LUMI SCRIPT - INTRODUÇÃO [UL601]

Luís Augusto Spranger

http://www.lumikit.com.br

LUMI SCRIPT - ANTES DE COMEÇAR

- Os exemplos estão mais no final do vídeo, o tempo exato você encontra na descrição do vídeo;
- NÃO é necessário dominar o LUMI SCRIPT para usar o Lumikit SHOW, todos os recursos do software desde o mais básico ao mais avançado continuam disponíveis, tais como Wizards, Funções extras etc… o LUMI SCRIPT é um complemento;
- O objetivo desse vídeo é apresentar de maneira geral o LUMI SCRIPT, não temos como ensinar uma linguagem de programação em apenas um vídeo, se tiver mais interesse no assunto busque fontes de estudo complementares a respeito da linguagem apresentada.

[UL601]

LUMI SCRIPT - HISTÓRIA

- História:
  - Presente nas primeiras versões do megaDMX (2003/2004):
    - Baseado em script interpretado: limitado e com baixo desempenho;
  - 2018: vários novos projetos com o Lumikit SHOW, necessidade de customização dentro da plataforma Lumikit SHOW:
    - Nova abordagem: código é compilado, garantindo máximo desempenho;

[UL601]

LUMI SCRIPT - OBJETIVOS

- Objetivos principais:
  - Uso interno: Permitir que a equipe da Lumikit consiga entregar uma solução mais customizada em determinados projetos, sem precisar fazer uma nova versão do software, ou sobrecarregar as funções existentes com coisas muito específicas;
  - Exclusividade: fazer com que os iluminadores tenham um diferencial, o iluminador pode ter os seus próprios SCRIPTS exclusivos, usados apenas em alguns shows por exemplo;

[UL601]

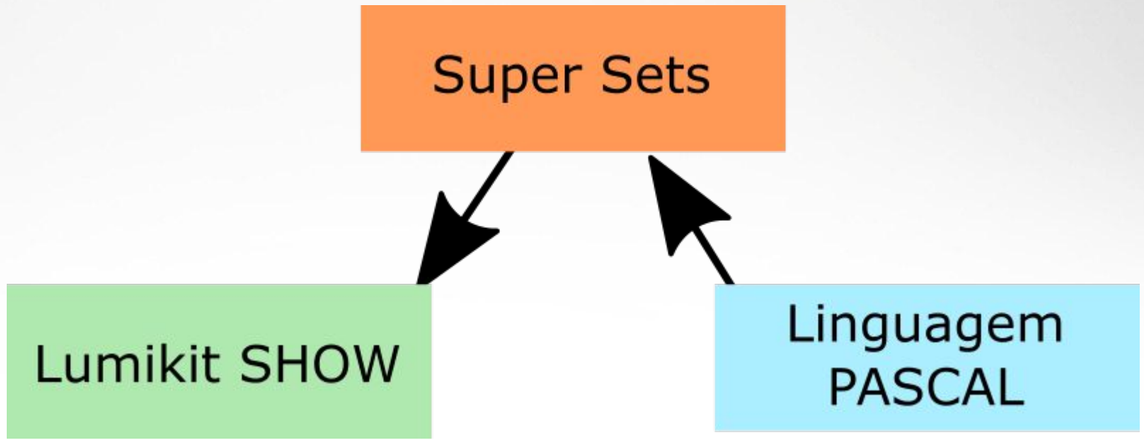
LUMI SCRIPT - PASCAL

- Linguagem usada:
  - PASCAL: tem todos os conceitos padrão usados em outras linguagens estão presentes:
    - Funções;
    - Métodos;
    - Variaveis;
    - Blocos ({} begin end);
    - Estruturas de comparação (if);
    - Repetição (while, for).

[UL601]

LUMI SCRIPT - SUPER SETS

- Super Sets: mais poder para resolver situações específicas:
  - A linguagem de programação por si só não permite fazer muita coisa além de guiar o fluxo do programa, declarar variáveis, etc.
  - Super Sets fazem a “cola” entre a linguagem e o Lumikit SHOW:



[UL601]

LUMI SCRIPT - SUPER SETS

- Super Set DMX - funções específicas para:
  - Ler valores DMX da entrada e escrever na saída DMX;
  - Alterar cores, pan/tilt e outros parâmetros dos canais DMX nos aparelhos/grupos configurados no show;
  - Acionar presets no gerador de imagens para LED/dimmer;
  - Enviar MIDI;
  - Trocar as cenas no modo Editar DMX;
- Super Set Gerador - funções específicas para:
  - Trabalhar com cores;
  - Desenhar formas geométricas;
  - Desenhar texto;
  - Ler dados do ACAP;

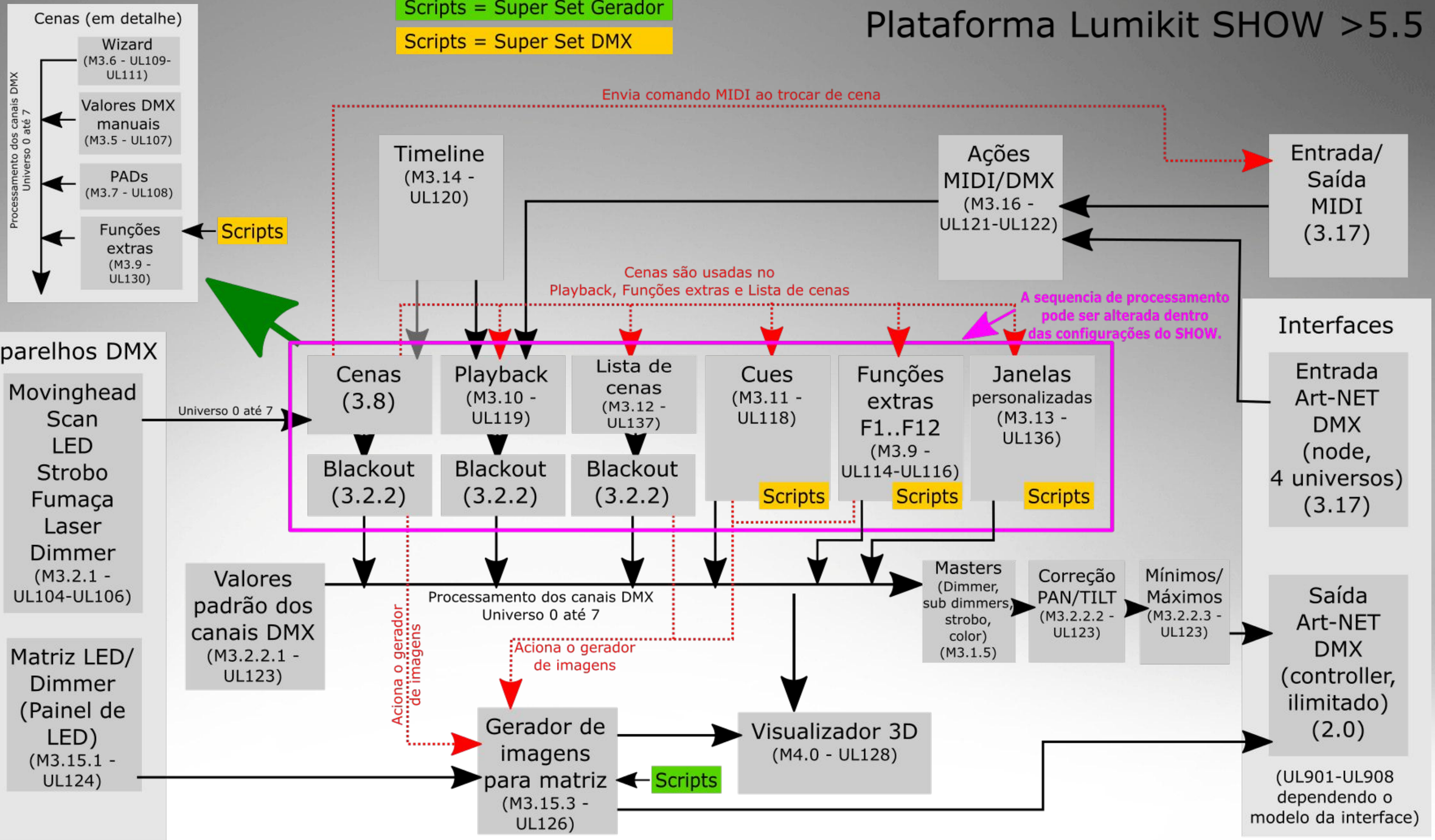
[UL601]

LUMI SCRIPT - SUPER SETS

Plataforma Lumikit SHOW >5.5

Scripts = Super Set Gerador

Scripts = Super Set DMX



LUMI SCRIPT - EXECUÇÃO

- Os scripts com Super Set DMX são executados a cada 40ms (25 vezes por segundo):
  - Dentro das funções extras: se a função extra estiver ativa:
    - Janela principal (F1..F12);
    - Janela de CUEs;
    - Janelas personalizadas;
  - Dentro de uma cena: se a cena estiver sendo executada:
    - Na janela principal;
    - Nos playbacks;
    - Lista de cenas;
    - Dentro de alguma função extra (F1..F12, CUEs, Janela Personalizada);

- Em outras palavras: código, longo, escrito de forma errada pode deixar o software LENTO ou mesmo travar o Lumikit SHOW!

[UL601]

LUMI SCRIPT - EXECUÇÃO

- Os scripts com Super Set Gerador são executados a cada 40ms (25 vezes por segundo):
  - Dentro de um efeito ativo no Gerador de imagem para LED/Dimmer na janela principal (L1, L2, L3 ou L4);
  - Dentro de um Wizard configurado como Matriz:
    - Em uma função extra (F1..F12, CUEs, Janela personalizada);
    - Em uma cena (Editar DMX, Playback, Lista de cenas, F1..F12, CUEs, Janela personalizada);

- Em outras palavras: código, longo, escrito de forma errada pode deixar o software LENTO ou mesmo travar o Lumikit SHOW!

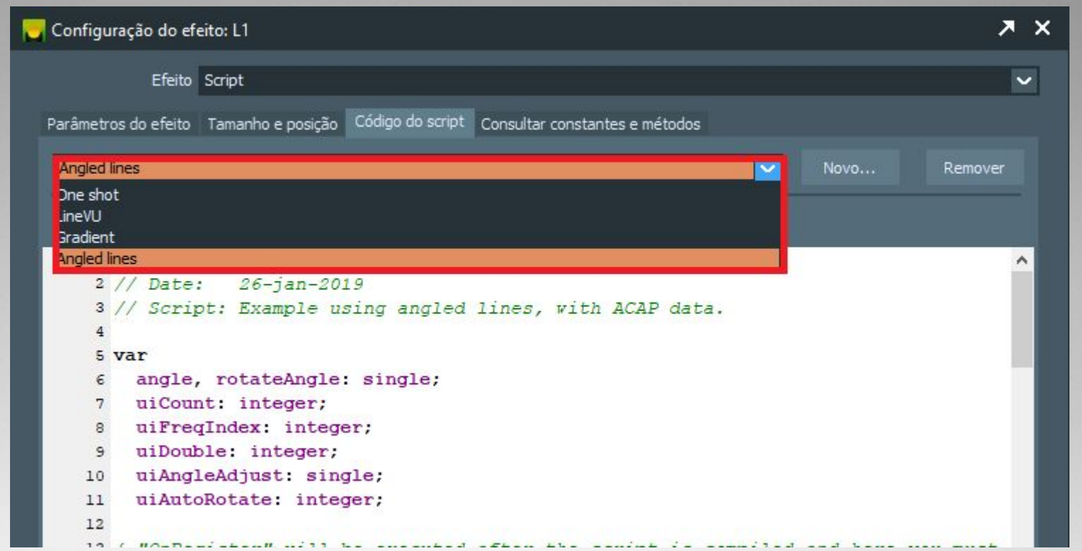
[UL601]

LUMI SCRIPT - BIBLIOTECAS DE CÓDIGO

- Para reaproveitar os scripts e otimizar o tempo de compilação do script (que pode ser demorado), mesmo que um script seja usado em várias funções extras/efeitos, ele é compilado apenas 1 (uma) vez;
- O script compilado é carregado apenas no momento em que se inicia a execução da função extra ou efeito;
- Isso quer dizer que se eu estiver executando um script dentro de uma função extra (chamada de A), alterar o script em outra função extra (por exemplo a B), a função extra inicial (A) só será atualizada se a mesma for desligada e ligada novamente!

[UL601]

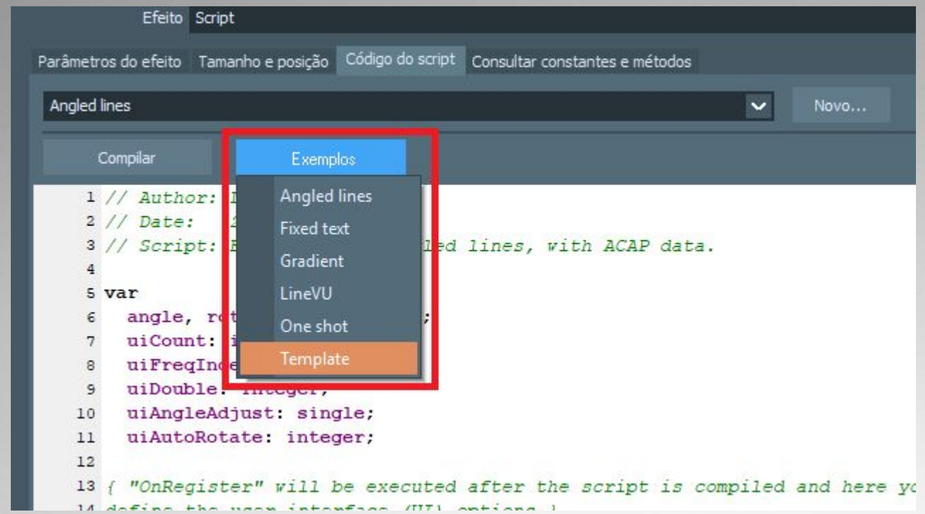
LUMI SCRIPT - BIBLIOTECAS DE CÓDIGO

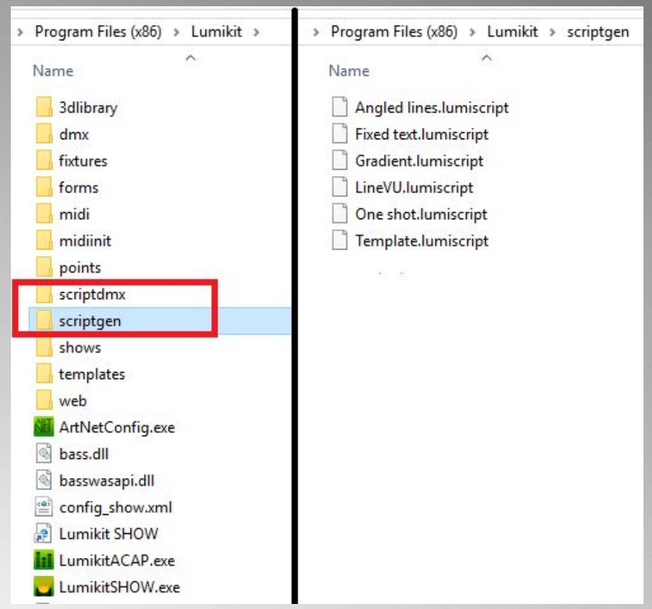


- Nesse exemplo a janela de configuração de efeitos no Gerador, na aba “Código do script” existe um ComboBox e por lá podem ser escolhidos os scripts, ou pelos botões ao lado, criar um novo ou excluir.
- 4 scripts disponíveis na imagem acima.

[UL601]

LUMI SCRIPT - EXEMPLOS



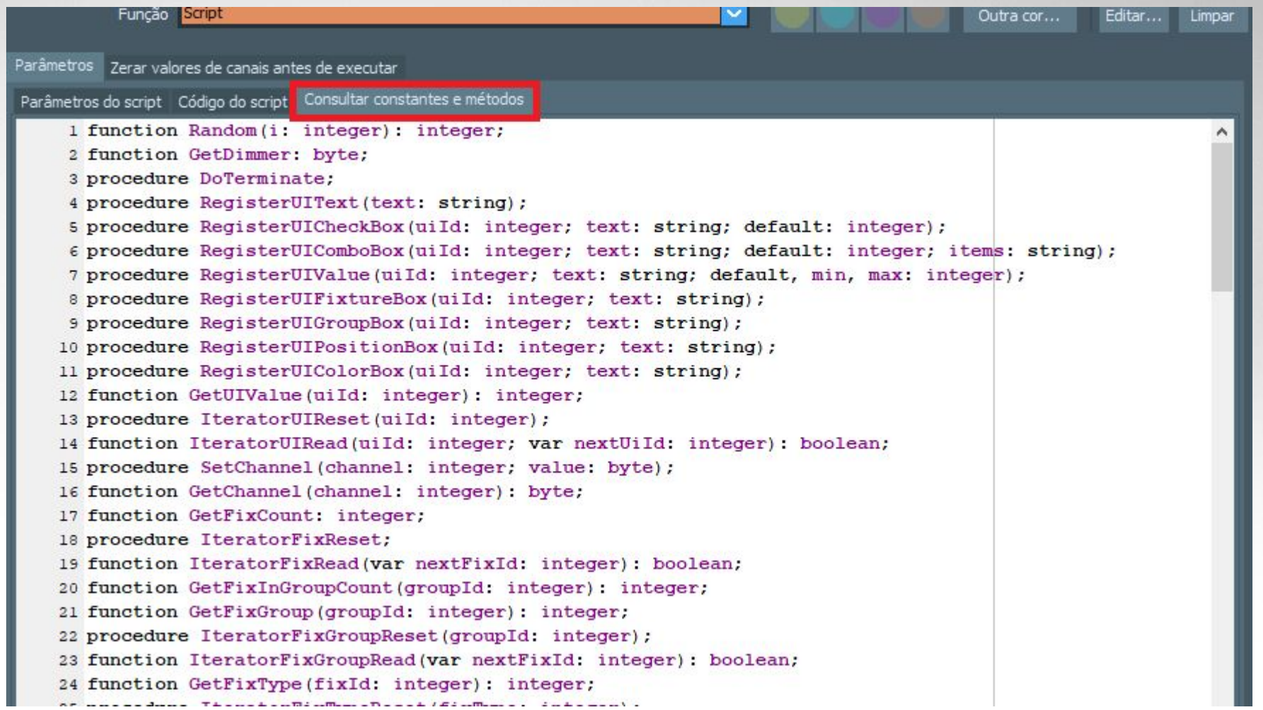


- Na janela de configuração das funções extras (se estiver selecionado a função “Script”) e na janela de configuração dos scripts (se estiver selecionado o efeito “Script”), você pode olhar os exemplos que acompanham o Lumikit SHOW no botão “Exemplos”;
- Se você quer fazer um script novo, um bom arquivo para usar como base é o “Template”;
- Os exemplos ficam na pasta de instalação do Lumikit SHOW.

[UL601]

LUMI SCRIPT - MÉTODOS, FUNÇÕES E CONSTANTES DOS SUPER SETS

- Dentro do Lumikit SHOW, se no momento da programação existirem dúvidas, os métodos/funções/constantes podem ser consultadas na aba “Consultar constantes e métodos”:

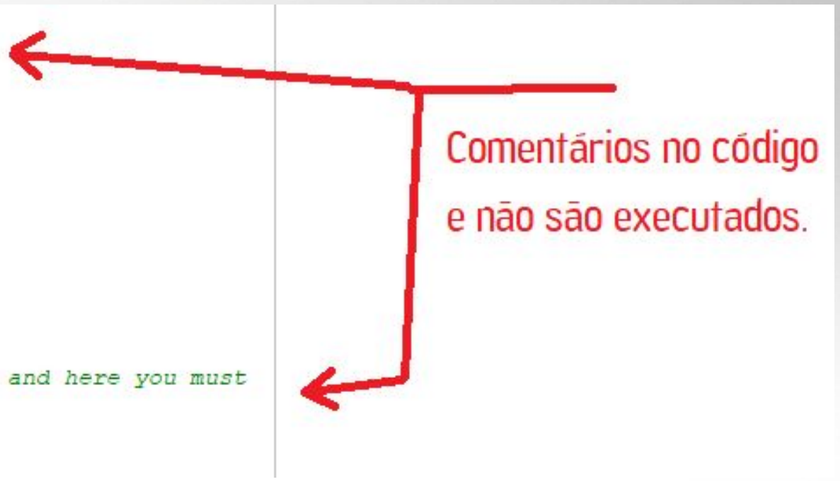


[UL601]

LUMI SCRIPT - PASCAL: COMENTÁRIOS

- Comentários no código são úteis para explicar a lógica, fazer observações ou até anotações;
- No PASCAL temos:
  - // comentário de uma linha : duas barras para comentar uma linha;
  - { comentário de mais linhas } : chaves para comentar um bloco com mais linhas;

```
1  // Author: Luis A. Spranger
2  // Date:   26-jan-2019
3  // Script: Example using angled lines, with ACAP data.
4
5  var
6    angle, rotateAngle: single;
7    uiCount: integer;
8    uiFreqIndex: integer;
9    uiDouble: integer;
10   uiAngleAdjust: single;
11   uiAutoRotate: integer;
12
13 { "OnRegister" will be executed after the script is compiled and here you must
14 define the user interface (UI) options }
15 procedure OnRegister;
16 begin
```



[UL601]

LUMI SCRIPT - PASCAL: VARIÁVEIS

Nome_da_variavel : tipo;

```
var
  // Integer data types :
  Int1 : Byte;     //                          0 to 255
  Int2 : ShortInt; //                       -127 to 127
  Int3 : Word;     //                          0 to 65,535
  Int4 : SmallInt; //                     -32,768 to 32,767
  Int5 : LongWord; //                          0 to 4,294,967,295
  Int6 : Cardinal; //                          0 to 4,294,967,295
  Int7 : LongInt;  //              -2,147,483,648 to 2,147,483,647
  Int8 : Integer;  //              -2,147,483,648 to 2,147,483,647
  Int9 : Int64;    // -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

  // Decimal data types :
  Dec1 : Single;   //  7  significant digits, exponent   -38 to +38
  Dec2 : Double;   // 15  significant digits, exponent  -308 to +308
  Dec3 : Extended; // 19  significant digits, exponent -4932 to +4932

  // Strings, the use of strings is not recommended
  Str1 : Char;         // Holds a single character, small alphabet
  Str2 : String;       // Holds strings of Char's of any size desired

  B: boolean;          // true or false
```

- Nome_da_variável sempre deve começar com uma letra ou _
- Ao atribuir um valor a uma variável, usar :=

[UL601]

LUMI SCRIPT - PASCAL: ARRAYS

- É possível criar uma lista com variáveis, no PASCAL isso se chama array, é útil para criar lista com valores de canais DMX ou de cores por exemplo.

```
nome_do_array: array [inicio..fim] of tipo;
```

```
lista_de_cores: array[0..19] of cardinal;
```

- O LUMI SCRIPT também suporta arrays dinâmicos que são aqueles que tem tamanho variável, olhe o exemplo no Super Set do Gerador chamado “Gradient” onde se cria uma lista de cores em tempo de execução baseado na altura da imagem a ser gerada;

[UL601]

LUMI SCRIPT - PASCAL: CONSTANTES

Nome_da_constante = valor;

- Nome_da_constante sempre deve começar com uma letra ou _
- Não é possível atribuir um novo valor a uma constante;
- Constantes ajudam na programação, por exemplo no Super Set DMX, você não precisa saber o código das cores do wizard, simplesmente pode usar uma constante:

```
CL_WHITE=0
CL_YELLOWL=1
CL_YELLOW=2
CL_YELLOWD=3
CL_REDLL=4
CL_REDL=5
CL_RED=6
CL_REDD=7
CL_MAGENTAL=8
CL_MAGENTA=9
CL_MAGENTAD=10
CL_PURPLE=11
CL_CYAN=12
CL_BLUEL=13
CL_BLUE=14
CL_BLUED=15
CL_GREENLL=16
CL_GREENL=17
CL_GREEN=18
CL_GREEND=19
CL_AMBARL=20
CL_AMBAR=21
CL_AMBARD=22
CL_OFF=23
CL_CUSTOM1=24
CL_CUSTOM2=25
CL_CUSTOM3=26
CL_CUSTOM4=27
```

[UL601]

LUMI SCRIPT - PASCAL: BLOCOS

- Um bloco de código sempre começa com “begin” e finaliza com “end”, em outras linguagens se usa { } por exemplo:

```
39 define the value of some variables or create another stuff here,
40 like arrays, tables }
41 procedure OnStart;
42 begin
43   angle := 0;
44   rotateAngle := 0;
45 end;
46
47 { "OnSetParameter" is called every time that a parameter is changed:
48 param = name of the parameter
49 value = value set by the user }
50 procedure OnSetParameterInteger(const param: string; const value: integer);
51 begin
52   if param='Count' then uiCount := value;
53   if param='Frequency' then uiFreqIndex := value;
54   if param='Double' then uiDouble := value;
55   if param='Angle adjust' then uiAngleAdjust := value;
56   if param='Auto rotate' then uiAutoRotate := value;
57 end;
58
59 { "OnExecute" runs every BASE_TIME (40ms) to render the images:
```

[UL601]

LUMI SCRIPT - PASCAL: COMPARADORES

- Comparadores são usados para desviar o fluxo de execução baseado em condições, por exemplo uma escolha do usuário, valor de variável, etc…, comparadores mais usados: IF..THEN (se alguma coisa então) e CASE, exemplo do IF:

```
106     DoLineAngle(cx, cy, size, angle, j);
107     if j<>0 then
108       DoLineAngle(cx, cy, size, angle, -j);
109
110     if uiDouble=1 then
111       begin
112         DoLineAngle(cx, cy, size, -angle, j);
113         if j<>0 then
114           DoLineAngle(cx, cy, size, -angle, -j);
115       end;
116
117     SetColor(GetColorCurrent(colorFade));
118
```

- Caso você queira executar mais que uma linha em determinada condição será necessário usar um bloco (BEGIN..END)!

[UL601]

LUMI SCRIPT - PASCAL: COMPARADORES

- ELSE: caso contrário:

```
74  if (r.movingUp) then
75    begin
76      r.currentValue := r.currentValue + r.step;
77      if (r.currentValue > r.maxValue) then
78        begin
79          r.currentValue := r.maxValue;
80          r.movingUp := false;
81          r.step := 20;
82        end;
83    end
84  else
85    begin
86      r.CurrentValue := r.CurrentValue - r.Step;
87      if (r.CurrentValue<=0) then
88        begin
89          RestartFakeData(i);
90          exit;
91        end;
92    end;
```

- Caso você queira executar mais que uma linha em determinada condição será necessário usar um bloco (BEGIN..END)!
- As condições do IF podem ser colocadas entre parenteses, se usar mais de uma condição o parenteses é obrigatório: IF ((a>1) AND (b>2)) THEN

[UL601]

LUMI SCRIPT - PASCAL: REPETIÇÃO

- Estruturas de repetição, executam um código várias vezes, mais usados:
  - FOR variavel:=valor_inicial TO valor_final DO
  - WHILE condição DO

```
128 var
129   i: integer;
130 begin
131   for i:=0 to 15 do
132     begin
133       if uiRealAudio=1 then
134         begin
135           data[i] := GetACAPData(i);
136         end
137       else
138         begin
139           ProcessFakeData(i);
140           data[i] := aFakeData[i].curre
141         end;
142     end;
```

[UL601]

LUMI SCRIPT - PASCAL: MÉTODOS E FUNÇÕES

- Métodos são conjuntos de código que são executados e NÃO RETORNAM VALOR!

```
procedure faz_alguma_coisa;
begin
  ..
end;
```

- Funções são conjuntos de código que são executados e RETORNAM UM VALOR!

```
function calcula_alguma_coisa: integer;
begin
  Result := 2;
end;
```

[UL601]

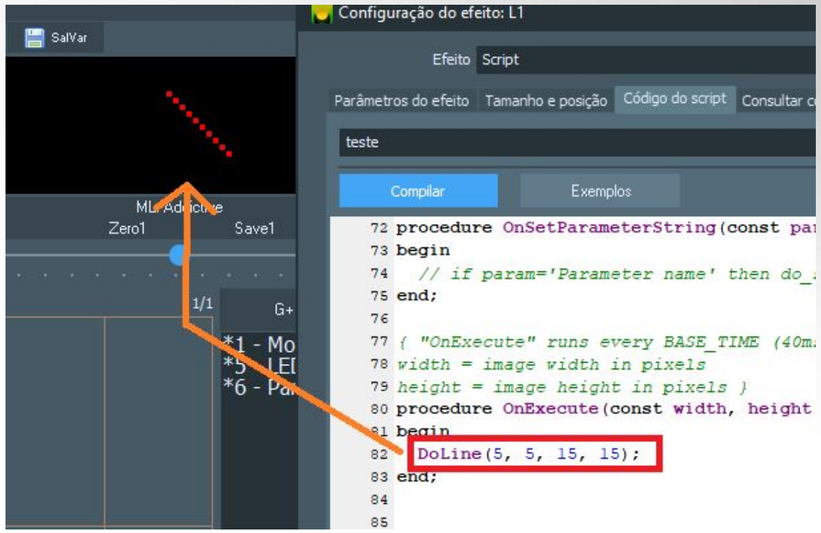
LUMI SCRIPT - PASCAL: MÉTODOS

- Exemplo de métodos dentro do Super Set do Gerador:

```
22 function GetACArData(Freq: integer): byte;
23 procedure DoClear;
24 procedure DoLine(x1, y1, x2, y2: integer);
25 procedure DoLineAngle(x1, y1, r: integer; angle: singl
26 procedure DoPixel(x1, y1: integer);
```

- começa com “procedure”, portanto é um método e não retorna nenhum valor!
- esse método desenha uma linha e tem parâmetros, que são as coordenadas da onde, para onde a linha será desenhada;

DoLine(5, 5, 15, 15);



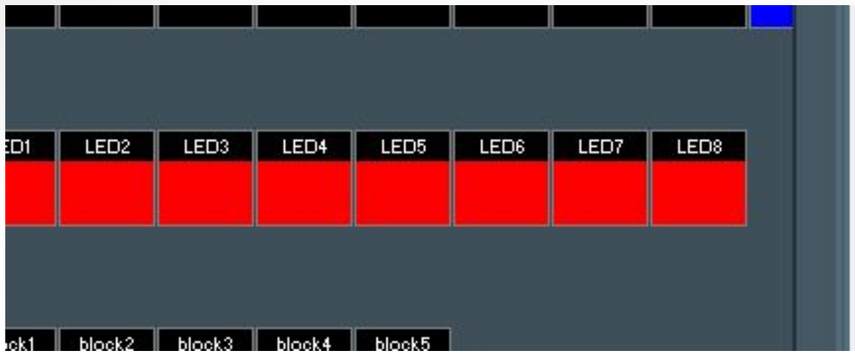
[UL601]

LUMI SCRIPT - PASCAL: FUNÇÕES

- Exemplo de função dentro do Super Set do DMX:

```
15 procedure SetChannel(channel: integer; value: byte);
16 function GetChannel(channel: integer): byte;
17 function GetFixCount: integer;
18 procedure IteratorFixReset;
19 function IteratorFixRead(var nextFixId: integer): boolean;
```

- começa com “function”, portanto é uma função e retorna valor do tipo inteiro (por causa do INTEGER!
- esse método retorna a quantidade de aparelhos dentro do show, e pode ser usado em uma repetição, por exemplo:




```
81 procedure OnExecute;
82 var
83   i: integer;
84 begin
85   for i:=0 to GetFixCount-1 do
86     begin
87       SetFixColor(i, CL_RED);
88       SetFixParam(i, CT_DIMMER, 255);
89     end;
90 end;
```

[UL601]

LUMI SCRIPT - SUPER SET METODOS/FUNÇÕES

- Os Super Sets então não são nada mais que métodos e funções, o nome deles segue uma lógica:
  - Get… geralmente é uma função e retorna um valor;
  - Set… geralmente é um método para atribuir um valor a um canal DMX por exemplo;
  - Do… executa alguma coisa, por exemplo envia um MIDI, desenha alguma coisa;
  - Register… esse é um método especial que cria objetos na tela para que o iluminador informe valores (será visto no vídeo UL602);

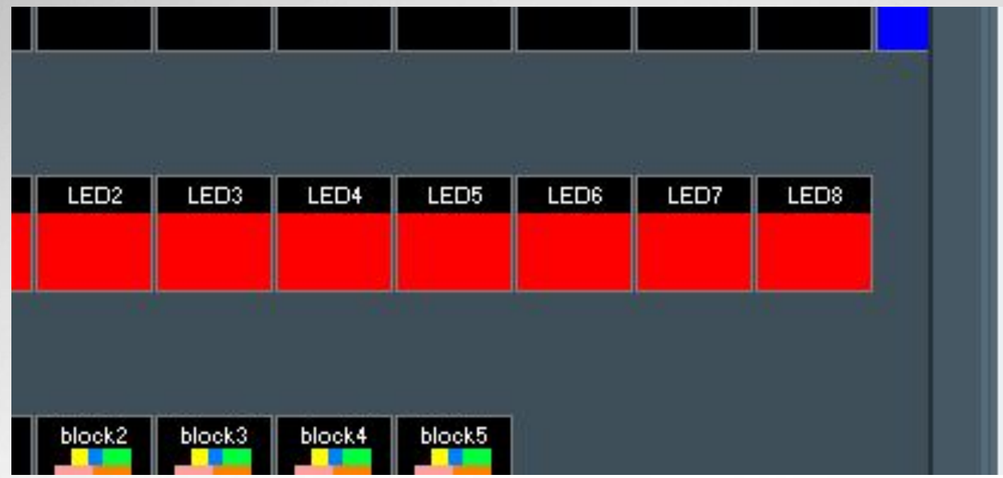
LUMI SCRIPT - SUPER METODOS ON...

- Para que um LUMI SCRIPT seja executado os Super Sets obrigam o programador a criar alguns métodos dentro dos scripts, por exemplo:
  - OnExecute (Quando executar), o programador deverá escrever esse método dentro do script e o Lumikit SHOW vai chamar esse método 25 vezes em 1 segundo; Lá dentro o programador deverá fazer tudo o que o script se propõe;
  - OnStart (Quanto iniciar), esse é um método chamado apenas no início da execução do script (é chamado apenas 1 vez) esse método pode ser usado para inicializar variáveis por exemplo;
  - OnRegister (Quando registrar o script) esse método é chamado quando o iluminador vai editar valores do script, mais detalhes no vídeo UL602;

[UL601]

LUMI SCRIPT - SUPER METODOS ON...

- Então o script que liga os LEDs na cor vermelha só é executado pois está dentro do método OnExecute!




```
80 }
81 procedure OnExecute;
82 var
83   i: integer;
84 begin
85   for i:=0 to GetFixCount-1 do
86     begin
87       SetFixColor(i, CL_RED);
88       SetFixParam(i, CT_DIMMER, 255);
89     end;
90 end;
91
```

[UL601]

LUMI SCRIPT - PRÁTICA 1

- Desenhar uma linha com o Super Set do Gerador:
  - Mostrar o conceito do OnExecute;
  - Mostrar como chamar um método do Super Set e passar parâmetros;
  - Bônus;

LUMI SCRIPT - PRÁTICA 2

- Deixar todos os aparelhos DMX do show na cor vermelha:
  - Reforçar o conceito do OnExecute;
  - Usar um “for” para percorrer os aparelhos DMX do show;
  - Reforçar como chamar um método do Super Set e passar parâmetros;
  - Bônus;

Obrigado!

Dúvidas:

suporte@lumikit.com.br