die Gesamtzahl der verfügbaren Zellen.
- Wird `GrowStyle` auf `AddRows` gesetzt, kann das `TableLayoutPanel` bei Bedarf automatisch weitere Zeilen hinzufügen.

RowStyles

RowStyles

Typ	<code>System.Windows.Forms.TableLayoutPanelRowStyleCollection</code>
Standardwert	leere Sammlung

Bestimmt die Höhe jeder Zeile.

Auch hier stehen

- `Absolute`
- `Percent`
- `AutoSize`

zur Verfügung.

Beispiel:

```
$table.RowStyles.Add(  
    [System.Windows.Forms.RowStyle]::new("AutoSize")  
)
```

Methoden

Methode	Beschreibung
<code>GetControlFromPosition()</code>	Liefert das Control einer bestimmten Zelle.
<code>GetPositionFromControl()</code>	Liefert die Position eines Controls.
<code>GetColumn()</code>	Liefert die Spalte eines Controls.
<code>GetRow()</code>	Liefert die Zeile eines Controls.

Methode	Beschreibung
<code>SetColumn()</code>	Verschiebt ein Control in eine andere Spalte.
<code>SetRow()</code>	Verschiebt ein Control in eine andere Zeile.
<code>SetColumnSpan()</code>	Lässt ein Control mehrere Spalten belegen.
<code>SetRowSpan()</code>	Lässt ein Control mehrere Zeilen belegen.

GetControlFromPosition()

GetControlFromPosition()

```
$table.GetControlFromPosition(1,0)
```

Beschreibung:

Liefert das Control zurück, das sich in der angegebenen Spalte und Zeile befindet.

Rückgabe

Rückgabetyyp `System.Windows.Forms.Control`

GetPositionFromControl()

GetPositionFromControl()

```
$table.GetPositionFromControl($button)
```

Beschreibung:

Ermittelt die aktuelle Position eines Controls innerhalb des Rasters.

Rückgabe:

Rückgabetyyp `System.Windows.Forms.TableLayoutPanelCellPosition`

SetColumn()

SetColumn()

```
$table.SetColumn($button,2)
```

Verschiebt ein Control in eine andere Spalte.

SetRow()

SetRow()

```
$table.SetRow($button,1)
```

Verschiebt ein Control in eine andere Zeile.

SetColumnSpan()

SetColumnSpan()

```
$table.SetColumnSpan($button,2)
```

Das Control erstreckt sich über mehrere Spalten.

SetRowSpan()

SetRowSpan()

```
$table.SetRowSpan($button,3)
```

Das Control erstreckt sich über mehrere Zeilen.

RowStyles

Insert

Insert()

```
$sizeType = [System.Windows.Forms.SizeType]::Absolute

$rowStyle = [System.Windows.Forms.RowStyle]::new( $sizeType, 50 )

$table.Rows.Insert(0, $rowStyle )
```

Die Methode `Insert()` fügt an der angegebenen Position einen neuen `RowStyle` in die `RowStyles`-Sammlung ein.

Alle vorhandenen Einträge ab diesem Index werden dabei um eine Position nach hinten verschoben. Dadurch kann die Reihenfolge der Zeilenstile geändert oder zwischen bestehenden Zeilen ein neuer Stil eingefügt werden.

Die Methode verändert lediglich die `RowStyles`-Sammlung. Damit der neue Eintrag einer Zeile zugeordnet werden kann, sollte die Anzahl der Einträge in der Regel mit `RowCount` übereinstimmen.

Die Methode `Insert()` erhöht **nicht** automatisch den Wert der Eigenschaft `RowCount`. Sie fügt lediglich einen neuen `RowStyle` in die `RowStyles`-Sammlung ein. Soll der eingefügte `RowStyle` einer zusätzlichen Zeile zugeordnet werden, muss `RowCount` entsprechend erhöht werden.

Rückgabe:

Kein Rückgabewert.

Rückgabetyt `System.Void`

Beispiel:

```
$table.RowCount = 3

$firstSizeType = [System.Windows.Forms.SizeType]::Absolute
$firstRowStyle = [System.Windows.Forms.RowStyle]::new( $firstSizeType, 40 )
$table.Rows.Add( $firstRowStyle )

$secondSizeType = [System.Windows.Forms.SizeType]::Percent
$secondRowStyle = [System.Windows.Forms.RowStyle]::new( $secondSizeType, 100 )
$table.Rows.Add( $secondRowStyle )

# Neuen RowStyle an Position 1 einfügen
```

```
$thirdSizeType = [System.Windows.Forms.SizeType]::Absolute
$thirdRowStyle = [System.Windows.Forms.RowStyle]::new( $thirdSizeType, 20 )
$table.RowStyles.Insert( 1, $thirdRowStyle )
```

Der neue `RowStyle` wird an Index `1` eingefügt. Der ursprünglich zweite Eintrag verschiebt sich automatisch an die nächste Position.

Events

Event	Beschreibung
<code>Layout</code>	Wird ausgelöst, wenn das Layout neu berechnet wird.
<code>ControlAdded</code>	Ein Control wurde hinzugefügt.
<code>ControlRemoved</code>	Ein Control wurde entfernt.

Layout

Layout

Das Event wird ausgelöst, wenn das `TableLayoutPanel` seine Controls neu anordnet.

Dies geschieht beispielsweise bei

- Größenänderungen
- Änderungen an `ColumnStyles`
- Änderungen an `RowStyles`
- Hinzufügen oder Entfernen von Controls

```
$table.Add_Layout({
    Write-Host "Layout aktualisiert"
})
```

ControlAdded

ControlAdded

```
$table.Add_ControlAdded({  
    param($sender,$e)  
  
    Write-Host $e.Control.Name  
})
```

Wird ausgelöst, sobald ein Control hinzugefügt wird.

ControlRemoved

ControlRemoved

```
$table.Add_ControlRemoved({  
    param($sender,$e)  
  
    Write-Host $e.Control.Name  
})
```

Wird ausgelöst, sobald ein Control entfernt wird.

Tipps & Tricks

Controls direkt einer Zelle hinzufügen

```
$table.Controls.Add($button,0,1)
```

Dadurch entfällt ein späterer Aufruf von `SetColumn()` und `SetRow()`.

Control über mehrere Spalten strecken

```
$table.SetColumnSpan($textBox,2)
```

Dies wird häufig für Überschriften oder TextBoxen verwendet.

Gesamten verfügbaren Platz ausfüllen

```
$table.Dock = "Fill"
```

Das `TableLayoutPanel` wächst und schrumpft automatisch mit seinem Parent-Container.

Gleichmäßige Spalten erzeugen

```
$table.ColumnCount = 2

$table.ColumnStyles.Add(
    [System.Windows.Forms.ColumnStyle]::new("Percent",50)
)

$table.ColumnStyles.Add(
    [System.Windows.Forms.ColumnStyle]::new("Percent",50)
)
```

Beide Spalten belegen dadurch jeweils die Hälfte der verfügbaren Breite

Revision #8

Created 2026-03-16 04:28:24 UTC by John-Andreas Borinas

Updated 2026-08-23 19:25:50 UTC by John-Andreas Borinas