# ninBot + Scripting Documentation

## Grundlegende Channelkommandos

- **Digitale Eieruhr:**
  `!timer MINUTEN EREIGNIS`
  z.B. "!timer 12 Pizza" würde einen 12 Minuten Timer starten und nach 12 Minuten *klingeln*

- **ninScript Code Interpretieren:**
  `!eval CODE`
  z.B. "!eval say &parnick;"

- **Verlassen der Raumes initiieren:**
  `!part`

- **Bannen des aktuellen Raumes in die Banlist:**
  `!ban`

- **Einen anderen Raum betreten** (*Userlevel >= 4*)**:**
  `!join CHANNEL`

- **Den Bot für 5 Minuten "stilllegen":** (Not implemented completely yet)
  `!mute`

- **Den Index aller Befehle aufrufen**
  `!index SEITE`

## Datensaetze anlegen und nutzen

**Die Grundlegenden Datensatzbefehle sind**

Gibt einen zufälligen Datensatz zurück
`data`

Legt einen Datensatz mit dem Namen "DATENNAME" und dem Inhalt "DATENINHALT" an
`data DATENNAME = DATENINHALT`

Legt einen fortlaufenden Datensatz mit dem Namen "DATENNAME" und dem Inhalt "DATENINHALT" an
`data+ DATENNAME = DATENINHALT`
Falls "DATENNAME" schon belegt wird eine fortlaufende Zahl angefügt z.B. "DATENNAME1"

Löscht den Datensatz mit dem Namen "DATENNAME"
`data- DATENNAME`

Setzt Flags für den Datensatz mit dem Namen "DATENNAME" und den Flags "FLAGS"

**dataflag *DATENNAME* = *FLAGS***

Flags können entweder 0, rw für frei, ro für nur lesen sein

Setzt das Userlevel für den Datensatz mit dem Namen "DATENNAME" und dem Level "LEVEL"

**datalevel *DATENNAME* = *LEVEL***

Userlevel gehen von 0 = alle bis hin zu 10 = nur Owner

**Formate Datensätze für weiteres Handling**

Um z.b. *!kommando* Kommandos zu erstellen, muss der Datensatzname folgendes Format haben:

> *com-kommando* als Datensatzname würde das Kommando *!kommando* anlegen

Wenn der Datensatzinhalt mit geschweiften Klammern umgeben ist "{}" wird der Inhalt als ninScript ausgeführt, die einzelnen Befehle werden durch Semikolon (;) getrennt

z.b. um ein Kommando !hallo anzulegen welches den aufrufenden Nickname Grüßt

```
data com-hallo = { say "Hallo &parnick"; }
```

Um ein !kommando zu nutzen das es bereits gibt, aber einen anderen Namen zusätzlich bekommen soll gibt es Links.
Links werden mit zwei Ausrufezeichen deklariert (!!) und muss nicht in geschweiften Klammern stehen.

z.b. um das oben genannte Kommando noch einmal zu nutzen mit dem Namen !hi

```
data com-hi = !!com-hallo
```

**Übersicht Formate Datensätze:**

**com-*NAME*** - Ein !kommando NAME
**data-var-*NAME*** - Eine Globale Variable NAME
**data-nvar-*NICKNAME-NAME***
- Eine Nickbasierte NICKNAME Variable NAME
**data-prg-NAME** - Eine Subroutine

**data-var-*NAME* = Eintrag1@@Eintrag2@@Eintrag3**
- Eine Globale Variable NAME als Array

**Events Format Datensätze:**

**data-event-onjoin**
- Globales onJoin Event

**data-event-#*CHANNEL*-onjoin**
- Globales Channel onJoin Event

**data-event-*NICKNAME*-onjoin**
- Nickname onJoin Event

**data-event-#*CHANNEL*-*NICKNAME*-onjoin**
- Nickname und Channel onJoin Event

## Scripting Befehle für ninScript

**cmd *KOMMANDO*** - Führt das ! Kommando KOMMANDO aus (com-KOMMANDO)

**dec *VARIABLE*** - Zählt die Variable VARIABLE um 1 runter

**inc *VARIABLE*** - Zählt die Variable VARIABLE um 1 hoch

**n_dec *VARIABLE*** - Zählt die Nickverknüpfte Variable VARIABLE um 1 runter

**n_inc *VARIABLE*** - Zählt die Nickverknüpfte Variable VARIABLE um 1 hoch

**say *TEXT*** - Sagt TEXT in den aufrufenden Channel

**me *TEXT*** - Action TEXT in den aufrufenden Channel

**run *SUBROUTINE*** - Führt Subroutine SUBROUTINE auf (data-prg-SUBROUTINE)

**set *VARIABLE* = *VALUE***
- Setzt Variable VARIABLE auf VALUE (data-var-VARIABLE)

**n_set *VARIABLE* = *VALUE***
- Setzt die Nickverknüpfte Variable VARIABLE auf VALUE (data-nvar-NICKNAME-VARIABLE)

**push *STACK* = *VALUE*** - Fügt einen Eintrag VALUE zum Stack STACK hinzu

**pop *STACK* = *VALUE*** - Entfernt den Eintrag VALUE vom Stack STACK

**s_flush *STACK*** - Löscht den Stack mit dem Namen STACK

**return *VALUE*** - Gibt Wert VALUE zurück (siehe Variablenersetzungen)

| | |
|---|---|
| **sleep *VALUE*** | - Wartet VALUE Sekunden bevor der nächste Befehl ausgeführt wird |
| **stackprint *STACK*** | - Gibt den Inhalt des Stacks STACK aus |
| **randcalc** | - Gibt einen zufälligen Datensatz wieder (Keine data- oder com- Datensätze) |
| **msg "*MSG*"** | - schickt eine private Nachricht MSG an den aufrufenden User |
| **msg *NICK* "*MSG*"** | - schickt eine private Nachricht MSG an den User NICK |
| **notice "*MSG*"** | - schickt eine notice Nachricht MSG an den aufrufenden User |
| **notice *NICK* "*MSG*"** | - schickt eine notice Nachricht MSG and den User NICK |

**Anweisungen für Bedingungen:**

```
if "VALUE" (eq/ne) "VALUE2" then { SCRIPT } else { SCRIPT}
nif "VALUE" (eq/ne) "VALUE2" then { SCRIPT } else { SCRIPT }
```

In Allen Variablen können auch Platzhalter verwendet werden um zur Laufzeit Parameter o.ä. nutzen zu können:

| | |
|---|---|
| **&nick** | - Wird durch den aufrufenden Nickname ersetzt |
| **&chan** | - Wird duch den aufrufenden Channel ersetzt |
| **&backend** | - Das aktuelle Calc Backend |
| **&userlevel** | - Das Userlevel des aufrufenden Nickname |
| **&1 / &2 / &3 / &4** | - Gibt die Parameter wieder (des aktuellen Kommandos) |
| **&devel[*0-2*]** | - Gibt die Anzahl der Zeilen/Wörter/Zeichen des Bots wieder |
| **&command** | - Gibt den Namen des aktuellen Kommandos wieder |
| **&return** | - Gibt den Returnwert von return an |
| **&allnicks** | - Alle Nicknames des aufrufenden Channels |
| **&param** | - Der aktuelle Parameter des Kommandos |
| **&parnick** | - Gibt entweder den Parameter oder den Nickname des Aufrufers aus |
| **if_stack(*NAME*)** | - Gibt true zurück wenn Stack NAME definiert ist |
| **num_stack(*NAME*)** | - Gibt die Anzahl der Einträge in Stack NAME zurück |
| **is_stack(*NAME*)[*VALUE*]** | - Gibt true zurück wenn der Eintrag VALUE in Stack NAME ist |
| **stack(*NAME*)** | - Gibt den Stack NAME aus |
| **rand_var(*NAME*)** | - Gibt einen Zufälligen Eintrag aus der Array Variable NAME aus |

| | |
|---|---|
| `rand_nick` | - Gibt einen Zufälligen Nickname aus dem aktuellen Channel aus |
| `var(`*`NAME`*`)` | - Gibt eine Variable NAME zurück |
| `var(`*`NAME`*`)[`*`INDEX`*`]` | - Gibt einen Arraywert INDEX aus dem Array NAME zurück |
| `n_var(`*`NAME`*`)` | - Gibt eine Nickverknüpfte Variable NAME zurück |
| `n_var(`*`NAME`*`)[INDEX]` | - Gibt einen Arraywert INDEX aus dem Nickverknüpften Array NAME zurück |
| `rand(`*`NUM`*`)` | - Gibt eine Zufallszahl von 1 bis NUM aus |
| `[[`*`NAME`*`]]` | - Gibt den Inhalt eines Datensatzes aus |