

# 1 Vorwort

Evtl. fehlt manchen Programmierern eine dynamische Datenstruktur in AutoIt. Die einfachste Möglichkeit so etwas zu realisieren ist der Stapelspeicher (STACK). Dieses Skript stellt ihnen bis zu fünf Stacks und entsprechende Bearbeitungsfunktionen zur Verfügung. Einige praktische Anwendungen sind:

- Schnell mal ein paar Daten ablegen, ohne sich um Variablen und Felder zu kümmern zu müssen
- Daten sortieren
- Schnelle Datenübergabe an (Unter)Funktionen und zurück

Auf die Stacks abgelegt werden können Zeichenketten und Zahlen.

## 1.1 Nutzungsbedingungen

1. stack[XX].au3 ist Freeware und darf nur kostenlos weitergegeben werden
2. Sie benötigen Autoit V3.2 oder höher
3. Die freie Verbreitung des Skripts ist erlaubt und sogar erwünscht
4. Für Funktionalität und Folgen der Nutzung von stack[XX].au3 wird keinerlei Haftung übernommen!
5. stack00.au3 darf nur mit diesem Text zusammen weitergegeben werden
6. Mit der Nutzung von stack[XX].au3 erkennen Sie die Punkte 1 bis 5 an

## 2 Änderungen

n.n.

## 3 Grundlagen

- direkter Zugriff nur auf das letzte eingereihte Element

- Last in – First Out (LIFO)

- Stapel von Tellern
- Stapel von Briefen

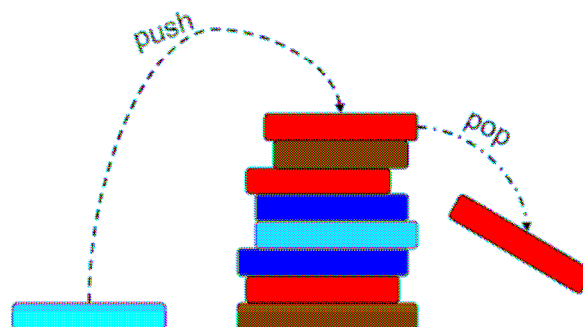
- Manche Leute sagen: „Keller“

- Zuletzt eingelagerte Kartoffeln werden zuerst gegessen

- Stacks sind sehr wichtige Datenstrukturen der Informatik

- Statt insert und remove sagt man

- push und pop



Stand: 15.10.2006

## 4 Befehle

### 4.1 drop(\$n)

Entfernt das 1. Element von Stack \$n: (x1 ) -> ( )

### 4.2 dup(\$n)

Dupliziert das 1. Element von Stack \$n: (x1) -> (x1 x1)

### 4.3 eos(\$n)

End Off Stack von Stack \$n: Ergibt:

- = 1 - Im Stack sind noch Elemente
- = 0 – Stack ist leer

### 4.4 inistack()

Erzeugt einen Stack von maimal fünf und gibt eine Stacknummer zurück.

z.B. \$nr = inistack()

### 4.5 minimax(\$n,\$f)

Verschiebt kleinstes (min.) Element (wenn \$f=0) oder größtes (max.) Element (wenn \$f=1) auf den Stack \$n

### 4.6 over(\$n)

Dupliziert das 2. Element von Stack \$n: (x2 x1) -> (x2 x1 x1)

Entspricht pick(\$n,2)

### 4.7 pick(\$n,\$m)

Kopiert m-te Element auf den Stack n\$: (... xm ...x1) -> (...xm ...x1 xm)

### 4.8 pop(\$n)

Holt das letzte eingegebene Element vom stack \$n (LIFO - Last In First Out)

### 4.9 push(\$n,\$var)

Legt \$var auf stack \$n

Stand: 15.10.2006

## 4.10 roll(\$n,\$m)

Verschiebt m-te Element auf den Stack n\$: (... xn xm-1 ...x1) -> (...xm-1 ...x1 xm)

## 4.11 rot(\$n)

Tauscht das 1. Element mit dem 3. Element von stack \$n: (x3 x2 x1) -> (x2 x1 x3)  
Entspricht roll(\$n,3)

## 4.12 stack2box(\$n,\$c)

Zeigt Stack \$n in einer msgbox. Wobei \$c ein Kommentar ist.

## 4.13 stacklen(\$n)

Zeigt die Anzahl der Elemente von stack \$n

## 4.14 stackrevers(\$n)

Kehrt die Reihenfolge der Elemente von Stack \$n um

## 4.15 stacksort(\$n,\$f)

Sortiert Stack \$n aufsteigend (wenn \$f=0) oder absteigend (wenn \$f=1)

## 4.16 swap(\$n)

Tauscht das 1. Element mit dem 2. Element von stack \$n: (x2 x1) -> (x1 x2)

Stand: 15.10.2006

## 5 ToDo Liste

- Quicksortrechenvorschrift für stacksort