Programmable **A**rhythmetic **I**nterface **S**cript

HANDBOOK>

Developer Fabian Müller

WAIT INTEGER

Requires a numeric parameter and waits for the time set by the parameter, in milliseconds.

EXAMPLE:

```
#PAIS
Echo LOADING . . .
wait 2000
Echo COMPIETE!
```

PAUSE

Pauses the script until an input inform of a keystroke occurs.

EXAMPLE:

```
#PAIS
Echo HELL
PAUSE
Echo O!
```

FOREGROUND=ARGUMENT

Changes the font color to a specified value. Affects only the following characters.

Arguments

- Red
- DarkRed
- Blue
- DarkBlue
- Green
- DarkGreen
- Cyan
- DarkCyan
- Magenta
- DarkMagenta
- Yellow
- DarkYellow
- White
- Black
- Gray

EXAMPLE:

```
#PAIS
Foreground=red
Echo ALARM!
```

BACKGROUND=ARGUMENT

Changes the background color to a set value. Affects only the following characters ..

Arguments

- **Red**
- **DarkRed**
- **Blue**
- **DarkBlue**
- **Green**
- **DarkGreen**
- **Cyan**
- **DarkCyan**
- **Magenta**
- **DarkMagenta**
- **Yellow**
- **DarkYellow**
- **White**
- **Black**
- **Gray**

EXAMPLE:

```
#PAIS
Background=cyan
Echo CRAZY!
```

TITLE=STRING

Changes the window title of the application to a specified parameter.

EXAMPLE:

```
#PAIS
Titel=MY FIRST PAIS :D!   JEY!
```

CALL STRING

Runs an external PAIS. (Class equivalent)

BEISPIEL:

```
#PAIS
Call Application.pais
```

CLEAR

Clears the contents of the application's output.

EXAMPLE:

```
#PAIS
Call Application.pais
Clear
```

DOWNLOADFILE:STRING<STRING>
Loads a file from the Internet or network and saves it.
EXAMPLE:

```
#PAIS
Downloadfile:https://www.google.de/<Google.txt>
Clear
```

FLOAT STRING
Displays the text specified by a parameter in consecutive characters at a specified rate.
EXAMPLE:

```
#PAIS
Float YEAAAYYYYYY!!!!!!!!
```

DIALOG STRING
Opens an information dialog with the text specified by a parameter.
EXAMPLE:

```
#PAIS
Dialog YEAAAYYYYYY!!!!!!!!
```

ECHO STRING
Writes a given text in the application output.
E:

```
#PAIS
Echo YEAAAYYYYYY!!!!!!!!
```

ECHOL STRING
Writes a given text to the application output followed by a line break
EXAMPLE:

```
#PAIS
Echol YEAAAYYYYYY!!!!!!!!
```

PRINT STRING
Writes a given text in the application output
EXAMPLE:

```
#PAIS
Print YEAAAYYYYYY!!!!!!!!
```

PRINTL STRING

Writes a given text to the application output followed by a line break.

EXAMPLE:

```
#PAIS
Printl YEAAAYYYYYY!!!!!!!!
```

RESET

Resets the application to the original state, but variables remain.

EXAMPLE:

```
#PAIS
Reset
```

RESETCOLOR

Resets the previously changed color to the default color.

EXAMPLE:

```
#PAIS
Resetcolor
```

POINT(INTEGER,INTEGER)

Sets the position of the cursor to a parameter set by two.

EXAMPLE:

```
#PAIS
POINT(19,5)
Printl YEAAAYYYYYY!!!!!!!!
```

CONSOLE_HEIGHT= INTEGER

Change the height of the application.

EXAMPLE:

```
#PAIS
CONSOLE_HEIGHT= 2
```

CONSOLE_WIDTH= INTEGER

Change the width of the application.

EXAMPLE:

```
#PAIS
CONSOLE_WIDTH= 2
```

WINDOW(INTEGER,INTEGER)
Resizes the window of the application
EXAMPLE:

```
#PAIS
WINDOW(40,40)
```

COMPUTER:ARGUMENT INTEGER
Performs a "PAIS computer function".
Arguments
Logout
Shutdown
Sleep
Restart

EXAMPLE:

```
#PAIS
COMPUTER:Shutdown 2000
```

BEEP INTEGER
Plays a processor beep.
EXAMPLE:

```
#PAIS
Beep 1400
```

PLAY STRING
Plays a WAV-file.
PLAY+ STRING Plays a file in the background

EXAMPLE:

```
#PAIS
PLAY FOLLOW THE TRAIN CJ.wav
```

SYSTEM:STRING
Executes a system command.
EXAMPLE:

```
#PAIS
System:ipconfig
```

RAINBOW STRING

Displays a parameter-defined text in chronological following and repeating colors.

EXAMPLE:

```
#PAIS
Rainbow WOOP!  WOOP!
```

GOTO STRING

Jump to a jump line within the script.

Jump lines are created as follows:

[STRING]

EXAMPLE:

```
#PAIS
[main]
echo Hello!
Goto main
```

ENCODING:ARGUMENT

Sets the output encoding.

Arguments

- **utf8**
- **utf7**
- **ascii**
- **default**
- **standard**

EXAMPLE:

```
#PAIS
Encoding:UTF8
```

ERROR:ARGUMENT

Enables / disables the error output.

Arguments

- **on**
- **off**

EXAMPLE:

```
#PAIS
Error:OFF
```

IF PARAMETER OPERANT PARAMETER>SCRIPT

Logical evaluation of arguments to logical procedures.

EXAMPLE:

```
#PAIS
IF %time @ :>Echo It contains ":"!    :O
```

REPLACE(STRING|STRING|STRING)

Replaces a string with a string from a string

EXAMPLE:

```
#PAIS
$Replaced=REPLACE(Hello! My name ist -NAME-|-NAME-|Jeff)
Echo $Replaced
```

RANDOM(INTEGER,INTEGER)

Creates a random number from integer to integer

EXAMPLE:

```
#PAIS
$Random=RANDOM(1,7)
Echo Du hast eine $Random gewürfelt
```

FOR EACH(PARAMETER1|PARAMETER2)COMMAND

For each PARAMETER1 in PARAMETER2, execute the command.

EXAMPLE:

```
#PAIS
$Text=Welcome!
$Treffer= 0
For each(m|%Text)$Treffer+1
Echo %Text Hits
```

O-PAIS 8 "Variables and Declarationen"

Declaration, setting, calculation and use of variables in constants scripts.

Declaration

Declarations of variables are optional. PAIS independently checks if a variable has been declared. If a variable is not declared, it is (in most cases) automatically declared. A self-declaration is an indication of a professional scripting method and offers third (other programmers) the ability to better understand a script, procedure, or function. In addition, errors are excluded! Declaring with $ is followed by the name / name, an operand, and in most cases a variable. A variable may only consist of letters and numbers and should never be given the same name as system variables such as% time or% path

Operants

=	**Sets the variable to an exact value**
|	**Saves the next keypress (a, b, c, 1, 2, 3, *, -, / , ...)**
+	**Adds the variable with the number following the operator**
-	**Subtracts the variable with the number following the operator**
*****	**Multiplies the variable by the number following the operator**
/	**Divides the variable with the number following the operator**

Variables are read out (eg within an echo command) with% followed by the identifier or the name of the variable.

EXAMPLE:

```
#PAIS
#Variablen kalkulationsbeispiel
$Wert1=0
$Wert2=%Wert1
$Wert2+199
$Wert2/4

Echo (0+199)/4=%Wert2
pause
```

The output would be in this example:

O-PAIS 9 "IF Operations"

If operations and logarithms and their use

If

If is a powerful operation and allows smaller logical decisions to be made within a script. From variable to static objects, everything is conceivable. If operations are created as follows:
IF, the first variable, followed by an operand, followed by a second variable, followed by>, followed by a command.
IF VARIABLEOPERANTVARIABLE>COMMAND

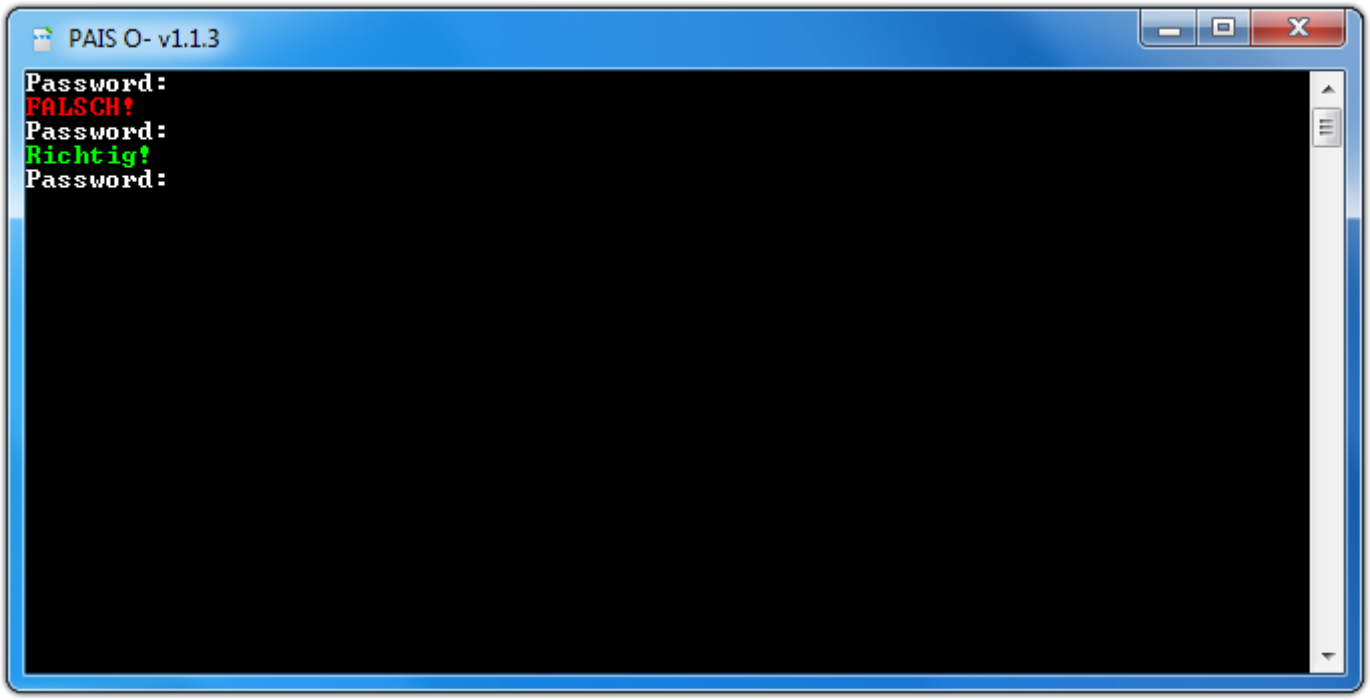
Operanten

=	Is exactly
!	Is not
<	Is smaller than
>	Is bigger than
@	Contains
$	Does'nt contains

EXAMPLE:

```
#PAIS
#IF-Operationen
[main]
$PasswordInput=0
$Password=12345
foreground=white
Echo Password:
foreground=black
$PasswordInput<
foreground=white
IF %Password=%PasswordInput>Goto Richtig
foreground=red
EchoL WRONG!
Goto main
[Richtig]
foreground=green
EchoL RIGHT!
Goto main
EchoL RRIGHT!
Pause
```

The output would be in this example: 1st input: "554321" 2nd input = "12345"



O-PAIS 10 "FOR EACH Operationen"

For each operations are used to evaluate fragments in an object

For Each

For Each is also a powerful operation that allows you to evaluate the contents of an object, such as string, integer, argument, etc. For each is constructed as follows:

FOR EACH(VARIABLE|VARIABLE)COMMAND

EXAMPLE:

```
#PAIS
#FOR-Operationen
[main]
$Text=Welcome!
$Treffer=0
For Each(m|%Text)$Treffer+1
EchoL %Treffer Hits
Pause
```

Die Ausgabe wäre bei diesem Beispiel: