| 피처 | 설명 | 예제 |
|---|---|---|
| 네임스페이스 | 모델 요소를 언급할 때는 접두사를 붙여야 합니다. 모델 요소에 따른 접두사는 다음과 같습니다.<br>**dp:** 데이터풀 항목, **ev:** 이벤트, **v:** 로컬 변수, **f:** 함수 | dp:x = 100; // set a datapool item<br>fire ev:back(); // fire an event<br>f:trace_string("hello world"); // call a function |
| 데이터풀 항목에 액세스 | 지정의 왼쪽에 배치하여 데이터풀 항목을 씁니다.<br>식의 모든 곳에서 사용하여 데이터풀 항목을 읽습니다.<br>리디렉션 참조(=>)는 데이터풀 항목 지정의 특수 형식입니다. | dp:x = 5; // writing to x<br>dp:x = dp:y + dp:z; // reading y and z<br>length dp:aList; // read the length of a list datapool item<br>dp:refX => dp:x; // redirect link |
| 이벤트 전송 | 구문:<br>**fire ev:**<identifier>**(**<parameter-list>**);**<br><br>이벤트가 시간 초과 후에 전송될 수 있습니다.<br>**cancel_fire** 식으로 연기된 이벤트를 취소할 수 있습니다.<br><br>구문:<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(); // send the event "back" in 3 seconds.<br>cancel_fire ev:back; // cancel the event |
| 이벤트에 반응 | **match_event**를 사용하여 이벤트에 반응합니다. **if-then-else** 문의 특수 형식입니다. **else** 분기의 유형이 동일해야 합니다. 지정의 오른쪽에 사용하면 else 분기가 필수입니다.<br><br>구문:<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; |
| 이벤트 매개변수에 액세스 | **match_event**의 **in** 식으로 이벤트 매개변수에 액세스할 수 있습니다. 점 표기법을 사용하여 이벤트 매개변수에 액세스합니다. | match_event v:event = ev:mouseClick in {<br>v:this.x = v:event.x;<br>v:this.y = v:event.y;<br>} |
| 위젯 속성에 액세스 | 스크립트는 위젯 작동 명령에 반응하는 입력 처리를 기술하는 위젯의 요소이며, 스크립트로 해당 위젯의 속성에 액세스할 수 있습니다.<br>**v:this**는 특수 로컬 변수로서 현재 위젯을 참조하여 사용할 수 있습니다. 점 표기법을 사용하여 위젯 속성의 주소를 지정합니다. | v:this.text = "hello world";<br>v:this.x = 10; |
| 위젯 트리 이동 | 위젯의 요소인 스크립트로 다른 위젯의 로컬 속성에 액세스합니다. 위젯 트리 내비게이션 연산자(->)를 사용합니다.<br>식별자(^)를 사용하여 상위 위젯에 액세스합니다 ^. | v:this->^->caption.text = "Play"; // goto parent, goto caption, property text<br>v:this->^.x-= 1; // goto parent, property x |
| 문자열 서식 지정 | **+** 연산자가 문자열을 연결합니다. 문자열 변환 함수에 대한 자세한 정보는 문서를 참조하십시오. | v:this.text = "current speed: " + f:int2string(dp:speed) + "km/h"; |
| 문자열 비교 | 대/소문자를 구분하여 두 문자열을 비교하려면 같음 연산자**(== or !=)**를 사용합니다.<br>대/소문자를 구분하지 않고 문자열을 비교하려면 같음 연산자**(=Aa=)**를 사용합니다. | "name" == "NAME" // false<br>"name" != "NAME" // true<br>"name" =Aa= "NAME" // true |
| 언어 변경 | EB GUIDE 모델의 모든 데이터풀 항목 언어를 변경하려면 **language**를 사용합니다. 이 작업은 비동기 방식으로 수행됩니다.<br>구문:<br>**f:language(l:**<identifier>**)** | f:language(l:Standard) // changes language to the standard language<br>f:language(l:German) // changes language to German |

| 피처 | 설명 | 예제 |
|---|---|---|
| 스킨 변경 | EB GUIDE 모델의 모든 데이터풀 항목 스킨을 변경하려면 **skin**을 사용합니다. 이 작업은 비동기 방식으로 수행됩니다.<br><br>구문:<br>**f:skin(s:**<identifier>**)** | f:skin(s:Standard) // changes to the standard skin<br>f:skin(s:“myskin”) // changes to a user-defined skin |
| 상수 | 문자열 상수는 따옴표를 사용하지 않고 쓸 수 있습니다.<br>색 상수는 RGBA 4중 문자입니다. | “hello world” // string constant<br>Napoleon // string constant<br>5 // integer constant<br>color:0,235,0,255 // EB green |
| 산술, 논리 및 지정 연산자 | 덧셈 및 문자열 연결: **+**, 뺄셈: **-**, 곱셈: *****, 나눗셈: **/**, 모듈로: **%**, 큼: **>**, 작음: **<**,<br>크거나 같음: **>=**, 작거나 같음: **<=**, 같음: **==**, 같지 않음: **!=**, 및: **&&**, 또는: **||**,<br>부정: **!**, 지정: **=**, 지정 증가: **+=**, 지정 감소: **-=** | dp:myString = “Hello” + “World”;<br>dp:count += 1; // increment one |
| 시퀀싱 | 시퀀스는 단일 식이거나 중괄호에 묶인 일련의 식입니다. 시퀀스의 마지막 식이 시퀀스 값입니다. | if( dp:something )<br>  dp:x = 5;  // single expression<br>if( dp:other ) {<br>  dp:x = 5;  // sequence enclosed<br>  dp:y = 10; // in curly braces<br>} |
| 로컬 변수 | **let** 바인딩을 사용하여 로컬 변수를 도입합니다. 초기화되지 않은 변수는 사용할 수 없습니다.<br>**let**바인딩은 중첩될 수도 있습니다.<br><br>구문:<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 루프 | **while** 루프는 조건 식과 본문 식으로 구성됩니다. 본문 식은 조건 식이 ‚false‘가 될 때까지 반복적으로 평가됩니다.<br><br>구문:<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** 는 C 및 **Java**의 3진 조건부 연산자와 비슷하게 작동합니다. 이 조건문을 지정의 오른쪽에 사용하는 경우에는 **else** 분기가 필수이며 두 분기 모두 같은 유형이어야 합니다.<br><br>구문:<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; |
| 주석 | C 스타일 블록 주석과<br>C++ 스타일 줄 주석을 사용합니다. | /* this is a C style block comment */<br>// this is a C++ style line comment |
| 반환 값 | 스크립트의 마지막 식이 반환 값입니다.<br>void 유형의 반환 값을 도출하려면 **unit** 또는 다음을 사용합니다. **{}** | dp:x + 2; // returns datapool item x plus 2 |