

Smart Material Changer 'Amber

版本: 2.1 PBR

[PDF 文档](#)

简介

- 基于脚本配置而不是notecard。更快的加载/传输速度，更自由的书写。
- 内核与产品功能是分离。可以支持菜单、HUD形式，本地与远端控制。
- 容易扩展且没有硬性束缚。
- 智能的匹配规则。

Ps: 没有使用 Notecard 作为配置的载体，是因为丫加载实在是太慢了，太他妈的慢了，实在是太他妈的慢了。

脚本列表

发送端（内核）

脚本	说明
SMC.KERNEL	内核，材质管理器，存储器
.SMC	用于KERNEL的配置

客户端（加载器）

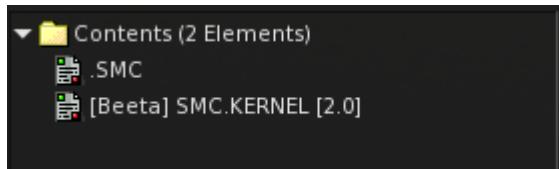
脚本	说明
SMC.Client	材质配色应用器，放在需要被替换材质的物体，接受KERNEL发出的信息
.SMC.Client	用于SMC.Client的配置

其他

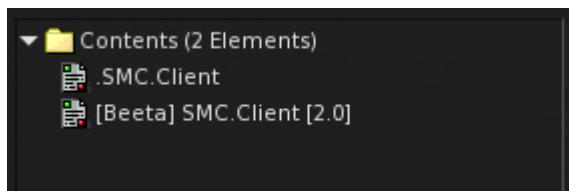
脚本	说明
SMC.HUD.TRIGGER	HUD专用，将Linkset中，Prim的描述以 PART.SET 的格式来发起材质替换
SMC.Menu	通过点击弹出菜单，选择PART与SET，实现材质的替换
.SMC.Menu	用于 SMC.Menu 的配置

脚本文件关系

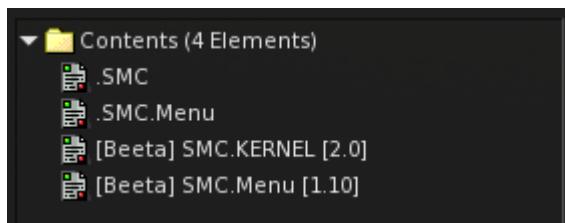
必须将 ".SMC" 与 "SMC.KERNEL" 放在一起



必须将 ".SMC.Client" 与 "SMC.Client" 放在一起



必须将 ".SMC.Menu" 与 "SMC.Menu" 放在一起, 而且它们必须伴随 KERNEL



配置文件

在红框内的部分都是可被修改的的配置项

.SMC

```

0 // version: 2.0
1 string DS="P";string PART="P";string SET="S";integer FULL=0;integer PREFIX=1;
2 integer DSUFFIX=12;integer DSMART=13;
3 integer D=0;integer DP=1;integer N=2;integer NP=3;integer S=4;integer SP=5;inte
integer GR=13;integer GB=14;integer GBC=15;integer GBA=16;integer GBM=17;intege
integer GMR = 24;integer GMP=25;integer GE=26;integer GET=27;integer GEP=28;
4 integer DEBUG = 0;
5 integer LOCAL = 0;
6 integer REMOTE = 0;
7 integer CACHE = 0;
8 integer RANGE = 0;
9
10 list LINES = [];
11
12 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
13 default {
14     state_entry(){llSleep(0.5);llMessageLinked(LINK_THIS,-643323340,llList2Jso
,RANGE),"");llSleep(0.5);llMessageLinked(LINK_THIS,-643323350,"","");
15     (i<l){unit=llList2String(LINES,i);if((unit==PART||unit==SET)&&llGetListLength(c
(cache,DS),"");cache=[];cache+=[unit];i++;}if(llGetListLength(cache)>0)llMess
llMessageLinked(LINK_THIS,-643323352,"","");
16     changed(integer change){if(change&&CHANGED_INVENTORY){llResetScript();}}
17 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //

```

.SMC.Client

```

6 // version: 2.0
7 integer DEBUG      = 0;
8 integer LOCAL      = 0;
9 integer REMOTE     = 0;
10 float   DEBOUNCE  = 0.0;
11 integer CACHE      = 0;
12
13 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
14 default {
15     state_entry(){llSleep(0.5);string _=llList2Json(JSON_OBJECT, ["DEBUG",DEB
16     llOwnerSay("SMC Client >> Submit configuration: "+_);llMessageLinked(LINK_TH
17     changed(integer change){if(change&CHANGED_INVENTORY){llResetScript();}}}
18 }
19 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //

```

.SMC.Menu

```

6 integer DEBUG          = 0;
7 integer TOUCH         = 0;
8 integer OWNER_ONLY    = 0;
9 integer SETS          = 0;
10 integer SETS_ON_TOP   = 0;
11 integer PARTS         = 0;
12 integer MENU_OPEN_LOCAL_NUM = 0;
13 integer MENU_BACK_LOCAL_NUM = 0;
14 string  MENU_BACK_OVERWRITE = "";
15 string  MENU_PREV_OVERWRITE = "";
16 string  MENU_NEXT_OVERWRITE = "";
17
18 list SETS_LIST = [];
19
20 string DIALOG_SETS = "";
21 string DIALOG_SET  = "";
22 string DIALOG_PART = "";
23
24 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
25 default {state_entry(){string _;
26     llSleep(0.5);_=llList2Json(JSON_OBJECT, ["SETS", DIALOG_SETS, "SET", DIALOG_SET, "PART", DIALOG_PART]);if(DEBUG)llo
27     +_);llMessageLinked(LINK_THIS,-643393341,_,"");
28     llSleep(0.1);_=llList2Json(JSON_OBJECT,[["DEBUG",DEBUG,"TOUCH","OWNER_ONLY",OWNER_ONLY,"SETS",SETS,"SETS_LIST"
29     "SETS_ON_TOP",SETS_ON_TOP,"MENU_OPEN_LOCAL_NUM",MENU_OPEN_LOCAL_NUM,"MENU_BACK_LOCAL_NUM",MENU_BACK_LOCAL_NUM,"PART
30     ,MENU_BACK_OVERWRITE,"MENU_PREV_OVERWRITE",MENU_PREV_OVERWRITE,"MENU_NEXT_OVERWRITE",MENU_NEXT_OVERWRITE]];if(DEBUG
31     +_);llMessageLinked(LINK_THIS,-643393340,_,"");
32 }changed(integer change){if(change&CHANGED_INVENTORY){llResetScript();}}}
33 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //

```

快速开始

菜单型

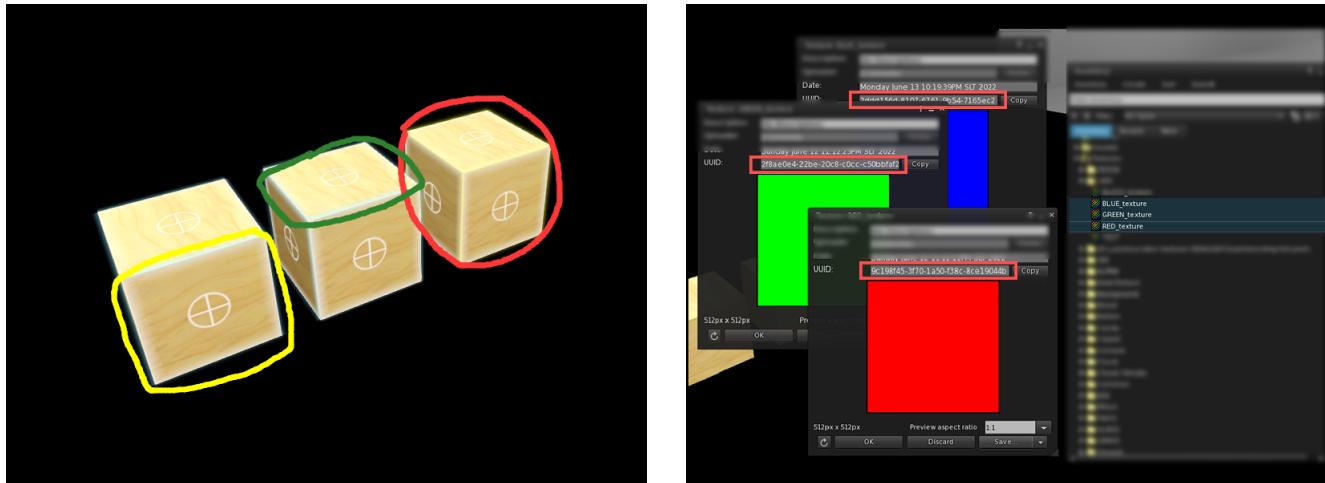
通过点击物品弹出的菜单控制替换材质

准备

准备三个连接在一起的盒子

准备三张图，得到它们的UUID

准备



替换红色轮廓内的方框的所有面、绿色轮廓内的方框的所有面以及黄色轮廓内的方框的所有面。让它们在**红色、绿色和蓝色漫反射纹理**之间切换。

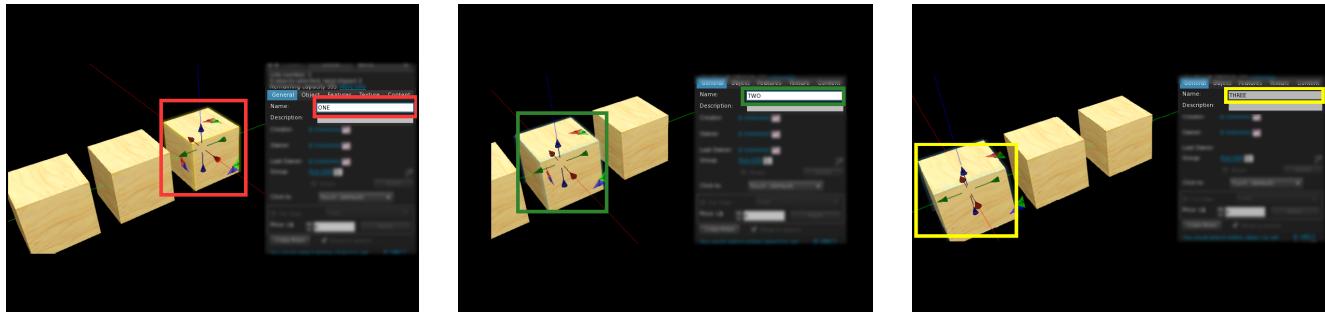
在编辑模式下，使用面选择器查看面编号。

重命名原始元素

将原始元素重命名为“ONE”

将原始元素重命名为“TWO”

将原始元素重命名为“THREE”



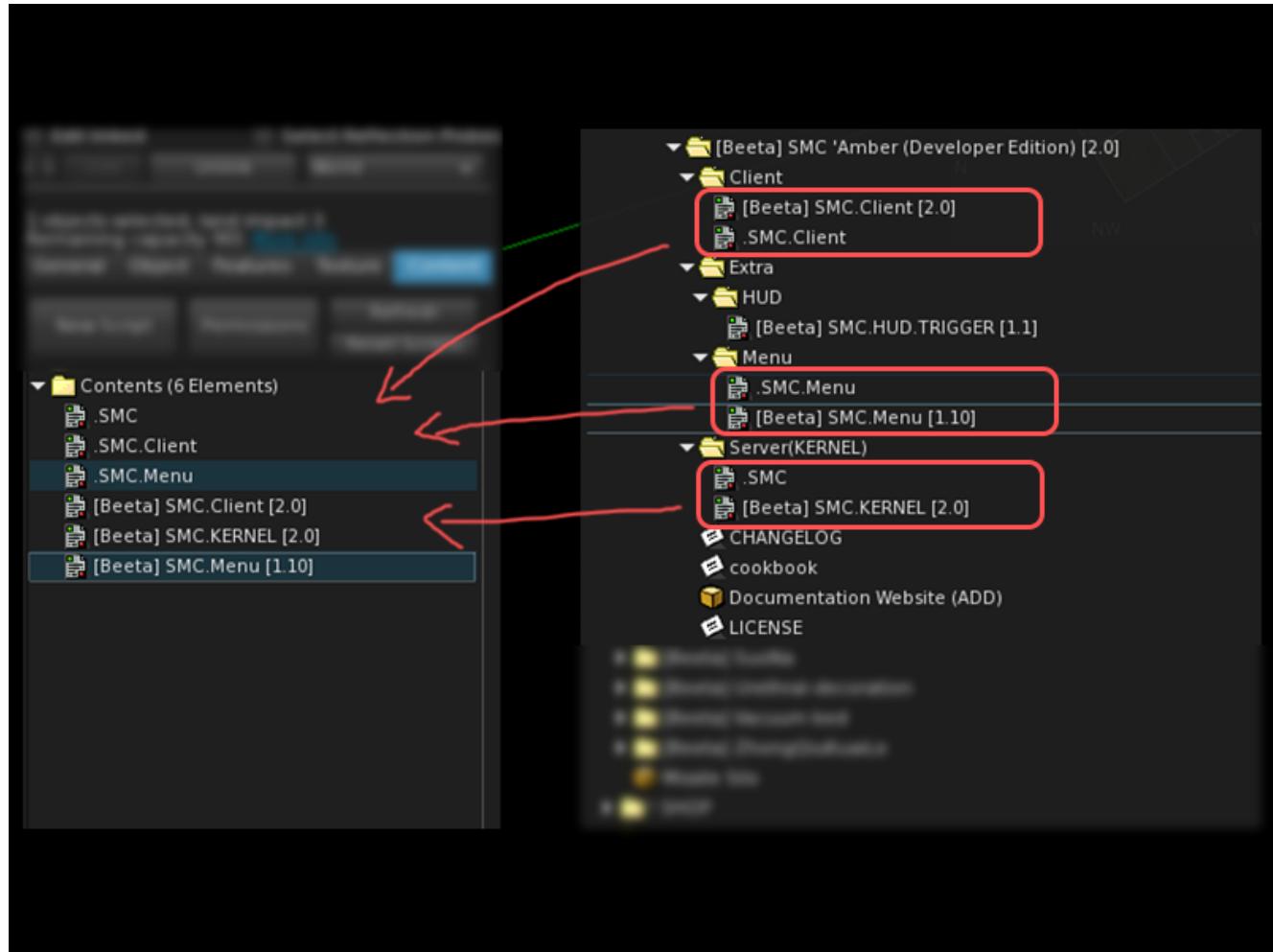
现在，得到了可以精确定位的面：

- 名为 "**ONE**" 的 prim 的 **所有面**
- 名为 "**TWO**" 的 prim 的 **0** 面
- 名为 "**THREE**" 的 prim 的 **4** 面

放置脚本

拖放所需脚本。由于使用菜单模式，KERNEL、Client、Menu 及其配置文件都将放置在物体的“contents”中。

放置脚本



编辑 .SMC

使用 **唯一** 名称来命名部件，指定匹配的模式和内容以及面。

PART 的含义就类似于 **选择器** 或 **定位器**，详细描述了如何找到目标面。

编辑 .SMC

The screenshot shows a software interface for editing Second Life scripts. The title bar says "Script: .SMC" and "Source object: ONE". The menu bar includes "File", "Edit", and "Help". Below the menu is a toolbar with icons for saving, opening, and other file operations. The main window displays the following LSL code:

```
3 integer DEBUG = 0;
4 integer LOCAL = 0;
5 integer REMOTE = 0;
6 integer CACHE = 0;
7 integer RANGE = 0;
8
9
10 // PARTS:
11 // ONE all
12 // TWO 0
13 // THREE 4
14 //
15 // SET TEXTURES:
16 // red 9c198f45-3f70-1a50-f38c-8ce19044b396
17 // blue 2ddd156d-8107-6761-9b54-7165ec249704
18 // green 2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871
19
20 list LINES = [
21     PART, "One", FULL, "ONE", ALL_SIDES,
22
23     PART, "Two", FULL, "TWO", 0,
24
25     PART, "Three", FULL, "THREE", 4
26 ];
27
28 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
29 default {
30     state_entry(){llSleep(0.5);llMessageLinked(LINK_THIS,-643323340,llList2Json(JSON_OBJECT,[{"DEBUG"
,DEBUG,"LOCAL",LOCAL,"REMOTE",REMOTE,"CACHE",CACHE,"RANGE",RANGE}],""));llSleep(0.5);llMessageLinked(

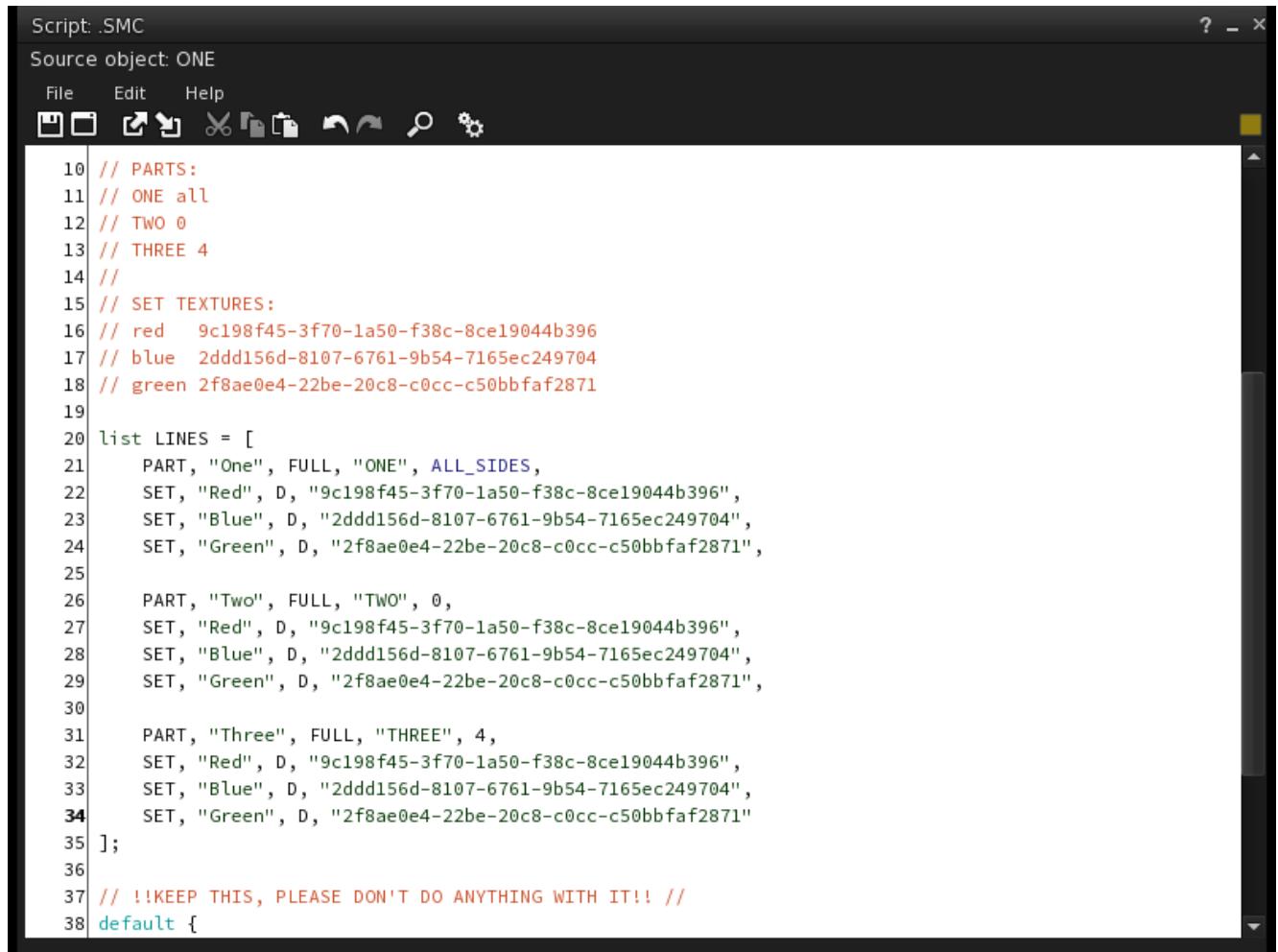
```

定义三种不同颜色的纹理并为其命名。

每个 PART 将在三种样式之间切换，因此需要完全指定所有三种变化。

注意：SET 的名称在一个 PART 内必须是唯一的。

编辑 .SMC



```
Script: .SMC
Source object: ONE
File Edit Help
? - ×

10 // PARTS:
11 // ONE all
12 // TWO 0
13 // THREE 4
14 //
15 // SET TEXTURES:
16 // red 9c198f45-3f70-1a50-f38c-8ce19044b396
17 // blue 2ddd156d-8107-6761-9b54-7165ec249704
18 // green 2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871
19
20 list LINES = [
21     PART, "One", FULL, "ONE", ALL_SIDES,
22     SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
23     SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",
24     SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871",
25
26     PART, "Two", FULL, "TWO", 0,
27     SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
28     SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",
29     SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871",
30
31     PART, "Three", FULL, "THREE", 4,
32     SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
33     SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",
34     SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871"
35 ];
36
37 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
38 default {
```

编辑 .SMC 和 .SMC.Client

将 LOCAL 调整为相同的非零值。

编辑 .SMC 和 .SMC.Client

The image shows two side-by-side Notepad windows. Both windows have dark themes and show LSL (Linden Scripting Language) code.

Script: .SMC

Source object: ONE

```

0 // version: 2.0
1 string DS="P";string PART="P";string SET="S";integer FULL=0;integer PREFIX=1;integer SUFFIX=2;integer
  SMART=3;integer CONST=4;integer DFULL=10;integer DPREFIX=11;integer DSUFFIX=12;integer DSMART=13.
2 integer D=0;integer DP=1;inte
  integer G=8;integer F=9;integ
    GBC=15;integer GBA=16;integ
    integer GM=22;integer GMM = 2
      GEP=28;
3
4 integer DEBUG = 0;
5 integer LOCAL = 666;
6 integer REMOTE = 0;
7 integer CACHE = 0;
8 integer RANGE = 0;
9
10 // PARTS:
11 // ONE all
12 // TWO 0
13 // THREE 4
14 //
15 // SET TEXTURES:
16 // red 9c198f45-3f70-1a50-f
17 // blue 2ddd156d-8107-6761-9
18 // green 2f8ae0e4-22be-20c8-c
19
20 list LINES = [
21   PART, "One", FULL, "ONE",
22   SET, "Red", D, "9c198f45-
23   SET, "Blue", D, "2ddd156d

```

Toggling the preprocessor will not take full eff.

Script: .SMC.Client

Source object: ONE

```

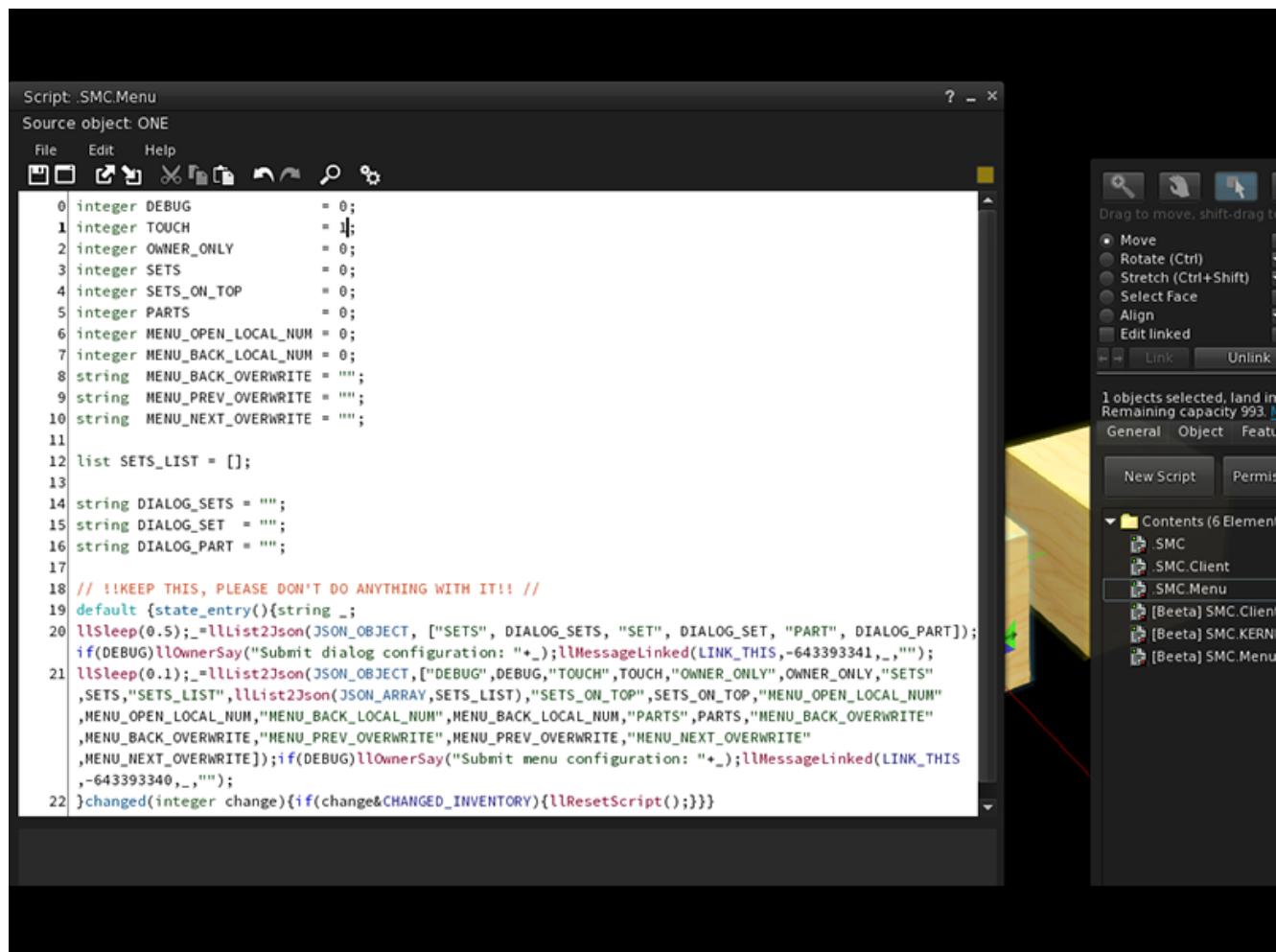
0 // version: 2.0
1 integer DEBUG = 0;
2 integer LOCAL = 666;
3 integer REMOTE = 0;
4 float DEBOUNCE = 0.0;
5 integer CACHE = 0;
6
7 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
8 default {
9   state_entry(){llSleep(0.5);string _=llList2Json(JSON_OBJECT, ["DEB
  ,REMOTE,"DEBOUNCE",DEBOUNCE,"CACHE",CACHE]);if(DEBUG)llOwnerSay("SMC C
+_");llMessageLinked(LINK_THIS,-643323341,_,"");}
10   changed(integer change){if(change&CHANGED_INVENTORY){llResetScript
11 }
12 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //

```

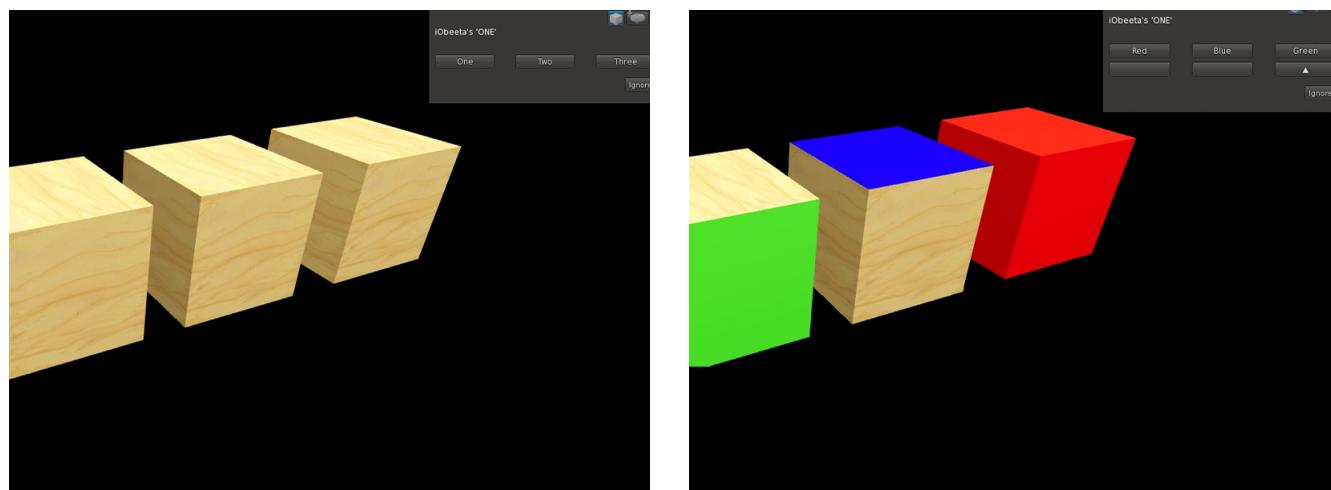
编辑 .SMC.Menu

设置 TOUCH = 1, 启用触摸。

编辑 .SMC.Menu



试试看



OWNER_ONLY = 1 将菜单限制为仅所有者触发。

关于 SETS

SETS 可让您方便地批量执行多个 PART 的替换。

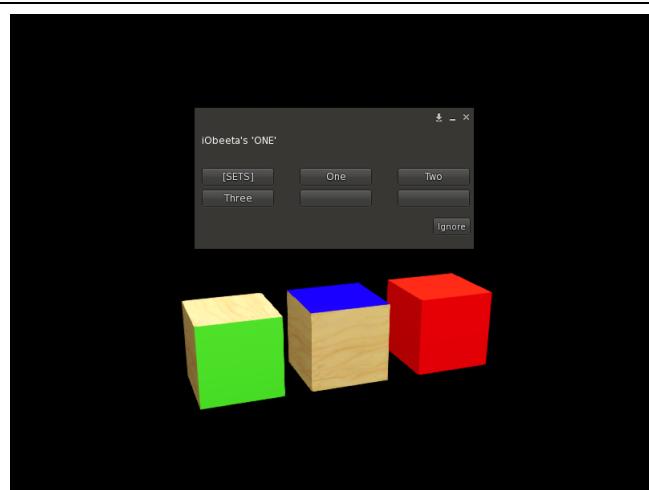
您可以在 .SMC.Menu 设置中启用 "SETS" 功能。以下是详细信息。

代码

预览

代码

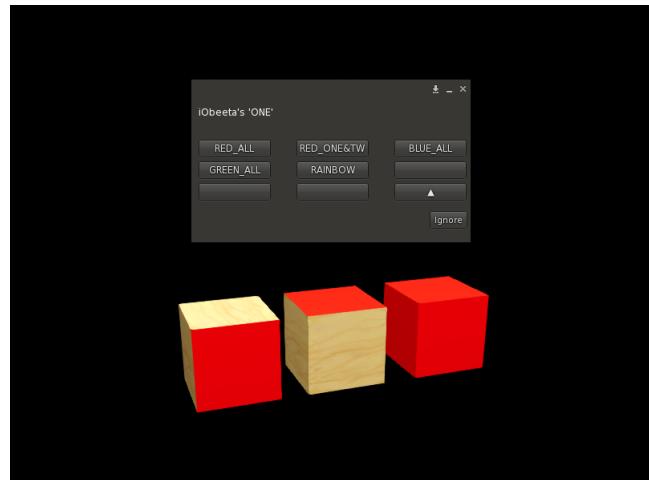
SETS = 1



定义 SETS_LIST

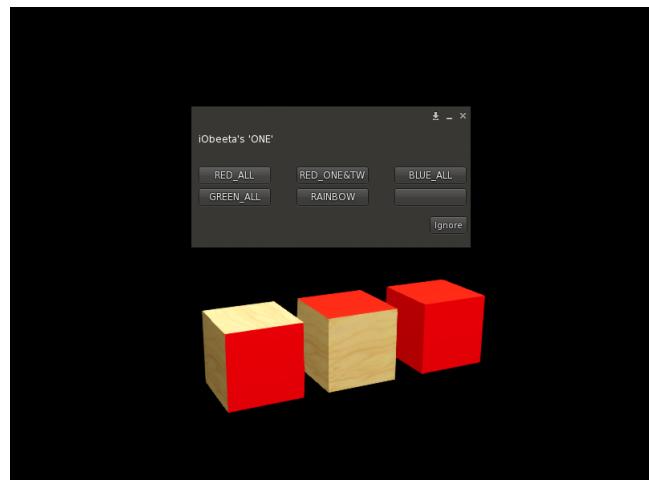
单击 [SETS] 显示所有集合。

```
Script: SMC.Menu
Source object: ONE
File Edit Help
integer DEBUG = 0;
integer TOUCH = 1;
integer OWNER_ONLY = 0;
integer SETS = 1;
integer SETS_ON_TOP = 0;
integer PARTS = 0;
integer MENU_OPEN_LOCAL_NUM = 0;
integer MENU_BACK_LOCAL_NUM = 0;
string MENU_BACK_OVERWRITE = "";
string MENU_PREV_OVERWRITE = "";
string MENU_NEXT_OVERWRITE = "";
list SETS_LIST = [
    "RED_ALL", ".Red",
    "RED_ONE&TWO", "One.Red.Two.Red",
    "BLUE_ALL", ".Blue",
    "GREEN_ALL", "One.Green.Two.Green.Three.Green",
    "RAINBOW", "One.Red.Two.Blue.Three.Green"
];
string DIALOG_SETS = "";
string DIALOG_SET = "";
string DIALOG_PART = "";
// !!!KICK THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
default {text_entry}()string {
    listed(0.5, [{"label": "Submit dialog configuration: ", "value": ""}], [MessageLinkedin(LINK_THIS, "643393341", "")], [OnEnter("Submit dialog configuration: ", "Submit dialog configuration", "Cancel dialog configuration", "Close dialog configuration", "Delete dialog configuration", "Edit dialog configuration", "Save dialog configuration", "Update dialog configuration")]);
}
```



SETS ON TOP = 1

在顶级菜单中显示集合列表（局部替换将不可用）。



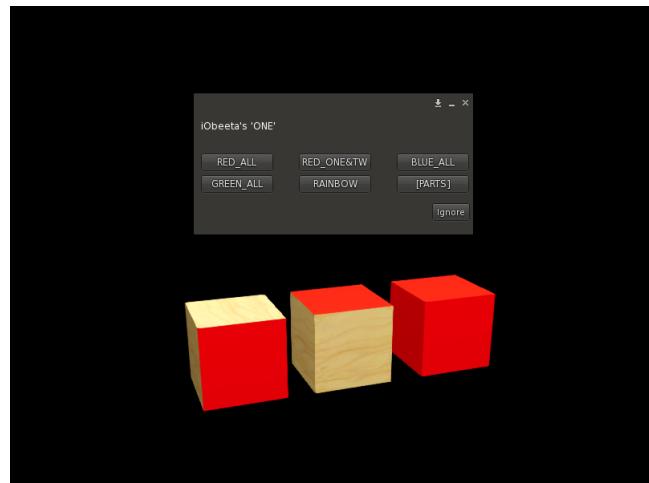
PARTS = 1

将选项 [PARTS] 添加到设置列表以启用并执行局部替换操作。

代码

```
Script: SMC.Menu
Source object: ONE
File Edit Help
□□□ DEBUG
1 Integer DEBUG = 0;
2 Integer TOUCH = 1;
3 Integer OWNER_ONLY = 0;
4 Integer SETS = 1;
5 Integer SETS_ON_TOP = 1;
6 Integer PARTS = 1;
7 Integer MENU_OPEN_LOCAL_NUM = 6;
8 Integer MENU_BACK_LOCAL_NUM = 6;
9 String MENU_BACK_OVERWRITE = "";
10 String MENU_PREV_OVERWRITE = "";
11 String MENU_NEXT_OVERWRITE = "";
12 List SETS_LIST = [
13     "RED_ALL", ".Red",
14     "RED_ONE&TW", "One.Red,Two.Red",
15     "BLUE_ALL", ".Blue",
16     "GREEN_ALL", "One.Green,Two.Green,Three.Green",
17     "RAINBOW", "One.Red,Two.Blue,Three.Green"
18 ];
19
20 String DIALOG_SETS = "";
21 String DIALOG_SET = "";
22 String DIALOG_PART = "";
23
24 // !KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
25 Default {state_entry}() {
26     lSleep(0.5); =!list2Json(Json_Object, ["SETS", DIALOG_SETS, "SET", DIALOG_SET, "PART", DIALOG_PART]); if(!
    llOwnerSay("Submit dialog configuration: "+.) || !MessageLinked(LINK_THIS, -643393341, "+"));
    adacty("on "+.); // 2024-07-23 10:22:22 [2024-07-23 10:22:22] [2024-07-23 10:22:22] [2024-07-23 10:22:22]
}
Compile successful!
```

预览

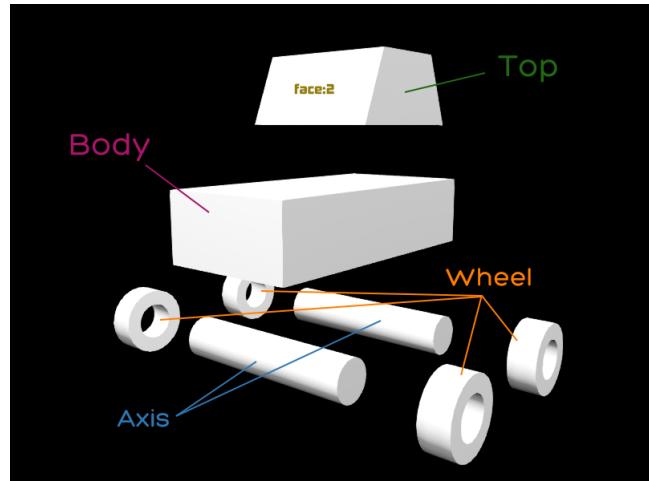
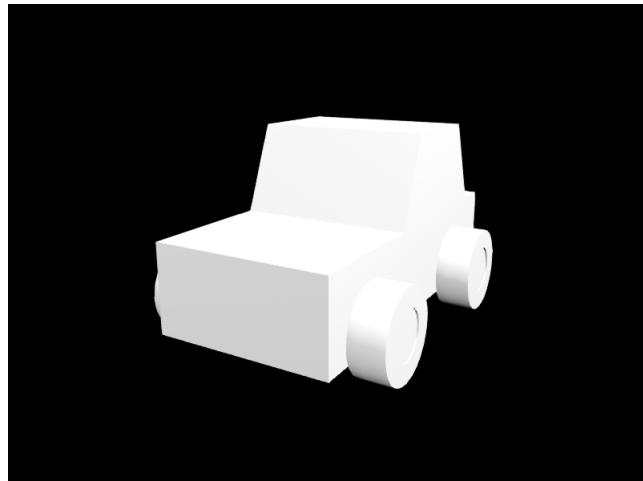


HUD 型

产品

准备产品

规划并命名其包含的子原件。



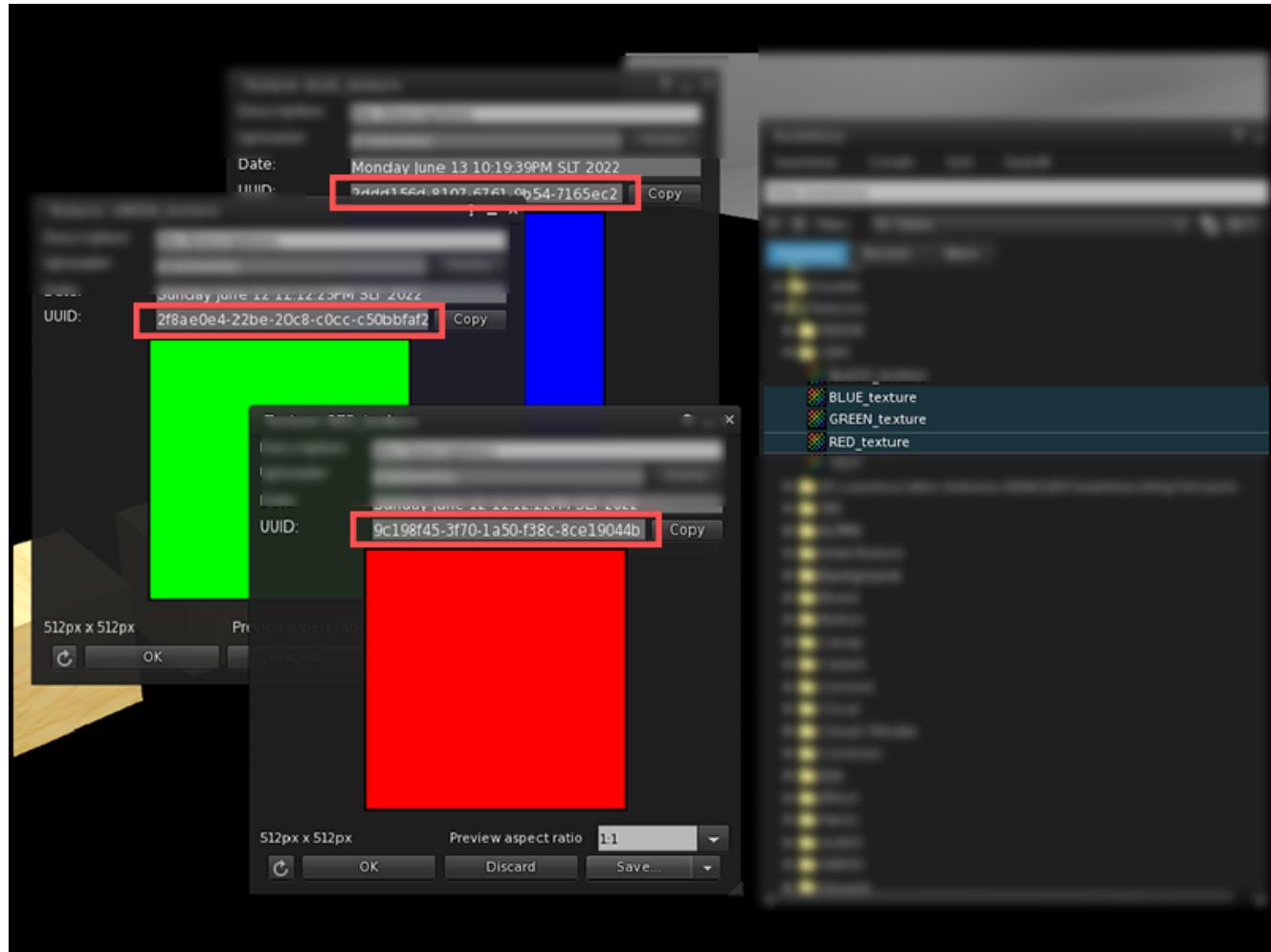
此外，顶部的第 2 个面将由玻璃制成。零件的规划如下：

- PART Top: Top, 01345
- PART Body: Body, ALL_SIDES
- PART Axis: Axis, ALL_SIDES
- PART Wheel: Wheel, ALL_SIDES
- PART Glass: Top, 2

纹理

使用 3 个纹理

纹理

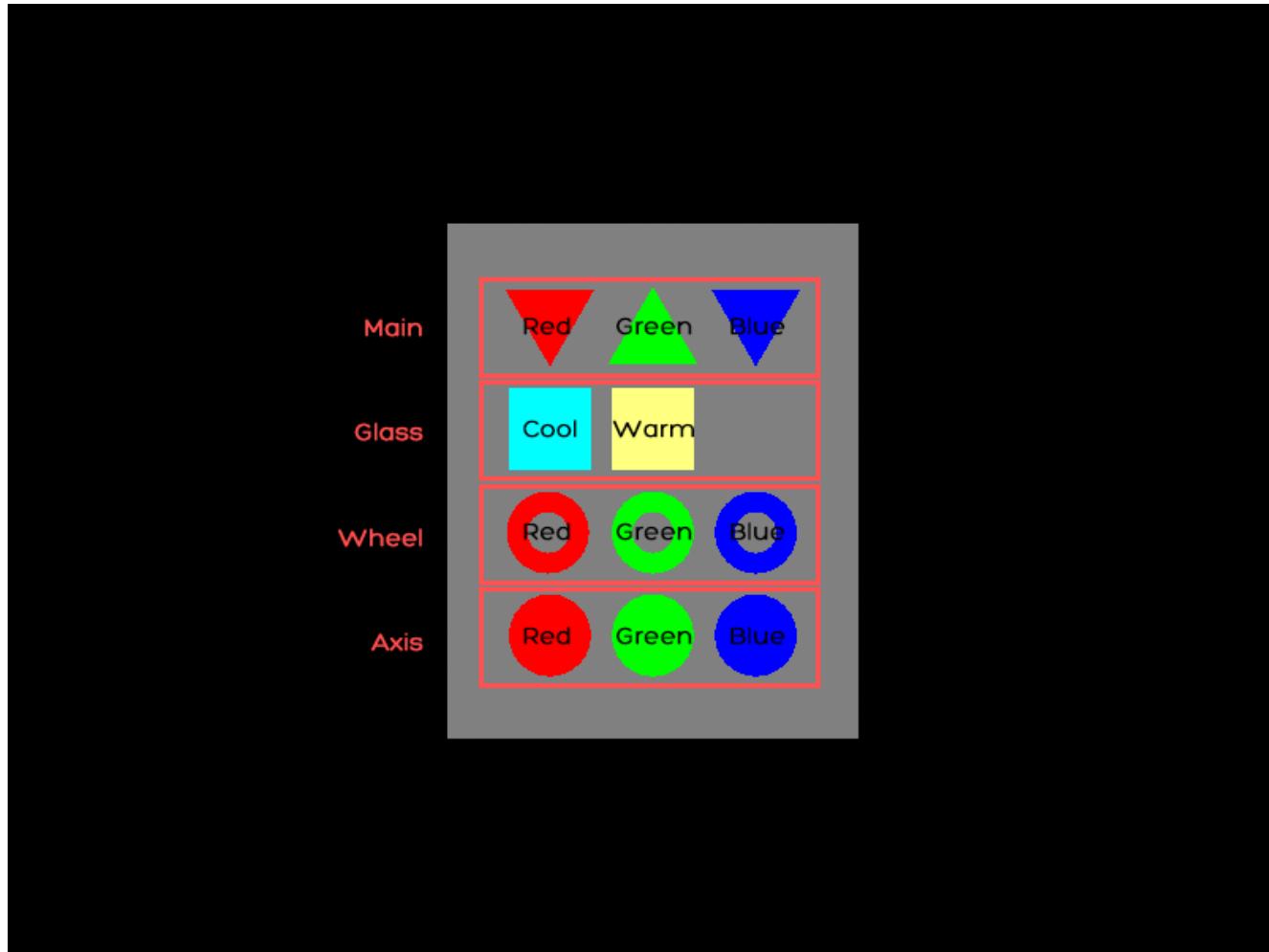


Body、**Top**、**Axis**、**Wheel** 分别具有 **Red**、**Green**、**Blue**。 **Glass** 准备使用 **TEXTURE_BLANK** 并同时进行着色。让它具有 2 种样式，**Cool** 和 **Warm**。

准备 HUD

创建 HUD，添加所需按钮并链接它们。

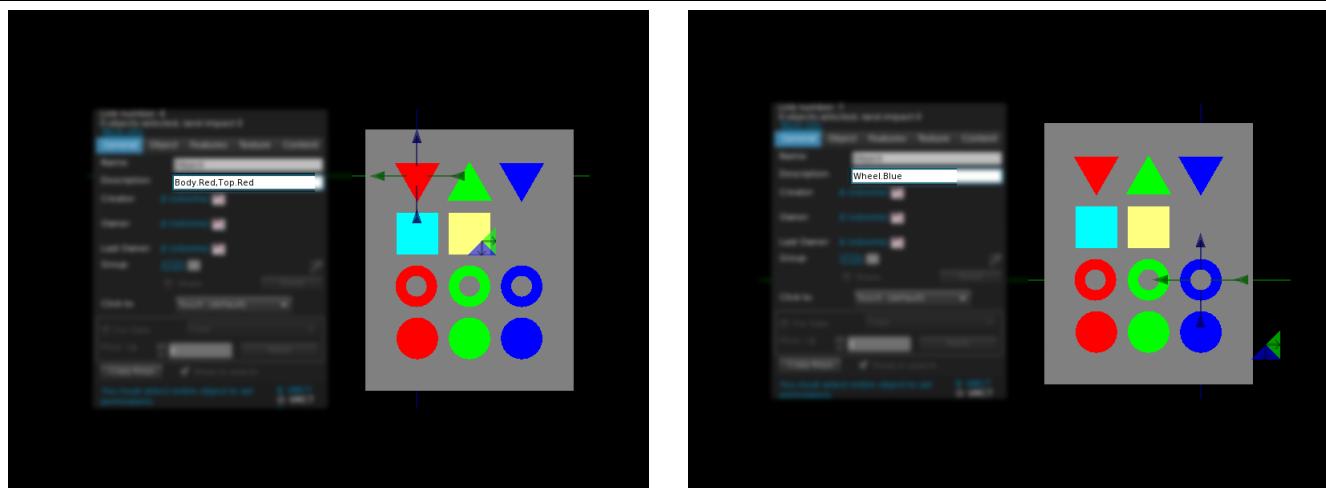
准备 HUD



编辑子元素的 description

Main 这组按钮将同时控制 Top 和 Body 部分。Red 中的 Main 包括 Top.Red 和 Body.Red。

Wheel.Blue。其他按钮类似。



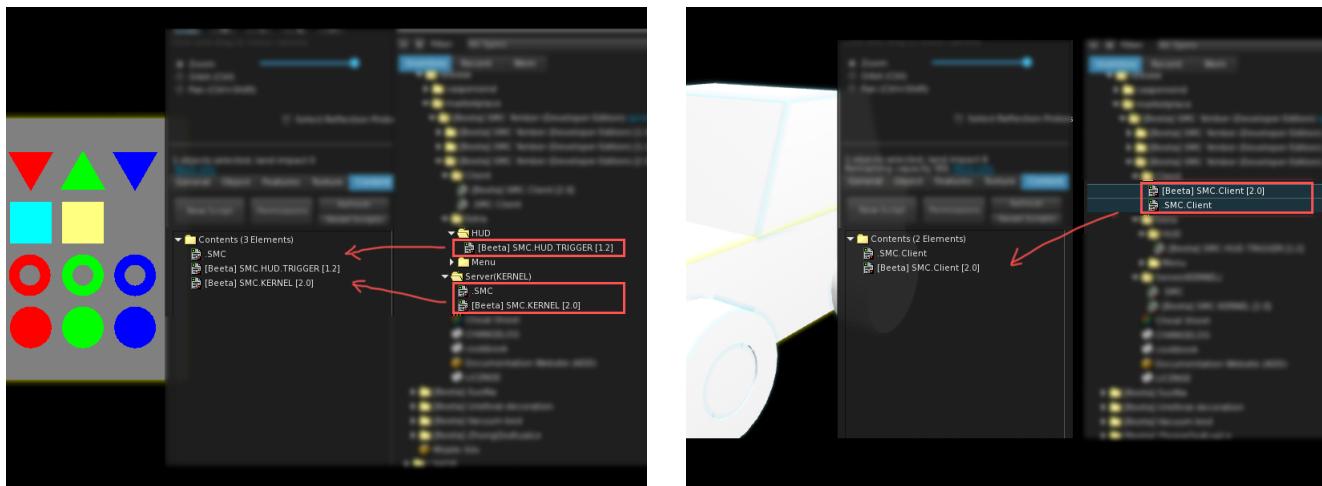
单个按钮可以执行批量操作，类似于菜单模式下的SETS。您也可以省略PART部分，只写**.SET**即可实现完全替换。

放置脚本

放置HUD脚本。

放置产品脚本。

放置脚本



编辑HUD中的**.SMC**

Script: .SMC

Source object: Example.Car.HUD

File Edit Help

```

integer RANGE = 0;

list LINES = [
    PART, "Top", FULL, "Top", "01345",
    SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
    SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871",
    SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",

    PART, "Body", FULL, "Body", ALL_SIDES,
    SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
    SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871",
    SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",

    PART, "Axis", FULL, "Axis", ALL_SIDES,
    SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
    SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871",
    SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",

    PART, "Wheel", FULL, "Wheel", ALL_SIDES,
    SET, "Red", D, "9c198f45-3f70-1a50-f38c-8ce19044b396",
    SET, "Green", D, "2f8ae0e4-22be-20c8-c0cc-c50bbfaf2871",
    SET, "Blue", D, "2ddd156d-8107-6761-9b54-7165ec249704",

    PART, "Glass", FULL, "Top", 2,
    SET, "Cool", D, TEXTURE_BLANK, C, <0.5, 1.0, 1.0>,
    SET, "Warm", D, TEXTURE_BLANK, C, <1.0, 1.0, 0.7>
];

```

Compile successful!

编辑 HUD 中的 .SMC 和产品中的 .SMC.Client

编辑 HUD 中的 .SMC 和产品中的 .SMC.Client

The image shows two side-by-side Notepad windows. The left window is titled 'Script: .SMC' and has a 'Source object: Example.Car.HUD' header. It contains LSL code for setting up various parameters like DS, PART, SET, CONST, etc., and lists of lines for different colors. A specific line 'integer REMOTE = 3333;' is highlighted with a red box. The right window is titled 'Script: .SMC.Client' and has a 'Source object: Body' header. It contains similar LSL code, also featuring a line 'integer REMOTE = 3333;' which is also highlighted with a red box. Both windows have standard Windows-style toolbars at the top.

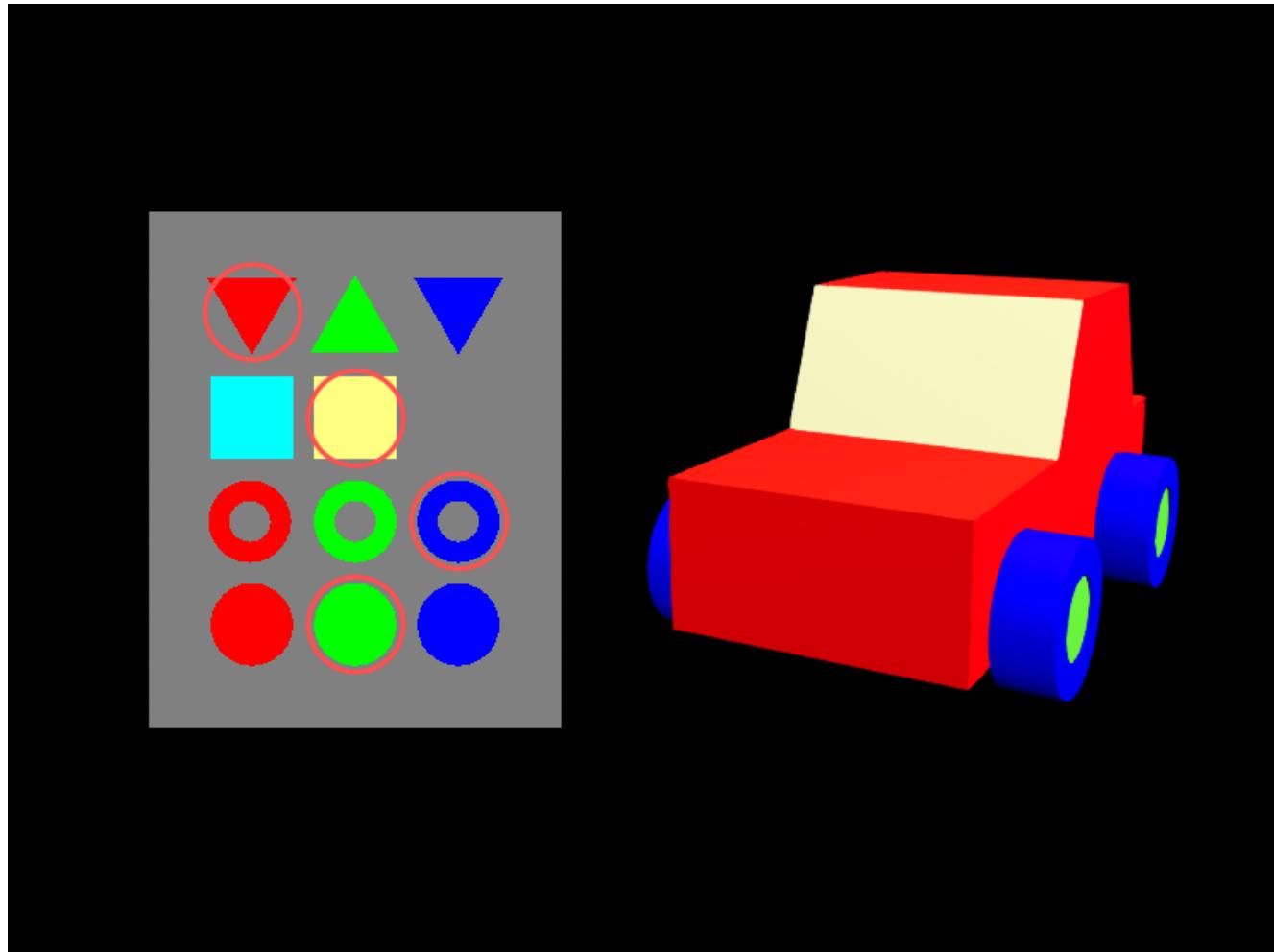
```
0 // version: 2.0
1 string DS="P";string PART="S";string SET="S";integer FULL=0;integer PREFTX=1;integer SUFFTX=2;integer SMAR
2 integer CONST=4;integer DFULL=10;integer D=0;integer DP=1;integer N=2;integer F=9;integer B=10;integer T=11;integer G
3 integer GBM=17;integer GBD=18;integer GBP=20;integer GMR = 24;integer GMP=25;integer GE=26;integer DEBOUNCE = 0.0;
4 integer DEBUG = 0;
5 integer LOCAL = 0;
6 integer REMOTE = 3333; // This line is highlighted with a red box.
7 integer CACHE = 0;
8 integer RANGE = 0;
9
10 list LINES = [
11     PART, "Top", FULL, "Top", "01345",
12     SET, "Red", D, "9c198f45-3f70-1a50-f38
13     SET, "Green", D, "2f8ae0e4-22be-20c8-0
14     SET, "Blue", D, "2ddd156d-8107-6761-9
15     PART, "Body", FULL, "Body", ALL_SIDES
16     SET, "Red", D, "9c198f45-3f70-1a50-f38
17     SET, "Green", D, "2f8ae0e4-22be-20c8-0
18     SET, "Blue", D, "2ddd156d-8107-6761-9
19
20     PART, "Axis", FULL, "Axis", ALL_SIDES
21     SET, "Red", D, "9c198f45-3f70-1a50-f38
22     SET, "Green", D, "2f8ae0e4-22be-20c8-0
23     SET, "Blue", D, "2ddd156d-8107-6761-9
24 ]
```

```
0 // version: 2.0
1 integer DEBUG = 0;
2 integer LOCAL = 0;
3 integer REMOTE = 3333; // This line is highlighted with a red box.
4 float DEBOUNCE = 0.0;
5 integer CACHE = 0;
6
7 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
8 default {
9     state_entry(){llSleep(0.5);string _=llList2Json(JSON_C
10     "DEBOUNCE",DEBOUNCE,"CACHE",CACHE]);if(DEBUG)llOwnerSay("S
11     LINK_THIS,-643323341,_,""");}
12     changed(integer change){if(change&CHANGED_INVENTORY){l
13 }
14 // !!KEEP THIS, PLEASE DON'T DO ANYTHING WITH IT!! //
```

试试看

点击按钮

试试看



场景示例

一件衣服，有独立的HUD

- SMC.KERNEL 放在HUD里。
 - 可使用 SMC.HUD.TRIGGER，如果您有脚本基础，可以自行编写，更加灵活。
- SMC.Client 放在衣服里。
- SMC.KERNEL 与 SMC.Client 定义相同的 REMOTE。

一件衣服，点击领口弹出菜单

- SMC.KERNEL、SMC.Client、SMC.Menu 放在衣服里。
 - 放在 ROOT 或者 领口 都可以，取决于您想点击哪里弹出菜单。
- SMC.KERNEL 与 SMC.Client 定义相同的 LOCAL。

一栋房子，上面有个触控板，并且触控板与房子是一体的，点触控板弹出菜单

- SMC.KERNEL、SMC.Client 放在房子的任意 PRIM，定义相同的 LOCAL。
- SMC.Menu 放在触控板并开启 TOUCH。

一栋房子，上面有个触控板，并且触控板与房子是分体的，点触控板弹出菜单

- SMC.Client 放在房子的任意 PRIM。

- SMC.KERNEL、SMC.Menu 放在触控板并开启 TOUCH。
- SMC.KERNEL 与 SMC.Client 定义相同的 REMOTE。

一栋房子，上面有2个触控板，触控板有与房子一体的，还有个分体的，点击弹出菜单。您兜里还有一个，能当HUD用

- SMC.Client、SMC.KERNEL、SMC.Menu 放在房子的任意 PRIM，SMC.Menu 开启 TOUCH
- SMC.KERNEL、SMC.Menu 放在分体的触控板，SMC.Menu 开启 TOUCH
- SMC.KERNEL、SMC.Menu 放在您兜里的触控板HUD。
- 房子里的 SMC.Client、SMC.KERNEL 定义相同的 LOCAL。
- 分体的和您兜里的 SMC.KERNEL 定义与房子里 SMC.Client 相同的 REMOTE。

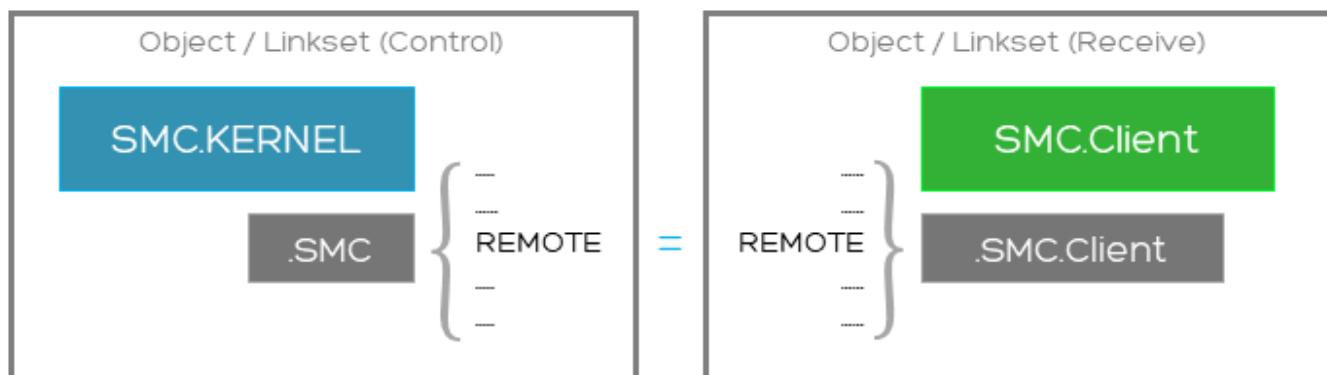
注意！SMC.HUD.TRIGGER 只能用于包含独立 PRIM 按钮的 HUD，它依赖于不同的名称或者备注。如果只有一个 PRIM，是不行的，它无法对面甚至面的触摸位置 (ST/UV) 进行识别。如果需要，您可以自行撰写。

部署

Smart Material Changer (SMC) 通过 REMOTE 或 LOCAL 配对操作，相同即可，“0”视为无效。

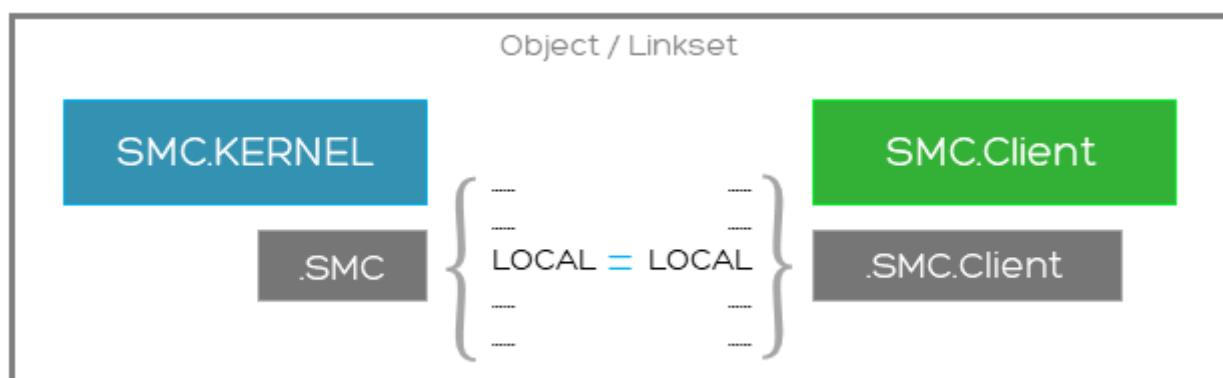
远端控制

远程控制是指一个object (linkset) 控制另一个object (linkset)，要求控制端 (kernel) **.SMC** 文件中的 **REMOTE** 和受控端 (client) **.SMC.Client** 文件中的 **REMOTE** 相匹配。



本地控制

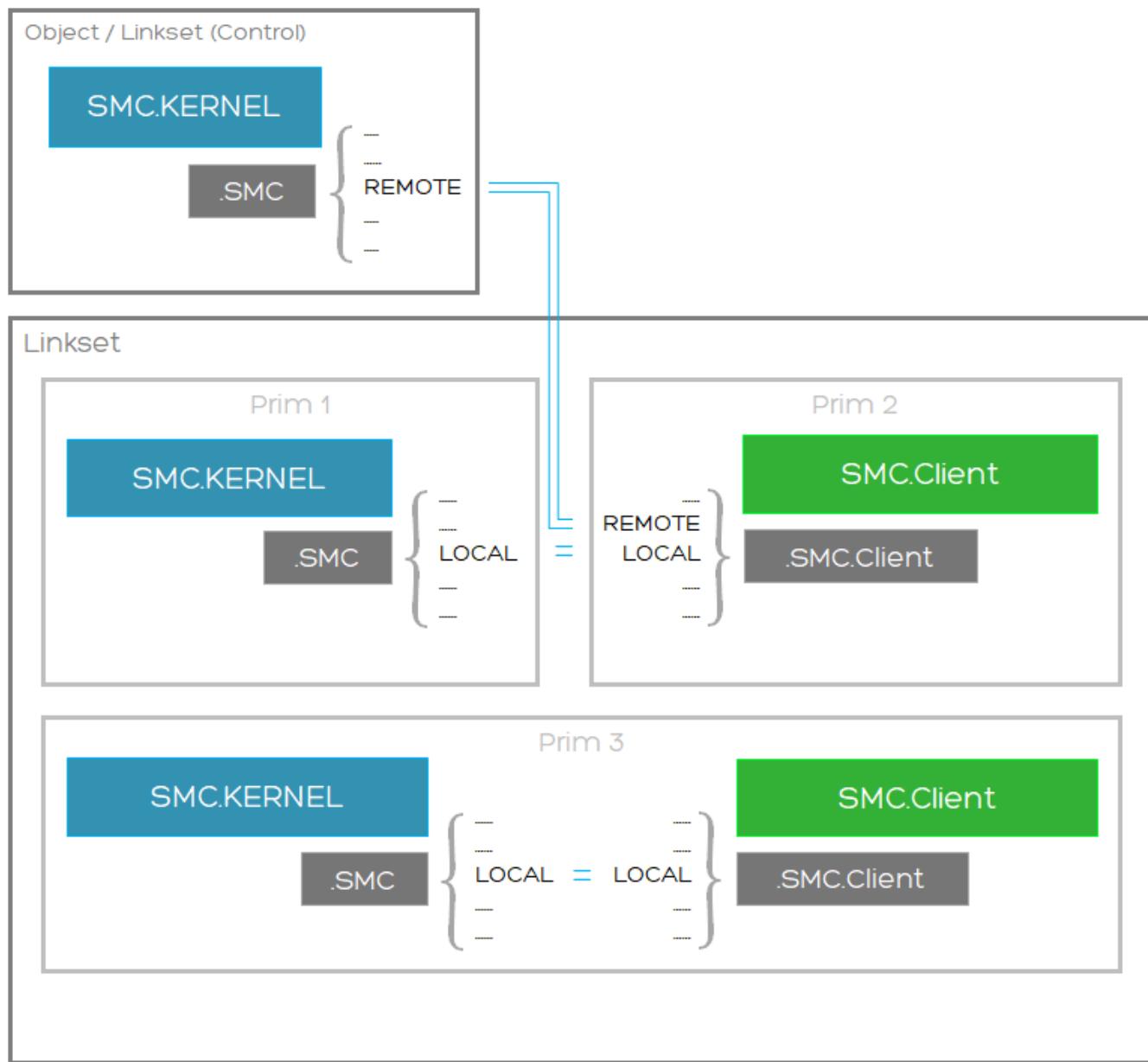
本地控制是指单个object (linkset)，将控制端 (kernel) 和受控端 (client) 放在一起，要求 **.SMC** 文件中的 **LOCAL** 和 **.SMC.Client** 文件中的 **LOCAL** 相匹配。



多重部署

在同一个linkset中，无论放在哪个prim里，只要它们的LOCAL相同，都可以发挥作用。当然，每个prim最多只能有一个kernel和一个client。

内核和客户端的关系可以是 1v1、1vN、Nv1 或 NvN。



配置

.SMC

配置项	类型	取值	默认	说明
DEBUG	integer	0 / 1	0	排障模式，开启时会输出较多信息
LOCAL	integer	-2147483648 ~ 2147483647 (0 无效)	0	本地通信频道，多用于菜单形式

配置项	类型	取值	默认	说明
REMOTE	integer	-10000 ~ 10000	0	远程通信频道偏移量（注意：这是私有频道偏移量，并不是确切的频道），一般用于HUD。
CACHE	integer	0/1	0	资源缓冲(UUID)，如果配置中使用的图片出现大量重用的情况，建议开启，可以节省大量内存。
RANGE	integer	0/1/2/3	0	控制距离，0:10米, 1:20米, 2:100米, 3:整个地区
LINES	list			详细书写规则会在下文中介绍

LINES

LINES 的书写规则较为复杂，您可以使用表格里为您提供的常量，也可以直接书写编号。

另外请注意参数的数量。

PART

部位/目标/选择器

- PART代表一个或者多个目标(prim + face)，会被更换材质的部位。可以理解为：PART所定义的是一个选择(查找)器。
- PART后面必须跟随**4个参数**

```
list LINES = [
    PART, "[名称]", {匹配类型}, "{匹配文本}", {面}
];
```

参数	类型	取值	说明
名称	string	任意	一组LINES的配置中，不可重复，这是用来换材质的依据之一，在本地菜单模式中也会作为选项来使用
匹配类型	integer	见下表	用于描述匹配的类型
匹配文本	string/integer	任何	用于匹配的名称或者描述，与参数2配合进行定义
面	string/integer	-1~7/"01234567"/ALL_SIDES	目标PRIM的哪个(些)面，PRIM的面编号(0~7)。 可以传递字符串比如"0267"，将会匹配多个面，不必按顺序，但不可重复。 也可以写 ALL_SIDES(-1)，此时不可再写其它面，因为ALL_SIDES代表所有面。

匹配类型

常量	对应值	说明
FULL	0	全文匹配PRIM名称
PREFIX	1	匹配PRIM名称的前缀
SUFFIX	2	匹配PRIM名称的后缀
SMART	3	智能匹配PRIM名称(暂时不可用)
CONST	4	匹配文本以SL中常量的方式，匹配文本可以是: LINK_SET, LINK_ALL_CHILDREN, LINK_ALL_OTHERS, LINK_ROOT, LINK_THIS
DFULL	10	全文匹配PRIM描述
DPREFIX	11	匹配PRIM描述的前缀
DSUFFIX	12	匹配PRIM描述的后缀
DSMART	13	智能匹配PRIM描述(暂时不可用)

举例

匹配名称为 "A" 的PRIM的第 3、4 面

```
list LINES = [
    PART, "Part A", FULL, "A", "34"
];
```

匹配 "名称前缀是 Rect" 的PRIM的所有面

```
list LINES = [
    PART, "All part starting with Rect", PREFIX, "Rect", ALL_SIDES
];
```

匹配 "名称末尾是 3" 的PRIM的第 0 面

```
list LINES = [
    PART, "All part ending with 3", SUFFIX, "3", 0
];
```

匹配 "除了脚本所在PRIM之外的其他PRIM" 的第 1、2、5 面

```
list LINES = [
    PART, "All others", CONST, LINK_ALL_OTHERS, "125"
];
```

匹配 "备注前缀为 top" 的第 所有 面

```
list LINES = [
    PART, "TOP", DSUFFIX, "top", ALL_SIDES
];
```

SET

配色/主题/材质方案

- SET代表着一套材质方案，它是非常自由的配置方式。
- SET的定义不能独立存在，**它必须跟在一个PART后面。**
- SET对应多种属性，属性所需参数数量也会有所不同。

```
list LINES = [
    PART, ...,
    SET, {属性}, ..., {属性}, ..., {属性}, ...
];
```

属性

参考 [PRIM_TEXTURE](#)

属性	对 应 值	对应属性	描述	参 数 数 量	值	说明
D	0	PRIM_TEXTURE	漫反 射贴 图	1	{贴图}	仅换图，其它参 数继承
DP	1	PRIM_TEXTURE	漫反 射(详 细)	4	{贴图}, {重复}, {位 置}, {旋转}	设置漫反射相关 的所有属性
N	2	PRIM_NORMAL	硬表 面贴 图	1	{贴图}	仅换图，其它参 数继承
NP	3	PRIM_NORMAL	硬表 面(详 细)	4	{贴图}, {重复}, {位 置}, {旋转}	设置硬表面相关 的所有属性
S	4	PRIM_SPECULAR	光泽 贴图	1	{贴图}	仅换图，其它参 数继承

属性	对 应 值	对应属性	描述	参 数 数 量	值	说明
SP	5	PRIM_SPECULAR	光泽 (详细)	7	{贴图}, {重复}, {位 置}, {旋转}, {反光颜 色}, {反光度}, {环境 光强度}	设置光泽相关的 所有属性
C	6	PRIM_COLOR	颜色	1	{颜色}	color 与 alpha 可以分开设置
A	7	PRIM_COLOR	透明 度	1	{透明度}	color 与 alpha 可以分开设置
G	8	PRIM_GLOW	发光	1	{强度}	灯泡一样的光
F	9	PRIM_FULLBRIGHT	全亮 模式	1	{布尔}	开启或者关闭
B	10	PRIM_BUMP_SHINY	硬表 面和 反光	2	{强度}, {模式}	系统自带的那个
T	11	PRIM_TEXGEN	贴图 模式	1	{模式}	默认/平面
M	12	PRIM_ALPHA_MODE	透明 模式	2	{模式}, {遮罩屏蔽}	不管用不用 遮 罩屏蔽, 第二个 参数都不能少
GR	13	PRIM_RENDER_MATERIAL	材质 设定	1	{render_material}	目录中的材质或 者它们的 UUID, 通常用 来开启或关闭 PBR模式
GB	14	PRIM_GLTF_BASE_COLOR	GLTF 漫反 射贴 图	1	{贴图}	仅换图, 其他参 数继承
GBC	15	PRIM_GLTF_BASE_COLOR	GLTF 漫反 射颜 色	1	{颜色}	仅换颜色, 其他 参数继承
GBA	16	PRIM_GLTF_BASE_COLOR	GLTF 漫反 射透 明度	1	{透明度}	仅换透明度, 其 他参数继承

属性	对 应 值	对应属性	描述	参 数 数 量	值	说明
GBM	17	PRIM_GLTF_BASE_COLOR	GLTF 漫反 射透 明模 式	2	{模式}, {遮罩屏蔽}	不管用不用 遮 罩屏蔽, 第二个 参数都不能少
GBD	18	PRIM_GLTF_BASE_COLOR	GLTF 漫反 射双 面模 式	1	{布尔}	仅换此项, 其他 参数继承。是否 显示一个单面的 背面。
GBP	19	PRIM_GLTF_BASE_COLOR	GLTF 漫反 射(详 细)	9	{贴图}, {重复}, {位 置}, {旋转}, {颜色}, {透明度}, {透明模 式}, {遮罩屏蔽}, {双 面}	设定漫反射全部 参数
GN	20	PRIM_GLTF_NORMAL	GLTF 硬表 面贴 图	1	{贴图}	仅换图, 其他参 数继承
GNP	21	PRIM_GLTF_NORMAL	GLTF 硬表 面(详 细)	4	{贴图}, {重复}, {位 置}, {旋转}	设定硬表面全部 参数
GM	22	PRIM_GLTF_METALLIC_ROUGHNESS	GLTF 金属 度和 粗糙 度贴 图	1	{贴图}	仅换贴图, 其他 参数继承
GMM	23	PRIM_GLTF_METALLIC_ROUGHNESS	GLTF 金属 度	1	{金属度}	仅变更金属度, 其他参数继承。 取值 0.0~1.0
GMR	24	PRIM_GLTF_METALLIC_ROUGHNESS	GLTF 粗糙 度	1	{粗糙度}	仅变更粗糙度, 其他参数继承。 取值 0.0~1.0

属性	对 应 值	对 应属性	描述	参 数 数 量	值	说明
GMP	25	PRIM_GLTF_METALLIC_ROUGHNESS	GLTF 金属度和粗糙度(详细)	6	{贴图}, {重复}, {位置}, {旋转}, {金属度}, {粗糙度}	设置金属度和粗糙度的全部参数
GE	26	PRIM_GLTF_EMISSIVE	GLTF 自发光贴图	1	{贴图}	仅换贴图，其他参数继承
GET	27	PRIM_GLTF_EMISSIVE	GLTF 自发光颜色	1	{颜色}	仅换颜色，其他参数继承
GEP	28	PRIM_GLTF_EMISSIVE	GLTF 自发光(详细)	5	{贴图}, {重复}, {位置}, {旋转}, {颜色}	设置自发光全部参数

如果值给予空字符串，表示不替换（继承当前的值）

举例

更换贴图，全部硬表面，透明度，发光

```
list LINES = [
    PART, ...,
    SET, "name_1", D, "{uuid}", NP, "{uuid}", <1.0, 1.0, 0.0>, <0.0, 0.0, 0.0>, 0.0,
    A, 0.6, G, 0.02
]
```

更换颜色，全亮模式，清空光泽贴图

```
list LINES = [
    PART, ...,
    SET, "name_2", C, <1.0, 0.0, 0.0>, F, TRUE, S, NULL_KEY
]
```

更换漫反射的位置和旋转，同时，不改变现有的贴图和重复

```
list LINES = [
    PART, ...,
    SET, "name_3", DP, "", "", <0.125, 0.4, 0.0>, 135.65
]
```

.SMC.Client

配置项	类型	取值	默认	说明
DEBUG	integer	0 / 1	0	排障模式，开启时会输出较多信息
LOCAL	integer	2147483647 (0 ~ 无效)	0	本地通信频道，多用于菜单形式
REMOTE	integer	-10000 ~ 10000	0	远程通信频道偏移量（注意：这是私有频道偏移量，并不是确切的频道），一般用于HUD。
DEBOUNCE	float	≥ 0.0	0.0	防抖时长，在这个时间内的变化均会累计，直到没有更换材质的操作并在本时长后开始生效，避免频繁切换带来的效率瓶颈。
CACHE	integer	0 / 1	0	选择器缓存，用缓存换取更高的匹配速度，注意：开启本选项后，不可以对物体进行link与unlink操作，否则会出现错误。

.SMC.Menu

配置项	类型	取值	默认	说明
DEBUG	integer	0 / 1	0	排障模式，开启时会输出较多信息
TOUCH	integer	0 / 1	0	菜单可否由点击弹出
OWNER_ONLY	integer	0 / 1	0	点击着是否必须为所有者
SETS	integer	0 / 1	0	套装选项，在部位（PART）列表中增加“[SETS]”选项，进入套装（SETS）列表菜单
SETS_ON_TOP	integer	0 / 1	0	顶级菜单 部位（PART）列表 被替换为 套装（SETS）列表
PARTS	integer	0 / 1	0	如果 SETS_ON_TOP 开启，在套装（SETS）菜单中增加一项 “[PART]”，作为部位（PART）菜单的入口
MENU_OPEN_LOCAL_NUM	integer	-2147483648 ~ 2147483647	0	触发菜单弹出的本地 num (0 无效)

配置项	类型	取值	默认	说明
MENU_BACK_LOCAL_NUM	integer	-2147483648 ~ 2147483647 (0 无效)	0	返回上一层菜单的回调
MENU_BACK_OVERWRITE	string	任意	空字符串	替换返回选项文案
MENU_PREV_OVERWRITE	string	任意	空字符串	替换前一页选项文案
MENU_NEXT_OVERWRITE	string	任意	空字符串	替换后一页选项文案
DIALOG_SETS	string	任意	空字符串	设置套装(SETS)菜单提示信息，换行请使用"\n"
DIALOG_SET	string	任意	空字符串	设置配色(SET)菜单提示信息，换行请使用"\n"
DIALOG_PART	string	任意	空字符串	设置部位(PART)菜单提示信息，换行请使用"\n"
SETS_LIST	list	键/值对	空数组	见下文样例

SETS_LIST

格式

```
list SETS_LIST = [
    "{套装名称}", "{PART}.{SET}",
    ...
];
```

```
list SETS_LIST = [
    "{套装名称}", ".{SET}",
    ...
];

list SETS_LIST = [
    "{套装名称}", ".{SET_A},.{SET_B},{PART1}.{SET_C},...",
    ...
];
```

举例

```
list SETS_LIST = [
    "BLACK", ".BLACK"
];
```

```
list SETS_LIST = [
    "BLACK&RED", ".BLACK,.RED"
];
```

```
list SETS_LIST = [
    "BLACK&TOP_RED", ".BLACK, TOP.RED"
];
```

```
list SETS_LIST = [
    "BTM_B&T_R", "BOTTOM.BLACK, TOP.RED"
];
```

核心 KERNEL 本地接口

消息字符串分隔符是 "◆"

```
llDumpList2String([...], "◆")
```

提交

-643323390

对预定义部位应用一个预定义属性，并支持自定义追加与覆盖

```
llMessageLinked(LINK_SET, -643323390, "{PART}◆{SET}[◆{DATA...}]", "");
```

- PART 与 SET 必须在配置中定义过，另外 SET 必须属于 PART，本条提交才会生效。
- DATA 部分为追加或覆盖属性，写法如 SET 中的属性，可选参数。

举例

```
// 最常用的(使用预定义配置 LINES)
llMessageLinked(LINK_SET, -643323390, "TOP◆BLACK", "");
// 带有自定义属性的
llMessageLinked(LINK_SET, -643323390, "TOP◆BLACK◆6◆<1.0, 0.0,
0.0>◆9◆TRUE◆4◆ee509dfd-0974-6fb5-3eea-2504fa13ef4c", "");
// 方便的写法
llMessageLinked(LINK_SET, -643323390, llDumpListToString([ "TOP", "BLACK", 6, <1.0,
0.0, 0.0>, 9, TRUE, 4, "ee509dfd-0974-6fb5-3eea-2504fa13ef4c"], "◆"), "");
// 建议使用常量，可写为
llMessageLinked(LINK_SET, -643323390, llDumpListToString([ "TOP", "BLACK", C, <1.0,
0.0, 0.0>, F, TRUE, S, "ee509dfd-0974-6fb5-3eea-2504fa13ef4c"], "◆"), "");
```

* 批量模式

```
llMessageLinked(LINK_SET, -643323390, "◆{SET}", "");
```

- 如果不给予 PART，此时将触发全量匹配模式，自动查找出包含 SET 的所有 PART，并批量生效。
- 此时再追加的 DATA，它们会应用在所有相关 PART。

举例

```
// 定义的 PART 中，比如有 TOP、MIDDLE、BOTTOM
// 其中 TOP、MIDDLE 包含 BLACK，那么，将会自动查找出 TOP、MIDDLE，应用 BLACK
// 相当于执行了 TOP◆BLACK 和 MIDDLE◆BLACK
llMessageLinked(LINK_SET, -643323390, "◆BLACK", "");
```

-643323392

对预定义部位应用一套自定义属性

```
llMessageLinked(LINK_SET, -643323392, "{PART}◆{DATA...}", "");
```

- PART 必须在配置中定义过，本条提交才会生效。
- DATA 的写法如 SET 中的属性，对 PART 应用自定义属性，与上面的区别是无需指定 SET（不验证 SET 的合法性）。

举例

```
llMessageLinked(LINK_SET, -643323392, "TOP◆6◆<1.0, 0.0,
0.0>◆9◆TRUE◆4◆ee509dfd-0974-6fb5-3eea-2504fa13ef4c", "");
// 方便的写法
llMessageLinked(LINK_SET, -643323392, llDumpListToString([["TOP", 6, <1.0, 0.0,
0.0>, 9, TRUE, 4, "ee509dfd-0974-6fb5-3eea-2504fa13ef4c"], "◆"], ""));
// 建议使用常量，可写为
llMessageLinked(LINK_SET, -643323392, llDumpListToString([["TOP", C, <1.0, 0.0,
0.0>, F, TRUE, S, "ee509dfd-0974-6fb5-3eea-2504fa13ef4c"], "◆"], ""));
```

-643323393

对自定义的部位应用一套自定义属性

```
llMessageLinked(LINK_SET, -643323393, "{DATA...}", "");
```

- DATA 的写法必须包含完整的 PART + SET 内容。
- 无需依照配置，这是一条完全独立的选择 + 属性 规则。

举例

```
llMessageLinked(LINK_SET, -643323393, "2◆top◆0123◆6◆<1.0, 0.0,
0.0>◆9◆TRUE◆4◆ee509dfd-0974-6fb5-3eea-2504fa13ef4c", "");
// 方便的写法
llMessageLinked(LINK_SET, -643323393, llDumpListToString([2, "top", "0123", 6,
<1.0, 0.0, 0.0>, 9, TRUE, 4, "ee509dfd-0974-6fb5-3eea-2504fa13ef4c"], "◆"), "");
// 建议使用常量，可写为
llMessageLinked(LINK_SET, -643323393, llDumpListToString([SUFFIX, "top", "0123", C,
<1.0, 0.0, 0.0>, F, TRUE, S, "ee509dfd-0974-6fb5-3eea-2504fa13ef4c"], "◆"), "");
```

请求(带回调)

-643323410

请求 PART 列表

```
llMessageLinked(LINK_SET, -643323410, "", id);
```

KERNEL 回调: **-643323411**

```
llMessageLinked({SENDER}, -643323411, "{PART1}◆{PART2}◆....", id);
```

```
link_message(integer sender_num, integer num, string str, key id)
{
    if(num == -643323411) {
        list parts = llParseString2List(str, ["♦"], []);
        // parts: ["PART1", "PART2", ....]
    }
}
```

-643323420

请求 SET 列表

```
llMessageLinked(LINK_SET, -643323420, "{PART}", id);
```

KERNEL 回调: **-643323411**

```
llMessageLinked({SENDER}, -643323421, "{SET1}♦{SET2}♦....", id);
```

```
link_message(integer sender_num, integer num, string str, key id)
{
    if(num == -643323421) {
        list sets = llParseString2List(str, ["♦"], []);
        // sets: ["SET1", "SET2", ....]
    }
}
```

菜单 SMC.Menu 的本地接口

.SMC.Menu 中定义的 MENU_OPEN_LOCAL_NUM 将用于通信**唤起菜单**

```
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "", {USER});
```

唤起菜单并展示 PART(部位) 列表

```
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "part", {USER});
```

// 使用按钮覆盖定义并提供回调，仅对此菜单有效。
// 选择一个选项或者点击返回按钮会发送一个本地消息。

```
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "part◆{BackButton}◆{LocalNum}",  
{USER});
```

唤起菜单并根据参数提供的 PART(部位名称) 展示 SET(样式) 列表

```
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "set◆{PART}", {USER});  
  
// 使用按钮覆盖定义并提供回调，仅对此菜单有效。  
// 选择一个选项或者点击返回按钮会发送一个本地消息。  
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "set◆{PART}◆{BackButton}  
◆{LocalNum}", {USER});
```

唤起菜单并展示 SETS(套装) 列表

```
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "sets", {USER});  
  
// 使用按钮覆盖定义并提供回调，仅对此菜单有效。  
// 选择一个选项或者点击返回按钮会发送一个本地消息。  
llMessageLinked(LINK_SET, {MENU_OPEN_LOCAL_NUM}, "sets◆{BackButton}◆{LocalNum}",  
{USER});
```

* 感谢亲爱的 **Amber0089**