| 特征 | 描述 | 示例 |
|---|---|---|
| 命名空间 | 引用模型元素时，您必须对这些元素添加前缀。存在下列前缀：<br><br>**dp:**（ 表示全局属性 ），**ev:**（ 表示事件 ），**v:**（ 表示局部变量 ），**f:**（ 表示函数 ） | dp:x = 100; // 设置全局属性<br>fire ev:back(); // 触发事件<br>f:trace_string(“hello world”); // 调用函数 |
| 访问全局属性 | 通过将全局属性放置在赋值左侧来编写此属性。通过在表达式中的任何其他位置使用全局属性来读取此属性。重定向引用 (=>) 是全局属性赋值的特殊形式。 | dp:x = 5; // 写入 x<br>dp:x = dp:y + dp:z; // 读取 y 和 z<br>length dp:aList; // 读取列表属性的长度<br>dp:refX => dp:x; // 重定向 |
| 发送事件 | 语法：<br><br>**fire ev:**<identifier>**(**<parameter-list>**);**<br><br>可以在超时后发送事件。<br>可以使用 **cancel_fire** 表达式取消此延迟事件。<br><br>语法：<br>**fire_delayed** <timeout>, **ev:**<identifier>**(**<parameter-list>**);**<br>**cancel_fire ev:**<identifier>**;** | fire ev:back();<br>fire ev:mouseClick(10, 20);<br><br>fire_delayed 3000, ev:back(); // 在 3 秒内发送时间“返回”。<br>cancel_fire ev:back; // 取消事件 |
| 对事件做出响应 | 要对事件做出响应，请使用 **match_event**。这是 **if-then-else** 语句的特殊形式。**If** 和 **else** 分支必须始终具有相同类型。如果在赋值右侧使用，则 **else** 分支是必需的。<br><br>语法：<br>**match_event v:**<identifier> **= ev:**<identifier><br>**in** <sequence><br>**else** <sequence> | match_event v:event = ev:back in {<br>f:trace_string(“back event received”);<br>}<br><br>v:this.x = match_event v:event = ev:back in 10 else 0; |
| 访问事件参数 | **match_event** 的 **in** 表达式可以访问事件参数。使用点标记来访问事件参数。 | match_event v:event = ev:mouseClick in {<br>v:this.x = v:event.x;<br>v:this.y = v:event.y;<br>} |
| 访问专用小组件属性 | 如果某个脚本是小组件（ 小组件操作、输入响应 ）的组成部分，则它可以访问该小组件的属性。通过引用当前小组件的变量，名为 **v:this** 的特殊局部变量可用。使用点标记对专用属性进行寻址。 | v:this.text = “hello world”;<br>v:this.x = 10; |
| 浏览小组件树 | 如果某个脚本是小组件的组成部分，则它可以访问其他小组件的属性。使用小组件树导航运算符：<br>**->**。要访问父代小组件，请使用标识符：**^**。 | v:this->^->caption.text = “Play”;<br>// 转到父代，转至标题，属性文本<br>v:this->^.x-= 1; // 转到父代，属性 x |
| 字符串格式 | **+** 运算符用于连接字符串。有关更多字符串转换函数，请参阅文档 | v:this.text = “current speed: ” +<br>f:int2string(dp:speed) + “km/h”; |

| 特征 | 描述 | 示例 |
|---|---|---|
| 常量 | 编写的字符串常量可以不带引号。<br>颜色常量是 RGBA 四重。 | “hello world” // 字符串常量<br>Napoleon // 字符串常量<br>5 // 整数常量<br>color:0,235,0,255 // EB 绿色 |
| 算术、逻辑和赋值运算符 | 加法和字符串连接:+，减法:-，乘法:*，除法:/，取模:%，大于:>，小于:<，大于或等于:>=，小于或等于:<=，等于:==，不等于:!=，与:&&，或:\|\|，非:!，赋值:=，赋值增加:+=，赋值减小:-= | dp:myString = “Hello” + “World”;<br>dp:count += 1; // 增加一 |
| 序列 | 序列可以是单个表达式，也可以括在大括号中的一系列表达式。序列中的最后一个表达式为序列的值。 | if( dp:something )<br>  dp:x = 5; // 单个表达式<br>if( dp:other ) {<br>  dp:x = 5; // 括在大括号<br>  dp:y = 10; // 中的表达式<br>} |
| 局部变量 | 使用 **let** 绑定引入局部变量。不允许使用未初始化的变量。<br>**let** 绑定可以进行嵌套。<br><br>语法:<br>**let v:**<identifier> **=** <expression>**;**<br>  **v:**<identifier2> **=** <expression>**;**<br>  ...<br>  **in** <sequence> | let v:x = 42;<br>  v:text = “hello world”;<br>in {<br>  v:this.x = v:x;<br>  v:this.text = v:text;<br>} |
| WHILE 循环 | **while** 循环包含两个表达式：条件表达式和主体表达式。在条件生成 false 之前，将对主体重复求值。<br><br>语法:<br>**while(** <expression> **)** <sequence> | dp:i = 0;<br>while( dp:i <= 10 ) {<br>  dp:sum += i;<br>  dp:i += 1;<br>} |
| IF-THEN-ELSE | **If-then-else** 的行为类似于 C 和 Java 中的三元条件运算符。如果在赋值右侧使用它，则 **else** 分支是必需的并且这两个分支必须具有相同类型。<br><br>语法:<br>**if(** <expression> **)** <sequence> **else** <sequence> | if( dp:buttonClicked ) {<br>  v:this.x = dp:x;<br>}<br>else {<br>  v:this.x = 0;<br>}<br><br>v:this.x = if( dp:buttonClicked ) dp:x else 0; |
| 注释 | 允许使用 C 样式块注释和<br>C++ 样式行注释。 | /* 此为 C 样式块注释 */<br>// 此为 C++ 样式行注释 |
| 返回值 | 脚本中的最后一个表达式是返回值。<br>要强制使用无效类型的返回值，请使用 **unit** 或 **{}** | dp:x + 2; // 返回数据池条目 x + 2 |