PGofer Versão 2



Rev.: 1.8

Autor: Franco Michel Almeida Caixeta.

E-mail: <francomichel@gmail.com>

**SUMARIO**

[1. APRESENTAÇÃO 8](#_Toc389570647)

[2. FORMULÁRIOS 9](#_Toc389570648)

[PGOFER 9](#_Toc389570649)

[AUTOCOMPLETES 9](#_Toc389570650)

[CLIPBOARDS 10](#_Toc389570651)

[CONSOLES 11](#_Toc389570652)

[HOTKEYS 11](#_Toc389570653)

[LINKS 12](#_Toc389570654)

[NOTEPADS 13](#_Toc389570655)

[SERVICES 14](#_Toc389570656)

[TIMERS 16](#_Toc389570657)

[3. TECLAS 18](#_Toc389570658)

[4. SIMBOLOS 19](#_Toc389570659)

[5. EXPRESSÕES NUMERICAS 20](#_Toc389570660)

[NOTAÇÃO CIENTIFICA 20](#_Toc389570661)

[PREFIXO INTERNACIONAL 20](#_Toc389570662)

[BINARIO 21](#_Toc389570663)

[HEXADECIMAL 22](#_Toc389570664)

[6. COMANDOS 23](#_Toc389570665)

[AND 23](#_Toc389570666)

[BEGIN 23](#_Toc389570667)

[COPY 24](#_Toc389570668)

[CONST 24](#_Toc389570669)

[DELETE 24](#_Toc389570670)

[DEC 24](#_Toc389570671)

[DO 25](#_Toc389570672)

[DOWNTO 25](#_Toc389570673)

[ELSE 25](#_Toc389570674)

[END 25](#_Toc389570675)

[FOR 25](#_Toc389570676)

[FUNCTION 26](#_Toc389570677)

[GLOBAL 26](#_Toc389570678)

[IF 27](#_Toc389570679)

[INC 27](#_Toc389570680)

[INSERT 27](#_Toc389570681)

[MOD 28](#_Toc389570682)

[NOT 28](#_Toc389570683)

[OR 29](#_Toc389570684)

[READ 29](#_Toc389570685)

[REPEAT 29](#_Toc389570686)

[ROOT 30](#_Toc389570687)

[THEN 30](#_Toc389570688)

[TO 30](#_Toc389570689)

[UNTIL 30](#_Toc389570690)

[VAR 30](#_Toc389570691)

[XOR 31](#_Toc389570692)

[WHILE 31](#_Toc389570693)

[WRITE 32](#_Toc389570694)

[7. CLIPBOARD 32](#_Toc389570695)

[COPYFROMTEXT 32](#_Toc389570696)

[CLEAR 33](#_Toc389570697)

[GETFORMAT 33](#_Toc389570698)

[LOADITEMFROMFILE 33](#_Toc389570699)

[LOADFROMITEM 34](#_Toc389570700)

[PASTETOTEXT 34](#_Toc389570701)

[SAVEITEMTOFILE 34](#_Toc389570702)

[SAVETOITEM 34](#_Toc389570703)

[8. FILE 36](#_Toc389570704)

[COPY 36](#_Toc389570705)

[DELETE 37](#_Toc389570706)

[DIREXISTS 37](#_Toc389570707)

[EXEC 38](#_Toc389570708)

[EXTRACTFILEEXT 38](#_Toc389570709)

[EXTRACTFILENAME 39](#_Toc389570710)

[EXTRACTFILEPATH 39](#_Toc389570711)

[FILEEXISTS 39](#_Toc389570712)

[FILEEXPANDPATH 39](#_Toc389570713)

[FILEUNEXPANDPATH 40](#_Toc389570714)

[GETACESSTIME 40](#_Toc389570715)

[GETATTRIB 40](#_Toc389570716)

[GETCREATETIME 41](#_Toc389570717)

[GETFILESIZE 41](#_Toc389570718)

[GETMODIFYTIME 42](#_Toc389570719)

[LOADFROMTEXT 42](#_Toc389570720)

[MKDIR 42](#_Toc389570721)

[MOVE 42](#_Toc389570722)

[OPENDIALOG 43](#_Toc389570723)

[RENAME 44](#_Toc389570724)

[SAVETOTEXT 44](#_Toc389570725)

[SCRIPT 45](#_Toc389570726)

[SEARCH 45](#_Toc389570727)

[SETATTRIB 45](#_Toc389570728)

[SETDATETIME 46](#_Toc389570729)

[9. FORM 47](#_Toc389570730)

[ALPHABLEND 47](#_Toc389570731)

[ALPHABLENDVALUE 47](#_Toc389570732)

[CLOSE 47](#_Toc389570733)

[ENABLED 48](#_Toc389570734)

[HEIGTH 48](#_Toc389570735)

[HIDE 48](#_Toc389570736)

[LEFT 49](#_Toc389570737)

[SHOW 49](#_Toc389570738)

[SETFOCUS 49](#_Toc389570739)

[TOP 50](#_Toc389570740)

[TRANSPARENT 50](#_Toc389570741)

[TRANSPARENTCOLOR 50](#_Toc389570742)

[WIDTH 50](#_Toc389570743)

[WINDOWSTATE 51](#_Toc389570744)

[10. KEY 52](#_Toc389570745)

[ALLUP 52](#_Toc389570746)

[CHARTOKEY 52](#_Toc389570747)

[GETPRESS 52](#_Toc389570748)

[KEYTOCHAR 53](#_Toc389570749)

[SETPRESS 53](#_Toc389570750)

[MACRO 53](#_Toc389570751)

[11. LINK 55](#_Toc389570752)

[DIRECTORY 55](#_Toc389570753)

[FILE 55](#_Toc389570754)

[ICON 55](#_Toc389570755)

[PARAMETER 56](#_Toc389570756)

[PARAMEXEC 56](#_Toc389570757)

[SHOWCONTROL 56](#_Toc389570758)

[12. MATH 57](#_Toc389570759)

[ARCCOS 57](#_Toc389570760)

[ARCSIN 57](#_Toc389570761)

[ARCTAN 58](#_Toc389570762)

[COS 58](#_Toc389570763)

[COSECANT 58](#_Toc389570764)

[COTAN 59](#_Toc389570765)

[BIN 59](#_Toc389570766)

[DEGTOGRAD 59](#_Toc389570767)

[DEGTORAD 59](#_Toc389570768)

[FORMAT 60](#_Toc389570769)

[GRADTODEG 61](#_Toc389570770)

[GRADTORAD 61](#_Toc389570771)

[HEX 61](#_Toc389570772)

[HYPOT 61](#_Toc389570773)

[LOG 62](#_Toc389570774)

[MODULE 62](#_Toc389570775)

[RANDOM 62](#_Toc389570776)

[RADTODEG 63](#_Toc389570777)

[RADTOGRAD 63](#_Toc389570778)

[SECANT 63](#_Toc389570779)

[SIN 64](#_Toc389570780)

[TAN 64](#_Toc389570781)

[13. NET 65](#_Toc389570782)

[CLIENT 65](#_Toc389570783)

[ACTIVE 65](#_Toc389570784)

[ADDRESS 65](#_Toc389570785)

[PASSWORD 65](#_Toc389570786)

[PORT 66](#_Toc389570787)

[SENDCOMMAND 66](#_Toc389570788)

[SERVER 66](#_Toc389570789)

[ACTIVE 67](#_Toc389570790)

[MAXCONNECT 67](#_Toc389570791)

[PASSWORD 67](#_Toc389570792)

[PORT 68](#_Toc389570793)

[SENDMESSAGE 68](#_Toc389570794)

[SETTCPIP 68](#_Toc389570795)

[14. PROCESS 69](#_Toc389570796)

[FINDFILE 69](#_Toc389570797)

[GETFILENAME 69](#_Toc389570798)

[GETFOREGROUND 69](#_Toc389570799)

[GETPRIORITY 70](#_Toc389570800)

[KILL 70](#_Toc389570801)

[SETPRIORITY 71](#_Toc389570802)

[15. REGISTRY 72](#_Toc389570803)

[DELETE 72](#_Toc389570804)

[READ 72](#_Toc389570805)

[WRITE 73](#_Toc389570806)

[16. SERVICE 74](#_Toc389570807)

[CREATE 74](#_Toc389570808)

[DELETE 74](#_Toc389570809)

[GETCONFIG 74](#_Toc389570810)

[GETDESCIPTION 75](#_Toc389570811)

[GETSTATE 75](#_Toc389570812)

[SETCONFIG 76](#_Toc389570813)

[SETDESCIPTION 76](#_Toc389570814)

[SETSTATE 77](#_Toc389570815)

[17. SOUND 78](#_Toc389570816)

[GETVOLUME 78](#_Toc389570817)

[MUTE 78](#_Toc389570818)

[PLAYSOUND 79](#_Toc389570819)

[SETVOLUME 79](#_Toc389570820)

[VOLUMESTEPDOWN 79](#_Toc389570821)

[VOLUMESTEPUP 80](#_Toc389570822)

[18. SYSTEM 81](#_Toc389570823)

[AUTOCLOSE 81](#_Toc389570824)

[CONSOLECLEAR 81](#_Toc389570825)

[CONSOLEDELAY 81](#_Toc389570826)

[CONSOLEMSG 82](#_Toc389570827)

[DATETIMENOW 82](#_Toc389570828)

[DIRCURRENT 83](#_Toc389570829)

[DELAY 83](#_Toc389570830)

[FILELISTMAX 84](#_Toc389570831)

[FINDWINDOW 84](#_Toc389570832)

[FORMATREPLY 84](#_Toc389570833)

[ICONLOADER 85](#_Toc389570834)

[LOCKWORKSTATION 85](#_Toc389570835)

[LOOPLIMITE 86](#_Toc389570836)

[MONITORPOWER 86](#_Toc389570837)

[NOOFF 86](#_Toc389570838)

[PRTSCREEN 87](#_Toc389570839)

[SENDMESSAGE 87](#_Toc389570840)

[SETSCREEN 87](#_Toc389570841)

[SETSUSPENDSTATE 88](#_Toc389570842)

[SHOWMESSAGE 88](#_Toc389570843)

[SHUTDOWN 89](#_Toc389570844)

# APRESENTAÇÃO

O nome "Gofer", que em linguagem informática representa a expressão "pau para toda obra". Isso porque o foco do programa é um utilitário de quebra galho para várias tarefas. Assim sendo uma ferramenta poderosa para controlar o computador. Um interpretador de linguagem pascal adaptado, podendo criar scripts para executar algoritmos complexos.

* **Cálculos matemáticos:**

Em qualquer momento pode calcular formulas complexas, escrevendo-as rapidamente usando expressões numéricas ou até prefixos.

* **Teclas de Atalho:**

Criando teclas de atalhos globais, pode executar qualquer comando desejado em qualquer lugar usando qualquer aplicativo.

* **Gerenciador de Serviços:**

Fica muito fácil gerenciar e salvar as configurações de todos os serviços do Windows, para levar para outro computador ou depois de reinstalar o mesmo.

* **Área de transferência:**

Para facilitar na hora de usar o famoso Copiar “ctrl+c” e Colar “ctrl+v”, existe uma ferramenta para salvar o que está na área de transferência, assim podemos usar o “ctrl+c” varias vezes sem perder a informação do ultimo.

* **Agenda de Tarefas:**

Muito mais fácil agendar um tempo, horário, dia, mês ou até dia da semana, para executar um programa ou comando.

# FORMULÁRIOS

Formulários com funções diversas.

* Lista

[PGOFER](#_PGOFER_2), [AUTOCOMPLETES](#_AUTOCOMPLETES), [CLIPBOARDS](#_CLIPBOARDS), [CONSOLES](#_CONSOLES), [HOTKEYS](#_HOTKEYS), [LINKS](#_LINKS), [NOTEPADS](#_NOTEPADS), [SERVICES](#_SERVICES), [TIMERS](#_TIMERS).

## PGOFER

Principal do programa, onde recebe os comandos e executa-os rapidamente.



1. **Bordas:** servem para arrastar a janela.
2. **Prompt:** é usado para digitar os comandos.

* Mais:

[FORM](#_FORM).

## AUTOCOMPLETES

Aparece apenas quando está digitando um comando, nela aparece as opções para auto completar comandos.



1. **Lista de comandos:** mostra a todos os comandos que correspondem com o que foi digitado.
2. **Comando:** Coluna que mostra os comandos.
3. **Valor:** Mostra o possível valor de uma variável.
4. **Prioridade:** E o contador de quantas vezes um comando é chamando, quanto maior o numero ele aparece primeiro na lista.
5. **Bordas:** usado para ajustar o tamanho da janela.

* Mais:

[FORM](#_FORM).

## CLIPBOARDS

Ferramenta para gerenciar a área de transferência do Windows.



1. **ClipBoards:** Mostra o que tem na área, permite salvar ou limpar ela.
2. **Salvar Automático:** Salvar automaticamente todos os dados que estiver na área, podendo definir a quantidade máxima, se ela exceder o mais antigo será apagado para caber o ultimo.
3. **Lista:** mostra todos os dados salvos, em formato e tamanho.
4. **Salvar como Arquivo:** Salva algum dado selecionando como arquivo “.clp”.
5. **Carregar do Arquivo:** carrega os dados apartir de um arquivo “.clp” para a lista.
6. **Roda Pé:** mostra a quantidade de dados salvos e o tamanho total.

* Mais:

[FORM](#_FORM), [CLIPBOARD](#_CLIPBOARD).

## CONSOLES

Mostra todas as mensagens e erros dos comandos e cálculos executados.



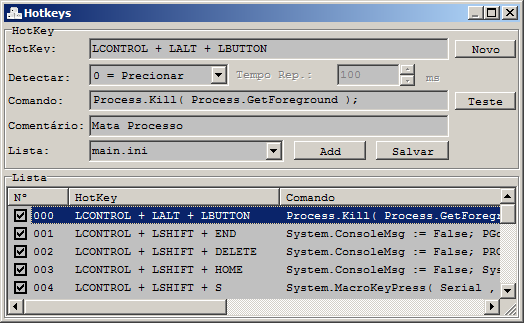
1. **Console:** mostra as mensagens de todos os comandos ou funções executadas.
2. **Bordas:** é usado para alterar o tamanho ou posição da janela.
3. **Fix:** Botão para fixar a janela para que ela não feche automaticamente.

* Mais:

[FORM](#_FORM).

## HOTKEYS

Ferramenta para criar e gerenciar teclas de atalho globais.



1. **Teclas de Atalho:** Define a tecla de atalho.
2. **HotKey:** a tecla de atalho.
3. **Detectar:** qual momento para ser detectado, o pressionamento ou soltura.
4. **Tempo de rep:** o tempo de repetição para “pressionado”.
5. **Comando:** a ser executado.
6. **Lista:** o arquivo que ficara salvo a tecla.
7. **Add:** adiciona na lista a baixo.
8. **Salvar:** Salva a lista a baixo.

* Mais:

[FORM](#_FORM).

## LINKS

Cria e gerencia os atalhos dos programas.



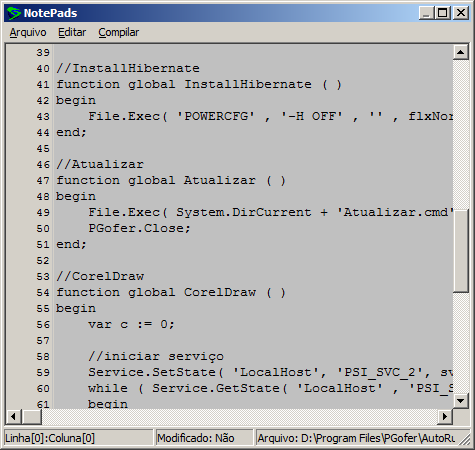
1. **Link’s:** define os atalhos.
2. **Comando:** o nome do atalho.
3. **Arquivo:** o endereço do arquivo.
4. **Parâmetro:** comando a ser enviado para o arquivo.
5. **Ícone:** o endereço do ícone a ser carregado para representar o arquivo.
6. **Diretório:** o endereço do diretório foco do arquivo.
7. **Lista:** arquivo a ser salvo os atalhos.
8. **Add:** Adiciona o atalho na lista a baixo.
9. **Salvar:** Salva a lista a baixo.

* Mais:

[FORM](#_FORM), [LINK](#_MATH).

## NOTEPADS

Ferramenta para editar arquivos “pas” para o PGofer.



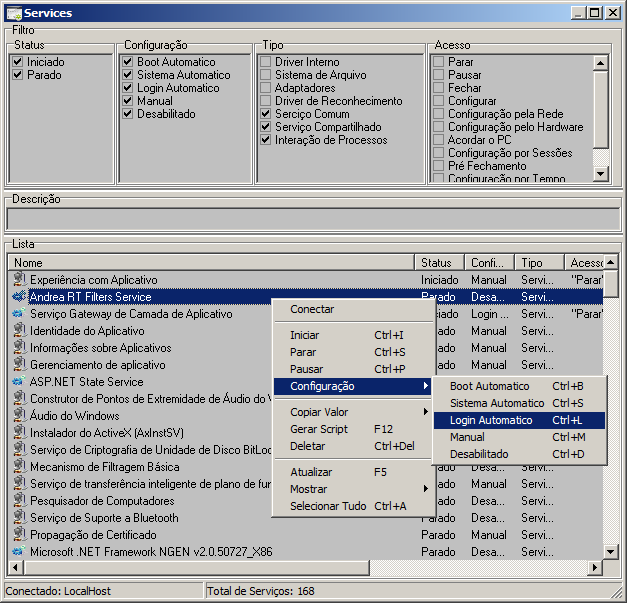
1. **Arquivo:** acessa opção de salvar ou abrir arquivos.
2. **Editar:** Acessa opção de editar o texto.
3. **Compilar:** Compila os comandos escritos a baixo.
4. **Roda Pé:** mostra a localização do cursor e qual arquivo está aberto.

* Mais:

[FORM](#_FORM).

## SERVICES

Gerencia todos os serviços do Windows.



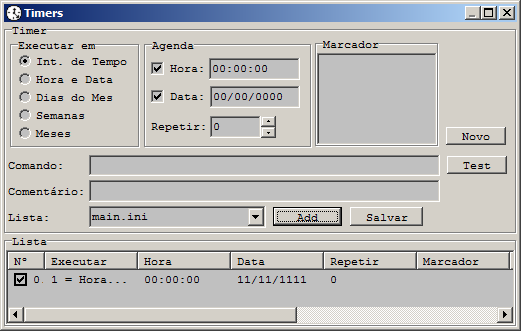
1. **Filtro:** Opções para mostrar os tipos de serviços.
2. **Descrição:** Mostra a descrição do serviço.
3. **Lista:** mostra todos os serviços listados.
4. **Conectar:** Conecta em uma determiada maquina.
5. **Iniciar, Parar, Pausar:** Define o estado do serviço.
6. **Configuração:** Define a configuração de um serviço.
7. **Copiar Valor:** Copia o nome do serviço.
8. **Gerar Script:** Gera um script de todos os serviços selecionados com seus estados e configuração.
9. **Deletar:** Deleta o serviço.
10. **Roda Pé:** Mostra em qual computador está conectado e a quantidade de serviços listados.

* Mais:

[FORM](#_FORM). [SERVICE](#_SERVICE).

## TIMERS

Ferramenta para agendar tarefas.



1. **Timer:** Define o agendamento.
2. **Executar em:** Define como deve ser agendado.
3. **Agenda:** Data e hora do agendamento.
4. **Marcador:** para Mês e dias da semana.
5. **Comando:** Comando a ser executado.
6. **Lista:** Arquivo com as tarefas.
7. **Add:** Adiciona a tarefa na lista a baixo.
8. **Salvar:** Salva a lista a baixo.

* Mais:

[FORM](#_FORM).

# TECLAS

Teclas para utilizar funções gerais.

|  |  |
| --- | --- |
| **TECLA** | **FUNÇÃO** |
| ENTER | PGofer: Compila o comando digitado.  NotePad: Insere o “ENTER”. |
| ESQ | Oculta o PGofer, Console, Auto Complete. |
| ↑ | Move o cursor para cima |
| ↓ | Move o cursor para baixo. |
| PGUP | Repete os últimos comandos compilados. |
| PGDOWN | Avança os últimos comandos compilados. |
| SHIFT+ENTER | Insere o “ENTER” junto com os comandos. |

# SIMBOLOS

|  |  |
| --- | --- |
| **SIMBOLOS** | **DESCRIÇÃO** |
| . | Separa o comando do sub-comando.  Expressões numéricas definem os décimos. |
| ; | Termina um comando. |
| , | Separa os parâmetros |
| = | Comparação booleana de igualdade.  Retorna o calculo de uma expressão. |
| > | Comparação booleana de maior. |
| < | Comparação booleana de menor. |
| >= | Comparação booleana de maior igual. |
| <= | Comparação booleana de menor igual. |
| <> | Comparação booleana de diferente. |
| + | Soma. |
| - | Subtrai. |
| \* | Multiplica |
| / | Divide |
| ( | Abre parente em uma expressão. |
| ) | Fecha parente em uma expressão. |
| ^ | Eleva a potencia um valor numérico. |
| := | Atribui um valor para uma variável. |

# EXPRESSÕES NUMERICAS

Ulilizando as feramentas matematicas, pode ser usado notações e prefixo para facilitar os calculos. Os valores numericos sao representados no padrão ingles, ou seja, a vírgula “,” é substituida por ponto “.”.

* Lista

[NOTAÇÃO CIENTIFICA](#_NOTAÇÃO_CIENTIFICA), [PREFIXO INTERNACIONAL](#_PREFIXO_INTERNACIONAL), [BINARIO](#_BINARIO), [HEXADECIMAL](#_HEXADECIMAL).

## NOTAÇÃO CIENTIFICA

Trabalhando com notação cientifica, podemos simplificar muito a quantidade de “ZEROS” digitados em um valor. Então utilizase a expressão “potencia de 10”.

M \* 10N = M\*10^N = MeN

* Onde:

**“M”** é o valor numerico significativo.

**“e”** é a abreviação da potencia de 10.

**“N”** é o valor da potencia.

* Exemplo:

= 5e3 \* 3e-3 Resposta do calculo: “5000\*0,003” ou “5\*103 \* 3\*10-3” = 15

## PREFIXO INTERNACIONAL

Para simplificar mais, pode ser usado prefixo que representa valores da potencia de 10, este é o único caso que letras maiusculas e minusculas são diferenciadas.

* Tabela

|  |  |  |
| --- | --- | --- |
| Simbolo | Nome | Valor |
| y | Yocto | e-24 |
| z | Zepto | e-21 |
| a | Atto | e-18 |
| f | Femto | e-15 |
| p | Pico | e-12 |
| n | Nano | e-9 |
| u | Micro | e-6 |
| m | Mili | e-3 |
| --- |  |  |
| k | Quilo | e3 |
| M | Mega | e6 |
| G | Giga | e9 |
| T | Tera | e12 |
| P | Peta | e15 |
| E | Exa | e18 |
| Z | Zetta | e21 |
| Y | Yotta | e24 |

* Exemplo:

= 5k \* 3m Resposta do calculo: 5e3\*3e-3 = 15

* Mais:

[REPLYPREFIX](#_REPLYPREFIX).

## BINARIO

Para escrever algum valor em binario e ser convertido para decimal, basta colocar a expressão “0b” antes do numero.

0bN

* Onde

“N” é o valor numerico em binario.

* Exemplo:

= 0b1010 Resposta do calculo: 1010 para decimal é 10

* Mais:

[BIN](#_BIN).

## HEXADECIMAL

Para escrever algum valor em hexadecimal e ser convertido para decimal, basta colocar a expressão “0h” antes do numero.

0hN

* Onde

“N” é o valor numerico em hexadecimal.

* Exemplo:

= 0h3F Resposta do calculo: 3F para decimal é 63

* Mais:

[HEX](#_HEX).

# COMANDOS

Comandos gerais.

* Lista

[AND](#_AND), [BEGIN](#_BEGIN), [COPY](#_COPY), [CONST](#_CONST), [DELETE](#_DELETE), [DEC](#_DEC), [DO](#_DO), [DOWNTO](#_DOWNTO), [ELSE](#_ELSE), [END](#_END), [FOR](#_FOR), [FUNCTION](#_FUNCTION), [GLOBAL](#_GLOBAL), [IF](#_IF_ELSE), [INC](#_INC), [INSERT](#_LINK), [MOD](#_MOD), [NOT](#_NOT), [OR](#_OR), [READ](#_READ), [REPEAT](#_REPEAT), [ROOT](#_ROOT), [THEN](#_THEN), [TO](#_TO), [UNTIL](#_UNTIL), [VAR](#_VAR), [XOR](#_XOR), [WHILE](#_WHILE), [WRITE](#_WRITE).

## AND

Expressão booleana “E” que retorna uma resposta booleana.

* Tabela Logica:

|  |  |  |
| --- | --- | --- |
| A | B | Saída |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

* Exemplo:

A variável “C” vale 1 e a “D” vale 0:

= C **and** D Resposta: 0

* Mais:

[NOT](#_NOT), [OR](#_OR) e [XOR](#_XOR).

## BEGIN

“**begin**” Inicia um bloco de instruções e “**end**” fecha-o.

* Exemplo:

**begin**

write(“hellow world”);

**end**;

## COPY

Copia um intervalo do texto e retorna o trecho.

* Parâmetros:

**copy**( <TEXTO> , <INICIO> , <FIM> );

* Exemplo:

**copy**( ‘hellow world’ , 1 , 6 ); Resposta: “hellow”

* Mais:

[DELETE](#_DELETE) e [INSERT](#_INSERT).

## CONST

Referencia [FUNCITION](#_FUNCTION) ou [VAR](#_VAR,_GLOBAL).

## DELETE

Apaga um intervalo do texto e retorna o resto.

* Parâmetros:

**delete**( <TEXTO> , <INICIO> , <FIM> );

* Exemplo:

Texto := **delete** ( ‘hellow world’ , 1 , 6 ); Resposta: “world”

* Mais:

[COPY](#_COPY) e [INSERT](#_INSERT).

## DEC

Decrementa 1 na variável especifica.

* Parâmetros:

**dec**( <VARIAVEIS> );

* Exemplo:

Variável “A” vale 4:

**dec**(A); “A” valerá 3.

* Mais:

[INC](#_INC).

## DO

Referencia em [FOR](#_FOR) ou [WHILE](#_WHILE).

## DOWNTO

Referencia em [FOR](#_FOR).

## ELSE

Referencia em [IF](#_IF).

## END

Referencia em [BEGIM](#_BEGIN).

## FOR

Cria um loop com uma variável contadora, no loop a variável não pode ser alterada, a condição de parada é definida e testada no inicio.

* Parâmetros:

**for** <VARIÁVEL> **to** <VALOR MÁXIMO A SER ATINGIDO> **do** <COMANDOS>

**for** <VARIÁVEL> **downto** <VALOR MÍNIMO A SER ATINGIDO> **do** < COMANDOS >

* Exemplo:

**for** c := 0 **to** 10 **do**

**begin**

**write(**‘valor de c é: ’+c**);**

**end;**

* Mais:

[REPEAT](#_REPEAT), [WHILE](#_WHILE).

## FUNCTION

Cria uma função ou procedimento e retorna ou não qualquer valor.

* Parâmetros:

**function** <NOME> **(** <VARIAVEIS> **) begin** <COMANDOS> **end;**

**function global** <NOME> **(**<VARIAVEIS>**) begin** < COMANDOS > **end;**

* Exemplo:

**function** Mostrar **(**c,d**)**

**begin**

**write(**‘Mostra valor: “’+c+’”’**);**

**return** :=c+d**;**

**end;**

* Mais:

[LINK](#_LINK).

## GLOBAL

Referencia [FUNCITION](#_FUNCTION), [LINK](#_LINK) ou [VAR](#_VAR,_GLOBAL).

## IF

Decide se pode ou não executar comandos.

* Parâmetros:

**if** <EXPRESSÃO BOOLEANA> **then** <COMANDOS>

**if** <<EXPRESSÃO BOOLEANA> **then** <COMANDOS> **else** <COMANDOS>

* Exemplo:

c := **read**(‘Mostrar a verdade: (S)im ou (N)ão’, ‘S’);

**if** c = ‘S’ **then**

**begin**

**write(**‘A verdade’**);**

**end else begin**

**write(**‘A falcidade’**);**

**end;**

## INC

Incrementa uma variável.

* Parâmetros:

**inc(** <VARIAVEL> **);**

* Exemplo:

Variavel “A” vale 4:

**inc(**A**);** “A” valerá 5.

* Mais:

[DEC](#_DEC).

## INSERT

Insere um texto em outro texto e retorna tudo.

* Parâmetros:

**insert(** <TEXTO ORIGEM> , <TEXTO NOVO> , <POSIÇÃO> **);**

* Exemplo:

**insert(** ‘Hello Wolrd’ , ‘welcome ’ , 7**);** Resposta: ‘Hello welcome Wolrd’

* Mais:

[COPY](#_COPY) e [DELETE](#_DELETE).

## MOD

Calcula o resto da divisão entre dois números.

* Exemplo:

= 6 **mod** 2; Resposta: 0.

* Mais:

[ROOT](#_ROOT) e [MATH](#_MATH).

## NOT

Expressão booleana “NÃO”, inverte um valor booleano.

* Tabela Logica:

|  |  |
| --- | --- |
| A | Saída |
| 0 | 1 |
| 1 | 0 |

* Exemplo:

Variável “C” vale 1:

= **not** C; Resposta: 0

* Mais:

[AND](#_AND), [OR](#_OR) e [XOR](#_XOR).

## OR

Expressão booleana “OU”, que retorna uma resposta booleana.

* Tabela Logica:

|  |  |  |
| --- | --- | --- |
| A | B | Saída |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

* Exemplo:

Variável “C” vale 1 e “D” vale 0:

= C **or** D; Resposta: 1

* Mais:

[AND](#_AND), [NOT](#_NOT) e [XOR](#_XOR).

## READ

Uma caixa de dialogo que mostra um texto e pede um valor a ser digitado retornando-o.

* Parâmetros:

read( <TEXTO> , <VALOR PADRÃO>)

* Exemplo:

c := **read**(‘Digite um valor:’, 0);

## REPEAT

Cria um loop com contador variável, define e testa a condição de parada no final.

* Parâmetros:

**repeat** <COMANDOS> **until** **(** <EXPRESSÃO BOOLEANA> **);**

* Exemplo:

**repeat**

c := **read(**‘Digite “S” para sair’,’N’**);**

**until (** c <> ‘S’ **);**

* Mais:

[FOR](#_FOR), [WHILE](#_WHILE).

## ROOT

Calcula a raiz de um numero.

* Exemplo:

= 27 **root** 3; Resposta da raiz cubica de 27: 3.

* Mais:

[MOD](#_MOD) e [MATH](#_MATH).

## THEN

Referencia [IF](#_IF_ELSE).

## TO

Referencia em [FOR](#_FOR).

## UNTIL

Referencia em [REPEAT](#_REPEAT).

## VAR

Declara variáveis, constantes e globais.

* Parâmetros:

**var** <NOME DA VARIAVEL>**;**

**var global** <NOME DA VARIAVEL GLOBAL>**;**

**var const** <NOME DA CONSTANTE>**;**

**var const global** <NOME DA CONSTANTE GLOBAL>**;**

* Exemplo:

**var** Contador := 0; A variável “Contador” é criada e inicializada com o valor 0;

## XOR

Expressão booleana “OU Exclusivo”, que retorna uma resposta booleana “0 ou 1”.

* Tabela Logica:

|  |  |  |
| --- | --- | --- |
| A | B | Saída |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

* Exemplo:

= 0 **xor** 1; Resposta: 1.

* Mais:

[AND](#_AND), [NOT](#_NOT) e [OR](#_OR).

## WHILE

Cria um loop com contador variável, define a condição de parada e o teste o seu inicio.

* Parâmetros:

**while** <EXPRESSÃO BOOLEANA> **do** <COMANDOS>

* Exemplo:

C := 0;

**while** C < 10 **do**

**begin**

**write(**‘Valor de c: ‘, C**);**

**end;**

* Mais:

[FOR](#_FOR), [REPEAT](#_REPEAT).

## WRITE

Mostra um texto no console.

* Parâmetros:

**write(** <TEXTO> **);**

* Exemplo:

**write(** ‘Hellow wolrd’ **);**

# CLIPBOARD

Controla a área de transferência.

* Lista

[COPYFROMTEXT](#_COPYFROMTEXT), [CLEAR](#_CLEAR), [GETFORMAT](#_GETFORMAT), [LOADITEMFROMFILE](#_LOADITEMFROMFILE), [LOADFROMITEM](#_LOADFROMITEM), [PASTETOTEXT](#_PASTETOTEXT), [SAVEITEMTOFILE](#_SAVEITEMTOFILE), [SAVETOITEM](#_SAVETOITEM).

## COPYFROMTEXT

Copiar texto para a área de transferência.

* Parâmetros:

ClipBoard.CopyFromText( <TEXTO> );

* Exemplos:

ClipBoard.CopyFromText( ‘Hellow Wolrd.’ );

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## CLEAR

Limpa a área de transferência.

* Parâmetros:

ClipBoard.Clear;

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## GETFORMAT

Pega o formato ou tipo do dado que esta na área de transferência.

* Parâmetros:

ClipBoard.GetFormat;

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## LOADITEMFROMFILE

Carrega um arquivo para a lista de clipboard e retorna o numero do item.

* Parâmetros:

ClipBoard.LoadItemFromFile( <ARQUIVO> );

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## LOADFROMITEM

Carrega um item da lista de clipboard para a área de transferência.

* Parâmetros:

ClipBoard.LoadFromItem( <NUMERO DO ITEM> );

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## PASTETOTEXT

Pega o texto que está na área de transferência.

* Parâmetros:

ClipBoard.PasteToText;

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## SAVEITEMTOFILE

Salva um item da lista de clipboard para um arquivo.

* Parâmetros:

ClipBoard.SaveItemToFile( <NUMERO DO ITEM> , <ARQUIVO> );

* Referencia

[CLIPBOARD](#_CLIPBOARD).

## SAVETOITEM

Salva o dado que está na área de transferência para a lista do clipboard e retorna o numero do item.

* Parâmetros:

ClipBoard.SaveToItem;

* Referencia

[CLIPBOARD](#_CLIPBOARD).

# FILE

Controla os arquivos.

* Lista

[COPY](#_COPY_1), [DELETE](#_DELETE_1), [DIREXISTS](#_DIREXISTS), [EXEC](#_EXEC), [EXTRACTFILEEXT](#_EXTRACTFILEEXT), [EXTRACTFILENAME](#_EXTRACTFILENAME), [EXTRACTFILEPATH](#_EXTRACTFILEPATH), [FILEEXISTS](#_FILEEXISTS), [FILEEXPANDPATH](#_FILEEXPANDPATH), [FILEUNEXPANDPATH](#_FILEUNEXPANDPATH), [GETATTRIB](#_GETATTRIB), [GETFILESIZE](#_GETFILESIZE), [LOADFROMTEXT](#_LOADFROMTEXT), [MKDIR](#_MKDIR), [MOVE](#_MOVE), [OPENDIALOG](#_OPENDIALOG), [RENAME](#_RENAME), [SAVETOTEXT](#_SAVETOTEXT), [SCRIPT](#_SCRIPT), [SEARCH](#_SEARCH), [SETATTRIB](#_SETATTRIB).

## COPY

Copia arquivo ou pasta.

* OPÇÕES:

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flcNull | 1 | Nulo. |
| flcMultiDestFiles | 1 | Para Especificar múltiplos diretórios para cada arquivo de origem. |
| flcConfirmMouse | 2 | Tela de confirmação. |
| flcSilent | 4 | Não mostrar um progresso de caixa de diálogo. |
| flcRenameOnCollision | 8 | Cria um novo nome para o arquivo de destino se já existir igual no destino. |
| flcNoConfirmation | 16 | Responder “Sim” para todos para qualquer caixa de diálogo que é exibida. |
| flcWantMappingHandle | 32 | Para fofRenameOnCollision, cria uma lista dos arquivos. |
| flcAllOwUndo | 64 | Para que a operação possa ser desfeita. |
| flcFilesOnly | 128 | Executa a operação apenas em arquivos, ignora os diretórios. |
| flcSimpleProgress | 256 | Exibe uma caixa de diálogo de progresso simples. |
| flcNoConfirmMKDir | 512 | Não pergunta a criação de um novo diretório se a operação requer um a ser criado. |
| flcNoErrorUi | 1024 | Não exibir uma caixa de diálogo se um erro ocorre. |

* Parâmetros:

File.Copy( <ORIGEM> , <DESTINO> , <OPÇÕES> );

* Referencia

[FILE](#_FILE).

## DELETE

Apaga arquivo ou pasta.

* OPÇÕES:

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flcNull | 1 | Nulo. |
| flcMultiDestFiles | 1 | Para Especificar múltiplos diretórios para cada arquivo de origem. |
| flcConfirmMouse | 2 | Tela de confirmação. |
| flcSilent | 4 | Não mostrar um progresso de caixa de diálogo. |
| flcRenameOnCollision | 8 | Cria um novo nome para o arquivo de destino se já existir igual no destino. |
| flcNoConfirmation | 16 | Responder “Sim” para todos para qualquer caixa de diálogo que é exibida. |
| flcWantMappingHandle | 32 | Para fofRenameOnCollision, cria uma lista dos arquivos. |
| flcAllOwUndo | 64 | Para que a operação possa ser desfeita. |
| flcFilesOnly | 128 | Executa a operação apenas em arquivos, ignora os diretórios. |
| flcSimpleProgress | 256 | Exibe uma caixa de diálogo de progresso simples. |
| flcNoConfirmMKDir | 512 | Não pergunta a criação de um novo diretório se a operação requer um a ser criado. |
| flcNoErrorUi | 1024 | Não exibir uma caixa de diálogo se um erro ocorre. |

* Parâmetros:

File.Delete( <ARQUIVO> , <OPÇÕES>);

* Referencia

[FILE](#_FILE).

## DIREXISTS

Verifica se um diretório existe.

* Parâmetros:

File.DirExists( <DIRETÓRIO> );

* Referencia

[FILE](#_FILE).

## EXEC

Executa um arquivo.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flxHide | 0 | Janela no estado Oculta. |
| flxNormal | 1 | Janela no estado Normal. |
| flxMinimized | 2 | Janela no estado Minimizado. |
| flxMaximized | 3 | Janela no estado Maximizado. |

* Parâmetros:

File.Exec( <ARQUIVO> , <PARAMETRO> , <DIRETÓRIO> , <OPÇÕES> );

* Referencia

[FILE](#_FILE).

## EXTRACTFILEEXT

Retorna apenas o nome da extensão do arquivo.

* Parâmetros:

File.ExtractFileExt( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## EXTRACTFILENAME

Retorna apenas o nome do arquivo.

* Parâmetros:

File.ExtractFileName( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## EXTRACTFILEPATH

Retorna apenas o endereço de um arquivo.

* Parâmetros:

File.ExtractFilePath(<ARQUIVO> );

* Referencia

[FILE](#_FILE).

## FILEEXISTS

Verifica se um arquivo existe.

* Parâmetros:

File.FileExists( | );

* Referencia

[FILE](#_FILE).

## FILEEXPANDPATH

Converte o endereço de um arquivo ou pasta para formato padrão.

* Parâmetros:

File.FileExpandPath( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## FILEUNEXPANDPATH

Converte o endereço de um arquivo ou pasta para formato registro de NT.

* Parâmetros:

File.FileUnExpandPath( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## GETACESSTIME

Pega a data e hora de acesso do arquivo ou pasta.

* Parâmetros:

File. GetAcessTime ( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## GETATTRIB

Pega o atributo de um arquivo ou pasta.

* Opções de retorno:

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flaReadOnly | 1 | Somente leitura. |
| flaHidden | 2 | Oculto |
| flaSysFile | 4 | Sistema |
| flaVolumeID | 8 | Volume ID |
| flaDirectory | 16 | Diretório |
| flaArchive | 32 | Ficheiro |
| flaSymLink | 64 | Link Simbólico |

* Parâmetros:

File.GetAttrib( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## GETCREATETIME

Pega a data e hora de criação do arquivo ou pasta.

* Parâmetros:

File. GetCreateTime ( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## GETFILESIZE

Pega o tamanho de um arquivo.

* Parâmetros:

File.GetFileSize( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## GETMODIFYTIME

Pega a data e hora de modificação do arquivo ou pasta.

* Parâmetros:

File. GetModifyTime ( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## LOADFROMTEXT

Carrega o texto de um arquivo.

* Parâmetros:

File.LoadFromText( <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## MKDIR

Cria um diretorio.

* Parâmetros:

File.MkDir( <DIRETÓRIO> );

* Referencia

[FILE](#_FILE).

## MOVE

Move arquivo ou pasta.

* OPÇÕES:

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flcNull | 1 | Nulo. |
| flcMultiDestFiles | 1 | Para Especificar múltiplos diretórios para cada arquivo de origem. |
| flcConfirmMouse | 2 | Tela de confirmação. |
| flcSilent | 4 | Não mostrar um progresso de caixa de diálogo. |
| flcRenameOnCollision | 8 | Cria um novo nome para o arquivo de destino se já existir igual no destino. |
| flcNoConfirmation | 16 | Responder “Sim” para todos para qualquer caixa de diálogo que é exibida. |
| flcWantMappingHandle | 32 | Para fofRenameOnCollision, cria uma lista dos arquivos. |
| flcAllOwUndo | 64 | Para que a operação possa ser desfeita. |
| flcFilesOnly | 128 | Executa a operação apenas em arquivos, ignora os diretórios. |
| flcSimpleProgress | 256 | Exibe uma caixa de diálogo de progresso simples. |
| flcNoConfirmMKDir | 512 | Não pergunta a criação de um novo diretório se a operação requer um a ser criado. |
| flcNoErrorUi | 1024 | Não exibir uma caixa de diálogo se um erro ocorre. |

* Parâmetros:

File.Move( <ORIGEM> , <DESTINO> , <OPÇÕES> );

* Referencia

[FILE](#_FILE).

## OPENDIALOG

Abre uma janela de dialogo para escolher arquivo ou pasta, retornando o mesmo.

* Parâmetros:

File.OpenDialog( <ARQUIVO PADRÃO> );

* Referencia

[FILE](#_FILE).

## RENAME

Renomeia arquivo ou pasta.

* OPÇÕES:

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flcNull | 1 | Nulo. |
| flcMultiDestFiles | 1 | Para Especificar múltiplos diretórios para cada arquivo de origem. |
| flcConfirmMouse | 2 | Tela de confirmação. |
| flcSilent | 4 | Não mostrar um progresso de caixa de diálogo. |
| flcRenameOnCollision | 8 | Cria um novo nome para o arquivo de destino se já existir igual no destino. |
| flcNoConfirmation | 16 | Responder “Sim” para todos para qualquer caixa de diálogo que é exibida. |
| flcWantMappingHandle | 32 | Para fofRenameOnCollision, cria uma lista dos arquivos. |
| flcAllOwUndo | 64 | Para que a operação possa ser desfeita. |
| flcFilesOnly | 128 | Executa a operação apenas em arquivos, ignora os diretórios. |
| flcSimpleProgress | 256 | Exibe uma caixa de diálogo de progresso simples. |
| flcNoConfirmMKDir | 512 | Não pergunta a criação de um novo diretório se a operação requer um a ser criado. |
| flcNoErrorUi | 1024 | Não exibir uma caixa de diálogo se um erro ocorre. |

* Parâmetros:

File.Rename( <ORIGEM> , <DESTINO> , <OPÇÕES> );

* Referencia

[FILE](#_FILE).

## SAVETOTEXT

Salva um texto em um arquivo.

* Parâmetros:

File.SaveToText( <ARQUIVO> , <TEXTO> );

* Referencia

[FILE](#_FILE).

## SCRIPT

Executa um script em Pascal PGofer.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flsIdle | 0 | Prioridade Inativa |
| flsLowest | 1 | Prioridade Mais Baixa |
| flsLower | 2 | Prioridade Baixa |
| flsNormal | 3 | Prioridade Normal |
| flsHigher | 4 | Prioridade Alta |
| flsHighest | 5 | Prioridade Mais Alta |
| flsTimeCritical | 6 | Prioridade Tempo Real |

* Parâmetros:

File.Script( <ARQUIVO> , <OPÇÕES> , <MENSAGEM> );

* Referencia

[FILE](#_FILE).

## SEARCH

Procura um arquivo ou pasta.

* Parâmetros:

File.Search( <DIRETÓRIO> , <ARQUIVO> );

* Referencia

[FILE](#_FILE).

## SETATTRIB

Seta atributos a um arquivo ou pasta.

* OPÇÕES:

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flaReadOnly | 1 | Somente leitura. |
| flaHidden | 2 | Oculto |
| flaSysFile | 4 | Sistema |
| flaVolumeID | 8 | Volume ID |
| flaDirectory | 16 | Diretório |
| flaArchive | 32 | Ficheiro |
| flaSymLink | 64 | Link Simbólico |

* Parâmetros:

File.SetAttrib( <ARQUIVO> , <ATRIBUTOS> );

* Referencia

[FILE](#_FILE).

## SETDATETIME

Seta a data de criação, modificação e acesso de um arquivo ou pasta.

* Parâmetros:

File. SetDateTime ( <ARQUIVO>, <CREATETIME>, <MODIFYTIME>, <ACESSTIME>);

* Referencia

[FILE](#_FILE).

# FORM

Controla e executa sub-comandos dos formulários: [PGOFER](#_PGOFER_2), [AUTOCOMPLETES](#_AUTOCOMPLETES) ,[CLIPBOARDS](#_CLIPBOARDS) ,[CONSOLES](#_HOTKEYS) ,[HOTKEYS](#_HOTKEYS) ,[LINKS](#_LINKS) ,[NOTEPADS](#_NOTEPADS) ,[SERVICES](#_SERVICES) ,[TIMERS](#_TIMERS).

* Lista

[ALPHABLEND](#_ALPHABLEND), [ALPHABLENDVALUE](#_ALPHABLENDVALUE), [CLOSE](#_CLOSE), [ENABLED](#_ENABLED), [HEIGTH](#_HEIGTH), [HIDE](#_HIDE), [LEFT](#_LEFT), [SHOW](#_SHOW), [SETFOCUS](#_SETFOCUS), [TOP](#_TOP), [TRANSPARENT](#_TRANSPARENT), [TRANSPARENTCOLOR](#_TRANSPARENTCOLOR), [WIDTH](#_WIDTH), [WINDOWSTATE](#_WINDOWSTATE).

## ALPHABLEND

Habilita o controle da transparência no formulário.

* Parâmetros:

<FORM>.AlphaBlend := <BOOLEANO> ;

* Referencia

[FORM](#_FORM).

## ALPHABLENDVALUE

Ajusta a transparência do formulário, 0 – 0%, 255 – 100% opaco.

* Parâmetros:

<FORM>.AlphaBlendValue := <VALOR> ;

* Referencia

[FORM](#_FORM).

## CLOSE

Fecha o formulário.

* Parâmetros:

<FORM>.Close;

* Referencia

[FORM](#_FORM).

## ENABLED

Habilita ou desabilita o formulário.

* Parâmetros:

<FORM>.Enabled := <BOOLEANO> ;

* Referencia

[FORM](#_FORM).

## HEIGTH

Ajusta a altura do formulário.

* Parâmetros:

<FORM>.Heigth := <VALOR> ;

* Referencia

[FORM](#_FORM).

## HIDE

Oculta o formulário.

* Parâmetros:

<FORM>.Hide;

* Referencia

[FORM](#_FORM).

## LEFT

Ajusta a posição do formulário no eixo X.

* Parâmetros:

<FORM>.Left := <VALOR> ;

* Referencia

[FORM](#_FORM).

## SHOW

Mostra o formulário.

* Parâmetros:

<FORM>.Show;

* Referencia

[FORM](#_FORM).

## SETFOCUS

Mostra e seta o foco no formulário.

* Parâmetros:

<FORM>.SetFocus;

* Referencia

[FORM](#_FORM).

## TOP

Ajusta a posição do formulário no eixo y.

* Parâmetros:

<FORM>.Top := <VALOR> ;

* Referencia

[FORM](#_FORM).

## TRANSPARENT

Habilita a transparência do formulário.

* Parâmetros:

<FORM>.Transparent := <BOOLEANO> ;

* Referencia

[FORM](#_FORM).

## TRANSPARENTCOLOR

Define a cor de transparência do formulário.

* Parâmetros:

<FORM>.TransparentColor := <COR> ;

* Referencia

[FORM](#_FORM).

## WIDTH

Ajusta a largura do formulário.

* Parâmetros:

<FORM>.Width := <VALOR> ;

* Referencia

[FORM](#_FORM).

## WINDOWSTATE

Ajusta o estado do formulário.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flxHide | 0 | Janela no estado Oculta. |
| flxNormal | 1 | Janela no estado Normal. |
| flxMinimized | 2 | Janela no estado Minimizado. |
| flxMaximized | 3 | Janela no estado Maximizado. |

* Parâmetros:

<FORM>.WindowState := <OPÇÃO> ;

* Referencia

[FORM](#_FORM).

# KEY

Controla teclas.

* Lista

[ALLUP](#_ALLUP), [CHARTOKEY](#_CHARTOKEY), [GETPRESS](#_GETPRESS), [KEYTOCHAR](#_KEYTOCHAR), [SETPRESS](#_SETPRESS), [MACRO](#_MACRO).

## ALLUP

Solta todas as teclas, até as virtuais.

* Parâmetros:

Key.AllUp;

* Referencia

[KEY](#_KEY).

## CHARTOKEY

Converte um caractere para valor numérico correspondente da tabela ASCII.

* Parâmetros:

Key.CharToKey( <CARACTERE> );

* Referencia

[KEY](#_KEY).

## GETPRESS

Verifica se uma tecla está pressionada.

* Parâmetros:

Key.GetPress( <VALOR> );

* Referencia

[KEY](#_KEY).

## KEYTOCHAR

Converte um valor numérico em caractere.

* Parâmetros:

Key.KeyToChar( <VALOR> );

* Referencia

[KEY](#_KEY).

## SETPRESS

Seta o estado de uma tecla.

* Parâmetros:

Key.SetPress( <TECLA> , <BOOLEANO> );

* Referencia

[KEY](#_KEY).

## MACRO

Executa uma sequencia de pressionamento de teclas por um texto.

* Parâmetros:

Key.Macro( <TEXTO> , <VELOCIDADE> );

* Referencia

[KEY](#_KEY).

# LINK

Cria ou acessa atalhos locais ou globais para arquivos.

* Parâmetros:

**link** <NOME>;

**link** **global** <NOME>;

## DIRECTORY

Define o endereço do diretorio corrente do arquivo.

* Parâmetros:

<LINK>.Directory := ‘<DIRETORIO>’;

* Referencia

[LINK](#_MATH).

## FILE

Define o endereço do arquivo.

* Parâmetros:

< LINK >.File := ‘<ARQUIVO>’;

* Referencia

[LINK](#_MATH).

## ICON

Define o endereço do ícone para o arquivo.

* Parâmetros:

< LINK >.Icon := ‘<ICONE>,<NUMERO DO ICONE>’;

* Referencia

[LINK](#_MATH).

## PARAMETER

Define o parâmetro do arquivo.

* Parâmetros:

< LINK >.Parameter := ‘<PARAMETRO>’;

* Referencia

[LINK](#_MATH).

## PARAMEXEC

Executa um link com o parâmetro definido, não alterar o parâmetro anterior.

* Parâmetros:

< LINK >.ParamExec := ‘<PARAMETRO>’;

* Referencia

[LINK](#_MATH).

## SHOWCONTROL

Define o estilo da janela.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flxHide | 0 | Janela no estado Oculta. |
| flxNormal | 1 | Janela no estado Normal. |
| flxMinimized | 2 | Janela no estado Minimizado. |
| flxMaximized | 3 | Janela no estado Maximizado. |

* Parâmetros:

< LINK >.ShowControl := <OPÇÕES>;

* Referencia

[LINK](#_MATH).

# MATH

Formulas matematicas.

* Lista

[ARCCOS](#_ARCCOS), [ARCSIN](#_ARCSIN), [ARCTAN](#_ARCTAN), [COS](#_COS), [COSECANT](#_COSECANT), [COTAN](#_COTAN), [BIN](#_BIN), [DEGTOGRAD](#_DEGTOGRAD), [DEGTORAD](#_DEGTORAD), [FORMAT](#_FORMATNUMBER), [GRADTODEG](#_GRADTODEG), [GRADTORAD](#_GRADTORAD), [HEX](#_HEX), [HYPOT](#_HYPOT), [LOG](#_LOG), [MODULE](#_MODULE), [RADTODEG](#_RADTODEG), [RADTOGRAD](#_RADTOGRAD), [SECANT](#_SECANT), [SIN](#_SIN), [TAN](#_TAN).

## ARCCOS

Arco coseno.

* Parâmetros:

Math.ArcCos( <VALOR> );

* Referencia

[MATH](#_MATH).

## ARCSIN

Arco seno.

* Parâmetros:

Math.ArcSin( <VALOR> );

* Referencia

[MATH](#_MATH).

## ARCTAN

Arco tangente.

* Parâmetros:

Math.ArcTan( <VALOR> );

* Referencia

[MATH](#_MATH).

## COS

Cosseno.

* Parâmetros:

Math.Cos( <VALOR> );

* Referencia

[MATH](#_MATH).

## COSECANT

Cosecante.

* Parâmetros:

Math.Cosecant( <VALOR> );

* Referencia

[MATH](#_MATH).

## COTAN

Co-Tangente.

* Parâmetros:

Math.CoTan( <VALOR> );

* Referencia

[MATH](#_MATH).

## BIN

Converte para Binario.

* Parâmetros:

Math.Bin( <VALOR> );

* Referencia

[MATH](#_MATH).

## DEGTOGRAD

Converte degrau para Gradiano.

* Parâmetros:

Math.DegToGrad( <VALOR> );

* Referencia

[MATH](#_MATH).

## DEGTORAD

Converte Degrau para Radiano.

* Parâmetros:

Math.DegToRad( <VALOR> );

* Referencia

[MATH](#_MATH).

## FORMAT

Converte valor para um formato em texto com ou sem prefixo.

* Formatos

|  |  |
| --- | --- |
| Simbolo | Descrição |
| d | Decimal (integer) |
| e | Scientific |
| f | Fixed |
| g | General |
| m | Money |
| n | Number (floating) |
| p | Pointer |
| s | String |
| u | Unsigned decimal |
| x | Hexadecimal |
| 0 | Quantidade de casas representadas, ou aparece “ZERO”. |
| # | Quantidade de casas, ou não aparece nada. |
| . | Define o ponto decimal |
| , | Define as centenas |
| E+ | Mostra em notação de potencia de 10 positivos |
| E- | Mostra em notação de potencia de 10 negativos. |
| ; | Separa positivo, negativo e zero. |

* Parâmetros:

Math.Format(<PREFIXO>, <FORMATO>, <VALOR> );

* Referencia

[MATH](#_MATH).

## GRADTODEG

Converte Gradiano para Degrau.

* Parâmetros:

Math.GradToDeg( <VALOR> );

* Referencia

[MATH](#_MATH).

## GRADTORAD

Converte Gradiano para Radiano.

* Parâmetros:

Math.GradToRad( <VALOR> );

* Referencia

[MATH](#_MATH).

## HEX

Converte para Hexadecimal

* Parâmetros:

Math.Hex( <VALOR> );

* Referencia

[MATH](#_MATH).

## HYPOT

Hypotenusa.

* Parâmetros:

Math.Hypot( <VALOR> , <VALOR> );

* Referencia

[MATH](#_MATH).

## LOG

Logaritmo.

* Parâmetros:

Math.Log( <VALOR> , <VALOR> );

* Referencia

[MATH](#_MATH).

## MODULE

Modulo absoluto.

* Parâmetros:

Math.Module( <VALOR> );

* Referencia

[MATH](#_MATH).

## RANDOM

Retorna um numero aleatório entre 0 (zero) ao valor de entrada, no segundo parâmetro pode recriar os números aleatórios antes de retornar o valor.

* Parâmetros:

Math.Random( <VALOR> ); Math.Random( <VALOR>, <BOLEANO>);

* Referencia

[MATH](#_MATH).

## RADTODEG

Converte Radiano para Degrau.

* Parâmetros:

Math.RadToDeg( <VALOR> );

* Referencia

[MATH](#_MATH).

## RADTOGRAD

Converte Radiano para Gradiano.

* Parâmetros:

Math.RadToGrad( <VALOR> );

* Referencia

[MATH](#_MATH).

## SECANT

Secante

* Parâmetros:

Math.Secant( <VALOR> );

* Referencia

[MATH](#_MATH).

## SIN

Seno

* Parâmetros:

Math.Sin( <VALOR> );

* Referencia

[MATH](#_MATH).

## TAN

Tangente

* Parâmetros:

Math.Tan( <VALOR> );

* Referencia

[MATH](#_MATH).

# NET

Configuração e acesso remoto.

* Lista

[CLIENT](#_Client), [SERVER](#_Server), [SETTCPIP](#_SETTCPIP) .

## CLIENT

Configura o cliente para conectar e enviar comandos ao servidor.

* Lista

[ACTIVE](#_ACTIVE), [ADDRESS,](#_ADDRESS) [PASSWORD](#_PASSWORD), [PORT](#_PORT), [SENDCOMMAND](#_SENDCOMMAND).

### ACTIVE

Ativa a ou desativa a conexção.

* Parâmetros:

Net.Client.Actrive := <BOOLEANO> ;

* Referencia

[NET](#_NET) , [CLIENT](#_Client).

### ADDRESS

Configura o endereço a ser conectado.

* Parâmetros:

Net.Client.Address := <ENDEREÇO> ;

* Referencia

[NET](#_NET) , [CLIENT](#_Client).

### PASSWORD

Configura o password para ser conectado.

* Parâmetros:

Net.Client.Password := <PASSWORD> ;

* Referencia

[NET](#_NET) , [CLIENT](#_Client).

### PORT

Configura a porta a ser conectada.

* Parâmetros:

Net.Client.Port := <PORTA> ;

* Referencia

[NET](#_NET) , [CLIENT](#_Client).

### SENDCOMMAND

Envia o comando a ser processado.

* Parâmetros:

Net.Client.SendCommand( <COMANDO> );

* Referencia

[NET](#_NET) , [CLIENT](#_Client).

## SERVER

Configura o servidor para ser conectado e processar os comandos.

* Lista

[ACTIVE](#_ACTIVE_1), [MAXCONNECT,](#_MAXCONNECT) [PASSWORD](#_PASSWORD_1), [PORT](#_PORT_1), [SENDMESSAGE](#_SENDMESSAGE_1).

### ACTIVE

Ativa a ou desativa a conexção.

* Parâmetros:

Net.Server.Actrive := <BOOLEANO> ;

* Referencia

[NET](#_NET) , [SERVER](#_Server).

### MAXCONNECT

Define o numero maximo de conecções simultaneas.

* Parâmetros:

Net.Server.MaxConnect := <Valor> ;

* Referencia

[NET](#_NET) , [SERVER](#_Server).

### PASSWORD

Configura o password para ser conectado.

* Parâmetros:

Net.Client.Password := <PASSWORD> ;

* Referencia

[NET](#_NET) , [SERVER](#_Server).

### PORT

Configura a porta a ser conectada.

* Parâmetros:

Net.Server.Port := <PORTA> ;

* Referencia

[NET](#_NET) , [SERVER](#_Server).

### SENDMESSAGE

Envia mensagens ao cliente.

* Parâmetros:

Net.Server.SendMessage( <Mensagem> );

* Referencia

[NET](#_NET) , [SERVER](#_Server).

## SETTCPIP

Configura IP, Mascara e Gateway de um determinado dispositivo de rede.

* Parâmetros:

Net.SetTCPIP( <DISPOSITIVO>, <IP>, <MASK>, <GATEWAY>);

* Referencia

[NET](#_NET).

# PROCESS

Controla os processos.

* Lista

[FINDFILE](#_FINDFILE), [GETFILENAME](#_GETFILENAME), [GETPRIORITY](#_GETPRIORITY), [KILL](#_KILL), [SETPRIORITY](#_SETPRIORITY).

## FINDFILE

Procura um processo e retorna seu PID.

* Parâmetros:

Process.FindFile( <ARQUIVO> );

* Referencia

[PROCESS](#_PROCESS).

## GETFILENAME

Converte PID para nome do arquivo.

* Parâmetros:

Process.GetFileName( <PID> );

* Referencia

[PROCESS](#_PROCESS).

## GETFOREGROUND

Retorna o PID do processo que esta em foco.

* Parâmetros:

Process.GetForeground;

* Referencia

[PROCESS](#_PROCESS).

## GETPRIORITY

Retorna a prioridade do arquivo.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flsIdle | 0 | Prioridade Inativa |
| flsLowest | 1 | Prioridade Mais Baixa |
| flsLower | 2 | Prioridade Baixa |
| flsNormal | 3 | Prioridade Normal |
| flsHigher | 4 | Prioridade Alta |
| flsHighest | 5 | Prioridade Mais Alta |
| flsTimeCritical | 6 | Prioridade Tempo Real |

* Parâmetros:

Process.GetPriority( <PID> );

* Referencia

[PROCESS](#_PROCESS).

## KILL

Mata o processo.

* Parâmetros:

Process.Kill( <PID> );

* Referencia

[PROCESS](#_PROCESS).

## SETPRIORITY

Seta prioridade para um processo.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| flsIdle | 0 | Prioridade Inativa |
| flsLowest | 1 | Prioridade Mais Baixa |
| flsLower | 2 | Prioridade Baixa |
| flsNormal | 3 | Prioridade Normal |
| flsHigher | 4 | Prioridade Alta |
| flsHighest | 5 | Prioridade Mais Alta |
| flsTimeCritical | 6 | Prioridade Tempo Real |

* Parâmetros:

Process.SetPriority( <PID> , <OPÇÃO> );

* Referencia

[PROCESS](#_PROCESS).

# REGISTRY

Controla registros.

* Lista

[DELETE](#_DELETE_2), [READ](#_READ_1), [WRITE](#_WRITE_1).

* Chaves

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| hkeyClassesRoot | 0X80000000 | HKEY\_CLASSES\_ROOT |
| hkeyCurrentUser | 0X80000001 | HKEY\_CURRENT\_USER |
| hkeyLocalMachine | 0X80000002 | HKEY\_LOCAL\_MACHINE |
| hkeyUsers | 0X80000003 | HKEY\_USERS |
| hkeyPerformData | 0X80000004 | HKEY\_PERFORMANCE\_DATA |
| hkeyCurrentConfig | 0X80000005 | HKEY\_CURRENT\_CONFIG |
| hkeyDynData | 0X80000006 | HKEY\_DYN\_DATA |

## DELETE

Apaga um registro ou pasta.

* Parâmetros:

Registry.Delete( <CHAVE> , <ENDEREÇO> , <REGISTRO> );

* Referencia

[REGISTRY](#_REGISTRY).

## READ

Le o valor de um registro.

* Parâmetros:

Registry.Read( <CHAVE> , <ENDEREÇO> , <REGISTRO>);

* Referencia

[REGISTRY](#_REGISTRY).

## WRITE

Escreve em um registro.

* Parâmetros:

Registry.Write( <CHAVE> , <ENDEREÇO> , <REGISTRO>, <VALOR> );

* Referencia

[REGISTRY](#_REGISTRY).

# SERVICE

Controla os serviços.

* Lista

[CREATE](#_CREATE), [DELETE](#_DELETE_3), [GETCONFIG](#_GETCONFIG), [GETDESCIPTION](#_GETDESCIPTION), [GETSTATE](#_GETSTATE), [SETCONFIG](#_SETCONFIG), [SETDESCIPTION](#_SETDESCIPTION), [SETSTATE](#_SETSTATE).

## CREATE

Cria um serviço.

* Parâmetros:

Service.Create( <MAQUINA> , <SERVIÇO> , <NOME> , <ARQUIVO>);

* Referencia

[SERVICE](#_SERVICE).

## DELETE

Apaga um serviço

* Parâmetros:

Service.Delete( <MAQUINA> , <SERVIÇO> );

* Referencia

[SERVICE](#_SERVICE).

## GETCONFIG

Retorna a configuração de um serviço.

* Opções de retorno

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| svcBootStart | 0 | Inicia com o Boot |
| svcSystemStart | 1 | Inicia com o Sistema |
| svcAutoStart | 2 | Inicia com o Login |
| svcDemandStart | 3 | Inicia Manualmente |
| svcDisabled | 4 | Desabilitado |

* Parâmetros:

Service.GetConfig( <MAQUINA> , <SERVIÇO> );

* Referencia

[SERVICE](#_SERVICE).

## GETDESCIPTION

Retorna a descrição de um serviço.

* Parâmetros:

Service.GetDesciption( <MAQUINA> , <SERVIÇO> );

* Referencia

[SERVICE](#_SERVICE).

## GETSTATE

Retorna o estado de um serviço.

* Opções de retorno

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| svsStop | 1 | Serviço parado |
| svsRunning | 2 | Iniciando o serviço |
| svsStoping | 3 | Parando o serviço |
| svsRun | 4 | Serviço iniciado |
| svsPending | 5 | Serviço Pendente |
| svsPausing | 6 | Pausando o serviço |
| svsPause | 7 | Serviço pausado |

* Parâmetros:

Service.GetState( <MAQUINA> , <SERVIÇO> );

* Referencia

[SERVICE](#_SERVICE).

## SETCONFIG

Seta a configuração de um serviço.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| svcBootStart | 0 | Inicia com o Boot |
| svcSystemStart | 1 | Inicia com o Sistema |
| svcAutoStart | 2 | Inicia com o Login |
| svcDemandStart | 3 | Inicia Manualmente |
| svcDisabled | 4 | Desabilitado |

* Parâmetros:

Service.SetConfig( <MAQUINA> , <SERVIÇO> , <OPÇÃO> );

* Referencia

[SERVICE](#_SERVICE).

## SETDESCIPTION

Seta a descrição de um serviço.

* Parâmetros:

Service.SetDesciption( <MAQUINA> , <SERVIÇO> , <DESCRIÇÃO> );

* Referencia

[SERVICE](#_SERVICE).

## SETSTATE

Seta o estado de um serviço.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| svsStop | 1 | Serviço parado |
| svsRunning | 2 | Iniciando o serviço |
| svsStoping | 3 | Parando o serviço |
| svsRun | 4 | Serviço iniciado |
| svsPending | 5 | Serviço Pendente |
| svsPausing | 6 | Pausando o serviço |
| svsPause | 7 | Serviço pausado |

* Parâmetros:

Service.SetState( <MAQUINA> , <SERVIÇO> , <OPÇÃO> );

* Referencia

[SERVICE](#_SERVICE).

# SOUND

Controla o som.

* Lista

[GETVOLUME](#_GETVOLUME), [MUTE](#_MUTE), [PLAYSOUND](#_PLAYSOUND), [SETVOLUME](#_SETVOLUME), [VOLUMESTEPDOWN](#_VOLUMESTEPDOWN), [VOLUMESTEPUP](#_VOLUMESTEPUP).

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| sndrvDefault | 0 | Placa de som padrão |
| sndrvMultimi | 1 | Placa de som multimídia padrão |
| sndrvComunic | 2 | Placa de som comunicação padrão |

## GETVOLUME

Retorna o volume.

* Parâmetros:

Sound.GetVolume( <PLACA DE SOM> );

* Referencia

[SOUND](#_SOUND).

## MUTE

Mudo.

* Parâmetros:

Sound.Mute( <PLACA DE SOM> );

* Referencia

[SOUND](#_SOUND).

## PLAYSOUND

Toca um arquivo de áudio.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| sndcSync | 0 | Espera terminar |
| sndcAsync | 1 | Não espera terminar |
| sndcNoDefault | 2 | Não usa a placa de som padrão |
| sndcMemory | 4 | Toca som da memoria (Não usado) |
| sndcLoop | 8 | Repete o som até tocar um próximo |
| sndcNoStop | 16 | Não para de tocar o som se vier tocar um próximo |

* Parâmetros:

Sound.PlaySound( <PLACA DE SOM> , <OPÇÃO> );

* Referencia

[SOUND](#_SOUND).

## SETVOLUME

Seta o volume.

* Parâmetros:

Sound.SetVolume( <PLACA DE SOM> , <VOLUME> );

* Referencia

[SOUND](#_SOUND).

## VOLUMESTEPDOWN

Decrementa o volume.

* Parâmetros:

Sound.VolumeStepDown( <PLACA DE SOM> );

* Referencia

[SOUND](#_SOUND).

## VOLUMESTEPUP

Incrementa o volume

* Parâmetros:

Sound.VolumeStepUp( <PLACA DE SOM> );

* Referencia

[SOUND](#_SOUND).

# SYSTEM

Controle do Windows e do PGofer.

* Lista

[AUTOCLOSE](#_AUTOCLOSE), [CONSOLECLEAR](#_CONSOLECLEAR), [CONSOLEDELAY](#_CONSOLEDELAY), [CONSOLEMSG](#_CONSOLEMSG), [DATETIMENOW](#_DATETIMENOW), [DIRCURRENT](#_DIRCURRENT), [DELAY](#_DELAY), [FILELISTMAX](#_FILELISTMAX), [FINDWINDOW](#_FINDWINDOW), [FORMATREPLY](#_FORMATREPLY), [ICONLOADER](#_ICONLOADER), [LOCKWORKSTATION](#_LOCKWORKSTATION), [LOOPLIMITE](#_LOOPLIMITE), [NOOFF](#_NOOFF), [PRTSCREEN](#_PRTSCREEN), [SENDMESSAGE](#_SENDMESSAGE), [SETSCREEN](#_SETSCREEN), [SETSUSPENDSTATE](#_SETSUSPENDSTATE), [SHOWMESSAGE](#_SHOWMESSAGE), [SHUTDOWN](#_SHUTDOWN).

## AUTOCLOSE

Oculta automaticamente o PGofer ao pressionar “Enter”.

* Parâmetros:

System.AutoClose := <BOOLEANO> ;

* Referencia

[SYSTEM](#_SYSTEM).

## CONSOLECLEAR

Limpa o console.

* Parâmetros:

System.ConsoleClear;

* Referencia

[SYSTEM](#_SYSTEM).

## CONSOLEDELAY

Tempo de amostra do console.

* Parâmetros:

System.ConsoleDelay := <VALOR> ;

* Referencia

[SYSTEM](#_SYSTEM).

## CONSOLEMSG

Habilita ou não mensagem no console.

* Parâmetros:

System.ConsoleMsg := <BOOLEANO> ;

* Referencia

[SYSTEM](#_SYSTEM).

## DATETIMENOW

Retorna a data e a hora do momento de acordo com o formato.

Formato:

|  |  |  |
| --- | --- | --- |
| **Simbolo** | **Descrição** | **Resultado** |
| c | data | 00/00/00 |
| t | tempo | 0:0:0 |
| tt | tempo | 00:00:00 |
| d | dia | (1-31) |
| dd | dia | 01-31 |
| ddd | semana | Dom-Sab |
| dddd | semana | Domingo-Sabado |
| m | mês | 1-12 |
| mm | mês | 01-12 |
| mmm | mês | Jan-Dez |
| mmmm | mês | Janero-Dezembro |
| yy | ano | 00-99 |
| yyyy | ano | 0000-9999 |
| h | hora | 0-23 |
| hh | hora | 00-23 |
| n | minuto | 0-59 |
| nn | minuto | 00-59 |
| s | segundo | 0-59 |
| ss | segundo | 00-59 |
| z | milisegundo | 0-999 |
| zzz | milisegundo | 000-999 |
| am/pm | am/pm | am-pm |

* Parâmetros:

System.DateTimeNow( <FORMATO> );

* Referencia

[SYSTEM](#_SYSTEM).

## DIRCURRENT

Constante com o endereço do diretório do PGofer.

* Parâmetros:

System.DirCurrent;

* Referencia

[SYSTEM](#_SYSTEM).

## DELAY

Espera um tempo em milissegundos.

* Parâmetros:

System.Delay( <TEMPO> );

* Referencia

[SYSTEM](#_SYSTEM).

## FILELISTMAX

Variável que define o tamanho máximo da lista de arquivos.

* Parâmetros:

System.FileListMax := <VALOR> ;

* Referencia

[SYSTEM](#_SYSTEM).

## FINDWINDOW

Procura se um aplicativo está rodando, e retorna seu Class ID.

* Parâmetros:

System.FindWindow( <NOME> );

* Referencia

[SYSTEM](#_SYSTEM).

## FORMATREPLY

Define o formanto do valor a ser mostrado após um calculo, com ou sem prefixo e um formato especifico.

* Formatos

|  |  |
| --- | --- |
| Simbolo | Descrição |
| d | Decimal (integer) |
| e | Scientific |
| f | Fixed |
| g | General |
| m | Money |
| n | Number (floating) |
| p | Pointer |
| s | String |
| u | Unsigned decimal |
| x | Hexadecimal |
| 0 | Quantidade de casas representadas, ou aparece “ZERO”. |
| # | Quantidade de casas, ou não aparece nada. |
| . | Define o ponto decimal |
| , | Define as centenas |
| E+ | Mostra em notação de potencia de 10 positivos |
| E- | Mostra em notação de potencia de 10 negativos. |
| ; | Separa positivo, negativo e zero. |

* Parâmetros:

System. FormatReply( <BOOLEANO>, <FORMAT>);

* Referencia

[FORMATNUMBER](#_FORMATNUMBER), [SYSTEM](#_SYSTEM).

## ICONLOADER

Habilita ou não ícones para os arquivos no PGofer.

* Parâmetros:

System.IconLoader := <BOOLEANO> ;

* Referencia

[SYSTEM](#_SYSTEM).

## LOCKWORKSTATION

Tranca o computador para a tela de senha.

* Parâmetros:

System.LockWorkStation;

* Referencia

[SYSTEM](#_SYSTEM).

## LOOPLIMITE

Variável que define a quantidade máxima de loops para comandos de loop.

* Parâmetros:

System.LoopLimite := <VALOR> ;

* Referencia

[SYSTEM](#_SYSTEM).

## MONITORPOWER

Liga ou desliga o monitor.

* Parâmetros:

System.MonitorPower( <BOOLEANO> );

* Referencia

[SYSTEM](#_SYSTEM).

## NOOFF

Permite ou não que outros programas além do PGofer possa desligar o computador.

* Parâmetros:

System.NoOff := <BOOLEANO> ;

* Referencia

[SYSTEM](#_SYSTEM).

## PRTSCREEN

Tira uma foto da tela.

* Parâmetros:

System.PrtScreen( <ALTURA> , <LARGURA> , <Y> , <X> , <ARQUIVO> );

* Referencia

[SYSTEM](#_SYSTEM).

## SENDMESSAGE

Envia uma mensagem para um aplicativo.

* Parâmetros:

System.SendMessage( <CLASSID> , <MESSAGE> , <WPAR> , <LPAR> );

* Referencia

[SYSTEM](#_SYSTEM).

## SETSCREEN

Define a resolução da tela.

* Parâmetros:

System.SetScreen( <ALTURA> , <LARGURA> , <MONITOR> );

* Referencia

[SYSTEM](#_SYSTEM).

## SETSUSPENDSTATE

Coloca o computador em Standby ou hibernado.

* Parâmetros:

System.SetSuspendState( <BOOLEANO> );

* Referencia

[SYSTEM](#_SYSTEM).

## SHOWMESSAGE

Mostra uma mensagem na tela.

* Opções

|  |  |  |
| --- | --- | --- |
| **Tipo** | | |
| **Constante** | **Valor** | **Descrição** |
| stmtWarning | 0 | Caixa em formato Aviso. |
| stmtError | 1 | Caixa em formato Erro. |
| stmtInformation | 2 | Caixa em formato Informação. |
| stmtConfirmation | 3 | Caixa em formato Confirmação. |
| stmtCustom | 4 | Caixa em formato Customizado. |

|  |  |  |
| --- | --- | --- |
| **Botões** | | |
| **Constante** | **Valor** | **Descrição** |
| stmbYes | 1 | Botão Sim. |
| stmbNo | 2 | Botão Não. |
| stmbOK | 4 | Botão Ok. |
| stmbCancel | 8 | Botão Cancelar. |
| stmbAbort | 16 | Botão Abortar. |
| stmbRetry | 32 | Botão Repetir. |
| stmbIgnore | 64 | Botão Ignorar. |
| stmbAll | 128 | Botão Tudo. |
| stmbNoToAll | 256 | Botão Não para Tudo. |
| stmbYesToAll | 512 | Botão Sim para Tudo. |
| stmbHelp | 1024 | Botão Ajuda. |
| stmbClose | 2048 | Botão Fechar. |

|  |  |  |
| --- | --- | --- |
| **Resultado** | | |
| **Constante** | **Valor** | **Descrição** |
| stmrNone | 0 | Retorna codigo do botão Nada |
| stmrOk | 1 | Retorna codigo do botão Ok |
| stmrCancel | 2 | Retorna codigo do botão Cancelar |
| stmrAbort | 3 | Retorna codigo do botão Abortar |
| stmrRetry | 4 | Retorna codigo do botão Repetir |
| stmrIgnore | 5 | Retorna codigo do botão Ignorar |
| stmrYes | 6 | Retorna codigo do botão Sim |
| stmrNo | 7 | Retorna codigo do botão Não |
| stmrClose | 8 | Retorna codigo do botão Fechar |
| stmrHelp | 9 | Retorna codigo do botão Ajuda |
| stmrTryAgain | 10 | Retorna codigo do botão Tentar Novamente |
| stmrContinue | 11 | Retorna codigo do botão Continuar |
| stmrAll | 12 | Retorna codigo do botão Tudo |
| stmrNoToAll | 13 | Retorna codigo do botão Não para Todos |
| stmrYesToAll | 14 | Retorna codigo do botão Sim para Todos |

* Parâmetros:

System.ShowMessage( <TEXTO>, <TIPO>, <BOTÕES>, <BOTÃO PADRÃO>);

* Referencia

[SYSTEM](#_SYSTEM).

## SHUTDOWN

Desliga, reinicia ou faz logoff.

* Opções

|  |  |  |
| --- | --- | --- |
| **Constante** | **Valor** | **Descrição** |
| stmsdLogOff | 0 | LogOff |
| stmsdShutDown | 1 | Desliga apenas o Windows, não o computador |
| stmsdReBoot | 2 | Reinicia o computador |
| stmsdForce | 4 | Desliga o computador forçadamente |
| stmsdPowerOff | 8 | Desliga o computador completamente |
| stmsdForceIfHung | 16 | Desliga o computador forçadamente de leve |

* Parâmetros:

System.ShutDown( <OPÇÃO> );

* Referencia

[SYSTEM](#_SYSTEM).