Elektrobit

EB GUIDE Studio

User manual

バージョン6.4.1.0

Elektrobit Automotive GmbH
Am Wolfsmantel 46
D-91058 Erlangen
GERMANY

Phone: +49 9131 7701-0
Fax: +49 9131 7701-6333
http://www.elektrobit.com

Legal notice

目次

1. 本書について 14
1.1. 対象者: モデラー 14
1.2. ユーザーマニュアルの構成 14
1.3. 本書で使用する表記スタイル 15
1.4. ネーミングルール 17
2. 安全で正しい使い方 18
2.1. 使用目的 18
2.2. 考えられる誤用 18
3. サポート 19
4. EB GUIDEの基本 20
4.1. EB GUIDE product line 20
4.2. EB GUIDE Studio 20
4.2.1. ヒューマンマシンインターフェイスの動作のモデル化 20
4.2.2. ヒューマンマシンインターフェイスの外観のモデル化 21
4.2.3. データの処理 21
4.2.4. EB GUIDEモデルのシミュレート 21
4.2.5. EB GUIDEモデルのエクスポート 22
4.3. EB GUIDE TF 22
5. チュートリアル: はじめに 24
5.1. EB GUIDEの起動 24
5.2. プロジェクトの作成 25
5.3. ヒューマンマシンインターフェイスの動作のモデル化 26
5.4. ヒューマンマシンインターフェイスの外観のモデル化 29
5.5. シミュレーションの開始 32
6. バックグラウンド情報 33
6.1. 3Dグラフィック 33
6.1.1. サポートされている3Dグラフィック形式 33
6.1.2. 3Dグラフィックファイルの設定 33
6.1.3. 3Dグラフィックファイルのインポート 34
6.2. アニメーション 35
6.2.1. ウィジェットのアニメーション 35
6.2.2. ビュー遷移のアニメーション 36
6.3. アプリケーションとモデルを結ぶアプリケーションプログラミングインターフェイス 36
6.4. 通信コンテキスト 37
6.5. グラフィカルユーザーインターフェイスの要素 37
6.5.1. プロジェクトセンター 37
6.5.1.1. ナビゲーションエリア 38
6.5.1.2. コンテンツエリア 38
6.5.2. プロジェクトエディター 39

6.5.2.1. ナビゲーションエリア 39
6.5.2.2. コンテンツエリア 41
6.5.2.3. コマンドエリア 42
6.5.2.4. ツールボックス 43
6.5.2.5. プロパティパネル 44
6.5.2.6. ステータスバー 45
6.5.2.7. 問題エリア 45
6.6. データプール 45
6.6.1. 概念 45
6.6.2. データプールアイテム 46
6.6.3. ウィンドウ表示リスト 46
6.7. EB GUIDEモデルとEB GUIDEプロジェクト 47
6.8. イベント処理 48
6.8.1. イベントシステム 48
6.8.2. イベント 48
6.9. 拡張機能 49
6.9.1. EB GUIDE Studio拡張機能 49
6.9.2. EB GUIDE GTF拡張機能 49
6.10. 言語 49
6.10.1. EB GUIDE Studioの表示言語 49
6.10.2. EB GUIDEモデルの言語 50
6.10.3. 言語依存テキストのエクスポートとインポート 50
6.11. リソース管理 50
6.11.1. フォント 51
6.11.2. イメージ 51
6.11.2.1. 9-patchイメージ 51
6.11.3. 3Dグラフィック用メッシュ 52
6.12. スクリプト言語EB GUIDEスクリプト 53
6.12.1. アプリケーションの機能とエリア 53
6.12.2. ネームスペースと識別子 53
6.12.3. コメント 54
6.12.4. データ型 54
6.12.5. 式 55
6.12.6. 定数と参照 55
6.12.7. 算術式と論理式 56
6.12.8. L値とR値 57
6.12.9. ローカル変数 57
6.12.10. Whileループ 59
6.12.11. If-then-else 59
6.12.12. 外部関数呼び出し 60
6.12.13. データプールアクセス 61
6.12.14. ウィジェットプロパティ 62

6.12.15. リスト 63
6.12.16. イベント 64
6.12.17. 文字列の書式設定 65
6.12.18. 標準ライブラリ 66
6.13. スクリプト値 66
6.14. ショートカット、ボタン、アイコン 68
6.14.1. ショートカット 68
6.14.2. ボタン 69
6.14.3. アイコン 69
6.15. ステートマシンとステート 70
6.15.1. ステートマシン 70
6.15.1.1. ハプティックステートマシン 70
6.15.1.2. ロジックステートマシン 70
6.15.1.3. 動的ステートマシン 71
6.15.2. ステート 71
6.15.2.1. 混合ステート 71
6.15.2.2. ビューステート 73
6.15.2.3. 初期ステート 73
6.15.2.4. 最終ステート 74
6.15.2.5. 選択ステート 75
6.15.2.6. 履歴ステート 76
6.15.3. 遷移 79
6.15.4. ステートマシンの実行 83
6.15.5. EB GUIDEの記法とUML記法の比較 87
6.15.5.1. サポートされている要素 88
6.15.5.2. サポートされない要素 88
6.15.5.3. 偏差 88
6.16. タッチ入力 89
6.16.1. 非パスジェスチャー 89
6.16.2. パスジェスチャー 89
6.16.3. 入力処理とジェスチャー 90
6.16.4. マルチタッチ入力 90
6.17. ウィジェット 91
6.17.1. ビュー 91
6.17.2. ウィジェットのカテゴリ 92
6.17.3. ウィジェットプロパティ 93
6.17.4. ウィジェットテンプレート 94
7. ヒューマンマシンインターフェイスの動作のモデル化 96
7.1. ステートマシンのモデリング 96
7.1.1. ステートマシンの追加 96
7.1.2. 動的ステートマシンの追加 96
7.1.3. ステートマシンに対するエントリーアクションの定義 97

7.1.4. ステートマシンに対する終了アクションの定義 98
7.1.5. ステートマシンの削除 98
7.2. モデリングステート 98
7.2.1. ステートの追加 98
7.2.2. 混合ステートへのステートの追加 99
7.2.3. 選択ステートの追加 100
7.2.4. ステートに対するエントリーアクションの定義 101
7.2.5. ステートに対する終了アクションの定義 102
7.2.6. ステートマシンからのモデル要素の削除 103
7.3. ステート間を遷移で接続 103
7.3.1. 2つのステート間に遷移を追加 103
7.3.2. 遷移の移動 104
7.3.3. 遷移に対するトリガーの定義 105
7.3.4. 遷移への条件の追加 106
7.3.5. 遷移へのアクションの追加 107
7.3.6. ステートへの内部遷移の追加 109
8. ヒューマンマシンインターフェイスの外観のモデル化 110
8.1. ウィジェットの操作 110
8.1.1. ビューの追加 110
8.1.2. ビューへのウィジェットの追加 111
8.1.3. ビューからのウィジェットの削除 111
8.1.4. ビューへのイメージの追加 112
8.1.5. ビューへのシーングラフの追加 112
8.1.6. ラベルのフォントの変更 113
8.1.7. コンテナーを使用したウィジェットのグループ化 114
8.1.8. ビューへのインスタンシエータの追加 114
8.2. ウィジェットプロパティの操作 115
8.2.1. ウィジェットの配置 115
8.2.2. ウィジェットのサイズの変更 116
8.2.3. ウィジェットプロパティ間のリンク設定 117
8.2.4. ウィジェットプロパティとデータプールアイテムのリンク設定 119
8.2.5. ウィジェットへのユーザー定義プロパティの追加 121
8.2.5.1. `Function ():bool`タイプのユーザー定義プロパティの追加 122
8.2.6. ユーザー定義プロパティの名前の変更 123
8.3. ウィジェット機能を追加してウィジェットを拡張する 124
8.3.1. ウィジェット機能の追加 124
8.3.2. ウィジェット機能の削除 126
8.4. EB GUIDEモデルへの言語の追加 127
8.4.1. 言語の追加 128
8.4.2. 言語の削除 128
8.5. アニメーションの追加 129
8.5.1. ウィジェットのアニメーション化 129

8.5.2. ビュー遷移のアニメーション化 131
8.6. ウィジェットの再利用 132
8.6.1. テンプレートの追加 132
8.6.2. テンプレートインターフェイスの定義 133
8.6.3. テンプレートの使用 134
8.6.4. テンプレートの削除 135
9. データの処理 136
9.1. イベントの追加 136
9.2. イベントへのパラメータの追加 136
9.3. イベントへの対応 137
9.4. イベントの削除 138
9.5. データプールアイテムの追加 138
9.6. リストタイプのデータプールアイテムの編集 139
9.7. プロパティのスクリプト値への変換 140
9.8. 外部通信の確立 141
9.9. データプールアイテム間のリンク設定 143
9.10. データプールアイテムの削除 144
10. プロジェクトの処理 146
10.1. プロジェクトの作成 146
10.2. プロジェクトを開く 146
10.2.1. ファイルエクスプローラからプロジェクトを開く 146
10.2.2. プロジェクトをEB GUIDE Studioに開く 147
10.3. EB GUIDEモデルのテストと改良 147
10.3.1. EB GUIDEモデルの検証 147
10.3.2. シミュレーションの開始と停止 148
10.4. EB GUIDEモデルのエクスポート 149
10.5. EB GUIDE Studioの表示言語の変更 150
10.6. プロファイルの設定 150
10.6.1. プロファイルの複製 151
10.6.2. ライブラリの追加 151
10.6.3. メッセージの追加 153
10.6.4. シーンの設定 154
10.7. 言語依存テキストのエクスポートとインポート 154
10.7.1. 言語依存テキストのエクスポート 155
10.7.2. 言語依存テキストのインポート 156
11. チュートリアル 158
11.1. チュートリアル: 動的ステートマシンの追加 158
11.2. チュートリアル: EB GUIDEスクリプトを使用したボタン動作のモデル化 166
11.3. チュートリアル: パスジェスチャーをモデル化する 172
11.4. チュートリアル: 動的コンテンツを使用したリストの作成 174
11.5. チュートリアル: 画面内での四角形の移動 181
11.6. チュートリアル: データプールアイテムへの言語依存テキスト追加 185

11.7. チュートリアル: 3Dグラフィックの操作 188
12. 参照 194
12.1. Androidイベント 194
12.2. データプールアイテム 195
12.3. データタイプ 196
12.3.1. メッシュ 196
12.3.2. ブール値 196
12.3.3. 色 196
12.3.4. 条件スクリプト 197
12.3.5. フロート 197
12.3.6. フォント 198
12.3.7. イメージ 198
12.3.8. 整数 198
12.3.9. リスト 199
12.3.10. 文字列 200
12.4. EB GUIDEスクリプト 200
12.4.1. EB GUIDEスクリプトのキーワード 200
12.4.2. EB GUIDEスクリプトの演算子の優先順位 201
12.4.3. EB GUIDEスクリプト標準ライブラリ 202
12.4.3.1. EB GUIDEスクリプトの関数A 202
12.4.3.1.1. abs 202
12.4.3.1.2. absf 202
12.4.3.1.3. acosf 203
12.4.3.1.4. animation_before 203
12.4.3.1.5. animation_beyond 203
12.4.3.1.6. animation_cancel 203
12.4.3.1.7. animation_cancel_end 204
12.4.3.1.8. animation_cancel_reset 204
12.4.3.1.9. animation_pause 204
12.4.3.1.10. animation_play 204
12.4.3.1.11. animation_reverse 205
12.4.3.1.12. animation_running 205
12.4.3.1.13. animation_set_time 205
12.4.3.1.14. asinf 205
12.4.3.1.15. atan2f 206
12.4.3.1.16. atan2i 206
12.4.3.1.17. atanf 206
12.4.3.2. EB GUIDEスクリプトの関数C～H 206
12.4.3.2.1. ceil 206
12.4.3.2.2. changeDynamicStateMachinePriority 207
12.4.3.2.3. character2unicode 207
12.4.3.2.4. clearAllDynamicStateMachines 207

12.4.3.2.5. color2string 207
12.4.3.2.6. cosf 208
12.4.3.2.7. deg2rad 208
12.4.3.2.8. expf 208
12.4.3.2.9. float2string 209
12.4.3.2.10. floor 209
12.4.3.2.11. focusNext 209
12.4.3.2.12. focusPrevious 209
12.4.3.2.13. format_float 210
12.4.3.2.14. format_int 210
12.4.3.2.15. getLineCount 211
12.4.3.2.16. getTextHeight 211
12.4.3.2.17. getTextLength 212
12.4.3.2.18. getTextWidth 212
12.4.3.2.19. has_list_window 212
12.4.3.2.20. hsba2color 212
12.4.3.3. EB GUIDEスクリプトの関数I～R 213
12.4.3.3.1. int2float 213
12.4.3.3.2. int2string 213
12.4.3.3.3. isDynamicStateMachineActive 213
12.4.3.3.4. language 214
12.4.3.3.5. localtime_day 214
12.4.3.3.6. localtime_hour 214
12.4.3.3.7. localtime_minute 214
12.4.3.3.8. localtime_month 215
12.4.3.3.9. localtime_second 215
12.4.3.3.10. localtime_weekday 215
12.4.3.3.11. localtime_year 215
12.4.3.3.12. log10f 216
12.4.3.3.13. logf 216
12.4.3.3.14. nearbyint 216
12.4.3.3.15. popDynamicStateMachine 216
12.4.3.3.16. powf 217
12.4.3.3.17. pushDynamicStateMachine 217
12.4.3.3.18. rad2deg 217
12.4.3.3.19. rand 217
12.4.3.3.20. request_runlevel 218
12.4.3.3.21. rgba2color 218
12.4.3.3.22. round 218
12.4.3.4. EB GUIDEスクリプトの関数S～W 218
12.4.3.4.1. seed_rand 218
12.4.3.4.2. sinf 219

12.4.3.4.3. sqrtf 219
12.4.3.4.4. string2float 219
12.4.3.4.5. string2int 220
12.4.3.4.6. string2string 220
12.4.3.4.7. substring 220
12.4.3.4.8. system_time 221
12.4.3.4.9. system_time_ms 221
12.4.3.4.10. tanf 221
12.4.3.4.11. trace_dp 221
12.4.3.4.12. trace_string 222
12.4.3.4.13. transformToScreenX 222
12.4.3.4.14. transformToScreenY 222
12.4.3.4.15. transformToWidgetX 223
12.4.3.4.16. transformToWidgetY 223
12.4.3.4.17. trunc 223
12.4.3.4.18. widgetGetChildCount 224
12.5. イベント 224
12.6. gtfStartup.cfg設定ファイル 224
12.6.1. マッピング規則の構造 224
12.6.2. 信号 225
12.6.3. アクション 225
12.6.4. メッセージ 227
12.7. シーン 231
12.8. EB GUIDE GTF がサポートするタッチスクリーンタイプ 233
12.9. ウィジェット 233
12.9.1. ビュー 233
12.9.2. 基本ウィジェット 234
12.9.2.1. ラベル 235
12.9.2.2. 四角形 235
12.9.2.3. イメージ 236
12.9.2.4. コンテナー 236
12.9.2.5. インスタンシエータ 237
12.9.3. アニメーション 237
12.9.3.1. アニメーション 237
12.9.3.2. コンスタント曲線 238
12.9.3.3. 高速開始曲線 239
12.9.3.4. 低速開始曲線 239
12.9.3.5. 二次曲線 240
12.9.3.6. 正弦曲線 240
12.9.3.7. スクリプト曲線 241
12.9.3.8. リニア曲線 241
12.9.3.9. リニア補間曲線 242

12.9.4. 3Dウィジェット 242
12.9.4.1. シーングラフ 243
12.9.4.2. シーングラフノード 243
12.9.4.3. カメラ 244
12.9.4.4. 指向性ライト 244
12.9.4.5. 材質 244
12.9.4.6. メッシュ 245
12.9.4.7. 点ライト 245
12.9.4.8. スポットライト 245
12.10. ウィジェット機能 246
12.10.1. 共通 246
12.10.1.1. 子の可視性の選択 246
12.10.1.2. 有効 246
12.10.1.3. フォーカス 247
12.10.1.4. 押下 247
12.10.1.5. 選択 247
12.10.1.6. 選択グループ 248
12.10.1.7. スピン 248
12.10.1.8. タッチ 249
12.10.1.9. テキストの切り捨て 250
12.10.1.10. 複数行 250
12.10.2. 効果 251
12.10.2.1. 枠 251
12.10.2.2. 配色 251
12.10.3. フォーカス 252
12.10.3.1. ユーザー定義フォーカス 252
12.10.3.2. 自動フォーカス 253
12.10.4. ジェスチャー 253
12.10.4.1. フリックジェスチャー 253
12.10.4.2. ホールドジェスチャー 254
12.10.4.3. ロングホールドジェスチャー 254
12.10.4.4. パスジェスチャー 255
12.10.4.4.1. ジェスチャーID 256
12.10.4.5. ピンチジェスチャー 257
12.10.4.6. 回転ジェスチャー 257
12.10.5. 入力処理 258
12.10.5.1. 内部で移動 258
12.10.5.2. 外部へ移動 258
12.10.5.3. 内部へ移動 258
12.10.5.4. タッチ押下 259
12.10.5.5. タッチリリース 259
12.10.5.6. タッチの喪失 259

12.10.5.7. タッチのステータス変更 260
12.10.5.8. タッチ移動 260
12.10.5.9. ジェスチャー 261
12.10.5.10. キー押下 261
12.10.5.11. キーUnicode 261
12.10.5.12. キーリリース 262
12.10.5.13. キーのステータス変更 262
12.10.5.14. 回転 262
12.10.5.15. 可動 263
12.10.6. レイアウト 263
12.10.6.1. 絶対レイアウト 263
12.10.6.2. ボックスレイアウト 263
12.10.6.3. フローレイアウト 264
12.10.6.4. グリッドレイアウト 264
12.10.6.5. レイアウト余白 265
12.10.6.6. リストレイアウト 265
12.10.6.7. 拡大縮小モード 266
12.10.7. リスト管理 266
12.10.7.1. ラインインデックス 266
12.10.7.2. リストインデックス 267
12.10.7.3. テンプレートインデックス 267
12.10.7.4. ポートの表示 267
12.10.8. 3D 268
12.10.8.1. カメラビューポート 268
12.10.8.2. アンビエントテクスチャ 268
12.10.8.3. ディフューズテクスチャ 268
12.10.8.4. エミッシブテクスチャ 269
12.10.8.5. ライトマップテクスチャ 269
12.10.8.6. ノーマルマップテクスチャ 270
12.10.8.7. 不透明テクスチャ 270
12.10.8.8. リフレクションテクスチャ 271
12.10.8.9. スペキュラテクスチャ 271
12.10.9. 変形 272
12.10.9.1. ピボット 272
12.10.9.2. 回転 273
12.10.9.3. 拡大縮小 273
12.10.9.4. せん断 273
12.10.9.5. 変換 274
13. インストール 275
13.1. バックグラウンド情報 275
13.1.1. 制限 275
13.1.2. システム要件 275

13.2. EB Commandからのダウンロード 276
13.3. EB GUIDEのインストール 277
13.4. EB GUIDEのアンインストール 278
用語集 279
インデックス 283

1. 本書について

1.1. 対象者: モデラー

モデラーは、ヒューマンマシンインターフェイス(HMI)を作るためにEB GUIDE Studioを使用します。EB GUIDEでは、HMIはEB GUIDEモデルと呼ばれます。アプリケーションと通信するには、イベントメカニズムを使ってイベントを確認する方法、データプールを使ってデータプールアイテムを利用する方法、ユーザー固有のEB GUIDEスクリプト関数を呼び出す方法のいずれかを使用します。

モデラーは、以下のタスクを担います。

- ウィジェットとビューのアーキテクチャを使用して、ディスプレイのグラフィカル要素を指定します。
- ヒューマンマシンインターフェイスの最適化について、デザイナーおよびユーザビリティの専門家と打ち合わせをします。
- グラフィカル要素をいつ表示するかをステートマシン機能を使って指定します。
- コントロールパネルやタッチスクリーンなどの装置から入力を受け取った要素がどのように反応するかを定義します。
- これらの要素が、情報をハードウェア、またはソフトウェアアプリケーション(ナビゲーションユニットのようなサービスを提供するもの)から受け取る方法を定義します。
- モデル要素間や、入力デバイス、出力デバイスとのインターフェイスを定義します。

モデラーには以下に関する深い知識があります。

- EB GUIDE Studio機能
- UMLステートマシンの概念
- ドメインの仕様と要件
- やり取りされるデータとEB GUIDE GTF通信メカニズム
- プロジェクトで3Dグラフィックを使用している場合、3Dグラフィックの仕様

1.2. ユーザーマニュアルの構成

次の構成で情報が記されています。

- バックグラウンド情報

  バックグラウンド情報では、具体的なトピックや重要な事実を紹介しています。こうした情報により、関連する手順を実行できます。

- 手引書

  この手引書では、具体的なタスクを段階的に説明し、EB GUIDEの使用方法を示します。手順は、タイトルで体言止め**(英語では動詞のing形)**になっているものです(例えば、「**EB GUIDE Studioの起動**」)。

- チュートリアル

  チュートリアルは手引書を拡張したものです。複雑なタスクの説明が記されています。チュートリアルの見出しは「**チュートリアル:**」で始まります(例えば、「**チュートリアル: ボタンの作成**」)。

- リファレンス

  リファレンスは、詳細な技術的パラメータおよび表、それにEB GUIDE Monitorアプリケーションプログラミングインターフェイスに関するドキュメントを提供します。

- デモ

  デモは、アプリケーションの記述方法や対話処理の流れを理解するのに役立ちます。デモはEB GUIDE GTF SDKに含まれています。

1.3. 本書で使用する表記スタイル

このドキュメントでは、一部の単語や語句を太字または斜体のフォント、または等幅フォントで表記しています。これらの表記の意味については、以下の表をご覧ください。デフォルトのテキストは、スタイル設定のないArial Regularフォントで表記します。

規則	目的	使用例
斜体	強調する	プロジェクトのリリースバージョンが混在する場合は、すべてのファイルの内容種別が使用可能です。このようなケースを**混在バージョン**と呼びます。
太字	メニュー、サブメニューを示す	[オプション]メニューを選択します。
太字	ボタンを示す	[OK]を選択します。
太字	キーボード上のキーを示す	F2キーを押します。
太字	キーボード上のキーの組み合わせを示す	Ctrl+Alt+Deleteを押します
太字	コマンドを示す	**legacy convert** コマンドを使って、`.xdm`ファイルを新しいバージョン用に変換します。
等幅フォント(Courier)	ファイル名やディレクトリ名、および章のタイトルを示す	スクリプトの配置先:`function_name\abcdirectory`。
等幅フォント(Courier)	コードを示す	`CC_FILES_TO_BUILD =(PROJECT_-PATH)\source\network\can_node.-`

規則	目的	使用例
		`c CC_FILES_TO_BUILD += $(PROJECT_-PATH)\source\network\can_config.c`
等幅フォント (Courier)	関数名、メソッド名、ルーチン名を示す	cos関数は、配列要素の余弦を求めます。構文の 使用例:`MLGetVar ML_var_name`
等幅フォント (Courier)	ユーザーからの入力や変数テキストを示す	メニュー行に`three-digit prefix`と入力してください。
角括弧[ ]	オプションパラメータ、またはオプションのパラメータがあるコマンド構文を示す	`insertBefore [<opt>]`
波括弧{}	必須パラメータ、または必須のパラメータがあるコマンド構文を示す	`insertBefore {<file>}`
三点リーダー	パラメータが続くこと、またはコマンド構文を示す	`insertBefore [<opt>…]`
警告	死亡または重傷の危険性を警告する	警告  これは警告ーを示しています。 警告の文面はこのように表記します。
注意	軽傷または物品破損の危険性を警告する	注意  これは注意ーを示しています。 注意の文面はこのように表記します。
メモ	補足の説明をする	注記  これはメモーを示しています。 メモの文面はこのように表記します。
ヒント	役に立つヒントやコツを提供する	ティップ  これはヒントーを示しています。 ヒントの文面はこのように表記します。
使用例	例を提示する	 **例1.1** これは例ーを示しています 例はこのように表記します。

これは操作方法一を示しています。

足あとの付いたバーの下には操作方法を段階的に説明する内容が続きます。

前提条件:

- この行には操作方法の前提条件を示します。

ステップ 1
操作方法の説明が入ります。

ステップ 2
操作方法の説明が入ります。

ステップ 3
操作方法の説明が入ります。

1.4. ネーミングルール

EB GUIDEドキュメンテーションでは、以下のディレクトリ名を使用します。

- EB GUIDEのインストール先ディレクトリは、`$GUIDE_INSTALL_PATH`と表されます。

  例えば、次のようになります。

  ```
  C:\Program Files\Elektrobit\EB GUIDE Studio 6.2
  ```

- EB GUIDE SDKプラットフォームのディレクトリは、`$GTF_INSTALL_PATH`と表されます。名前のパターンは、`$GTF_INSTALL_PATH\platform\<platform name>`となります。

  例えば、次のようになります。

  ```
  C:\Program Files\Elektrobit\EB GUIDE Studio 6.4\platform\win32
  ```

- EB GUIDEプロジェクトの保存先ディレクトリは、`$GUIDE_PROJECT_PATH`と表されます。

  例えば、次のようになります。

  ```
  C:\Users\[user name]\Documents\EB GUIDE 6.4\projects\
  ```

2. 安全で正しい使い方

2.1. 使用目的

- EB GUIDE StudioおよびEB GUIDE GTFは、インフォテイメントヘッドユニット、計器パネル、特定の工業アプリケーション向けにユーザーインターフェイスを開発するプロジェクトで使うことを意図しています。
- ライセンスでカバーされる範囲によりますが、主なユースケースは、大量生産、受注生産、プロトタイピングでの使用です。

2.2. 考えられる誤用

警告

誤用の可能性と賠償責任

お客様は、このソフトウェアを、意図された用途に従い、 適用可能なライセンス使用許諾契約書で許可されたとおりに常に使用するものとします。Elektrobit Automotive GmbHは、いっさいの賠償責任を負わないものとし、このソフトウェアが 適用可能なライセンス使用許諾契約書に従わずに使用された場合も、それについて責任を保持しません。

- EBから提供されるEB GUIDE product lineは、ISO 26262/A-SILに規定された安全関連システムにヒューマンマシンインターフェイスを実装する目的には使用しないでください。
- EB GUIDE product lineは、DO-178B、SIL、A-SILなどの認定が必要とされる安全関連システムでの使用を意図していません。

このような環境でEB GUIDE GTFを使用することを禁じます。具体的なアプリケーションに対する適合性が不確かな場合は、第3章「サポート」の説明に従ってEBまでお問い合わせください。

3. サポート

EB GUIDEサポートは次のような形で利用できます。

- コミュニティエディションの場合:

  当社の記事、ブログ、およびフォーラム内の包括的な情報を検索します。

- エンタープライズエディションの場合:

  サポート上の合意事項に従って当社までお問い合わせください。

サポートが必要な場合は、EB GUIDEインストールのバージョン番号をお手元にご用意ください。 バージョン番号を確認するには、プロジェクトセンターに移動し、[ヘルプ]をクリックします。 ダイアログの右下隅にバージョン番号が記されています。

4. EB GUIDEの基本

EB GUIDEは、ヒューマンマシンインターフェイス(HMI)の開発プロセスに携わるユーザーをさまざまな局面で支援します。EB GUIDE商品ラインは、グラフィカルユーザーインターフェイスまたは音声ユーザーインターフェイス向けのツールとプラットフォームを提供しています。EB GUIDE商品ラインは、インフォテイメントヘッドユニット、計器パネル、特定の工業アプリケーション向けにユーザーインターフェイスを開発するプロジェクトで使うことを意図しています。主に、大量生産、受注生産、プロトタイピングで使用されます。

4.1. EB GUIDE product line

EB GUIDE product lineは、以下のソフトウェア要素から構成されます。

- EB GUIDE Studio
- EB GUIDE TF

EB GUIDE Studioは、PCで動作するモデリングツールです。EB GUIDE Studioを使うと、機能へのアクセスをユーザーに提供する中心的な制御要素としてヒューマンマシンインターフェイス機能全体をモデリングできます。

EB GUIDE TFは、EB GUIDE Studioで作成されたEB GUIDEモデルを実行します。EB GUIDE TFは、開発用PCやその他の組み込みプラットフォームで使用できます。

EB GUIDE Studioで作成されたEB GUIDEモデルと、EB GUIDE TFで実行されるエクスポートされたEB GUIDEモデルは完全に分離されます。相互に対話は行われますが、相手側の動作をブロックすることはできません。

4.2. EB GUIDE Studio

4.2.1. ヒューマンマシンインターフェイスの動作のモデル化

EB GUIDEモデルの動的な動作は、ステートマシンにステートを配置し、複数のステートを組み合わせることで指定します。

ステートマシン
: ステートマシンは、決定性有限オートマトンであり、システムの動的な動作を表します。EB GUIDE Studioでは、ハプティックステートマシンなど、複数のタイプのステートマシンを使用できます。ハプティックステートマシンを使うと、グラフィカルユーザーインターフェイスを指定できます。

ステート
: ステートは遷移でリンクされます。遷移は、ステート間の接続であり、ステートの変化をトリガーします。

4.2.2. ヒューマンマシンインターフェイスの外観のモデル化

EB GUIDE Studioで、EB GUIDEモデルのグラフィカルユーザーインターフェースと音声ユーザーインターフェースを定義します。

ウィジェット
: グラフィカルユーザーインターフェイスを作成するために、EB GUIDE Studioはウィジェットを提供しています。ウィジェットは、見え方を定義するモデル要素です。テキストラベルやイメージなどの情報を表示するのに主に使用します。また、ボタンやスライダーなど、システムの動作を操作する手段も提供できます。複数のウィジェットを組み合わせて、ビューと呼ばれる構造を作成します。

スピジェット
: 音声ユーザーインターフェイスを作成するために、EB GUIDE Studioはスピジェットを提供しています。スピジェットを使用して、音声ダイアログの基本要素を指定します。ユーザー入力としてのスピーチ認識とシステム出力としてのスピーチ合成です。プロンプトスピジェットを使って、Text–to-Speechシンセサイザ(TTS)で再生されるテキストをモデリングできます。コマンドスピジェットは、スピーチ認識機能が理解できる文法のモデリングに使用します。関連性のあるスピージェットは、モデル要素全体でグループ化されています。このグループをトークと呼びます。

4.2.3. データの処理

ヒューマンマシンインターフェイスとアプリケーションが交わす通信は、データプールとイベントシステムを使って実装されています。

データプール
: データプールは、すべての表示データとその他の内部情報を格納する組み込みのデータベースです。データプールアイテムはデータを格納し、やり取りします。

イベントシステム
: イベントは一時的なトリガーです。イベントは双方に送信され、特定の事態が発生したことを通知します。

アプリケーションソフトウェアは、アプリケーションプログラミングインターフェイスを通じてイベントやデータプールにアクセスできます。

4.2.4. EB GUIDEモデルのシミュレート

EB GUIDE Studioでは、シミュレーション中にEB GUIDEモデルの機能をテストできます。マウスでクリックするだけでシミュレーションを開始し、EB GUIDEモデルがどのように見えるかすぐに確認できます。

シミュレーションは、マウス、キーボード、タッチスクリーンなどの入力デバイスを使用して操作することができます。

4.2.5. EB GUIDEモデルのエクスポート

EB GUIDEモデルをターゲットデバイスで使うには、EB GUIDEモデルをEB GUIDE Studioからエクスポートし、ターゲットデバイスで認識できる形式に変換する必要があります。エクスポートの過程で、すべての関連データが一連のASCIIファイルとしてエクスポートされます。

4.3. EB GUIDE TF

EB GUIDE TFは、EB GUIDEモデルの実行に必要な`GtfStartup`実行可能ファイルとライブラリで構成されています。

EB GUIDE Studioで選択したプロジェクトタイプに応じて、以下を実行します。

- EB GUIDE GTF

  EB GUIDE Graphics Trarget Frameworkは、グラフィカルHMIモデルを実行するランタイム環境です。
- EB GUIDE STF

  EB GUIDE Speech Target Frameworkは、HMIに含まれるスピーチ機能を実行するランタイム環境です。

EB GUIDE TFのプログラムコードは、大半がプラットフォームに依存しません。別のシステムへのコードの移植は、非常に簡単です。

EB GUIDEモデルファイルを入れ替えるだけで、ヒューマンマシンインターフェイス全体を入れ替えることが可能です。EB GUIDE TFの再コンパイルは不要です。変更したEB GUIDEモデルをEB GUIDE Studioから再エクスポートするだけで、必要な作業は完了です。

EB GUIDE TFでは、以下のプラットフォーム抽象化を使用します。

- OSの抽象化

  オペレーティングシステム(OS)のプラットフォーム依存関係は、OSアブストラクションレイヤー(`GtfOSAL`)にカプセル化されています。EB GUIDE TFで利用されるオペレーティングシステム機能には、ファイルシステムやTCPソケットなどがあります。
- GLの抽象化

  グラフィックサブシステムのプラットフォーム依存関係は、レンダラーにカプセル化されています。EB GUIDEモデルには、ジオメトリやライティングなどの情報を持つ要素プロパティが含まれます。エクスポートされたEB GUIDEモデルに含まれるデータは、レンダラーに渡され、処理されてデジタルイメージに出力されます。レンダラーは、ハードウェアで提供される実際のグラフィカルシステムを抽象化したものです。EB GUIDE TFは、異なるプラットフォームに対応するさまざまなレンダラーをサポートしています。
- オーディオの抽象化

スピーチユーザーインターフェイスは、オーディオハードウェアにアクセスする必要があります。オーディオの抽象化によって、マイクとスピーカーへのアクセスが可能になります。EB GUIDE STFには、スピーチ認識とText-to-Speech音声合成機能が実装されています。この目的で、EB GUIDE STFにはサーパーティのスピーチエンジンが組み込まれています。

5. チュートリアル: はじめに

このセクションでは、EB GUIDE Studioによるヒューマンマシンインターフェイスのモデル化について、概要を簡単に紹介します。EB GUIDE Studioの起動方法、プロジェクトの作成方法、EB GUIDEモデルの動作および外観のモデル化の方法、EB GUIDEモデルの実行方法を説明しています。

5.1. EB GUIDEの起動

EB GUIDEの起動

Prerequisite:

- EB GUIDEのインストールが完了していること。

Step 1
Windowsの[スタート]メニューで、[すべてのプログラム]をクリックします。

Step 2
[Elektrobit]メニューで、起動するバージョンをクリックします。

EB GUIDE Studioが起動します。プロジェクトセンターが表示されます。

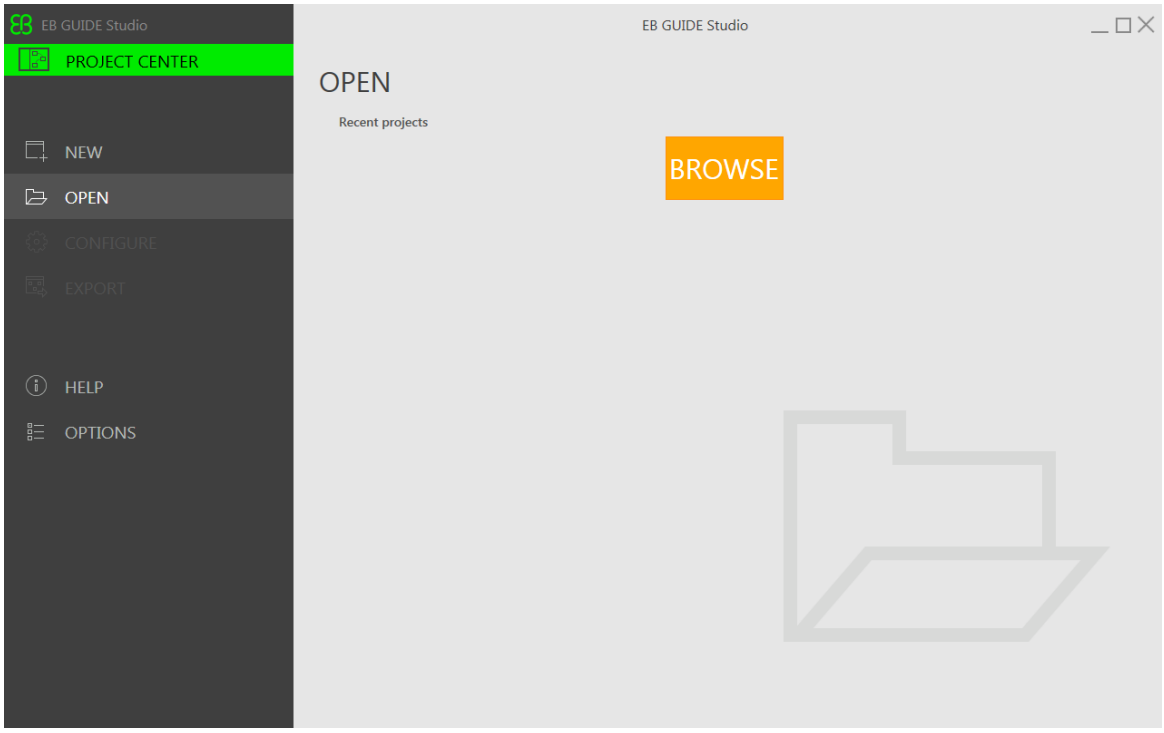

Figure 5.1. プロジェクトセンター

5.2. プロジェクトの作成

プロジェクトの作成

Prerequisite:

- EB GUIDE Studioが起動されていること。
- `C:\temp`のディレクトリが作成されていること。

Step 1
ナビゲーションエリアで、[新規]をクリックします。

Step 2
コンテンツエリアで、`C:\temp`ディレクトリを選択します。

Step 3
`MyProject`というプロジェクト名を入力します。

Step 4
[作成]をクリックします。

プロジェクトが作成されます。プロジェクトエディターが開き、空プロジェクトが表示されます。

[メイン]ステートマシンがデフォルトで追加され、コンテンツエリアに表示されます。

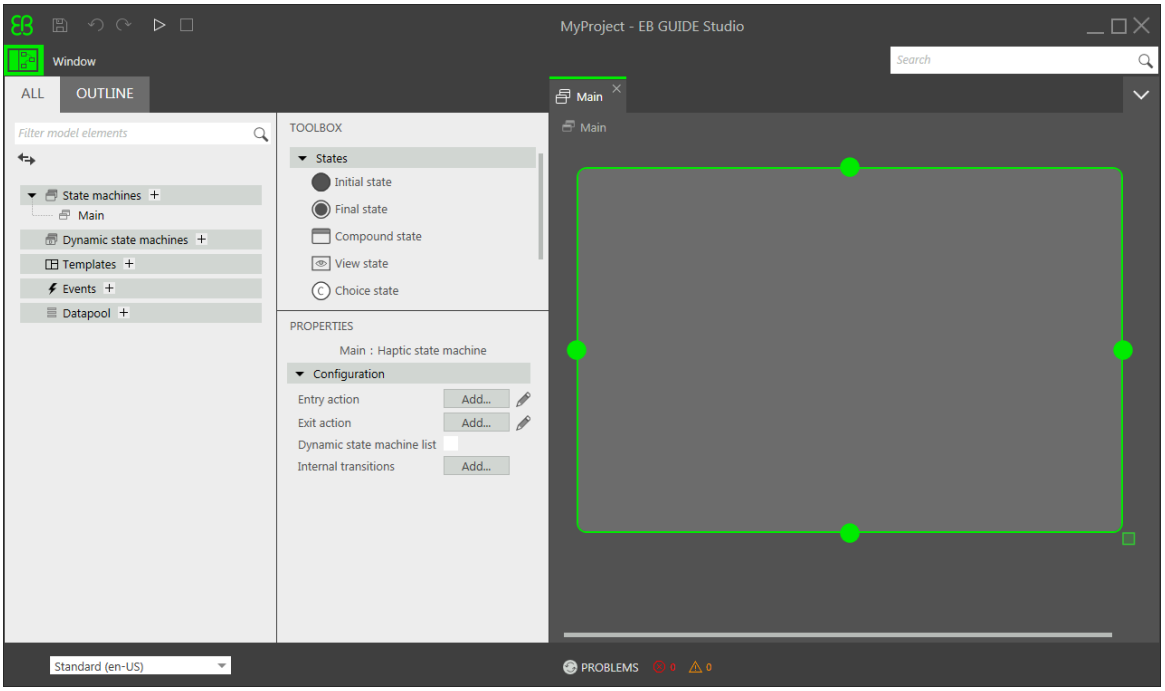

Figure 5.2. [メイン]ステートマシンが表示されたプロジェクトエディター

5.3. ヒューマンマシンインターフェイスの動作のモデル化

EB GUIDEモデルの動作は、ステートマシンによって定義されます。EB GUIDEでは、UMLに似た構文を使用してこれを行います。

このセクションでは、定義されたビューを起動時に表示し、ボタンを押すと別のビューに変わるステートマシンをモデル化する方法を説明します。

ステートマシンへのステートの追加

EB GUIDEには、さまざまなステートが用意されています。このセクションでは、3種類のステートを示します。初期ステートは、ステートマシンのスタート地点となります。ビューステートは、デフォルトのビューを表示します。また、ステートマシンの最終ステートは、ステートマシンの終了処理を行います。

Prerequisite:

- プロジェクト`MyProject`が作成されていること。
- コンテンツエリアに、[メイン]ステートマシンが表示されていること。

Step 1
[ツールボックス]からビューステートをドラッグし、ステートマシンにドロップします。

[ビューステート1]と共に、ビューがEB GUIDEモデルに追加されます。

Step 2
ステップ1を繰り返します。

[ビューステート2]が追加されていること。

Step 3
[ツールボックス]から初期ステートをドラッグし、ステートマシンにドロップします。

Step 4
[ツールボックス]から最終ステートをドラッグし、ステートマシンにドロップします。

[メイン]ステートマシンに追加した4つのステートは、コンテンツエリアとナビゲーションエリアの両方に、それぞれステートチャートと階層的ツリービューとして表示されます。

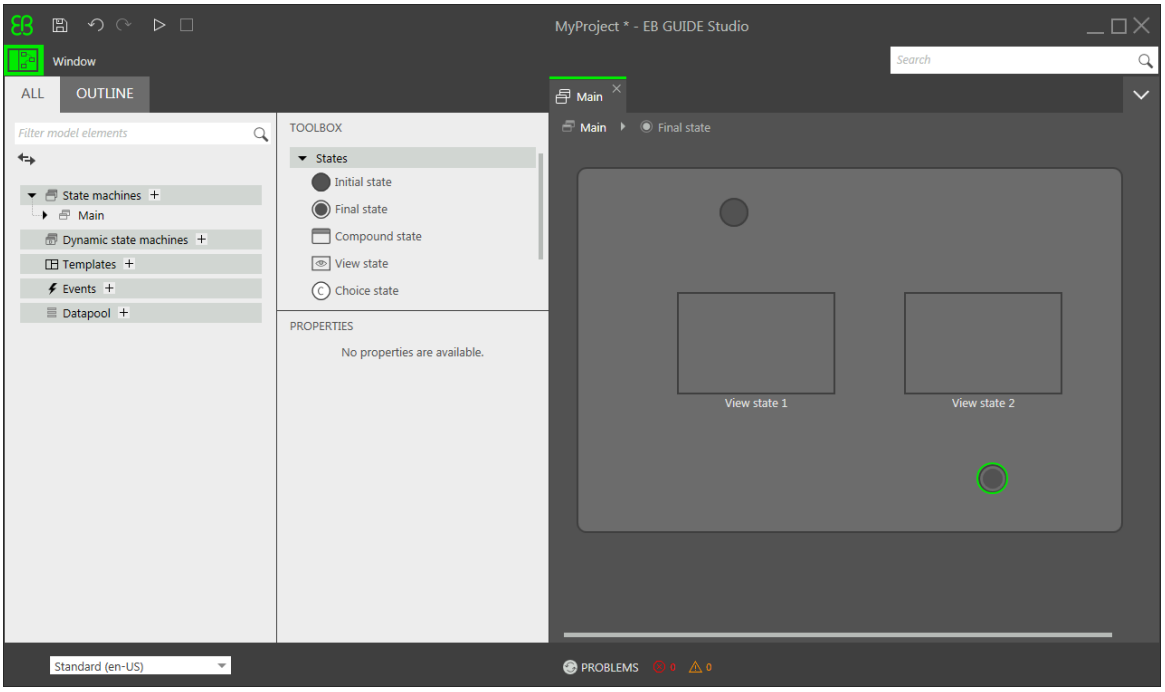

Figure 5.3. ステートが表示されたプロジェクトエディター

遷移の追加

遷移は、ステート間の接続であり、ステートの変化をトリガーします。各種の遷移タイプがあります。このセクションでは、デフォルト遷移とイベントトリガー遷移を扱います。

Prerequisite:

- コンテンツエリアに、[メイン]ステートマシンが表示されていること。

- [メイン]ステートマシンに、初期ステート、2つのビューステート、および最終ステートが含まれていること。

Step 1
初期ステートを遷移のソースステートとして選択します。

Step 2
緑色のドラッグ点をクリックし、マウスボタンを押したままにします。

Step 3
ターゲットステートである[ビューステート1]までマウスをドラッグします。

Step 4
ターゲットステートが緑色で強調表示されたら、マウスボタンを離します。

遷移が作成され、緑色の矢印として表示されます。

Step 5
[ビューステート1]と[ビューステート2]の間に遷移を追加します。

[ビューステート1]を選択し、ステップ2～4を繰り返します。

Step 6
[ビューステート1]と[ビューステート2]の間の遷移を選択します。

次のステップでは、遷移をイベントに関連付けます。

Step 7
[トリガー]コンボボックスに`Event 1`と入力し、[イベントの追加]をクリックします。f

[イベント1]というイベントが作成され、遷移トリガーとして追加されます。[イベント1]が発火したときには、必ずこの遷移が実行されます。

Step 8
[ビューステート2]と最終ステートの間に遷移を追加します。

[ビューステート2]を選択し、ステップ2～4を繰り返します。

新しく、[イベント2]をトリガーとして追加します。

この時点で、ステートマシンは次の図のようになっています。

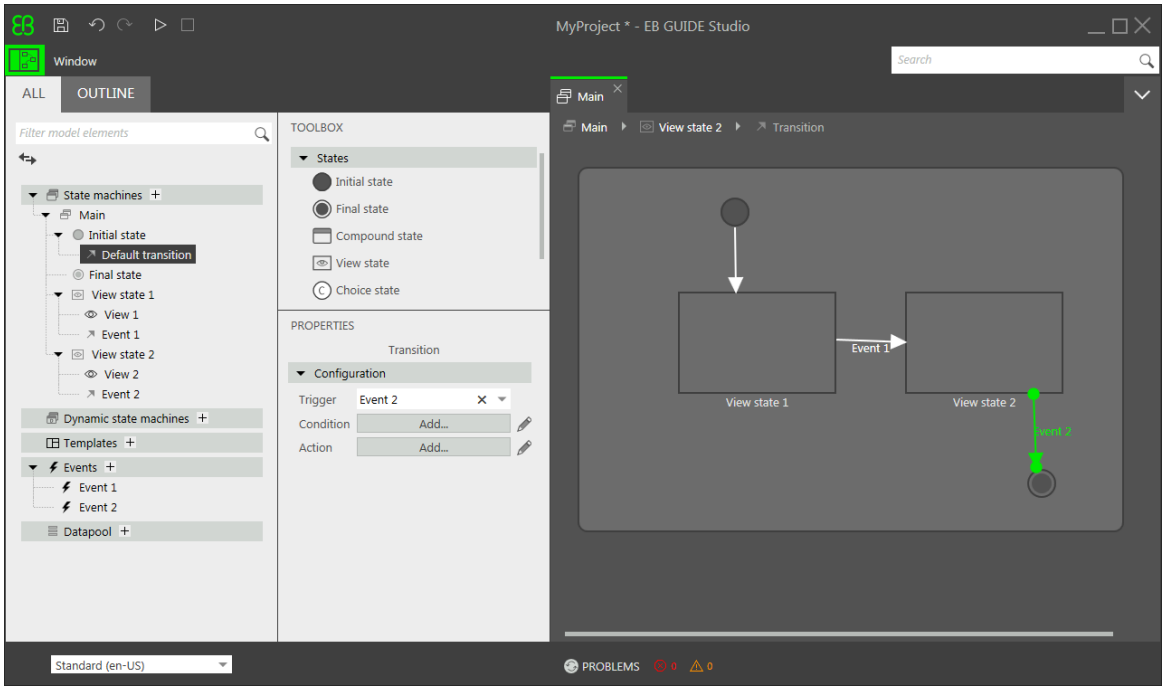

Figure 5.4. イベントを伴う遷移でリンクされたステート

これで、基本的なステートマシンの動作が定義されました。

5.4. ヒューマンマシンインターフェイスの外観のモデル化

上記のセクションで作成したステートマシンには、2つのビューステートが含まれています。このセクションでは、ビューをモデル化する方法を説明します。

ビューを開く

Prerequisite:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- [ビューステート1]が追加されていること。

Step 1
[ビューステート1]をダブルクリックします。

コンテンツエリアに、[ビュー1]が表示されます。

ビューへのボタンの追加

EB GUIDE Studioには、ビューの外観をモデル化するさまざまなオプションがあります。

1つの例として、このセクションでは四角形をビューに追加する方法を示します。この四角形は、ユーザーの入力に反応し、ボタンとして機能します。

Prerequisite:

- コンテンツエリアに[ビューステート1]が表示されていること。

Step 1
[ツールボックス]から四角形をドラッグし、ビューにドロップします。

Step 2
[プロパティ]パネルで[ウィジェット機能プロパティ]カテゴリに移動し、[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

Step 3
[使用可能なウィジェット機能]から[入力処理]カテゴリを展開し、[タッチリリース]を選択します。

[承認]をクリックします。

関連するウィジェット機能のプロパティが[プロパティ]パネルに追加されます。

Step 4
[プロパティ]パネルで、`touchPolicy`ドロップダウンリストボックスから`Press then react`を選択します。

この四角形はタッチ入力に反応します。

Step 5
`touchShortReleased`プロパティに移動し、[編集]をクリックします。

Step 6
次のEB GUIDEスクリプトを入力します。

```
function(v:touchId::int, v:x::int, v:y::int, v:fingerId::int)
    {
    fire_delayed 500, ev:"Event 1"()
    true
    }
```

四角形にタッチすると、500ミリ秒後に[イベント1]が発火します。

Step 7
[承認]をクリックします。

Step 8
[プロパティ]パネルで、`fillColor`プロパティとして赤を選択します。

Step 9
ナビゲーションエリアで、[ビュー2]をダブルクリックします。

コンテンツエリアに、[ビュー2]が表示されます。

Step 10
ステップ1～5を繰り返します。

Step 11
次のEB GUIDEスクリプトを入力します。

```
function(v:touchId::int, v:x::int, v:y::int, v:fingerId::int)
    {
    fire_delayed 500, ev:"Event 2"()
    true
    }
```

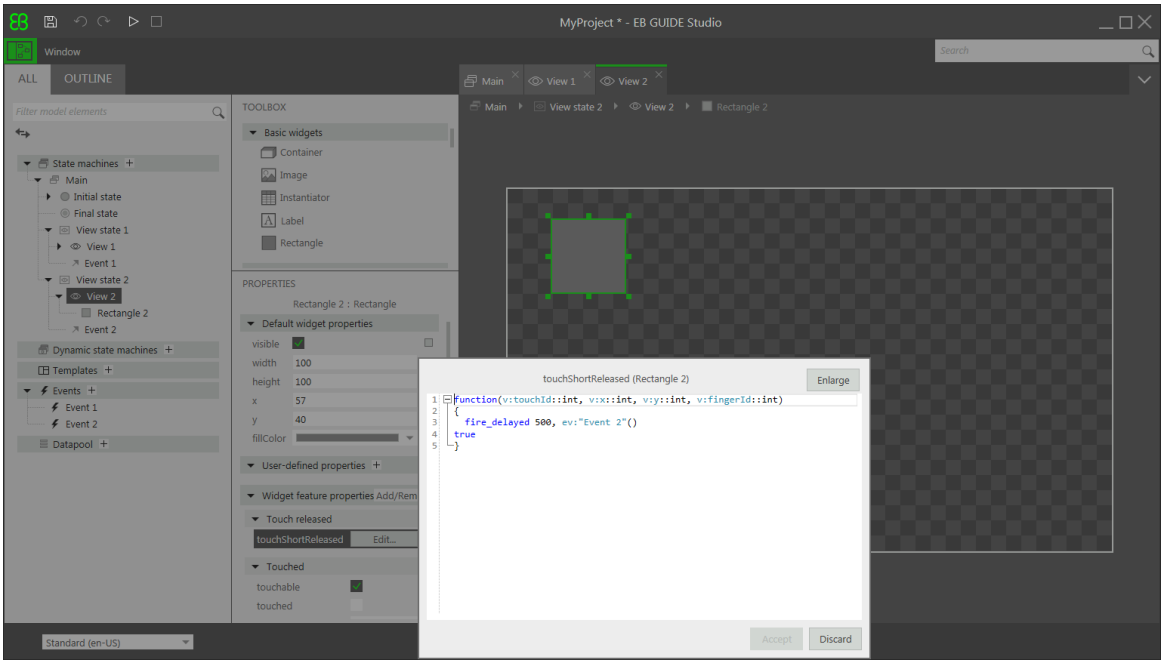

Figure 5.5. EB GUIDEスクリプトを持つウィジェットプロパティ

Step 12
[承認]をクリックします。

四角形にタッチすると、500ミリ秒後に[イベント2]が発火します。

Step 13
[プロパティ]パネルで、`fillColor`プロパティとして青を選択します。

5.5. シミュレーションの開始

EB GUIDEでは、対象デバイスにエクスポートする前にモデルをPC上で実行できます。

シミュレーションの開始

Step 1
プロジェクトを保存するには、コマンドエリアで をクリックします。

Step 2
コマンドエリアで▷をクリックします。

EB GUIDEモデルが開始され、モデル化した動作と外観が表示されます。

最初に、[ビュー1]が表示されます。赤い四角形をクリックすると、画面が[ビュー2]に変化します。これは、クリックによって[イベント1]が発火し、[イベント1]によって[ビューステート1]から[ビューステート2]への遷移が実行されるためです。

次に、[ビュー2]が表示されます。[ビュー2]で青い四角形をクリックすると、ステートマシンの終了処理が行われます。これは、クリックによって[イベント2]が発火し、[イベント2]によって[ビューステート2]から最終ステートへの遷移が実行されるためです。シミュレーションのウィンドウは開いたままです。シミュレーションを停止するには、□をクリックします。

6. バックグラウンド情報

この章では、トピックスをアルファベット順で並べています。

6.1. 3Dグラフィック

EB GUIDE Studioでは、EB GUIDEプロジェクトに3Dグラフィックを使用できます。

6.1.1. サポートされている3Dグラフィック形式

OpenGL ES 2.0およびDirectX 11レンダラーのみが3Dグラフィックを表示できます。サポートされている3Dグラフィック形式は、COLLADA (`.dae`)およびFilmbox (`.fbx`)です。失敗を防ぐため、Filmbox形式を使用します。

6.1.2. 3Dグラフィックファイルの設定

3DオブジェクトをEB GUIDE Studioのビューに表示するには、次のオプションで3Dグラフィックファイルを作成する必要があります。

- 透視図カメラ
- 1つ以上のオブジェクト
- 1つ以上の光源

3Dグラフィックファイルを作成するには、サードパーティの3Dモデリングソフトウェアを使用します。

3Dグラフィックファイルは、以下に示す多種多様な追加コンテンツをサポートしています。

- 位置、法線、従法線、接線、および1つのテクスチャチャネルを含む3Dオブジェクト
- 指向性光源
- 定数、線形、二次、および三次減衰を含む点光源
- 円錐角、定数、線形、二次、および三次減衰を含むスポット光源
- ビュー、ニアプレーン、およびファープレーンのフィールドに対応する透視図カメラのサポート
- テクスチャ: エミッシブ、ディフューズ、スペキュラ、ノーマルマップ、オパシティ、リフレクションキューブ、およびライトマップ

ティップ

3Dグラフィックファイルのセットアップ
オパシティマップには有効なアルファチャネルが必要であることに注意してください。

6.1.3. 3Dグラフィックファイルのインポート

3Dグラフィックをビューに追加するには、シーングラフを使用して3Dグラフィックファイルをインポートする必要があります。インポート中、EB GUIDE Studioは3Dグラフィックファイルをシーングラフを親ノードに持つウィジェットツリーに変換します。例えばカメラ、材質、メッシュなどの3Dグラフィックファイルのコンテンツに対し、EB GUIDE Studioはそれぞれウィジェットを作成します。

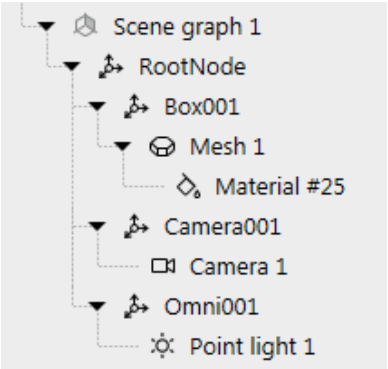

図6.1 ナビゲーションエリアに表示されるシーングラフの例

注記

メッシュごとに1つの材質のみ指定可能
EB GUIDE Studioでは、メッシュごとに1つの材質のみ指定できます。3Dグラフィックでメッシュ1つにつき材質が複数指定されている場合、EB GUIDE Studioは材質ごとにメッシュを追加で作成します。

3Dグラフィックファイルをインポートすると、ディレクトリ`$GUIDE_PROJECT_PATH/<project name>/resources`にサブディレクトリが作成されます。サブディレクトリには、インポートされた`.fbx`ファイルの名前が付けられます。サブディレクトリの名前に作成した日時を追加することもできます。

例6.1
インポートディレクトリの命名

3Dグラフィックファイルは、`car.fbx`と呼ばれます。EB GUIDE Studioに3Dグラフィックファイルをインポートすると、`$GUIDE_PROJECT_PATH/<project name>/resources`に`car_20160102_103029`という名前のサブディレクトリが作成されます。

サブディレクトリには、以下の項目が収められます。

- `.ebmesh`ファイルとなったメッシュ
- `.png`または`.jpg`ファイルとなったテクスチャ

3Dグラフィックに追加でテクスチャを使用するには、`$GUIDE_PROJECT_PATH/<project name>/resources`にテクスチャをコピーします。テクスチャには`.png`または`.jpg`イメージを使用してください。

複数の3Dグラフィックを1つのシーングラフにインポートすることもできます。

3Dウィジェットはインポート後に追加、変更、削除することができます。

詳細については、6.17「ウィジェット」、12.9.4「3Dウィジェット」および12.10.8「3D」をご覧ください。

手順については、8.1.5「ビューへのシーングラフの追加」および11.7「チュートリアル: 3Dグラフィックの操作」をご覧ください。

6.2. アニメーション

アニメーションを使用すると、EB GUIDEモデルに動きと視覚効果を付けることができます。EB GUIDEでは、さまざまな使用例でアニメーションを使用できます。ビュー内のウィジェットをアニメーション化することや、あるビューから別のビューへの遷移をアニメーション化することができます。

6.2.1. ウィジェットのアニメーション

ウィジェットのアニメーション化とは、ウィジェットをビュー上で移動させることです。この移動は曲線で定義されます。このため、[ツールボックス]の[アニメーション]カテゴリには、アニメーションと呼ばれるウィジェットと一連の曲線が含まれています。例えば、定数曲線、リニア補間曲線、正弦曲線があります。曲線は、`target`ウィジェットプロパティを持ち、その`target`プロパティの時間的な変化を記述したものです。

各アニメーションには、1つ以上の曲線が関連付けられています。

とりわけ、ウィジェットにアニメーション化すると次の操作が行えます。

- ビュー内でウィジェットを移動する
- ウィジェットのサイズを変更する
- ウィジェットの色を段階的に変更する

アニメーションは、EB GUIDEスクリプト `f:animation_play`, `f:animation_pause`,`f:animation_cancel`, などの関数によって制御されます。.

ティップ

並列アニメーション

EB GUIDEでは、アニメーションが並列アニメーションになっており、複数の曲線が並行して実行されます。これは次のことを意味します。複数のアニメーションの曲線が同じウィジェットプロパティを対象として使用している場合、それらの曲線は同じタイミングでその`target`プロパティの値を上書きすることになります。

アニメーションと曲線のプロパティについては、12.9.3「アニメーション」をご覧ください。

手順については、8.5.1「ウィジェットのアニメーション化」をご覧ください。

6.2.2. ビュー遷移のアニメーション

ビュー遷移のアニメーション化とは、ビューの開始時または終了時にムーブまたはフェードするアニメーションを定義することです。ビューを変更すると、それらのアニメーションがトリガーされます。

表示遷移アニメーションはビューテンプレートで定義します。ビューテンプレートを再利用するたびに、インスタンスは開始アニメーションと終了アニメーションを継承します。

表示遷移アニメーションにはさまざまなタイプがあります。開始アニメーションは、右からのムーブインや下からのムーブインなどです。終了アニメーションは、上から下へのムーブアウトなどです。

ビューテンプレートのアニメーションのプロパティについては、12.9.1「ビュー」をご覧ください。

手順については、8.5.2「ビュー遷移のアニメーション化」をご覧ください。

6.3. アプリケーションとモデルを結ぶアプリケーションプログラミングインターフェイス

EB GUIDEでは、アプリケーションとEB GUIDE TFとの間の通信データは、アプリケーションプログラミングインターフェイス(API)に抽象化されています。アプリケーションとは、例えばメディアプレーヤーであったりナビゲーションであったりします。

アプリケーションプログラミングインターフェイスは、データプールアイテムとイベントで定義します。イベントはヒューマンマシンインターフェイスとアプリケーションの間を送信されます。

例6.2
アプリケーションプログラミングインターフェイスの内容

- START_TRACK: アプリケーションに送信されるイベント。パラメータ`track`には、再生するトラックの数が設定されます。

- TRACK_STOPPED: 再生されたトラックが終了したときに、アプリケーションからヒューマンマシンインターフェイスに送信されるイベント。
- MEDIA_CURRENT_TRACK: アプリケーションから書き込まれる動的なデータプールアイテム。
- MEDIA_PLAY_SPEED: 再生速度を定義する動的なデータプールアイテム。この値は、ヒューマンマシンインターフェイスを通じてユーザーが設定します。

6.4. 通信コンテキスト

通信コンテキストは、通信が行われる環境を記述します。例えばメディアやナビゲーションアプリケーションなどは通信コンテキストです。これらはヒューマンマシンインターフェイスモデルを使って通信を行います。ある通信コンテキストによる変更は、ライター通信コンテキストで発行され、リーダー通信コンテキストで更新されるまで、他の通信コンテキストには見えません。

通信コンテキストには、識別のため、プロジェクトの設定で固有の名前と数値ID(0～255)が割り振られます。

データプールのアイテムには、通信コンテキストが値を書き込む1つのプロパティと、通信コンテキストが値の変更について通知を受け取り、その変更に対して応答するためのもう1つのプロパティがあります。

手順については、9.8「外部通信の確立」をご覧ください。

6.5. グラフィカルユーザーインターフェイスの要素

EB GUIDE Studioのグラフィカルユーザーインターフェイスは、プロジェクトセンターとプロジェクトエディターという2つの要素に分割されています。プロジェクトセンターでは、EB GUIDEプロジェクトの管理、オプションの設定、対象デバイスにコピーするためのEB GUIDEモデルのエクスポートを行います。プロジェクトエディターでは、ヒューマンマシンインターフェイスの外観と動作をモデリングします。

6.5.1. プロジェクトセンター

プロジェクトセンターは、EB GUIDE Studioの起動後に表示される最初の画面です。プロジェクト関連のすべての機能が、プロジェクトセンターにあります。プロジェクトセンターは、ナビゲーションエリアとコンテンツエリアという2つの要素で構成されています。

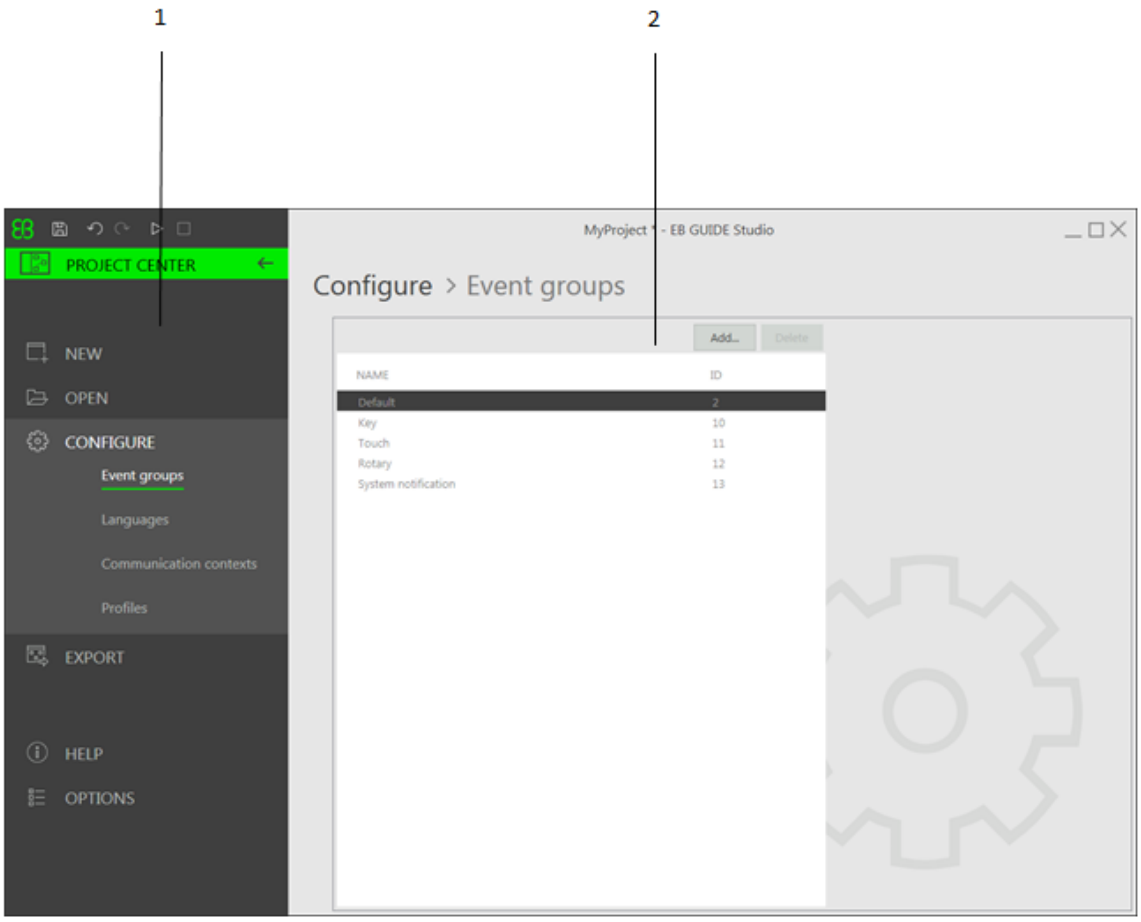

図6.2 ナビゲーションエリア(1)とコンテンツエリア(2)で構成されるプロジェクトセンター

6.5.1.1. ナビゲーションエリア

プロジェクトセンターのナビゲーションエリアは、[設定]や[エクスポート]といった機能タブで構成されます。ナビゲーションエリア内のタブをクリックすると、対応する機能と設定がコンテンツエリアに表示されます。

6.5.1.2. コンテンツエリア

プロジェクトセンターのコンテンツエリアでは、プロジェクト管理と設定を行います。例えば、プロジェクトを保存するディレクトリを選択したり、EB　GUIDEモデルの起動時の動作を定義したりします。コンテンツエリアの外観は、ナビゲーションエリアで選択されているタブによって異なります。

6.5.2. プロジェクトエディター

プロジェクトを作成すると、プロジェクトエディターが表示されます。プロジェクトエディターでは、ヒューマンマシンインターフェイスの動作と外観のモデリングを行います。ステートマシンをモデリングし、ビューを作成し、イベントとデータプールを管理します。プロジェクトエディターは、以下のエリアで構成されています。

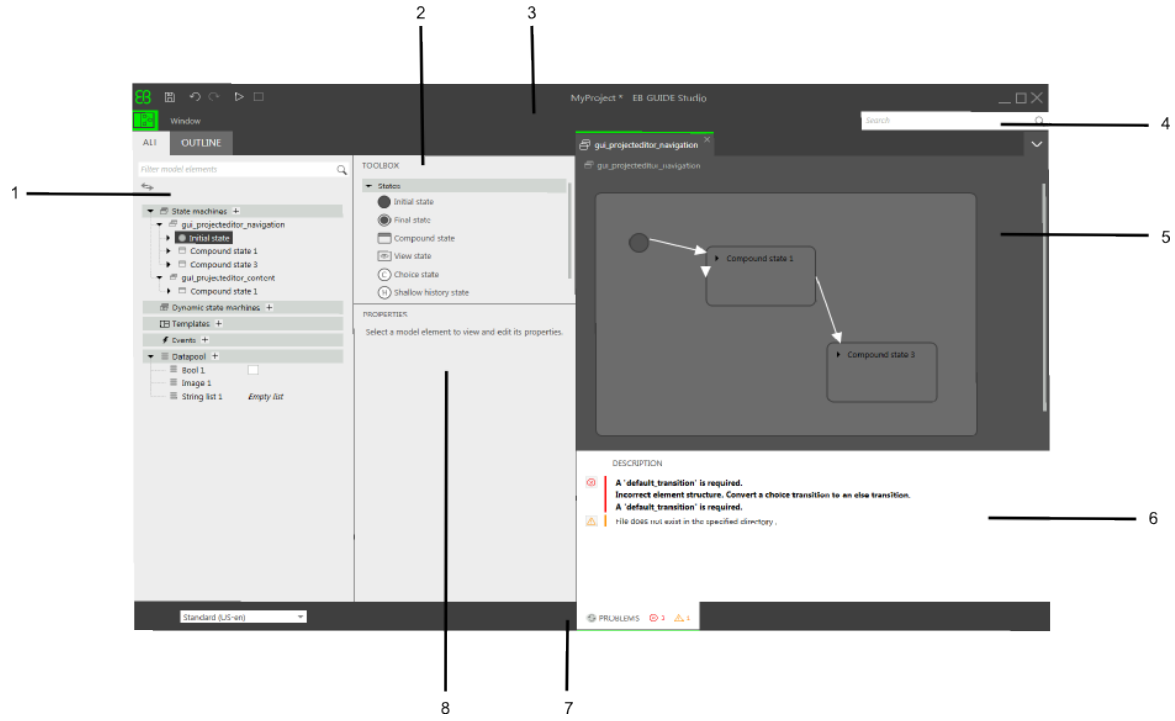

図6.3 プロジェクトエディターとそのエリア

1 ナビゲーションエリア

2 [ツールボックス]

3 コマンドエリア

4 検索ボックス

5 コンテンツエリア

6 [問題]エリア

7 ステータスバー

8 [プロパティ]パネル

6.5.2.1. ナビゲーションエリア

ナビゲーションエリアでは、EB GUIDEモデルのモデル要素が階層構造として表示され、任意の要素に移動することができます。ナビゲーションエリア内でモデル要素をダブルクリックすると、そのモデル要素がコンテンツエリアに表示されます。

ナビゲーションエリアは、[すべて]タブと[概要]タブという2つのタブに分かれています。

- [すべて]タブには、EB GUIDEモデルのすべてのグラフィカル要素と非グラフィカル要素の概要が表示され、ステートマシン階層が反映されます。

  [すべて]タブは、ステートマシン、ビュー、イベント、データプールアイテムなどの要素をEB GUIDEモデルに追加する際にも使用します。
- [概要]タブには、選択されたビューツリー要素とそのサブ要素の構造が表示されます。

ナビゲーションエリアの最上部には、ナビゲーションエリア内のモデル要素を検索するためのフィルタボックスがあります。

ナビゲーションエリア内のモデル要素をクリックしてF3キーを押すと、参照検索が始まります。検索結果ウィンドウが開き、選択したモデル要素のEB GUIDEモデル内での出現箇所がすべてリストされます。

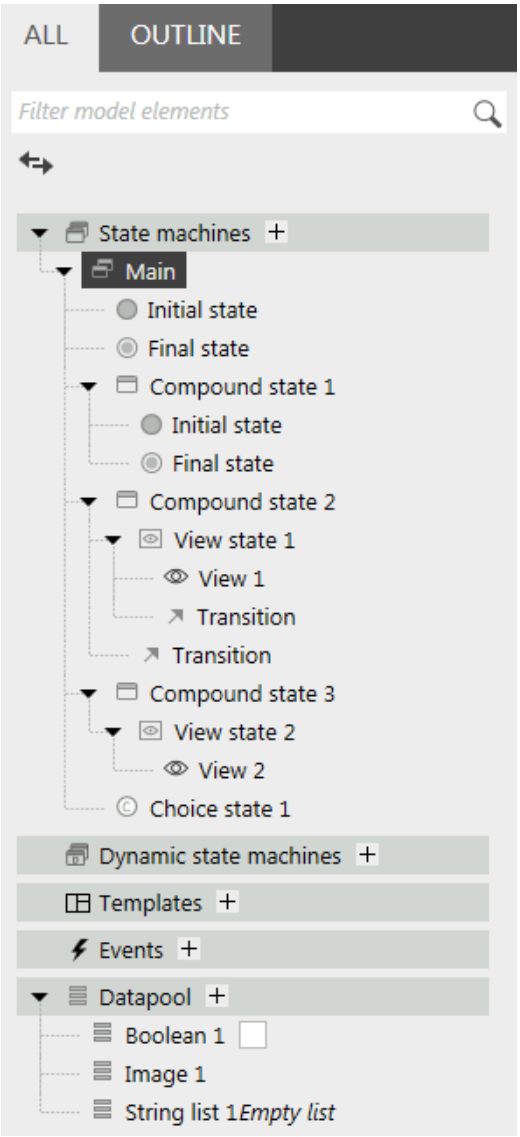

図6.4 プロジェクトエディターのナビゲーションエリア

6.5.2.2. コンテンツエリア

コンテンツエリアの表示内容は、ナビゲーションエリアでの選択内容によって異なります。モデル要素を編集する場合、ナビゲーションエリアでそのモデル要素をダブルクリックするとコンテンツエリアにそれが表示されます。例えば、ステートマシンのステートをモデリングする場合は、ビュー内にウィジェットを整列させるか、コンテンツエリアでEB GUIDEスクリプトを編集します。

コンテンツエリア内のステートまたはウィジェットをクリックしてF3キーを押すと、参照検索が始まります。検索結果ウィンドウが開き、選択したステートまたはウィジェットのEB GUIDEモデル内での出現箇所がすべてリストされます。

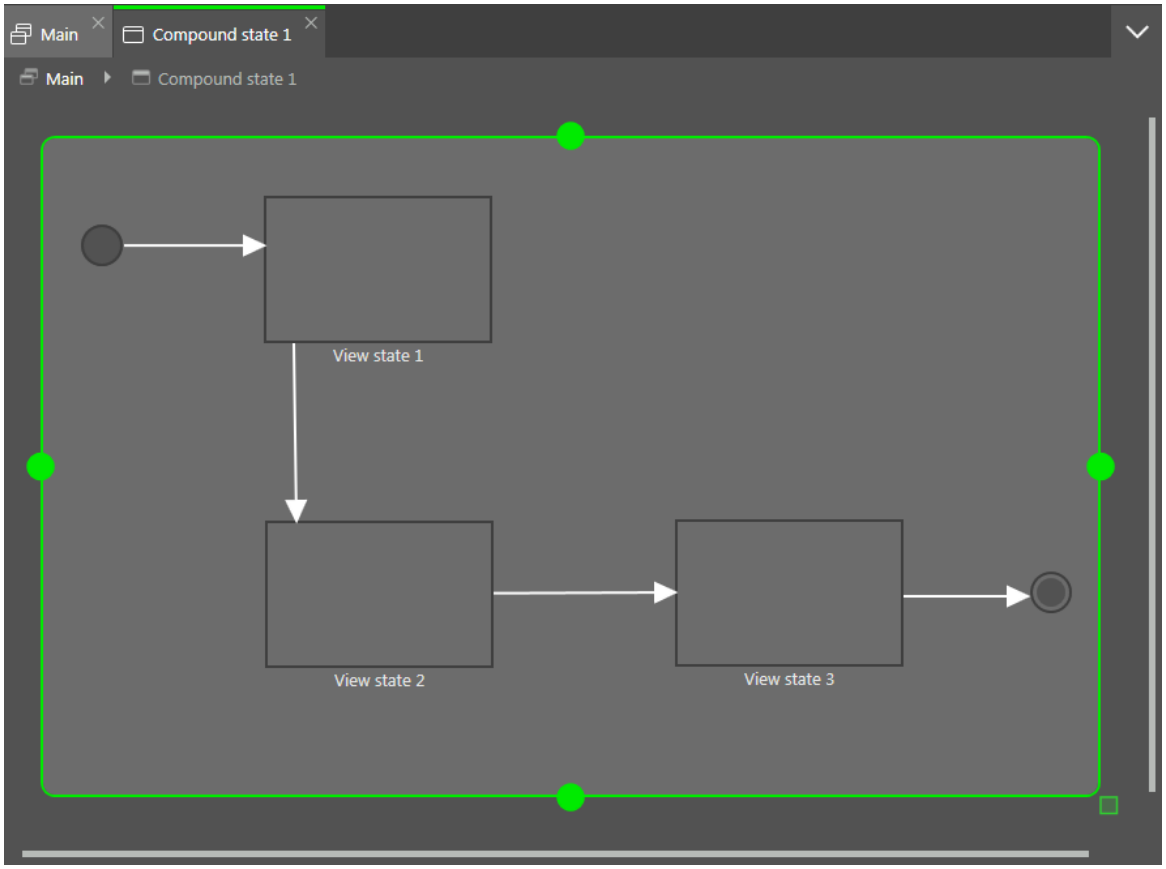

図6.5 プロジェクトエディターのコンテンツエリア

6.5.2.3. コマンドエリア

コマンドエリアにあるボタンを使うと、プロジェクトセンター、検索ボックス、その他のメニューを開くことができます。

検索ボックス

検索ボックスを使うと、モデル要素を検索することができます。検索ボックスは次のように使用します。

- 検索ボックスをクリックするか、Ctrl +Fキーのショートカットを使用して検索ボックスにジャンプします。検索するモデル要素の名前を入力します。
- ヒットリスト内のモデル要素をダブルクリックし、ジャンプします。

検索結果ウィンドウの左側には、見つかったモデル要素がカテゴリ別に分類された状態でリストされます。上部のフィルタボタンを使用して、カテゴリの表示/非表示を切り替えます。モデル要素を選択して、プレビューを表示するか、読み取り専用モードでモデル要素のプロパティを表示します。

検索結果ウィンドウを閉じると最後に使用した検索語、フィルタ設定、対応するヒットリストが保存され、次に検索結果ウィンドウを開いたときに表示されます。次に開くまでの間にモデル要素が変更された場合、再度検索を実行する必要があります。

検索では大文字と小文字は区別されません。

アスタリスク*を使用してワイルドカード検索を行う場合、次のルールが適用されます。

- **t**と入力して検索すると、tが含まれる要素の名前が返されます。
- ***t**と入力して検索すると、tで終わる要素の名前が返されます。
- **t***と入力して検索すると、tで始まる要素の名前が返されます。

検索できるのは以下のモデル要素カテゴリです。

表6.1 検索ボックス内のカテゴリ

カテゴリ	説明
ステート	ヒットリストには、見つかったステートの子要素も表示されます。
ビュー	ヒットリストには、見つかったビューの子要素も表示されます。
テンプレート	ヒットリストには、見つかったテンプレートの子要素も表示されます。
イベント	プレビューには、イベントのプロパティが表示されます。
データプールアイテム	プレビューには、データプールアイテムのプロパティが表示されます。
スクリプト	プレビューには、テキストを含むスクリプトのコンテンツが表示されます。見つかったテキストは強調表示されます。
プロパティ	プレビューには、プロパティが属するウィジェットが表示されます。

6.5.2.4. ツールボックス

モデリングに必要なツールはすべて、[ツールボックス]にあります。[ツールボックス]には、コンテンツエリアに表示される要素によって異なるツールセットが提供されます。例えば、[ツールボックス]には次のものが含まれます。

- コンテンツエリアにステートマシンが表示されている場合、[ツールボックス]には、ステートマシンに追加できるステートが含まれます。
- コンテンツエリアにビューが表示されている場合、[ツールボックス]には、ビューに整列できるウィジェットやアニメーションが含まれます。
- コンテンツエリアにスクリプト値プロパティが表示されている場合、[ツールボックス]には、挿入可能なEB GUIDEスクリプト関数が含まれます。

[ツールボックス]からモデル要素をドラッグし、コンテンツエリアにドロップします。

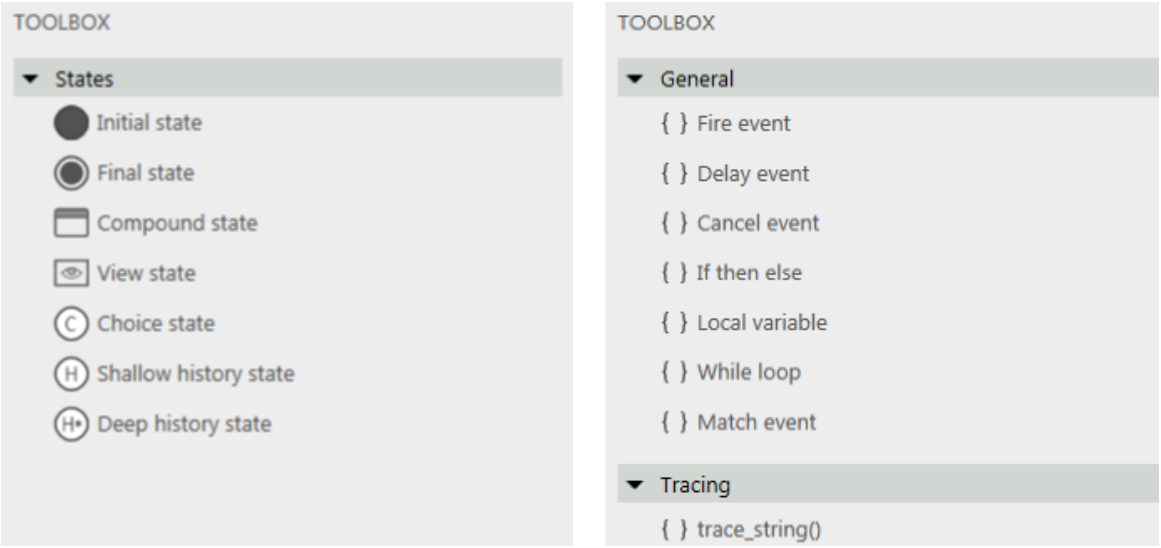

図6.6 プロジェクトエディターのツールボックス

6.5.2.5. プロパティパネル

[プロパティ]パネルには、ウィジェットやステートなど、選択されたモデル要素のプロパティが表示されます。[プロパティ]パネルのプロパティは、カテゴリによって分類されています。モデル要素が選択されている場合は、そのプロパティを[プロパティ]パネルで編集することができます。

[プロパティ]パネル内のプロパティをクリックしてF3キーを押すと、参照検索が始まります。検索結果ウィンドウが開き、選択したプロパティのEB GUIDEモデル内での出現箇所がすべてリストされます。

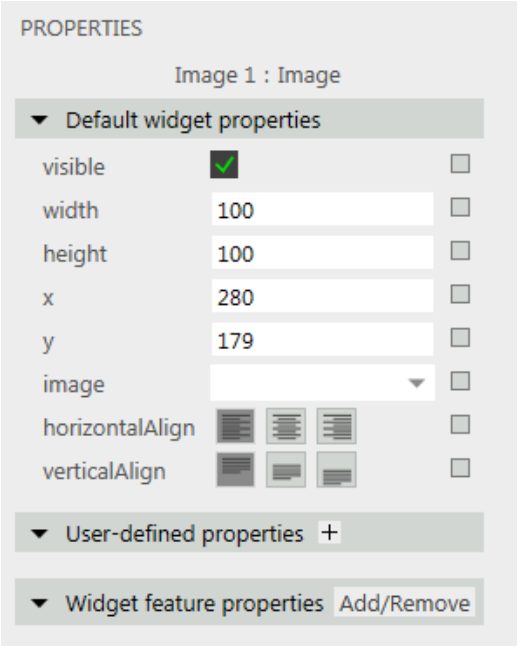

図6.7 ウィジェットのプロパティが表示された[プロパティ]パネル

6.5.2.6. ステータスバー

ステータスバーには、EB GUIDE Studioに関するステータス情報が表示されます。

6.5.2.7. 問題エリア

[問題]エリアには、EB GUIDEモデルのエラーと警告が表示されます。

6.6. データプール

6.6.1. 概念

モデルは実行中にさまざまなアプリケーションと通信します。この通信を可能にするため、EB GUIDEモデルはインターフェイスを提供しなければなりません。データプールは、データプールアイテムにアクセスしてデータを交換できるようにするインターフェイスです。データプールアイテムは、値を格納し、ヒューマンマシンインターフェイスとアプリケーションとの間で通信を成立させます。データプールアイテムは、EB GUIDEモデルで定義されます。

6.6.2. データプールアイテム

データプールアイテムは、以下の目的に使用します。

- データをアプリケーションからヒューマンマシンインターフェースへ送信します
- データをヒューマンマシンインターフェースからアプリケーションへ送信します
- ヒューマンマシンインターフェースまたはアプリケーションだけで使用されるデータを格納します

手順については、9.5「データプールアイテムの追加」をご覧ください。

通信チャネルを確立するには、通信コンテキストを使用します。

新しい値を書き込む通信コンテキストは、`Writer context`プロパティで定義します。

値の変更に関する通知を受け取り、その変更に対して応答する通信コンテキストは、`Reader context`プロパティで定義します。

内部通信の場合は、1つの通信コンテキストがデータプールアイテムのリーダーとライターを兼ねます。内部通信は、データを格納するために使用されます。例えば、内部通信に関わるデータプールアイテムはウィジェットプロパティで使用されます。

2つの通信コンテキストを使うと、外部通信が確立されます。外部通信は、データプールアイテムの`Read-only`プロパティが空白の場合にのみ実行可能です。

手順については、9.8「外部通信の確立」をご覧ください。

6.6.3. ウィンドウ表示リスト

[ウィンドウ表示]というデータプールアイテムのプロパティを使用すると、EB GUIDE product lineでウィンドウ表示リストの概念がサポートされます。多くの場合、ウィンドウ表示リストの操作モードは、大きなリスト(例えば、ディレクトリ内のすべての MP3タイトル)を表示する場合のメモリ消費量を削減するために使用されます。通常、こうしたリストは1つの通信コンテキスト(例えば、メディアアプリケーション)によって提供され、別の通信コンテキスト(例えば、ヒューマンマシンインターフェイス)によってその一部のみが表示されます。

注記

[ウィンドウ表示]プロパティが有効になっているデータプールアイテムは、1つのライター通信コンテキストと、それぞれに異なるリーダー通信コンテキストを必要とします。

ライター通信コンテキストは、仮想リストの長さとウィンドウの数を定義します。こうしたウィンドウには、そのリストの一部だけが含まれる可能性があります。リーダー通信コンテキストは、各ウィンドウの対象範囲内にある場所からの

みデータの読み取りを行います。他の場所からの読み取りは失敗します。そのようなユースケースでは、現在必要とされているリストの部分に関する情報をリーダー通信コンテキストがライター通信コンテキストに通知する必要があります。例えば、ヒューマンマシンインターフェイスは、完全なリスト内での現在のカーソル位置を提供するアプリケーション呼び出しを行うことができます。

例**6.3**
ウィンドウ表示リスト

オーディオプレーヤーデバイスのMP3 タイトルリストには、1,000,000 の要素があります。ヒューマンマシンインターフェイスは、ヘッドユニットディスプレイ、計器パネルディスプレイ、ヘッドアップディスプレイという3つの異なるディスプレイにこのリストを並行して表示する必要があります。

各ディスプレイは、別々に制御され、表示される行の数や完全なリスト内でのカーソル位置が異なります。

3つのカーソルのいずれかが動くたびに、ヒューマンマシンインターフェイスは新しい位置をイベントによってメディアアプリケーションに非同期で送信します。メディアアプリケーションは、3つのウィンドウをリストに提供します。3つのウィンドウのそれぞれは、3つのディスプレイの1つに関連付けられています。イベントとデータプールの更新に基づいた非同期の通信であるため、ウィンドウはカーソルの移動後、少し遅れて更新されます。そのため、特定のディスプレイによって表示される行の周辺まで対象範囲を拡げるようなウィンドウ位置とウィンドウサイズの使用をお勧めします。

6.7. EB GUIDEモデルとEB GUIDEプロジェクト

EB GUIDEモデルとは、ヒューマンマシンインターフェイスの外観と動作を記述するすべての要素をまとめたものです。このモデルは全体がEB GUIDE Studio内で構築されます。EB GUIDEモデルは、PCでシミュレートできます。

EB GUIDEモデルを対象デバイスで実行するには、EB GUIDEモデルをエクスポートし、結果として得られるバイナリファイルを対象デバイスにコピーします。

EB GUIDEプロジェクトは、EB GUIDEモデルと、モデリングに必要な設定で構成されます。このプロジェクトには、プロジェクト特有のオプション、カスタマイズ、リソース、またグラフィカルプロジェクトの場合にはハプティックパネルも含まれます。

EB GUIDEプロジェクトには、EB GUIDEモデル内で設定され、リンクされたオブジェクトが含まれます。これらのオブジェクトをEB GUIDEモデル要素と呼びます。例えば、以下のものはEB GUIDEモデル要素です。

- データプールアイテム
- イベント
- ステート
- ステートマシン
- ウィジェット

- リソース
- 言語

6.8. イベント処理

6.8.1. イベントシステム

イベントシステムは、通信コンテキスト内部での通信、または通信コンテキスト同士での通信をサポートする非同期メカニズムです。

EB GUIDEイベントシステムでは、すべてのイベントを送信された順序どおりに伝達します。イベントを、異なるサブスクライバに事前に定義された順序で伝達することはできません。

6.8.2. イベント

EB GUIDEのイベントは一意のイベント IDを持ち、イベントグループに属しています。イベント IDは、イベントを送受信するためにEB GUIDE TFによって使用されます。

イベントグループIDの0～65535は、EB GUIDE product lineでの内部使用のために予約されています。次の表に示すイベントグループは例外です。

表6.2 許可されるイベントグループとID

イベントグループ	**ID**
デフォルト	2
キー入力イベント	10
タッチ入力イベント	11
回転入力イベント	12
システム通知イベント	13

残りの範囲のグループIDは、顧客固有のアプリケーションで使用できます。

手順については、以下をご覧ください。

- 9.1「イベントの追加」
- 9.3「イベントへの対応」

6.9. 拡張機能

6.9.1. EB GUIDE Studio拡張機能

EB GUIDE Studioの機能を拡張し、すべてのEB GUIDEモデルで有効なソフトウェアをEB GUIDE Studio拡張機能と呼びます。EB GUIDE Studio拡張機能はEB GUIDE GTFとは無関係です。

以下はEB GUIDE Studio拡張機能の主な例です。

- 追加のツールバーボタン
- 追加のデータエクスポーター

6.9.2. EB GUIDE GTF拡張機能

EB GUIDE GTF機能拡張はEB GUIDE Studioを拡張するソフトウェアですが、1つのEB GUIDEモデルに対してのみ有効です。EB GUIDE GTF機能拡張はEB GUIDE GTFをベースにしています。

以下はEB GUIDE GTF拡張機能の主な例です。

- ウィジェットの新機能
- 新しいEB GUIDEスクリプト関数

EB GUIDE GTF拡張機能はダイナミックリンクライブラリ(`.dll`)、または共有オブジェクト(`.so`)ファイルです。

EB GUIDE GTF拡張機能は、サードパーティライブラリを含め、以下のディレクトリに配置してください。

```
$GUIDE_PROJECT_PATH/<project name>/resources/target
```

6.10. 言語

6.10.1. EB GUIDE Studioの表示言語

EB GUIDE Studioにはグラフィカルユーザーインターフェイスを表示する言語が数多く用意されています。表示言語はプロジェクトセンターの[オプション]タブで選択できます。

手順については、10.5「EB GUIDE Studioの表示言語の変更」をご覧ください。

6.10.2. EB GUIDEモデルの言語

ほとんどのヒューマンマシンインターフェイスでは、テキストをユーザーの優先する言語で表示できます。そのような言語の管理機能もEB GUIDEによって提供されます。EB GUIDEモデルの言語の追加は、プロジェクト設定で行います。

手順については、8.4.1「言語の追加」をご覧ください。

データプールアイテムを言語依存にすることができます。データプールアイテムは、各言語の値を定義します。言語をサポートするには、[言語サポート]プロパティを選択します。

例6.4
言語依存テキスト

プロジェクト設定には、3つの言語が追加されています。英語、ドイツ語、フランス語が追加されていること。データプールアイテムには、英語の**Welcome**、ドイツ語の**Willkommen**、フランス語の**Bienvenue**という値があります。

手順については、11.6「チュートリアル: データプールアイテムへの言語依存テキスト追加」をご覧ください。

エクスポートされるEB GUIDEモデルの現在の言語は、ランタイムに設定できます。

6.10.3. 言語依存テキストのエクスポートとインポート

すべての言語依存テキストのエクスポート、編集、およびインポートを行うには、EB GUIDE Studioのエクスポート機能およびインポート機能を使用します。テキストを`.xliff`ファイルにエクスポートし、そのファイルを翻訳会社に転送します。`.xliff` (XML Localization Interchange File Format)は、抽出されたテキストを格納し、ローカライズプロセスのステップ間でデータを搬送するためのXMLベースの形式です。

翻訳終了後、翻訳された`.xliff`ファイルをEB GUIDE Studioの対応する言語にインポートします。

手順については、10.7「言語依存テキストのエクスポートとインポート」をご覧ください。

6.11. リソース管理

リソースは、EB GUIDE内で作成されないものの、プロジェクトで必要とされるコンテンツのことです。EB GUIDE Studioプロジェクトのすべてのリソースは、リソースディレクトリ内に配置します。

リソースディレクトリは、`$GUIDE_PROJECT_PATH/<project name>/resources`にあります。

EB GUIDEでは、リソースに3つのタイプがあります。

1. フォント
2. イメージ
3. 3Dグラフィック用メッシュ

リソースをプロジェクトで使うには、リソースファイルを上記のディレクトリに追加する必要があります。

6.11.1. フォント

フォントをプロジェクトで使うには、フォントをディレクトリ`$GUIDE_PROJECT_PATH/<project name>/resources`に追加します。

サポートされているフォントタイプは、TrueTypeフォント(`*.ttf`,`*.ttc`)とOpenTypeフォント(`*.otf`)です。

手順については、8.1.6「ラベルのフォントの変更」をご覧ください。

6.11.2. イメージ

イメージをプロジェクトで使うには、イメージを`$GUIDE_PROJECT_PATH/<project name>/resources`ディレクトリに追加します。別のディレクトリにあるイメージを選択した場合は、そのイメージがこのディレクトリにコピーされます。

サポートされているイメージ形式は、Portable Network Graphic (`*.png`)、Portable Pixel Map (`*.ppm`)、JPEG (`*.jpg`、`*.jpeg`)、9-patchイメージ(`*.9.png`)です。

手順については、8.1.4「ビューへのイメージの追加」をご覧ください。

6.11.2.1. 9-patchイメージ

EB GUIDE Studioでは、9-patchイメージ方式に準拠する追加のメタ情報が含まれるイメージをサポートしています。9-patchイメージは伸縮可能な`.png`イメージです。9-patchイメージには2つの黒色マーカーがあり、1つはイメージの上端を、もう1つはイメージの左端を示します。マーカーが示す範囲の外側は、拡大縮小の対象となりません。マーカーの範囲内が拡大縮小されます。マーカーはEB GUIDE Studioに表示されません。

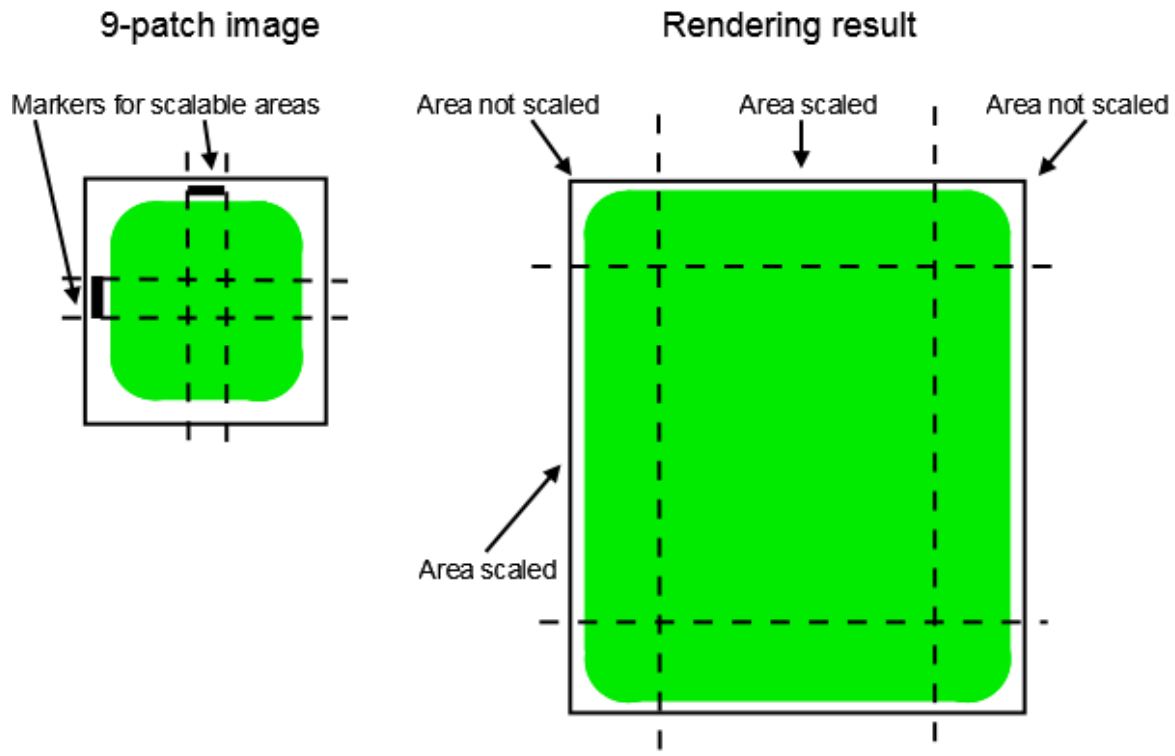

図6.8 9-patchの例

9-patchイメージを操作する際の注意事項を以下に示します。

- 9-patch処理には、OpenGL ES 2.0およびDirectXレンダラーを使う必要があります。
- 9-patch処理には、.pngを使う必要があります。.ppmイメージは9-patch処理をサポートしていません。
- 9-patchイメージの場合は、*.9.png拡張子が必須となります。
- イメージの上端と左端を示すマーカーを0個、1個、またはそれ以上指定できます。9-patch定義では、イメージの右端および下端の位置にテキストエリア用のマーカーを含めることもできます。これらのマーカーは、EB GUIDE Studioでは評価されません。

6.11.3. 3Dグラフィック用メッシュ

3DグラフィックファイルをEB GUIDE Studioにインポートすることができます。EB GUIDE Studioに3Dグラフィックファイルをインポートすると、$GUIDE_PROJECT_PATH/<project name>/resourcesにサブディレクトリが作成されます。メッシュは3Dグラフィックファイルで定義されていたとおりに.ebmeshファイルとしてインポートされます。詳細については、6.1.3「3Dグラフィックファイルのインポート」をご覧ください。

手順については、8.1.5「ビューへのシーングラフの追加」をご覧ください。

6.12. スクリプト言語EB GUIDEスクリプト

EB GUIDEスクリプトはEB GUIDEの組み込み型のスクリプト言語です。この章では、EB GUIDEスクリプト言語の機能、構文、使用方法を説明します。

6.12.1. アプリケーションの機能とエリア

EB GUIDEスクリプトは、例えば以下のようなプロジェクトのさまざまな場所で使用できます。

- ウィジェットのプロパティ内
- ステートマシン内(遷移またはステートの一部として)
- データプールアイテム内

EB GUIDEスクリプトの全機能がすべてのクラスで使用できるわけではありません。例えば、ローカルウィジェットプロパティにアクセスできるのは、スクリプトがウィジェットの一部である場合に限られます。一方、データプールへのアクセスはどこからでも可能です。

EB GUIDEスクリプトを使うと、モデル要素を直接操作できます。例えば、次の操作が可能です。

- イベントの発行
- データプールアイテムの書き込み
- ウィジェットプロパティの変更

6.12.2. ネームスペースと識別子

EB GUIDEでは、種類の異なるオブジェクトに同じ名前を付けることができます。例えば、イベントとデータプールアイテムに**Napoleon**という同じ名前を付けてもかまいません。EB GUIDEスクリプトネームスペースがこのような命名を可能にします。EB GUIDEスクリプトでは、オブジェクトの名前に相当する識別子は、先頭にネームスペースとコロンが必ず付きます。

EB GUIDEスクリプトで使われる一連のネームスペースは固定で、新しいネームスペースを追加することはできません。次のネームスペースがあります。

- `ev`: イベント
- `dp`: データプールアイテム
- `f`: ユーザー定義アクション(外部関数)
- `v`: ローカル変数

例えば、`ev:Napoleon`なら**Napoleon**という名前のイベントで、`dp:Napoleon`なら**Napoleon**という名前のデータプールアイテムという意味になります。

ネームスペースのプレフィックスがない識別子は、文字列定数として扱われます。

EB GUIDEでは、識別子にスペースや句読点を含め、多数の文字を使用できます。そのため、EB GUIDEスクリプトでは識別子を引用符で囲むことができます。例えば文字、数字、アンダースコアのみで構成される有効なC言語識別子のように、特殊記号が含まれない識別子であれば、引用符で囲む必要はありません。

例6.5
EB GUIDEスクリプトの識別子

```
dp:some_text = foo; // foo is a string here
dp:some_text = "foo"; // this statement is identical to the one above
dp:some_text = v:foo; // foo is the name of a local variable
// of course you can quote identifiers, even if it is not strictly necessary
dp:some_text = v:"foo";
// again, a string constant
dp:some_text = "string with spaces, and -- punctuation!";
// identifiers can also contain special characters, but you have to quote them
dp:some_text = v:"identifier % $ with spaces @ and punctuation!";
```

6.12.3. コメント

EB GUIDEスクリプトでは、C言語式のブロックコメントと、C++式の行コメントを使用できます。ブロックコメントは入れ子で記述できません。

例6.6
EB GUIDEスクリプトのコメント

```
/* this is a C style block comment */
// this is a C++ style line comment
```

「todo」という文字列を含むすべてのEB GUIDEスクリプトコメントについて、EB GUIDE Studioではプロジェクトを検証したとき問題エリアに警告が表示されます。この機能を使用し、すべての未処理タスクをマークしてそれらを一覧表示することができます。

注記

条件スクリプトのデフォルトコメント
デフォルトでは、`Conditional script`タイプのデータプールアイテムまたはプロパティには、`// todo: auto generated return value, please adapt`というコメントが含まれています。警告が表示されないようにするには、必要なEB GUIDEスクリプトコードを入力した後、コメントから`todo`の文字列を削除します。

6.12.4. データ型

EB GUIDEスクリプトは、強く型付けされた、静的型付けプログラミング言語です。すべての式は、明確に定義されたデータ型を持ちます。予期しないデータ型を与えるとエラーになります。

EB GUIDEスクリプトでは、次のデータ型がサポートされています。

- 整数
- Unicode文字列(string)
- 参照カウント付きのオブジェクト
- 上記および以下のデータ型への型定義
  - 色(32ビットのRGBA値を表す整数)
  - ブール値
  - 各モデル要素のID: データプールアイテム、ビュー、ステートマシン、ポップアップ(いずれも整数型)
- void(別名unit型)。例えば、Haskellなどの関数型言語で使用
- ウィジェット参照、イベント参照。これらはレコード型で、そのフィールドにアクセスするには、CやJavaで一般的な方法である**ドット**記法を使用します。これらのデータ型で新規のオブジェクトを直接作成することはできません。必要となった時点で自動的に作成されます。

すべてのデータ型とデータ型定義には互換性がなく、型のキャストはありません。そのため、スクリプトのコンパイルが正常に完了した後で型の安全性が保証されます。

6.12.5. 式

EB GUIDEスクリプトは式ベースです。すべての言語構造は式です。複数の式を演算子で連結して、大きな式を記述できます。

式を評価するとは、式をそれに相当する値に置き換えることを意味します。

例6.7
整数値の評価

```
1 + 2 // when this expression is evaluated, it yields the integer 3
```

6.12.6. 定数と参照

基本的な式は、整数、色、ブール値、文字列定数、そしてモデル要素への参照です。

void型も定数値を持ち、その書き込み方法は次の2とおりありますが、評価上の意味は同じです。

- 左右の波括弧{}を使用
- キーワードunitを使用

例6.8
定数の使用

```
"hello world"  // a string constant
true           // one of the two boolean constants
ev:back        // the event named "back" of type event_id
dp:scrollIndex // the datapool item named "scrollIndex",
               // the type is whichever type the dp item has
5              // integer constants have a dummy type "integer constant"
5::int     // typecast your constants to a concrete type!
color:255,255,255,255 // the color constant for white in RGBA format


 // the following are two ways to express the same
                       if( true )
{
}
else
{
}

if( true )
    unit
else
    unit
```

6.12.7. 算術式と論理式

EB GUIDEスクリプトでは、次の算術式がサポートされています。

- 加算(+)、減算(-)、乗算(*)、除算(/)、剰余(%)を整数型の式に実行できます。
- 論理演算子のOR(||)、AND(&&)、NOT(!)は、ブール型の式に実行できます。
- 整数または文字列は、比較演算子のより大きい(>)、より小さい(<)、以上(>=)、以下(<=)を使って比較できます。
- データ型は、等価演算子(==)または(!=)を使って比較できます。

  文字列は、等価演算子を使って比較できます。大文字と小文字は区別されません(例: (=Aa=))。

注記

等価演算子の使用

イベントおよび例えば3Dグラフィック、フォント、イメージなどのリソースデータ型は、 等価演算子 (==) または (!=) を使って比較できません。

- 文字列は (+) 演算子で連結できます。

例6.9
算術式と論理式

```
10::int + 15::int // arithmetic expression of type int
dp:scrollIndex % 2        // arithmetic expression of type int,
                          // the concrete type depends on the type
                          // of dp:scrollIndex
"Morning Star" == "Evening Star" // type bool and value false (wait, what?)
"name" =Aa= "NAME" // type bool and value true
!true              // type bool, value false
!(0 == 1)          // type bool, value true
// as usual, parenthesis can be used to group expressions
((10 + dp:scrollIndex) >= 50) && (!dp:buttonClicked)
// string concatenation
"Napoleon thinks that " + "the moon is made of green cheese"
f:int2string(dp:speed) + " km/h" // another string concatenation
```

6.12.8. L値とR値

EB GUIDEスクリプトの式には、**L値**と**R値**の2種類があります。L値はアドレスを持ち、代入演算の左辺に配置できます。R値はアドレスを持たず、代入演算の左辺に配置できません。

- L値はデータプール参照、ローカルウィジェットプロパティ、またはローカル変数です。
- R値はイベントパラメータまたは定数式(文字列定数や整数定数など)です。

6.12.9. ローカル変数

`let`式でローカル変数を導入します。この式は一連の変数宣言と`in`式で構成され、ローカル変数はこの`in`式内のみで参照できます。変数はL値であり、代入演算の左辺で使用できます。変数のネームスペースは`v:`です。`let`式の構文には、次の特徴があります。

```
let v:<identifier> = <expression> ;
```

```
    [ v:<identifier> = <expression> ; ]...
in
    <expression>
```

let式のデータ型と値は、in式の使用時と同じです。

let式は入れ子にすることができます。外側のlet式の変数は、内側の式でも参照できます。

例6.10
let式の使用方法

```
// assign 5 to the datapool item "Napoleon"
let v:x = 5 in dp:Napoleon = v:x;

// define several variables at once
let v:morning_star = "Venus";
    v:evening_star = "Venus";
in
    v:morning_star == v:evening_star; // Aha!

let v:x = 5;
    v:y = 20 * dp:foo;
in
{
    // Of course you may have a sequence as the in expression,
    // but parenthesis or braces are required then.
    v:x = v:y * 10;
    dp:foo = v:x;
}
// Because let expression also have types and values, we can have them
// at the right hand side of assignments.
dp:x = let v:sum = dp:x + dp:y + dp:z
       in v:sum; // this is the result
                 // of the let expression

// A nested let expression
let v:x = dp:x + dp:y;
v:a = 5;
in
{
    let v:z = v:x + v:a;
    in
    {
        dp:x = v:z;
    }
}
```

6.12.10. Whileループ

EB GUIDEスクリプトのwhileループ構文は、CやJavaと似ています。条件式と実行式から構成されます。以下の構文を使用します。

```
while (<condition expression> ) <do expression>
```

条件式がfalseとみなされるまで、実行式が繰り返し評価されます。condition expressionはブール型で、実行式はvoid型でなければなりません。while式はvoid型であり、代入演算の左辺または右辺に記述することはできません。

例6.11
whileループの使用方法

```
// Assume dp:whaleInSight is of type bool
while( ! dp:whaleInSight )
{
    dp:whaleInSight = f:lookAtHorizon();
}
```

6.12.11. If-then-else

EB GUIDEスクリプトのif-then-elseは、CやJavaの三項条件演算子(?:)と似た動作をします。

if-then-else式は、次のサブ式で構成されます。

- 条件式
- then式
- else式

以下の構文を使用します。

```
if ( < condition expression> ) <then expression> else <else expression>
```

if-then-elseは、次のように動作します。

1. 最初に条件式が評価されます。この式はブール型でなければなりません。
2. 条件がtrueであれば、then式が評価されます。
3. 条件がfalseであれば、else式が評価されます。

if-then-elseそのものが1つの式です。式全体のデータ型は、then式とelse式のデータ型です。この2つは一致していなければなりません。if-then-else式の値は、then式の値またはelse式の値で、どちらの値になるかは上記の規則に従います。

if-then-elseでは、else分岐を省略できる特殊なケースがあります。この記述が許されるのは式がvoid型の場合で、スクリプトから戻り値を帰すことはできません。

例6.12
if-then-elseの使用方法

```
// Assume dp:whaleInSight is of type bool
// and dp:user is of type string.
if( dp:whaleInSight && dp:user == "Captain Ahab" )
{
    dp:mode = "insane";
}
else
{
    dp:mode = "normal";
}

// Because if-then-else is also an expression,
// we may simplify the previous example:
dp:mode = if( dp:whaleInSight && dp:user == "Captain Ahab" )
            "insane"
          else
            "normal"

if ( <expression> ) <expression> // This is the reduced way of
                writing if-then-else
            //It is an alternative to the following
            if( <expression> ) { <expression> ; {} } else {}
```

6.12.12. 外部関数呼び出し

C言語で書かれた関数(外部関数と呼ぶ)を使って、EB GUIDEスクリプトの機能を拡張できます。

f:を先頭に付けた識別子は、外部関数の名前として扱われます。外部関数には引数リストと戻り値があり、これはC言語の関数と同じです。外部関数の構文は以下のとおりです。

```
f:<identifier> ( <expression> [ , <expression> ] ... )
```

例6.13
外部関数の呼び出し

```
// write some text to the connection log
f:trace_string("hello world");
// display dp:some_index as the text of a label
v:this.text = f:int2string(dp:some_index);
```

```
// passing different parameters of matching type
f:int2string(v:this.x)
f:int2string(4)
f:int2string(dp:myInt)
f:int2string(v:myVar)


//passing parameters of different types
// starts an animation (parameter type GtfTypeRecord) from a script
// located in its parent widget
f:animation_play(v:this->Animation);

// checks the number of child widgets of a widget (parameter type widget)
f:widgetGetChildCount(v:this);

// traces debugging information about a datapool item (parameter type dp_id)
// to the connection log; uses the address of the datapool item as parameter
f:trace_dp(&dp:myFlag);
```

6.12.13. データプールアクセス

EB GUIDEスクリプトで書かれたスクリプトは、データプールアイテムを読み書きできます。ネームスペースdp:が先頭に付けられた識別子は、データプールアイテム式と呼ばれます。この式のデータ型は**X型のデータプールアイテム**で、この「X」は参照先のデータプールエントリーのデータ型です。

X型のデータプールアイテムを代入演算の左辺に使い、X型の式を右辺に使った場合、データプールアイテムの値が書き込まれます。

プログラム内でデータプールアイテムを代入演算の左辺ではない場所に使った場合、データプールアイテムの値が読み取られます。

例6.14
データプールアイテム値の代入

```
// Assume intA to be of type int. Assign 10 to it.
dp:intA = 10;
// Assume strA to be of type string. Assign the string "blah" to it.
dp:strA = blah; // Yes, we can omit the quotes, remember?
dp:strA = 42; // Error: integer cannot be assigned to string

// Assign the value of the datapool item intB to intA.
// Both datapool items must have the same type.
dp:intA = dp:intB;
// Multiply the value of intB by two and assign it to intA.
dp:intA = 2 * dp:intB;
```

```
// Use the value of a datapool item in an if-clause.
if( dp:speed > 100 )
{
    // ...
}
```

以下の演算子がデータプールアイテムに使用できます。

- 参照演算子(&)はデータプールアイテムに使用できます。この演算子を使うと、データプールアイテムの値ではなくアドレスを参照できます。参照演算子は、dp_id型のパラメータを渡すために外部関数で使用します。
- リダイレクト参照演算子(=>)は、データプール参照に別のデータプールアイテムを代入します。これが行われるのはデータプール参照のみです。

6.12.14. ウィジェットプロパティ

スクリプトがウィジェットの一部である場合は、スクリプトからそのウィジェットのローカルプロパティにアクセスできます。EB GUIDEスクリプトでは、このプロパティにドット記法でアクセスするためにv:thisというローカル変数が作成されます。

スクリプトは、ウィジェットのローカルプロパティに接続されると、そのウィジェットの一部となります。クリックやボタン操作などの入力アクションにスクリプトを接続するケースがこれに該当します。

例6.15
ウィジェットプロパティの設定

```
// assume this script is part of a widget
v:this.x = 10; // if the widget has an x-coordinate

v:this.text = "hello world"; // if the widget is a label and has a text property
// assume testEvent has one integer parameter
fire ev:testEvent(v:this.x);
```

スクリプトがウィジェットの一部である場合は、スクリプトからウィジェットツリーにある他のウィジェットのプロパティにもアクセスできます。

アロー演算子(->)を使って、ウィジェットツリー内の他のウィジェットを参照します。以下の構文を使用します。

```
<expression> -> <expression>
```

式の左辺はウィジェットを参照し、右辺は子ウィジェットの名前を文字列で記述する必要があります。親ウィジェットへ移動するには、右側に^記号を付けます。アロー式全体が1つのウィジェットを参照します。

ウィジェットツリーの別の場所を参照すると、ランタイムのパフォーマンスが低下することがあります。複数のプロパティを効率よく操作するには、ウィジェットをローカル変数に代入します。

例6.16
ウィジェットプロパティへのアクセス

```
v:this.x       // access the properties of the current widget
v:this->^.x    // access the x property of the parent widget
v:this->^->caption.text // access the text property of a label called caption,
                        // read: "go-to parent, go-to caption, text"

// Modify several properties of the caption.
// This way, the navigation to the caption is only performed once.
let v:cap = v:this->^->caption
in
{
  v:cap.textColor = color:0,0,0,255;
  v:cap.x += 1;
  v:cap.y += 1;
}
```

6.12.15. リスト

データプールアイテムとウィジェットプロパティには、リストを格納できます。添字演算子(`[]`)を使うと、リスト要素にアクセスできます。以下の構文を使用します。

```
<expression> [ <expression> ]
```

最初の式ではリストの型を評価し、2番目の式では整数値を評価する必要があります。リストが`list A`型である場合、リスト添字式全体は`A`型でなければなりません。

リスト添字式を代入演算の左辺に使うと、参照先リスト要素の値が書き込まれます。

`length`キーワードは、リストにある要素の数を返します。このキーワードをリスト式の前面に配置すると、式全体が整数型になります。

例6.17
リスト

```
// Assume this widget is a label and dp:textList is a list of strings
v:this.text = dp:textList[3];

dp:textList[1] = v:this.text; // writing the value of the list element


v:this.width = length dp:textList;// checking the length of the list
dp:textList[length dp:textList - 1] = "the end is here";
```

EB GUIDEスクリプトでは、現在、リスト要素の追加と削除はサポートされていません。

リストの終端を超えてリスト要素にアクセスしようとすると、スクリプトの実行が即座に中止されます。リストへのアクセスが常に範囲を超えないように注意してください。

6.12.16. イベント

EB GUIDEスクリプトには、イベントを処理する以下の式が用意されています。

- `fire`式はイベントを送信します。以下の構文を使用します。

  ```
  fire ev:<identifier> ( <parameter list> )
  ```

  イベントにパラメータを指定できますが、必須ではありません。`fire`式のパラメータリストは、発火したイベントのパラメータと一致する必要があります。イベントにパラメータがなければ、括弧の中は空白にします。

  

  例6.18
  `fire`式の使用

  ```
  fire ev:toggleView(); // the event "toggleView" has no parameters
  fire ev:mouseClick(10, 20); // "mouseClick" has two integer parameters
  fire ev:userNameEntered("Ishmael"); // string event parameter
  ```

- `fire_delayed`式は、指定された時間が経過した後でイベントを送信します。以下の構文を使用します。

  ```
  fire_delayed <time> , ev:<identifier> ( <parameter list> )
  ```

  `time`パラメータは、この遅延時間をミリ秒単位で指定する整数値です。

  

  例6.19
  `fire_delayed`式の使用

  ```
  fire_delayed 3000, ev:mouseClick(10, 20); // send the event "mouseClick"
                  //in 3 seconds.
  ```

- `cancel_fire`式は、遅延中のイベントを取り消します。以下の構文を使用します。

  ```
  cancel_fire ev:<identifier>
  ```

- `match_event`式では、スクリプトの実行がイベントによってトリガーされたものかどうかをチェックします。以下の構文を使用します。

  ```
  match_event v:<identifier> = ev:<identifier>
  in
      <expression>
  else
      <expression>
  ```

  `match_event`式の型は、`in`式と`else`式の型です。この2つは一致していなければなりません。

match_event式では、else分岐を省略できる特殊なケースがあります。この記述が許されるのは式がvoid型の場合で、スクリプトから戻り値を帰すことはできません。

例6.20
match_event式の使用

```
match_event v:theEvent = ev:toggleView in
{
    // this code will be executed when the "toggleView" event
    // has triggered the script
    dp:infoText = "the view has been changed";
}
else {}

match_event ( <expression> ) in <expression> //special form
                  //without an else branch
               //The special form is an alternative way to express the following
               match_event ( <expression> ) in { <expression> ; {} } else {}
```

スクリプトがパラメータ付きのイベントによってトリガーされる場合は、match_event式のin式でこのパラメータにアクセスできます。C言語で構造体のフィールドにアクセスする場合と同じように、ドット記法でパラメータから値を読み取ります。イベントのパラメータは、else式で使用できません。

例6.21
イベントパラメータ

```
// assume that "mouseClick" has two parameters: x and y
match_event v:event = ev:mouseClick in
{
    dp:rectX = v:event.x;
    dp:rectY = v:event.y;
}
```

6.12.17. 文字列の書式設定

EB GUIDEスクリプトでは、文字列の書式設定に、連結演算子(+)やさまざまなデータ文字列変換関数を使います。EB GUIDEスクリプト標準ライブラリには、整数から文字列への簡易な変換を実行するint2string関数が付属します。

例6.22
文字列の書式設定

```
// Assume this widget is a label and has a text property.
// Further assume that the datapool item dp:time_hour and
```

```
// dp:time_minute hold the current time.
v:this.text = "the current time is: " + f:int2string(dp:time_hour)
    + ":" + f:int2string(dp:time_minute);
```

6.12.18. 標準ライブラリ

EB GUIDEスクリプトには、例えば以下に示すような一連の外部関数で構成される標準ライブラリが付属します。

- 文字列の書式設定
- 言語管理
- トレース
- 時刻と日付
- 乱数生成

詳細については、12.4.3「EB GUIDEスクリプト標準ライブラリ」をご覧ください。

6.13. スクリプト値

スクリプト値は、ウィジェットプロパティまたはデータプールアイテムの値を、別の表記に変換したものです。この変換によって、ウィジェットプロパティまたはデータプールアイテムは、他のモデルの要素を使用して、自身の値を評価したり、イベントやプロパティの更新に反応したりしています。スクリプト値はEB GUIDEスクリプトのスクリプト言語で記述されます。

EB GUIDEのプロパティは、スクリプト値に変換することも、スクリプト値から元の値に戻すこともできます。

手順については、9.7「プロパティのスクリプト値への変換」をご覧ください。

スクリプト値を編集するには、EB GUIDE Studioのスクリプトエディターを使用します。このエディターはいくつかに分割されています。

図6.9 EB GUIDE StudioのEB GUIDEスクリプト

- [Read]スクリプトはスクリプト値のプロパティが読み取られると呼び出されます。プロパティがリストタイプの場合は、パラメータにリストインデックスが含まれます。

  [Read]スクリプトの戻り値は、プロパティの現在の値を表します。
- [Write]スクリプトはスクリプト値のプロパティが書き込まれると呼び出されます。

  新しいプロパティ値は[Write]　スクリプトのパラメータになります。プロパティがリストタイプの場合は、パラメータにリストインデックスが含まれます。

  [Write]スクリプトの戻り値によって、プロパティの変更通知が次のように制御されます。
  - true: 変更通知をトリガーする
  - false: 変更通知をトリガーしない
- [Trigger]リストには、[On　trigger]スクリプトの実行をトリガーするイベント、データプールアイテム、およびウィジェットプロパティのリストが含まれます。
- [On　trigger]スクリプトはイベントをトリガーしたのち、またはプロパティを更新したのち、初期化時に呼び出されます。

  [On　trigger]スクリプトのパラメータは、スクリプトを実行する要因を表します。実行は初期化、または[Trigger]リストに含まれるいずれかのトリガーによって引き起こされます。

  [On trigger]スクリプトの戻り値によって、プロパティの変更通知が次のように制御されます。
  - true: 変更通知をトリガーする
  - false: 変更通知をトリガーしない
- [Length]スクリプトはリストタイプのプロパティでのみ使用されます。

[Length]スクリプトの戻り値は、リストの現在の長さを表します。

6.14. ショートカット、ボタン、アイコン

6.14.1. ショートカット

次の表に、EB GUIDE Studioで使用できるショートカットとその意味を示します。

表6.3 ショートカット

ショートカット	説明
Ctrl+C	選択内容をコピー
Ctrl+F	検索ボックスにジャンプ
Ctrl+S	保存
Ctrl+V	コピーした選択内容を貼り付け
Ctrl+Y	やり直し
Ctrl+Z	元に戻す
Alt+F4	アクティブなウィンドウを閉じる
Shift+F1	EB GUIDE TFのユーザーマニュアルを開く
F1キー	EB GUIDE Studioのユーザーマニュアルを開く
F2キー	ナビゲーションエリア内の選択したモデル要素の名前を変更
F3キー	EB GUIDEモデル内における選択した要素の出現箇所をすべて検索
F5キー	シミュレーションの開始
F6キー	検証
Deleteキー	選択したモデル要素をコンテンツエリアまたはナビゲーションエリアから削除
-	ナビゲーションエリア内の選択したモデル要素を折りたたむ
*と+	ナビゲーションエリア内の選択したモデル要素を展開
上/下/左/右方向キー	コンテンツエリア内の選択したステートまたはウィジェットを上/下/左/右の方向に1ピクセル移動

次の表に、EB GUIDE Studioで使用できるコマンドラインオプションとその意味を示します。

表6.4 コマンドラインオプション

オプション	説明
`-h`	ヘルプメッセージを表示

オプション	説明
`-e <project file, destination dir, profile>`	`project file`として保存されたEB GUIDEモデルを、選択した`profile`を使用して出力先ディレクトリ`destination dir`にエクスポート
`-m`	エクスポート中のプロジェクトの移行を許可

6.14.2. ボタン

次の表に、EB GUIDE Studioで使用されているボタンとその意味を示します。

表6.5 EB GUIDE Studioのボタン

ボタン	説明
	元に戻す
	やり直し
	保存
	プロジェクトを検証
	シミュレーションを開始
	シミュレーションを停止
	プロジェクトセンターを開く
	追加のエディターを開く
	コンテンツエリアとナビゲーションエリアの同期を取る
	イベント、データプールアイテム、またはステートマシンを追加
	プロパティに関連付けられたコンテキストメニューを開く 次のようにボタンが色分けされています。 プロパティがローカルである プロパティが別のプロパティにリンクされている プロパティがデータプールアイテムにリンクされている プロパティ値がテンプレート値と等しい

6.14.3. アイコン

次の表に、EB GUIDE Studioで使用されているアイコンとその意味を示します。

表6.6 EB GUIDE Studioのアイコン

アイコン	説明
	ビューテンプレートの終了アニメーションを示す
	ビューテンプレートの開始アニメーションを示す
	ステートマシンまたはステートのエントリーアクションを示す
	ステートマシンまたはステートの終了アクションを示す
	エントリーアクションまたは終了アクションを削除するコンテキストメニューを開く
	動的ステートマシンリストが有効になっていることを示す
	テンプレートを示す
	遷移を示す
	内部遷移を示す
	ウィジェットテンプレート: プロパティがウィジェットテンプレートインターフェイスに追加されていることを示す

6.15. ステートマシンとステート

6.15.1. ステートマシン

ステートマシンは、決定性有限オートマトンであり、システムの動的な動作を表します。EB GUIDEのステートマシンは、階層的に順序付けられた任意の数のステートと、ステート間の遷移から構成されます。

EB GUIDEでは、以下のタイプのステートマシンを作成できます。

6.15.1.1. ハプティックステートマシン

ハプティックステートマシンを使うと、GUIの仕様を提供できます。

6.15.1.2. ロジックステートマシン

ロジックステートマシンを使うと、GUIなしでロジックの仕様を提供できます。

6.15.1.3. 動的ステートマシン

動的ステートマシンは、他のステートマシンと並行して動作します。

動的ステートマシンは、システムの起動時に自動的に開始されません。動的ステートマシンの開始と停止は、別のステートマシンから実行されます。

動的ステートマシンには、次の2種類があります。

- ハプティック動的ステートマシン
- ロジック動的ステートマシン

手順については、11.1「チュートリアル: 動的ステートマシンの追加」をご覧ください。

6.15.2. ステート

EB　GUIDEでは、ステートという概念を使用します。ステートは、ステートマシンのステータスと動作を決定します。ステートは遷移でリンクされます。遷移は、ステート間の接続であり、ソースステートからターゲットステートへのステート変更を定義します。

ステートには次のプロパティがあります。

- エントリーアクション
- 終了アクション
- 内部遷移

6.15.2.1. 混合ステート

混合ステートは、内部に他のステートを子ステートとして持つことができます。混合ステートは階層構造で、任意の数の子ステートを作成できます。どのタイプのステートでも混合ステートに入れ子にすることができます。

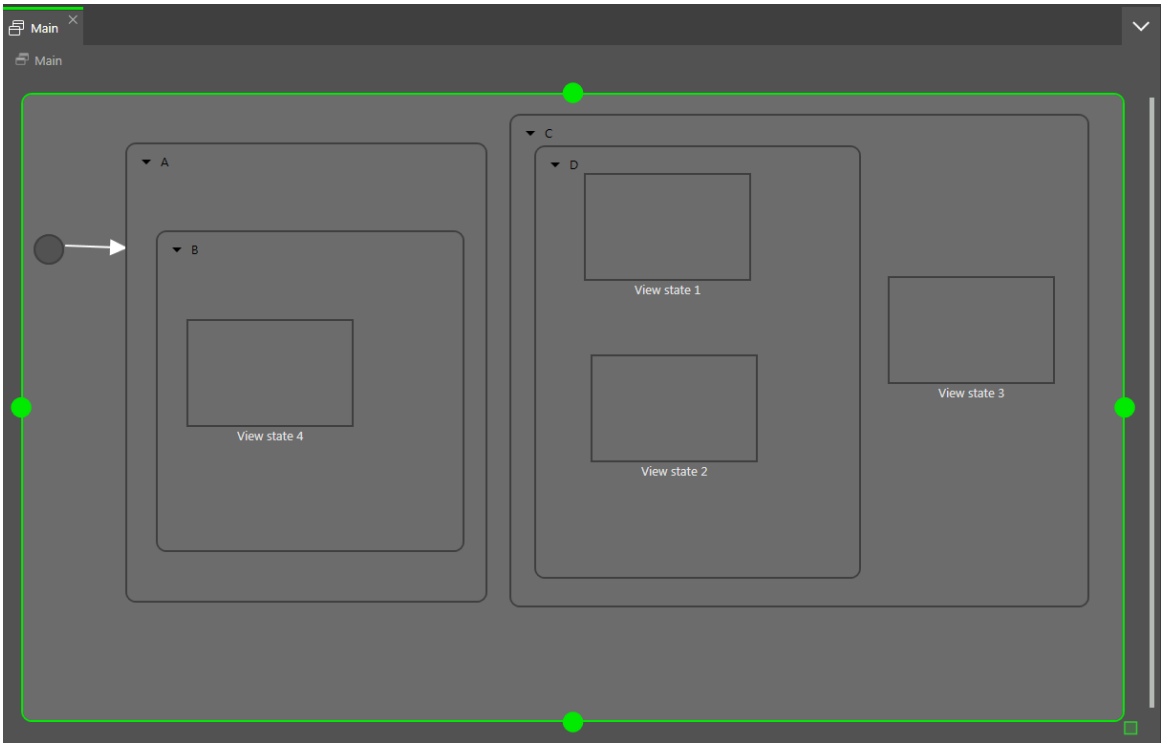

図6.10 混合ステート

ナビゲーションエリアには、ステートの階層がツリー構造で表示されます。

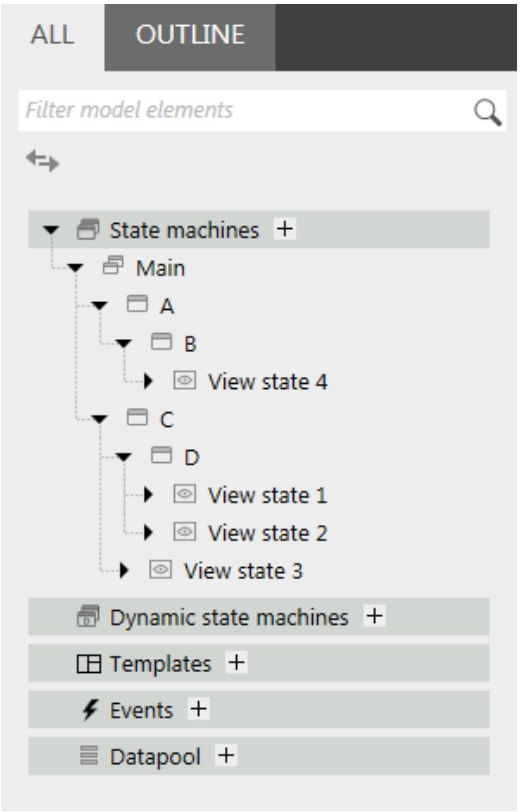

図6.11 ツリー構造で表示されたステート階層

混合ステートには、入力または出力遷移、および内部遷移をいくつでも作成できます。子ステートは親ステートの遷移を継承します。

6.15.2.2. ビューステート

ビューステートはビューを格納します。ビューはプロジェクト固有のヒューマンマシンインターフェイス画面を提供します。ビューは、対応するビューステートがアクティブになっている場合に表示されます。ビューは、ユーザーとシステムを結ぶインターフェイスであるウィジェットから構成されます。

6.15.2.3. 初期ステート

初期ステートは、ステートマシンのスタート地点となります。初期ステートには、最初のステートを指すデフォルト遷移があります。初期ステートへエントリーする遷移はありません。

初期ステートは、混合ステートのスタート地点として使うことができます。混合ステートへは次の方法でエントリーします。

- 混合ステートへの遷移を使う(初期ステート必須)

- 混合ステートの子ステートへの遷移を使う

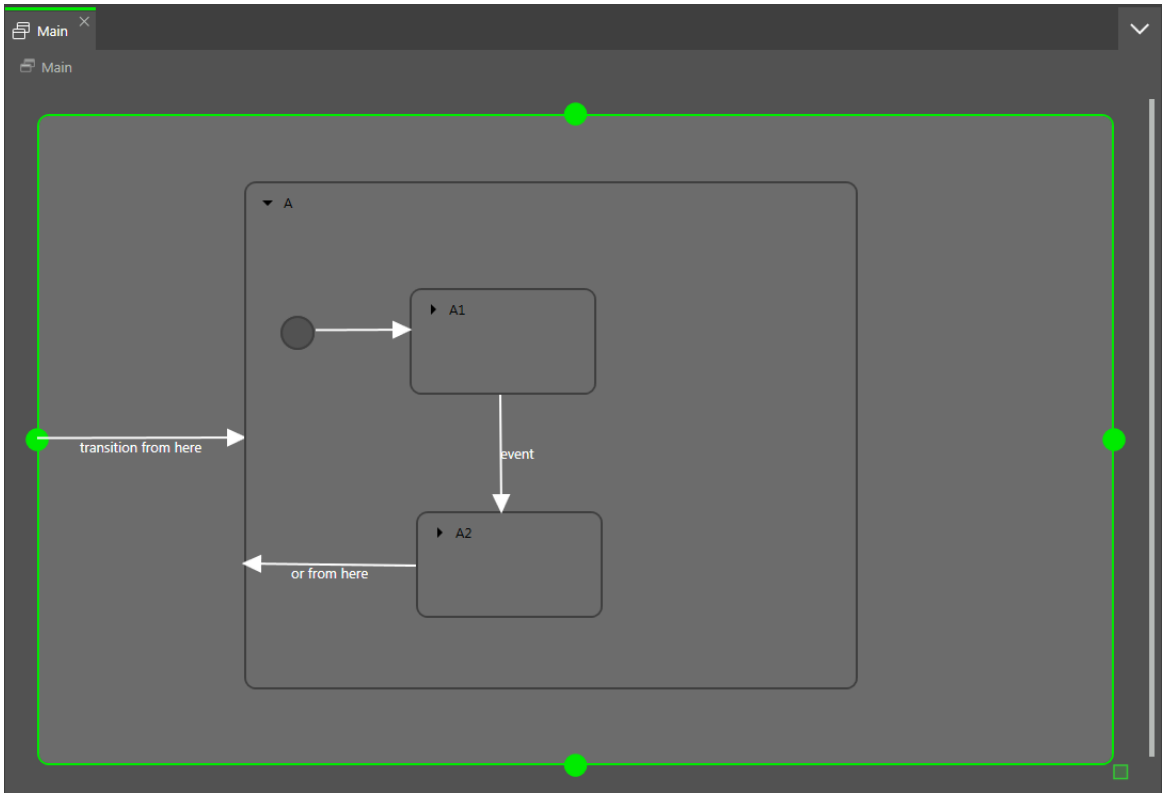

図6.12 初期ステートの例

6.15.2.4. 最終ステート

最終ステートは混合ステートを終了するために使用します。ステートマシンの最終ステートへエントリーすると、ステートマシンの動作は終了します。混合ステート内の履歴ステートはリセットされます。最終ステートから他のステートへ向かう遷移はありません。

混合ステートには最終ステートを1つだけ作成できます。最終ステートがトリガーされるのは、以下の状況です。

- 子ステートから混合ステートの外部への遷移が発生した(イベントzによる遷移)
- 混合ステートからの出力遷移が発生した(イベントyによる遷移)
- 混合ステート内で最終ステートへの遷移が発生した(イベントxによる遷移)

最終ステートを含む混合ステートには、出力遷移が必要です。

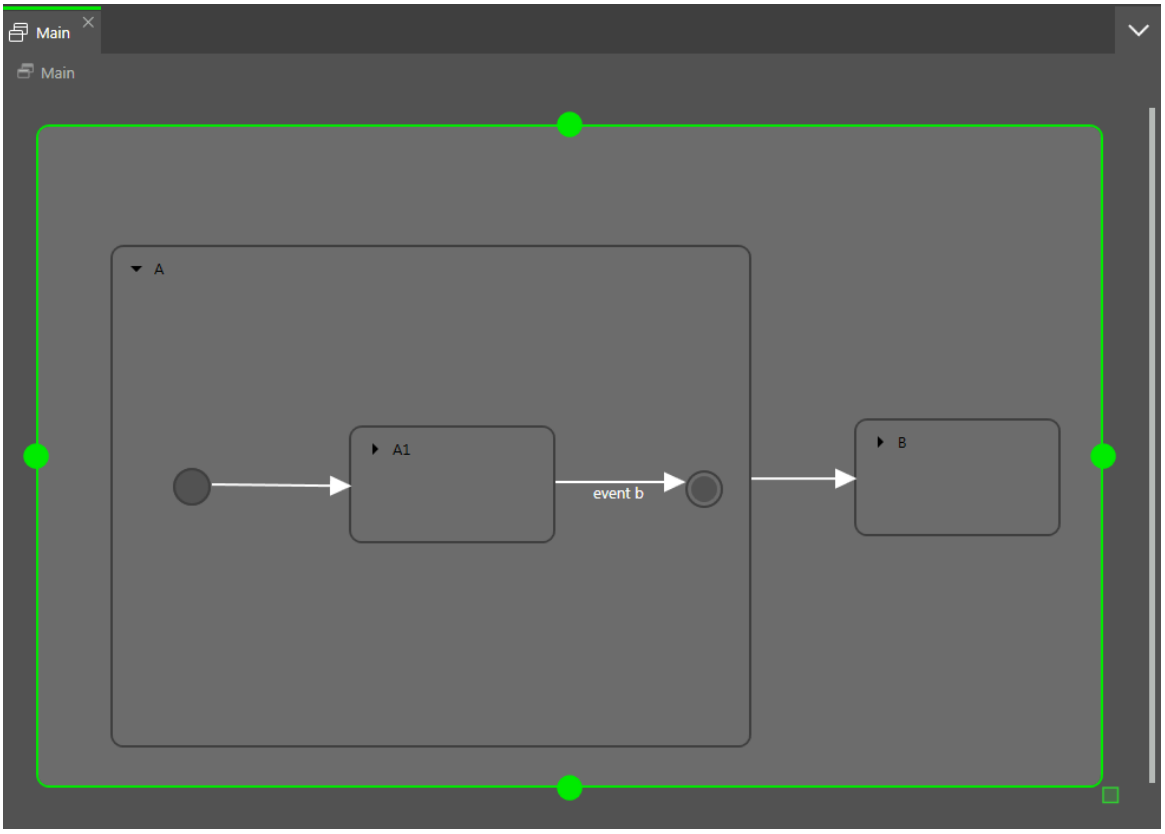

図6.13 混合ステート内での最終ステートの使用方法

6.15.2.5. 選択ステート

選択ステートは、動的な条件分岐を実現します。条件に基づいてイベントを発行する場合に使用します。選択ステートは、遷移元のステートと遷移先のステートを接続するものです。1つの選択ステートに複数の入出力遷移を設定できます。出力遷移には条件が与えられ、その条件が`true`と評価されない限り、遷移は実行されません。出力遷移の1つに`else`遷移があります。他のすべての条件が`false`と評価されると、この遷移が実行されます。`else`遷移は必須です。

外へ向かう複数の遷移がtrueと評価されることがあるため、遷移を評価する順序を定義する必要があります。

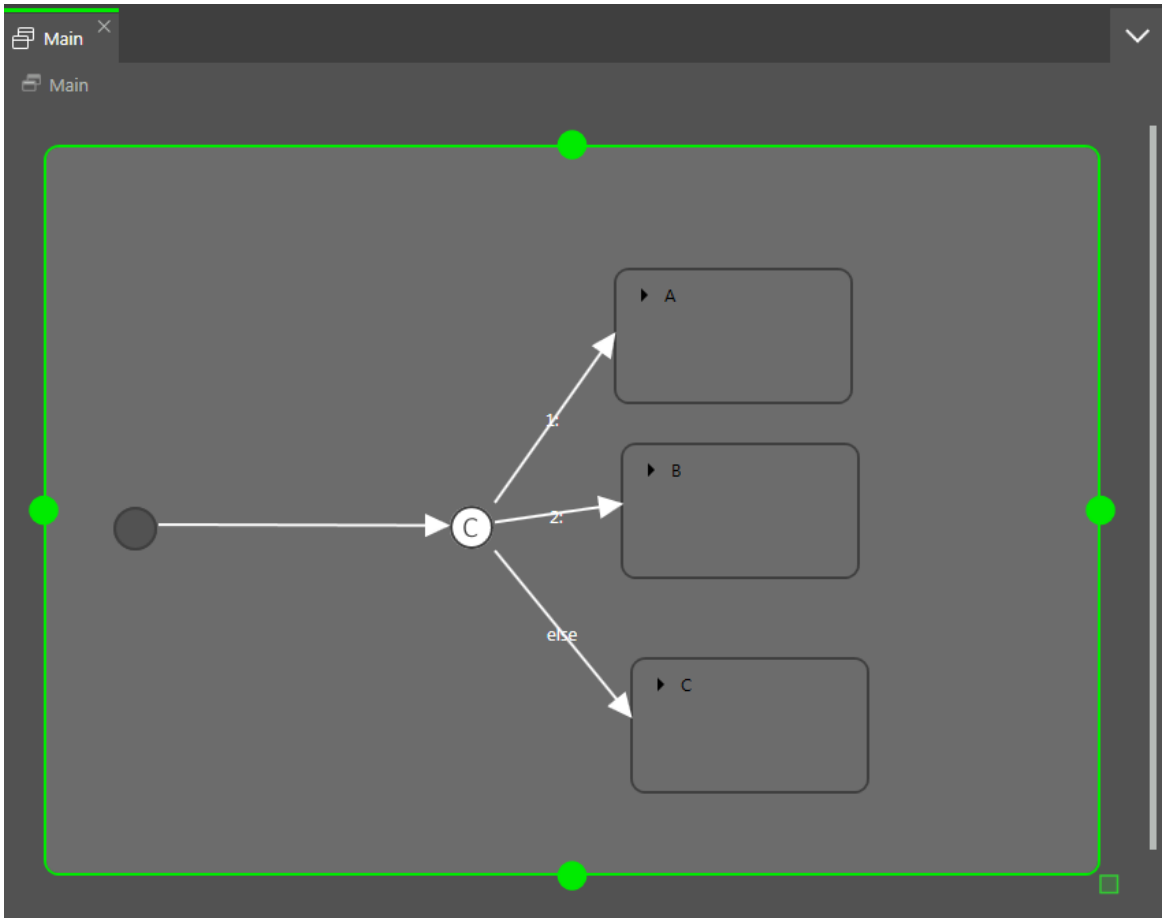

図6.14 入力遷移と出力遷移を持つ選択ステート

6.15.2.6. 履歴ステート

EB GUIDEには、2種類の履歴ステートがサポートされています。

- 浅い履歴ステートには、最新のアクティブなサブステート(混合ステートを終了する直前にアクティブだったサブステート)が保存されます。
- 深い履歴ステートには、混合ステートと、混合ステートが終了する直前のそのサブ階層全体が保存されます。

履歴ステートの親ステートへ初めてエントリーしたときに、最後にアクティブだった子ステートが復帰されます。

浅い履歴ステートは、混合ステートが終了する直前にアクティブだったステートだけを覚えています。ステートの階層は記憶できません。

浅い履歴ステートは、混合ステート内に記録された最後のアクティブステートを復帰します。この履歴ステートには、外へ向かう無条件のデフォルト遷移があるほか、複数のエントリー遷移を指定できます。

混合ステートへ初めてエントリーする時点では、浅い履歴ステートは空です。空の浅い履歴ステートへエントリーすると、そのステートのデフォルト遷移によって次のステートが決定されます。

例6.23
浅い履歴ステート

浅い履歴ステートは次のように使用できます。

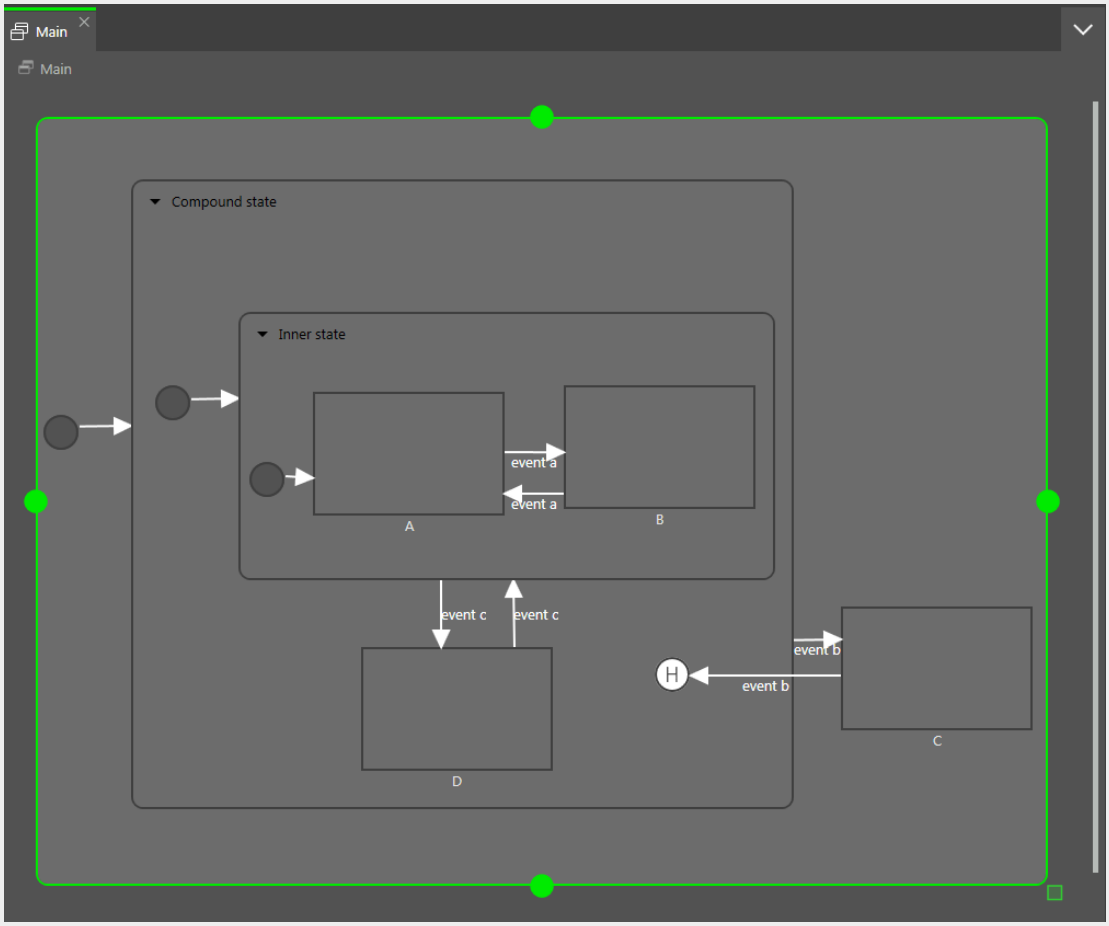

図6.15 浅い履歴ステート

- ケース1: アクティブなステートはDです。
  1. `event b`が発行され、ステートCにエントリーします。
  2. `event b`が再び発行され、浅い履歴ステートにエントリーします。
  3. ステートマシンは、浅い履歴ステートからステートDにエントリーします。ステートDが`Compound State`内の最後のアクティブステートだったからです。
- ケース2: アクティブなステートはBです。
  1. `event b`が発行され、ステートCにエントリーします。
  2. `event b`が再び発行され、浅い履歴ステートにエントリーします。
  3. ステートマシンは、浅い履歴ステートから`Inner state`にエントリーします。浅い履歴ステートは、最後にアクティブだったステートを記憶していますが、階層を記憶できないからです。

4. `Inner state`にエントリーするとステート`A`に遷移します。

深い履歴ステートでは階層履歴も記録されます。

例6.24
深い履歴ステート

深い履歴ステートは次のように使用できます。

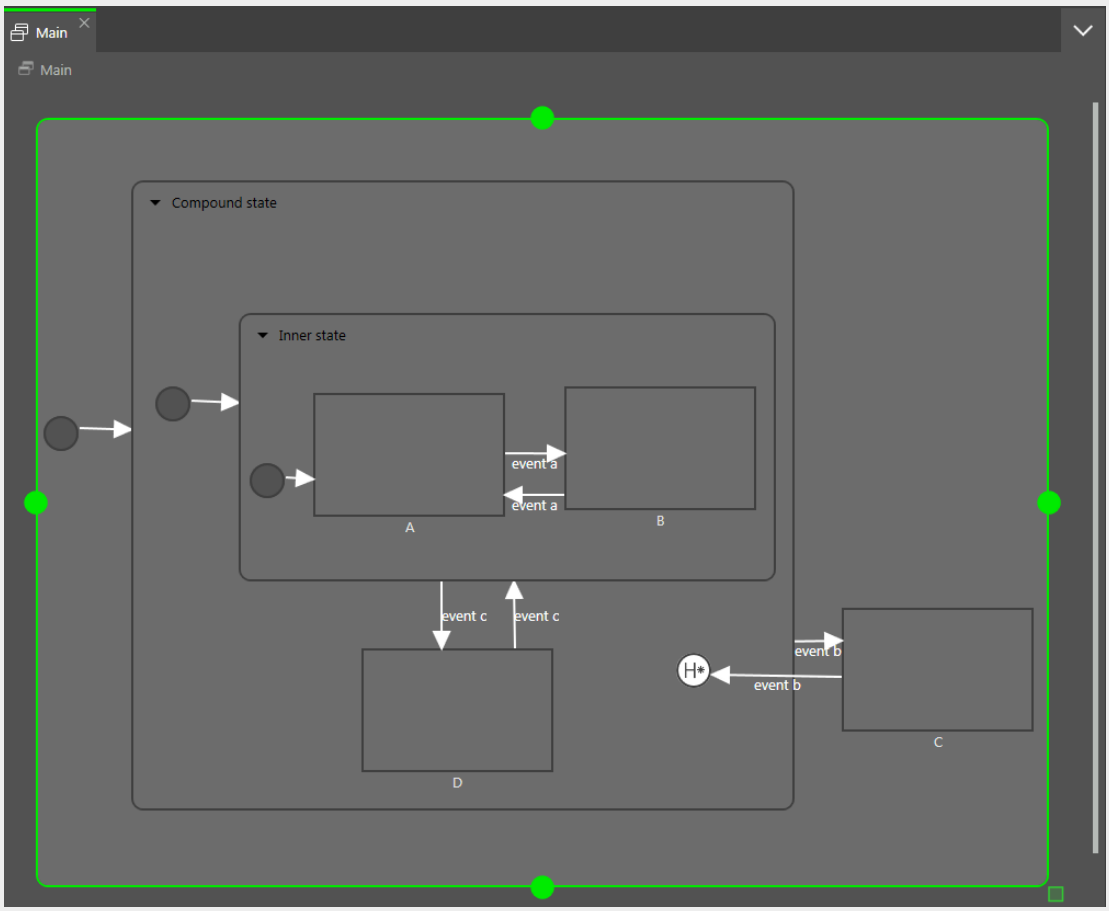

図6.16 深い履歴ステート

- ケース1: アクティブなステートは`D`です。
  1. `event b`が発行され、ステート`C`にエントリーします。
  2. `event b`が再び発行され、深い履歴ステートにエントリーします。
  3. ステートマシンは、深い履歴ステートからステート`D`にエントリーします。ステート`D`が`Compound State`内の最後のアクティブステートだったからです。
- ケース2: アクティブなステートは`B`です。
  1. `event b`が発行され、ステート`C`にエントリーします。

2. `event b`が再び発行され、深い履歴ステートにエントリーします。
3. ステートマシンは、深い履歴ステートからステート`B`にエントリーします。ステート`B`が最後のアクティブステートであり、深い履歴ステートはステート階層を記憶しているからです。

ステートには浅い履歴ステートと深い履歴ステートのどちらかを設定できます。親ステートに履歴ステートを設定した場合も、その子ステートに別の履歴ステートを設定できます。

6.15.3. 遷移

遷移は、ソースステートとターゲットステートを直接結ぶ関連付けです。ステートマシンを現在のステートから別のステートへと導きます。遷移には次のプロパティがあります。

- 遷移を実行するトリガー

  トリガーはイベント、またはデータプールアイテムの変更です。
- 遷移を実行するために`true`と評価される必要がある条件
- 遷移時に実行されるアクション

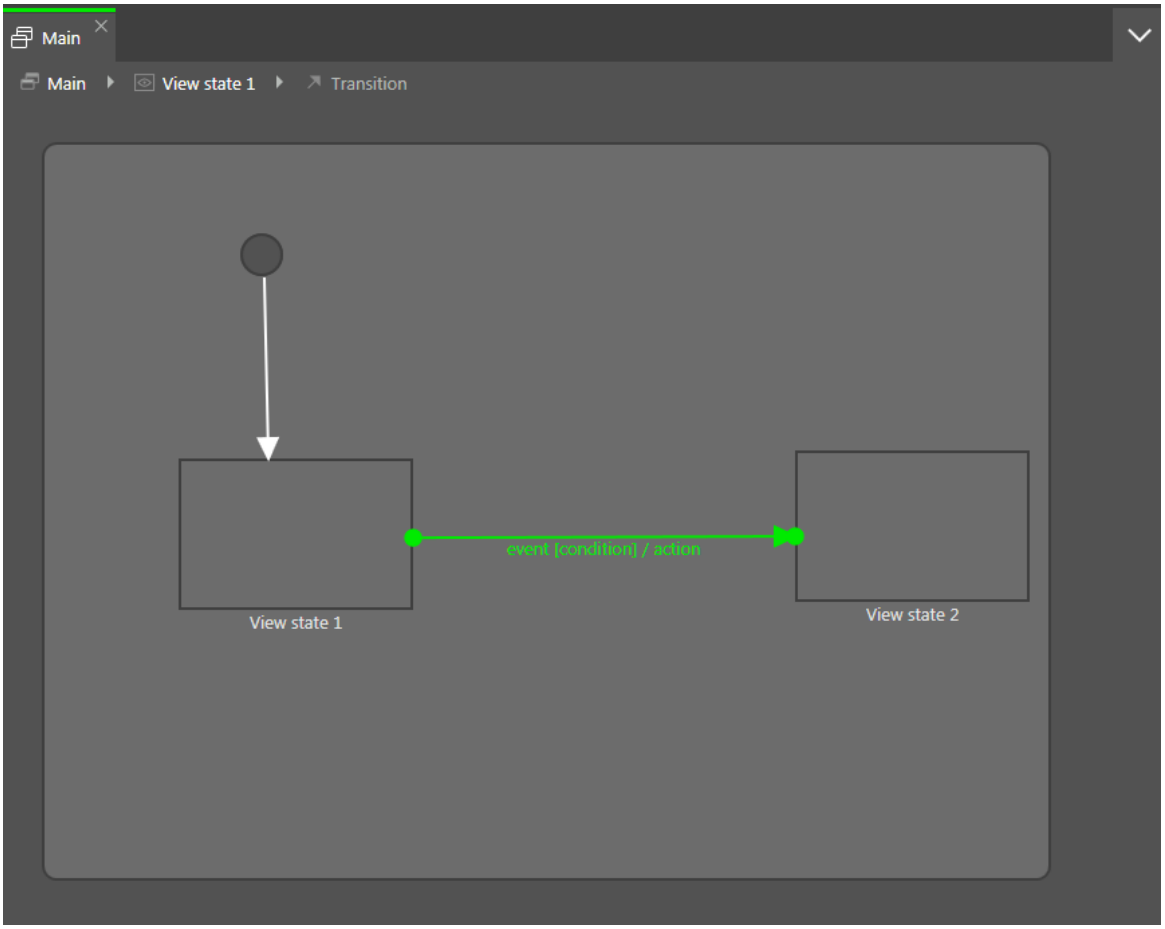

図6.17 遷移

注記

遷移の決定性

特定のソースステートから同じイベントに複数の遷移を設定することは、たとえ条件に違いがあるとしても不可能です。複数の条件を使ってステートマシンを複数の遷移先ステートにジャンプさせたい場合は、選択ステートを使ってください。

ステートは親ステートからすべての遷移を継承します。複数のステートが別のステートへの遷移を共有する場合、混合ステートを使って遷移をまとめると条件の数を減らせます。

例**6.25**

遷移の継承

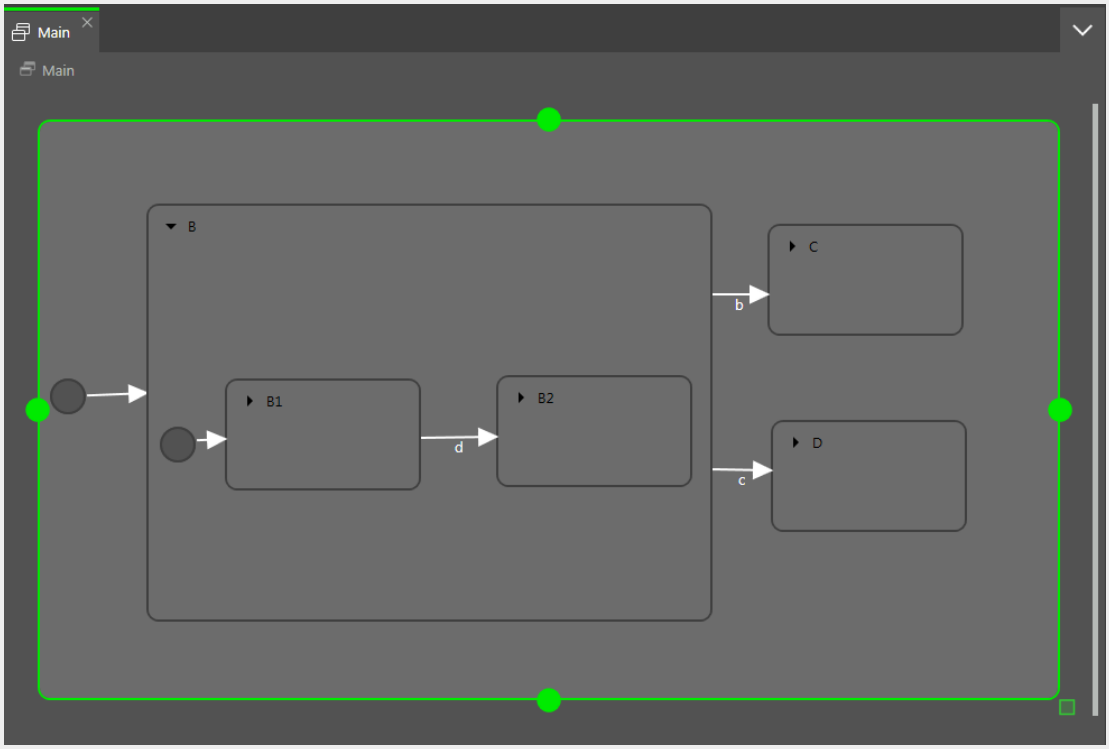

図6.18 遷移の継承

イベントbの発行時にステートマシンがState B1にある場合、State Cへの遷移が実行されます。子ステートState B1とState B2が、ステートState Bの遷移を継承するためです。

子ステートからの内部遷移に使われるイベントが、親ステートからの外部遷移のイベントと同じ場合、遷移の継承がオーバーライドされます。

例**6.26**

遷移のオーバーライド

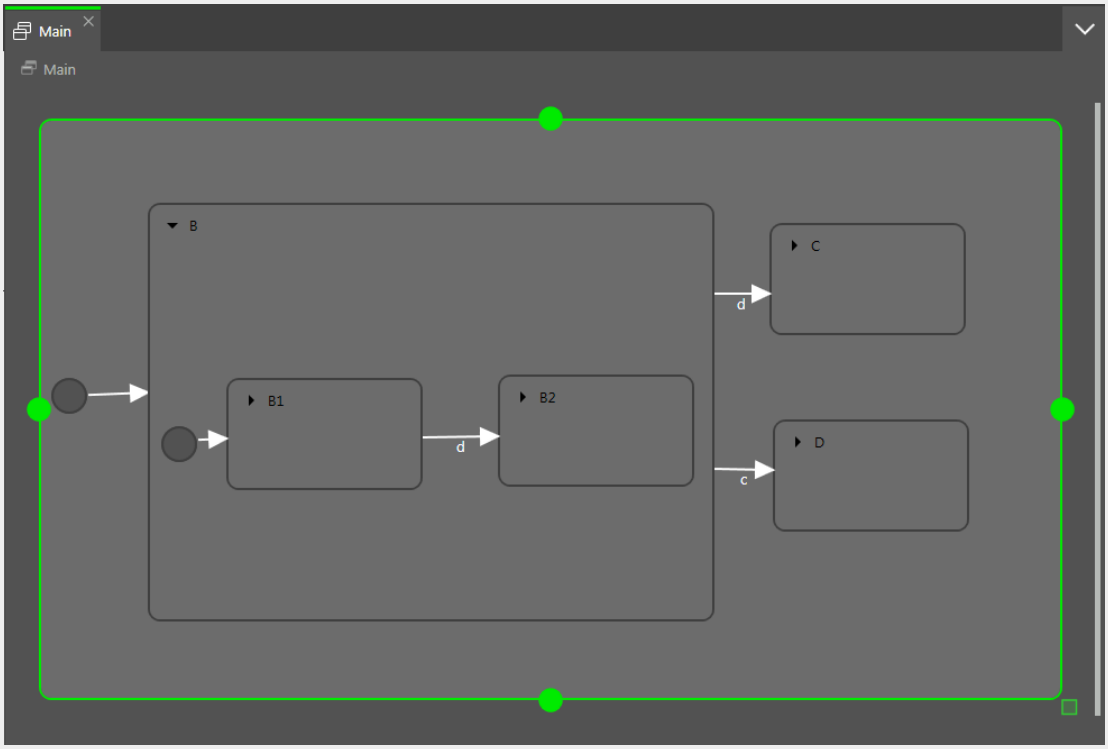

図6.19 遷移のオーバーライド

イベントdの発行時にステートマシンがState Bのステートにある場合、State Cへの遷移が実行されます。

イベントdの発行時にステートマシンがState B1のステートにある場合、State B2への遷移が実行されます(State Cには遷移しません)。2つの遷移の名前が同一であるため、内側の遷移が外側の遷移よりも優先されます。

注記

実行の階層

ステートマシンでは、同じイベントを使う遷移の実行階層は、常に内側から外側へ向かいます。つまり、内部遷移が外部遷移より優先されるということです。

遷移には次のタイプがあります。

- デフォルト遷移

  デフォルト遷移は自動的にトリガーされます。イベントやデータプールアイテムの更新ではトリガーされません。条件はありませんが、アクションを指定できます。この遷移は、初期ステート、最終ステート、選択ステート、履歴ステートで使用されます。

- 選択遷移

選択遷移は、与えられた条件によって外部へ遷移します。ソーススートは選択ステートです。選択遷移は、条件の評価によってトリガーされます。その結果としてアクションが実行されます。条件が`true`と評価された最初の選択遷移が実行されます。

- else遷移

  else遷移は、選択遷移と対になる必須の遷移です。選択ステートには1つのelse遷移が必要で、すべての選択遷移の条件が`false`と評価された場合にelse遷移が実行されます。

- 内部遷移

  内部遷移は、ターゲットステートを持たない遷移です。したがって、アクティブステートを変更しません。内部遷移の目的は、現在のステートに留まったままイベントに反応することです。この遷移に条件を指定し、その結果としてアクションを実行できます。

  ステートの同じイベントに対して複数の内部遷移を指定できます。実行の順序を定義します。

- 自己遷移

  自己遷移は、ソースステートとターゲットステートが同じステートである遷移です。内部遷移とは違い、自己遷移はステートを脱してから同じステートへ再度エントリーします。したがって、エントリーアクションと終了アクションが実行されます。

6.15.4. ステートマシンの実行

ステートマシンが実行されると、実行中は常に1つのアクティブステートが存在します。ステートマシンはイベント主導型です。

ステートマシンの動作サイクルは、次のようなものです。

1. ステートマシンは、初期ステートへエントリーすると開始されます。
2. ステートマシンは、イベントの発生を待ちます。
   a. 内部遷移が見つかります。
      i. 現在のステートを出発点に、現在のイベントでトリガーされ、条件が`true`と評価される最初の内部遷移を探します。該当する遷移が見つかったら、それが実行されます。
      ii. 遷移が見つからない場合は、親ステートに移動し、現在のイベントでトリガーされ、条件が`true`と評価される最初の内部遷移を探します。
      iii. 遷移が見つからなければ、最上位のステートに達するまで前のステップを繰り返します。
   b. 内部遷移が処理されます。

      内部遷移を実行すると、それに関連付けられたアクションのトリガーだけが行われます。ステートの終了と再エントリーは行われません。

c. 遷移が見つかります。

   i. 現在のステートを出発点に、現在のイベントでトリガーされ、条件が`true`と評価される最初の遷移を探します。該当する遷移が見つかったら、それが実行されます。

   ii. 遷移が見つからなければ、親ステートに移動し、遷移を探します。

   iii. 条件を満たす遷移が見つかるまで、前のステップを繰り返します。

d. 遷移が処理されます。

   遷移を実行すると、ステートマシンは現在のステートから別のステートに変わります。ソースステートが終了し、ターゲットステートへエントリーします。

   遷移が実行されるのは、対応するイベントが発生し、条件が`true`と評価された場合だけです。

   遷移は、ステート階層内の複数の混合ステートを終了またはエントリーできます。終了とエントリーの段階的な処理に挟まれる形で遷移のアクションが実行されます。

   ステートへエントリーするには、後続の遷移が必要です。例えば、混合ステートへエントリーする場合、初期ステートの遷移を後続の遷移として実行する必要があります。後続の遷移を連続して複数実行することが可能です。

3. 最終ステートに達した時点で、ステートマシンは停止します。

遷移がステート階層内で複数のステートを横断して使われる場合、段階的な終了とエントリーのアクションが実行されます。

例6.27

遷移の実行

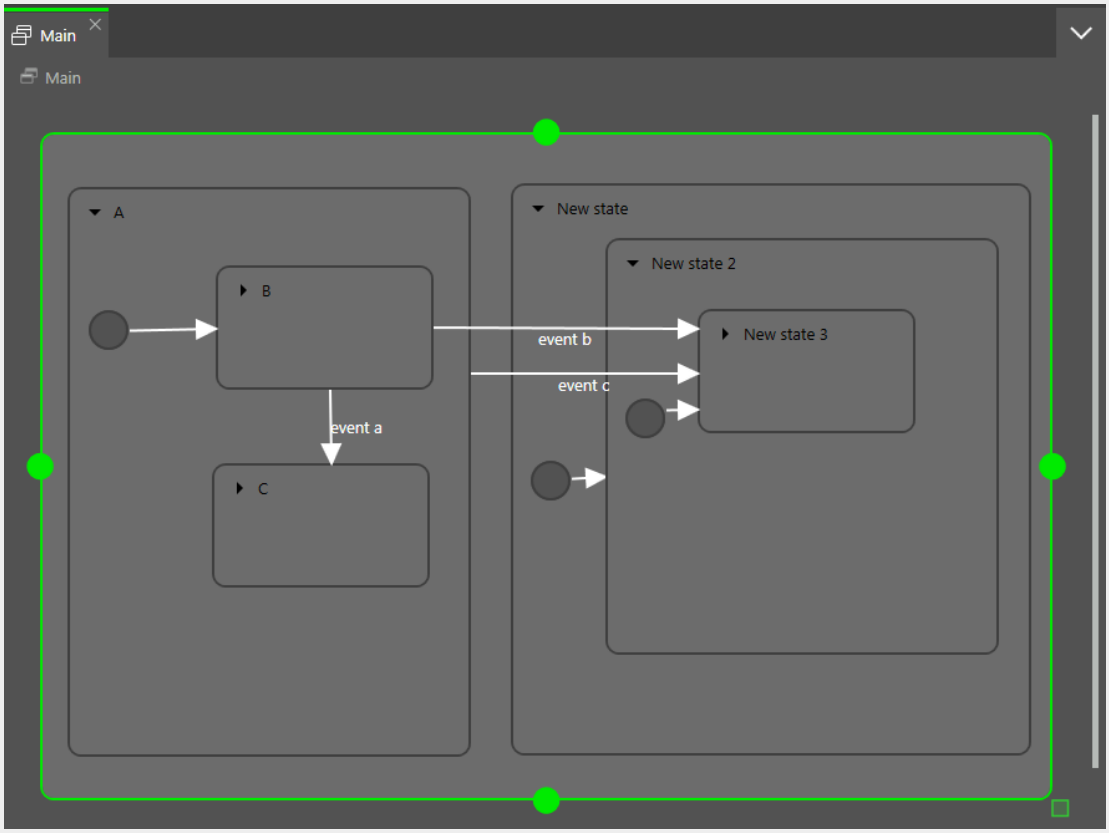

図6.20 遷移の実行

`event a`が発行されると、次のことが生じます。

1. ステート`B`が終了します。
2. ステート`C`が開始します。

`event b`が発行されると、次のことが生じます。

1. ステート`B`が終了します。
2. ステート`A`が終了します。
3. ステート`New state`が開始します。
4. ステート`New state 2`が開始します。
5. ステート`New state 3`が開始します。

`event c`が発行されると、次のことが生じます。

1. ステート`B`またはステート`C`がアクティブの場合は、ステート`B`またはステート`C`が終了します。
2. ステート`A`が終了します。

3. ステートNew stateが開始します。
4. ステートNew state 2が開始します。
5. ステートNew state 3が開始します。

例6.28
遷移の実行

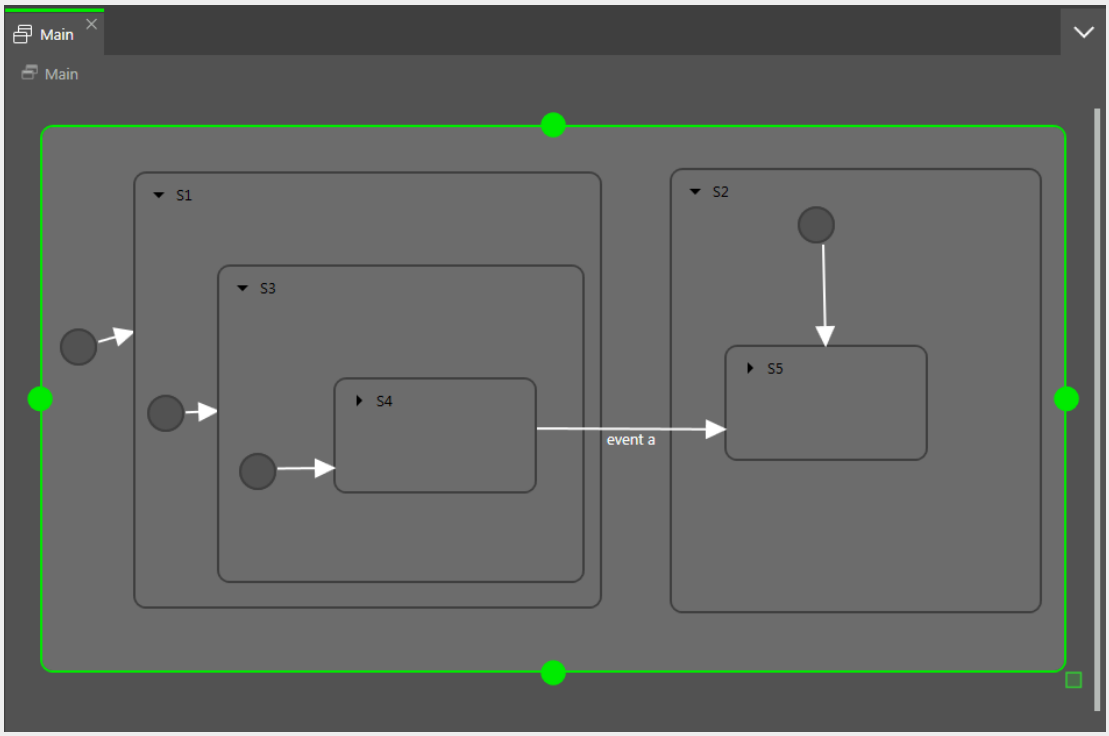

図6.21 遷移の実行

event aによって遷移がトリガーされた場合、次のことが生じます。

1. ステートS4が終了します。
2. ステートS3が終了します。
3. ステートS1が終了します。
4. ステートS2が開始します。
5. ステートS5が開始します。

例6.29

遷移の実行

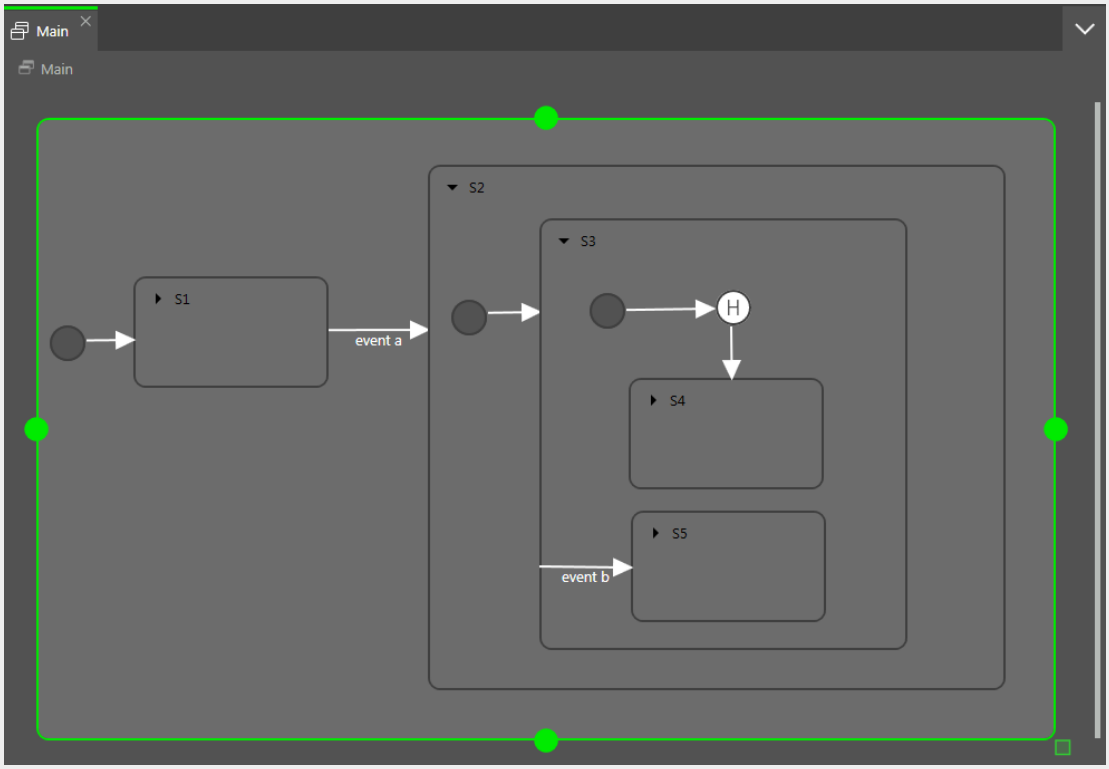

図6.22 遷移の実行

`event a`によって遷移がトリガーされると、次の遷移シーケンスが生じます。

1. ステートマシンはステート`S2`に移動します。
2. デフォルト遷移によってステート`S3`に遷移します。
3. 次のデフォルト遷移によって浅い履歴ステートにエントリーします。
4. 浅い履歴ステートは、ステート`S3`の最後のアクティブステートである、ステート`S4`またはステート`S5`を復帰します。

それぞれのステップで、開始-終了カスケードが個別に実行されます。

6.15.5. EB GUIDEの記法とUML記法の比較

ここではEB GUIDE記法を統一モデリング言語(Unified Modeling Language; UML) 2.5記法と比較して紹介します。

6.15.5.1. サポートされている要素

以下の表に、EB GUIDEでサポートされているすべてのUML 2.5要素を示します。一部の要素はUML 2.5の命名規則に従っていませんが、機能はまったく同じです。

EB GUIDEでの名前	**UML 2.5**での名前
初期ステート	初期(仮ステート)
最終ステート	最終ステート
混合ステート	ステート
選択ステート	選択(仮ステート)
深い履歴ステート	深い履歴(仮ステート)
浅い履歴ステート	浅い履歴(仮ステート)
内部遷移	内部遷移
遷移	外部/ローカル遷移 [a]

[a] EB GUIDEでは、外部遷移とローカル遷移を区別しません。

6.15.5.2. サポートされない要素

以下のUML 2.5要素はEB GUIDEでサポートされません。

- ジョイン
- フォーク
- 交差
- エントリーポイント
- イグジットポイント
- 停止

6.15.5.3. 偏差

UML 2.5記法の一部の要素はEB GUIDEに実装されていません。ただし、それらの機能はEB GUIDEの概念によってモデリングされています。

UML 2.5での概念	**EB GUIDE**での回避策
並行ステート	この概念は、動的ステートマシンを使って実装されています。
遷移ごとのトリガー数	この概念は、EB GUIDEスクリプトを使ってデータプールアイテムまたはビューとして実装されています。

UML 2.5での概念	**EB GUIDE**での回避策
遷移時のタイムトリガー	この概念は、EB GUIDEスクリプト(`fire_delayed`)を使ってステートマシン、データプールアイテム、遷移、またはビューとして実装されています。

6.16. タッチ入力

EB GUIDEでは、2種類のタッチ入力がサポートされています。タッチジェスチャーとマルチタッチ入力です。

各タッチジェスチャーは、EB GUIDE Studioにおいてウィジェット機能として表されます。ウィジェット機能を有効にすると、一連のプロパティがウィジェットに追加されます。

ジェスチャーは、次の2つの基本タイプに分類されます。

- 非パスジェスチャー
- パスジェスチャー

6.16.1. 非パスジェスチャー

EB GUIDEには、以下の非パスジェスチャーが実装されています。

- フリック
- ピンチ
- 回転
- ホールド
- ロングホールド

パスジェスチャーには、マルチタッチジェスチャーとシングルタッチジェスチャーがあります。マルチタッチジェスチャーには、マルチタッチ入力をサポートする入力デバイスが必要です。シングルタッチジェスチャーには、サポートされている任意の入力デバイスが使用できます。

各ジェスチャーは、互いに独立して反応します。複数のジェスチャーが有効である場合、モデラーはEB GUIDEの動作の一貫性を保証する必要があります。

6.16.2. パスジェスチャー

パスジェスチャーとは、タッチスクリーン上に指で描画されたか、その他の入力デバイスによって入力された形状のことです。ウィジェットに対してウィジェット機能が有効である場合、ユーザーはそのウィジェット上で開始する形状を入

力できます。設定可能な最小矩形よりも大きい形状でなければ、パスジェスチャーとしての認識対象にはなりません。入力形状が既知の形状集合とマッチングされ、マッチすればジェスチャーとして認識されます。

手順については、11.3「チュートリアル: パスジェスチャーをモデル化する」をご覧ください。

6.16.3. 入力処理とジェスチャー

ジェスチャー認識は、通常の入力処理と並行して実行されます。各ジェスチャーは、ジェスチャー関連のコンタクトを通常の入力処理から除外するように要求できます。ジェスチャーがコンタクトの除外を要求するタイミングは、実際のジェスチャーによって異なり、一部のジェスチャーに対してはこれを設定することが可能です。

コンタクトの除外は、ジェスチャーを行う指に対してのみ適用されます。コンタクトが除外されると、そのコンタクトに対するリリースイベントが受信されるまで、通常の入力処理においてそれが無視されます。つまり、近接性をサポートしないタッチスクリーン上では、除外されたコンタクトによってその他のタッチ反応がトリガーされることはありません。

ティップ

通常の入力処理からのコンタクトの除外

ボタンとウィジェット機能を備えるウィンドウに対して、ジェスチャーを行う場合を考えます。コンタクトがジェスチャー関連のものである場合、そのコンタクトがボタン上でリリースされたとしても、それによってボタンに関連付けられたアクションがトリガーされることがあってはいけません。

6.16.4. マルチタッチ入力

EB GUIDEでは、互換性のあるマルチタッチ入力デバイスが使用されれば、マルチタッチ入力を処理することができます。

マルチタッチとは、入力デバイスの2点以上のコンタクトを画面上で認識してトラッキングする機能のことです。典型的な例としては、タッチスクリーンを複数の指でタッチする場合が挙げられます。

- マルチタッチイベントの処理

  マルチタッチイベントは、マウスやシングルタッチのタッチスクリーンからのイベントと同じように、タッチイベントのメカニズムを使用してディスパッチされます。唯一の相違点は、各コンタクトが互いに独立してタッチ反応をトリガーすることです。個々のコンタクトを区別できるように、各タッチ反応に`fingerid`というパラメータが与えられます。
- Finger ID

  入力デバイスによってトラッキングされる各コンタクトには、それを識別する番号が割り当てられます。この識別子は`fingerid`と呼ばれ、入力デバイスごとに一意です。ただし、もう使用されていない同じ値が、後で別のコンタクトに割り当てられることはあります。

マルチタッチ入力が有効である場合に、エンドユーザーが入力可能なその他のタッチ操作シーケンスについて説明します。そのようなシーケンスとしては、以下のものがあります。

- エンドユーザーは、リストをスクロールしながらボタンを押すなど、インターフェイスの複数の要素を同時に操作することができます。
- エンドユーザーは、1つのウィジェット上に複数の指を置くことができます。

これが生じる2つの典型的な操作は、スクロールとドラッグです。`fingerid`を使用することによって、これらの操作を正しく処理することができます。求められる動作によって、考えられるソリューションとしては以下のようなものがあります。

- ウィジェットを最初に押した指のみに、スクロールとドラッグの操作を許可する。
- 必ず最後にウィジェット上に置かれた指によって、スクロールとドラッグの操作を行う。上の方法を少し変更するだけで、簡単にこれが実現できます。

6.17. ウィジェット

ウィジェットは、EB GUIDEモデルを構成する基本的なグラフィカル要素です。

ウィジェットはカスタマイズできます。ウィジェットのプロパティを編集すれば、ニーズに合ったウィジェットに変更できます。サイズ、色、レイアウトのほか、タッチ時や移動時の動作といったプロパティがあります。

ウィジェットは組み合わせて使えます。小さなブロックを積み重ねて複雑な構造物を作成できます。例えば、ボタンは四角形、イメージ、ラベルから成り立っています。

ウィジェットは入れ子にすることができます。ウィジェット階層では、下位のウィジェットを「子ウィジェット」と呼び、上位のウィジェットを「親ウィジェット」と呼びます。

6.17.1. ビュー

ビューは、シーンの最上位ウィジェットです。モデリング時には、基本ウィジェット、3Dウィジェット、アニメーション、ウィジェットテンプレートがビュー内に配置されます。ビューは、1つのビューステートに関連付けられます。ビューは、ビューステートなしで存在できません。

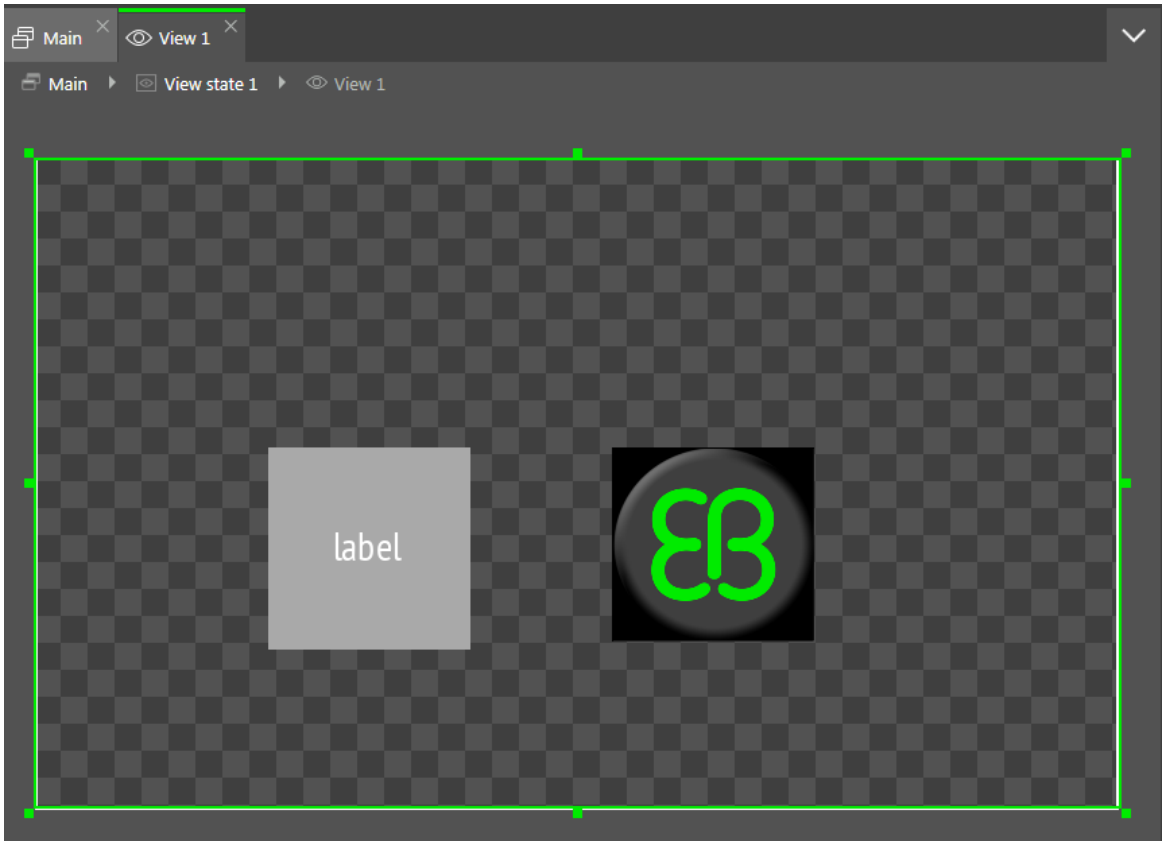

図6.23 四角形、ラベル、イメージを含むビュー

6.17.2. ウィジェットのカテゴリ

[ツールボックス]内で、ウィジェットはカテゴリによって分類されています。以下のカテゴリがあります。

- 基本ウィジェット

  基本ウィジェットにはラベル、四角形、コンテナー、イメージ、インスタンシエータがあります。

- アニメーション

  [アニメーション]カテゴリは、アニメーションと、アニメーションの詳細を指定するための一連の曲線を提供します。それぞれの曲線には、サポートされているデータタイプごとに1つのウィジェットが存在します。

- 3Dウィジェット

  [3Dウィジェット]カテゴリには、3Dグラフィックを表示するウィジェットが含まれます。3Dウィジェットにはシーングラフ、シーングラフノード、材質、メッシュ、カメラ、指向性ライト、点ライト、スポットライトがあります。

注記

サポートされているレンダラー
3Dグラフィックを表示するには、OpenGL ES 2.0またはDirectX 11のレンダラーが必要です。

- ウィジェットテンプレート

  [テンプレート]カテゴリには、ウィジェットテンプレートが含まれます。このカテゴリは、ウィジェットテンプレートが定義されている場合にのみ表示されます。

6.17.3. ウィジェットプロパティ

ウィジェットは、外観や動作を指定する一連のプロパティによって定義されます。[プロパティ]パネルには、現在フォーカスがあるウィジェットのプロパティが表示され、プロパティを編集する機能が提供されています。

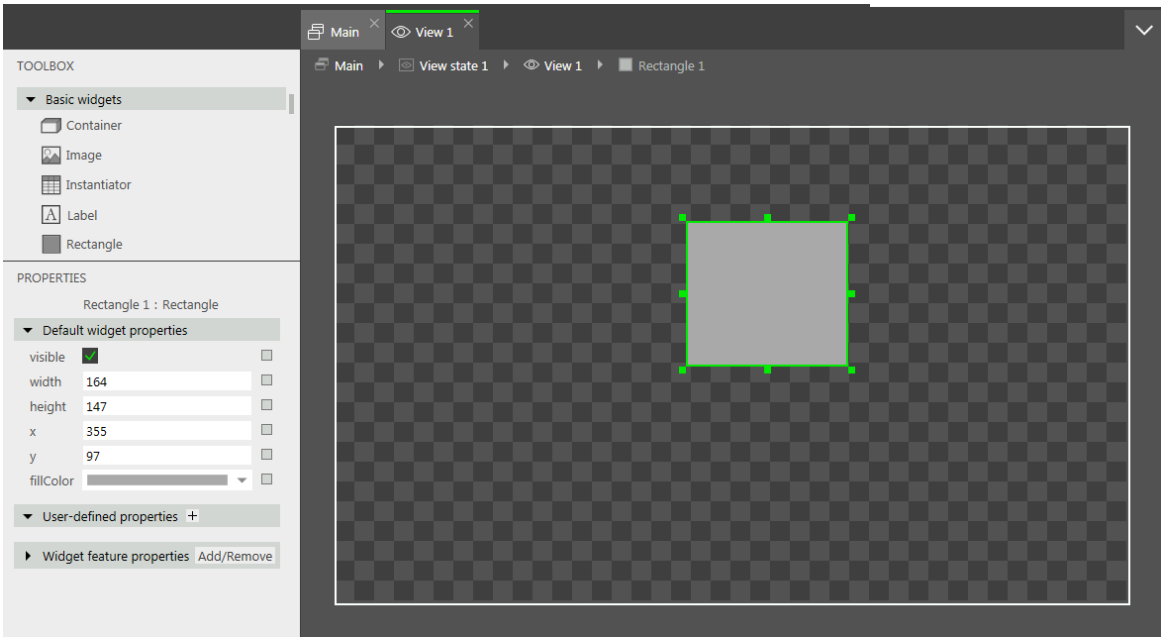

図6.24 四角形とそのプロパティ

ウィジェットプロパティには3つのタイプがあります。

- デフォルトウィジェットプロパティは、ウィジェットインスタンスと共に作成されます。すべてのウィジェットのデフォルトプロパティについては、12.9「ウィジェット」をご覧ください。
- ユーザー定義ウィジェットプロパティは、デフォルトプロパティに加えて、モデラーが作成するプロパティです。
- ウィジェット機能プロパティは、モデラーがウィジェット機能をウィジェットに追加するときに、EB GUIDE Studioによって作成されます。ウィジェット機能プロパティは、カテゴリに分かれています。ウィジェット機能は、ウィジェットの外観と動作に関して別の機能を追加します。

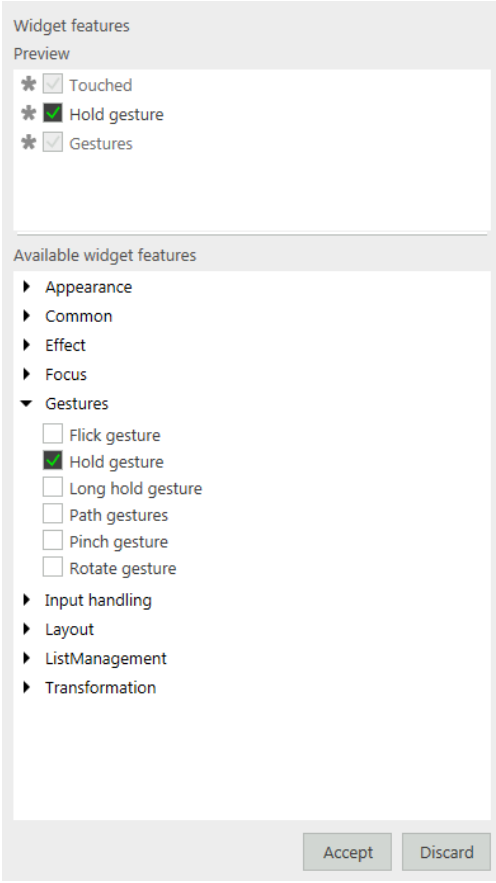

図6.25 ウィジェット機能

例6.30
[タッチ]ウィジェット機能

[タッチ]ウィジェット機能は、ウィジェットがタッチに反応するか、反応する場合はどう反応するかを定義します。プロパティは4つ追加されます。ブール値プロパティ`touchable`は、ウィジェットがタッチ入力に反応するかどうかを定義します。ブール値プロパティ`touched`は、ウィジェットが現在タッチされている場合、EB GUIDEによってランタイムに設定されます。2つの整数プロパティ、`touchPolicy`および`touchBehavior`は、ウィジェットがタッチ入力にどう反応するかを定義します。

6.17.4. ウィジェットテンプレート

ウィジェットテンプレートにより、EB GUIDEモデルで何度も使用できるようにカスタマイズされたウィジェットを定義できます。既存のウィジェットに基づいてテンプレートを定義したり、既存のテンプレートから新しいテンプレートを派生させたりすることができます。テンプレートを作成した後は、必要に応じて、例えばプロパティやウィジェット機能の追加などで変更します。このため、ウィジェットテンプレートでは、複雑なウィジェットのライブラリを構築できます。

ウィジェットテンプレートは、テンプレートインターフェイスを備えています。テンプレートインターフェイスにはテンプレートのプロパティが含まれます。これらのプロパティは、ウィジェットインスタンスで表示およびアクセスが可能です。こうして、ウィジェットインスタンスはそのテンプレートのインターフェイスからプロパティを継承します。継承されたプロパティは、テンプレートプロパティと呼ばれます。テンプレートプロパティは■ボタンでマークされます。

テンプレートプロパティの値を変更すると、そのプロパティはローカルプロパティに変わります。ローカルプロパティは■ボタンでマークされます。

例6.31
ウィジェットテンプレートのプロパティとそのインスタンスの関係

`Square`ウィジェットテンプレートをEB GUIDEモデルに追加します。`Square`に、`color`というプロパティを追加します。`color`がテンプレートインターフェイスに追加されます。`color`の値を`red`に設定します。

`Square`ウィジェットテンプレートのインスタンスをビューに追加します。インスタンスは`BlueSquare`という名前にします。

- `BlueSquare`には、`color`が値`red`で継承されます。
- `Square`テンプレートで`color`の値を`green`に変更します。

  => `BlueSquare`でも`color`の値が`green`に変わります。
- `BlueSquare`で`color`の値を`blue`に変更します。

  `Square`テンプレートで`color`の値を`yellow`に変更します。

  => `BlueSquare`の`color`値は`blue`のままです。

手順については、8.6「ウィジェットの再利用」をご覧ください。

7. ヒューマンマシンインターフェイスの動作のモデル化

7.1. ステートマシンのモデリング

7.1.1. ステートマシンの追加

ステートマシンの追加

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで[ステートマシン]に移動し、⊞をクリックします。

メニューが展開されます。

ステップ 2
ステートマシンのタイプをクリックします。

選択したタイプの新しいステートマシンが追加されます。

ステップ 3
ステートマシンの名前を変更します。

7.1.2. 動的ステートマシンの追加

動的ステートマシンは他のステートマシンと平行して動作し、ランタイムの間に開始(プッシュ)したり終了(ポップ)したりできます。

動的ステートマシンの追加

動的ステートマシンは、例えば通常の画面をオーバーレイしてエラーメッセージを表示する場合に使用します。

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- ステートマシンがあり、ビューステートまたは混合ステートがEB GUIDEモデルに追加されていること。

ステップ 1
ナビゲーションエリアで[動的ステートマシン]に移動し、⊞をクリックします。

メニューが展開されます。

ステップ 2
動的ステートマシンのタイプをクリックします。

選択したタイプの新しい動的ステートマシンが追加されます。

ステップ 3
ナビゲーションエリアで、動的ステートマシンと平行して動作させるステートマシン、ビューステート、または混合ステートをクリックします。

ステップ 4
[プロパティ]パネルで、`Dynamic state machine list`チェックボックスを選択します。

以上の操作が終了したら、動的ステートマシンに関連付けられたEB GUIDEスクリプト関数を使用します。

詳細については、11.1「チュートリアル: 動的ステートマシンの追加」をご覧ください。

7.1.3. ステートマシンに対するエントリーアクションの定義

ステートマシンに対するエントリーアクションの定義

ステップ 1
ステートマシンを選択します。

ステップ 2
[プロパティ]パネルで[エントリーアクション]プロパティに移動し、[追加]をクリックします。

ステップ 3
EB GUIDEスクリプトを使用してアクションを入力します。

バックグラウンド情報については、6.12「スクリプト言語EB GUIDEスクリプト」をご覧ください。

ステップ 4
[承認]をクリックします。

ステートマシンに対するエントリーアクションが定義されました。

7.1.4. ステートマシンに対する終了アクションの定義

ステートマシンに対する終了アクションの定義

ステップ 1
ステートマシンを選択します。

ステップ 2
[プロパティ]パネルで`Exit action`プロパティに移動し、[追加]をクリックします。

ステップ 3
EB GUIDEスクリプトを使用してアクションを入力します。

バックグラウンド情報については、6.12「スクリプト言語EB GUIDEスクリプト」をご覧ください。

ステップ 4
[承認]をクリックします。

ステートマシンに対する終了アクションが定義されました。

7.1.5. ステートマシンの削除

ステートマシンの削除

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで、ステートマシンを右クリックします。

ステップ 2
コンテキストメニューの[削除]をクリックします。

ステートマシンが削除されました。

7.2. モデリングステート

7.2.1. ステートの追加

ステートの追加

前提条件:

- コンテンツエリアにステートマシンが表示されていること。

ステップ 1
[ツールボックス]からステートをドラッグし、ステートマシンにドロップします。

ステートがステートマシンに追加されます。

注記

初期ステートと最終ステートはそれぞれ1つずつ
初期ステートと最終ステートは、1つの混合ステートに1回だけ挿入できます。

ティップ

ステートのコピーと検索
既存のステートをコピーして貼り付けるには、コンテキストメニューまたはCtrl+CキーとCtrl+Vキーを使用するという方法もあります。

EB GUIDEモデル内で特定のステートを検索するには、ステートの名前を検索ボックスに入力するか、Ctrl+Fキーを使用します。ステートにジャンプするには、ヒットリスト内のステートをダブルクリックします。

7.2.2. 混合ステートへのステートの追加

混合ステートへのステートの追加

ステート階層を作成するには、ステートを別のステートの子として作成します。混合ステートにステートを追加することによって、これを行います。

前提条件:

- コンテンツエリアにステートマシンが表示されていること。
- ステートマシンには混合ステートが含まれていること。

ステップ 1
コンテンツエリアで▶をクリックして、コンパウンドステートを展開します。

ステップ 2
[ツールボックス]からステートをドラッグし、混合ステートにドロップします。

ステートは、混合ステートの子ステートとして追加されます。

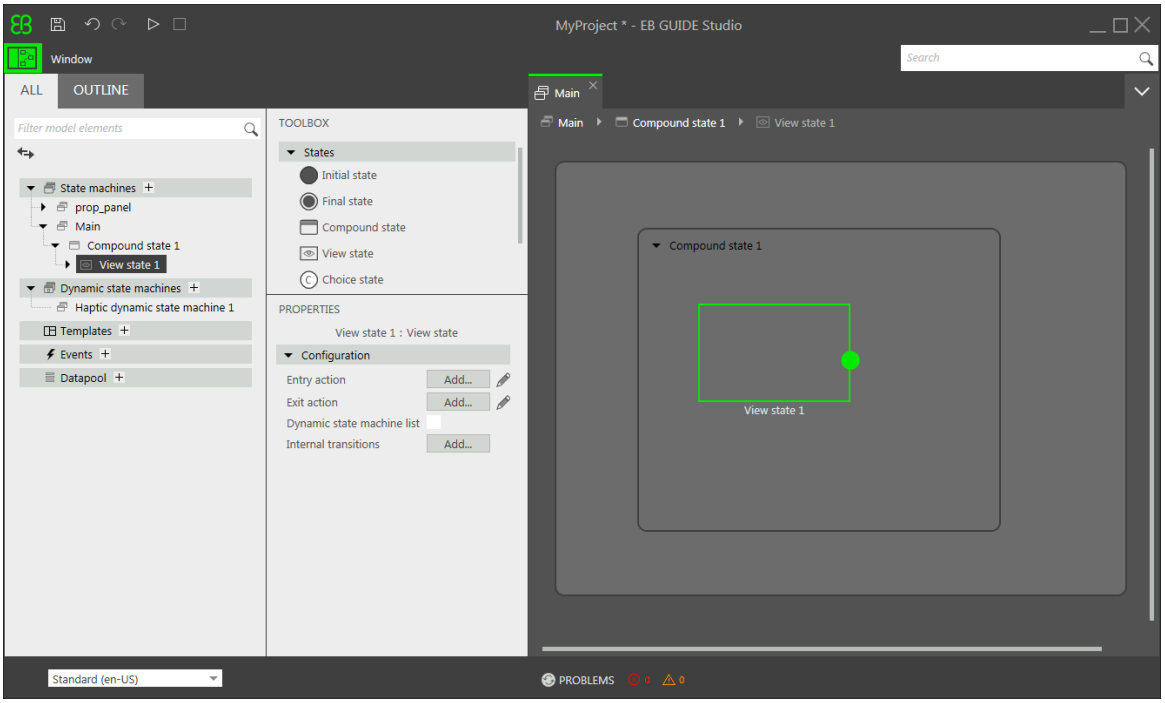

図7.1 入れ子のビューステートを持つ混合ステート

7.2.3. 選択ステートの追加

選択ステートの追加

前提条件:

- コンテンツエリアにステートマシンが表示されていること。
- ステートマシンには、少なくとも2つのステートが含まれています。

ステップ 1
[ツールボックス]から選択ステートをドラッグし、ステートマシンにドロップします。

ステップ 2
選択ステートからの出力遷移を追加します。

ステップ 3
出力遷移に条件を追加します。詳細については、7.3.4「遷移への条件の追加」をご覧ください

この条件には優先度1が割り当てられます。ステートマシンが選択ステートにエントリーすると、優先度1の条件が最初に評価されます。

ステップ 4
選択遷移をさらに追加するには、上記の2つのステップを繰り返します。

新しい選択遷移には、以前に作成された遷移よりも低い優先度が割り当てられます。

ステップ 5
選択ステートからの出力遷移を追加します。

ステップ 6
ナビゲーションエリアで、遷移を右クリックします。コンテキストメニューの[elseに変換]をクリックします。

else遷移が追加されました。else遷移は、外へ向かう選択遷移に割り当てられているすべての条件が`false`と評価されたときに実行されます。

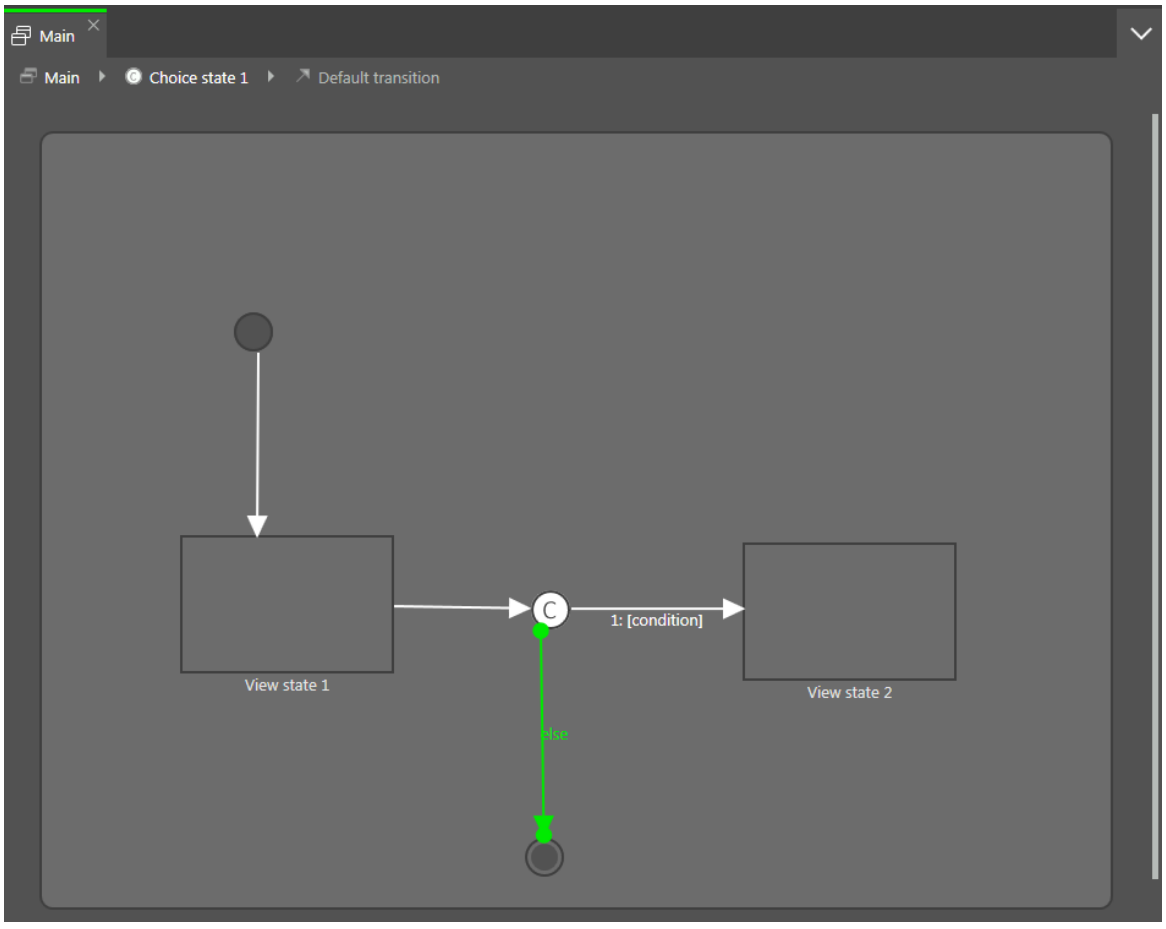

図7.2 選択遷移を持つ選択ステート

7.2.4. ステートに対するエントリーアクションの定義

ステートに対するエントリーアクションの定義

ビューステートと混合ステートに対し、エントリーアクションを定義することができます。エントリーアクションは、そのステートの開始時に必ず実行されます。

前提条件:

- ステートマシンにビューステートまたは混合ステートが含まれていること。

ステップ 1
ステートを選択します。

ステップ 2
[プロパティ]パネルで`Entry action`プロパティに移動し、[追加]をクリックします。

ステップ 3
EB GUIDEスクリプトを使用してアクションを入力します。

バックグラウンド情報については、6.12「スクリプト言語EB GUIDEスクリプト」をご覧ください。

ステップ 4
[承認]をクリックします。

7.2.5. ステートに対する終了アクションの定義

ステートに対する終了アクションの定義

ビューステートおよび混合ステートに対し、終了アクションを定義することができます。終了アクションは、そのステートの終了時に必ず実行されます。

前提条件:

- ステートマシンにビューステートまたは混合ステートが含まれていること。

ステップ 1
ステートを選択します。

ステップ 2
[プロパティ]パネルで`Exit action`プロパティに移動し、[追加]をクリックします。

ステップ 3
EB GUIDEスクリプトを使用してアクションを入力します。

バックグラウンド情報については、6.12「スクリプト言語EB GUIDEスクリプト」をご覧ください。

ステップ 4
[承認]をクリックします。

7.2.6. ステートマシンからのモデル要素の削除

ステートマシンからのモデル要素の削除

前提条件:

- ステートマシンに、少なくとも1つのモデル要素が含まれていること。

ステップ 1
ナビゲーションエリアで、モデル要素を右クリックします。

ステップ 2
コンテキストメニューの[削除]をクリックします。

モデル要素が削除されます。

7.3. ステート間を遷移で接続

7.3.1. 2つのステート間に遷移を追加

2つのステート間に遷移を追加

遷移によって、ソースステートとターゲットステートを接続します。

前提条件:

- コンテンツエリアにステートマシンが表示されていること。
- ステートマシンには、少なくとも2つのステートが含まれています。

ステップ 1
遷移のソースステートとなるステートを選択します。

ステップ 2
緑色のドラッグ点をクリックし、マウスボタンを押したままにします。

ステップ 3
ターゲットステートまでマウスをドラッグします。

ステップ 4
ターゲットステートが緑色で強調表示されたら、マウスボタンを離します。

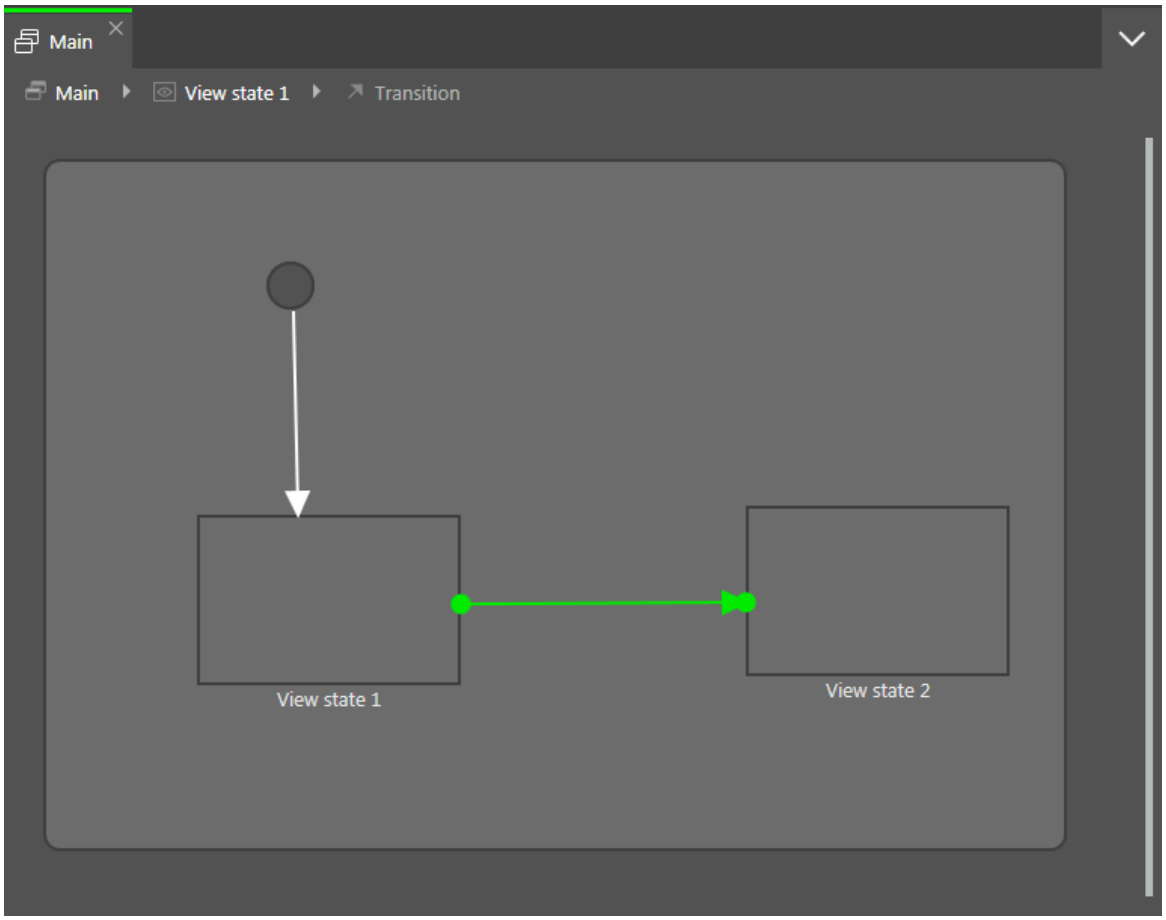

図7.3 遷移

遷移が追加され、緑色の矢印として表示されます。

ティップ

ステートマシンに遷移を接続

ステートマシンは、最上層の混合ステートです。したがって、ステートマシンの枠に対して入出力する遷移を作成できます。そのような遷移は、ステートマシン内のすべてのステートに継承されます。

7.3.2. 遷移の移動

遷移の移動

遷移の移動は、一方の終点を動かすことによって行います。

前提条件:

- コンテンツエリアにステートマシンが表示されていること。
- ステートマシンには、少なくとも2つのステートが含まれています。
- ステート間は遷移によって接続されていること。

ステップ 1
コンテンツエリアで、遷移をクリックします。

2つの緑色のドラッグ点が表示されます。

ステップ 2
移動させるドラッグ点をクリックし、マウスボタンを押したままにします。

ステップ 3
異なるステートまでマウスをドラッグします。

ステップ 4
ステートが緑色で強調表示されたら、マウスボタンを離します。

すると、遷移が移動します。

7.3.3. 遷移に対するトリガーの定義

遷移に対するトリガーの定義

遷移に対し、それをトリガーするイベントを定義できます。

前提条件:

- ステートマシンに少なくとも2つのステートが含まれていること。
- ステート間は遷移によって接続されていること。

ステップ 1
遷移をクリックします。

ステップ 2
[プロパティ]パネルで、[トリガー]コンボボックスを展開します。

ステップ 3
イベントをクリックします。

ステップ 4
新しいイベントを作成する場合は、[トリガー]コンボボックスに名前を入力し、[イベントの追加]をクリックします。

イベントが、遷移トリガーとして追加されます。

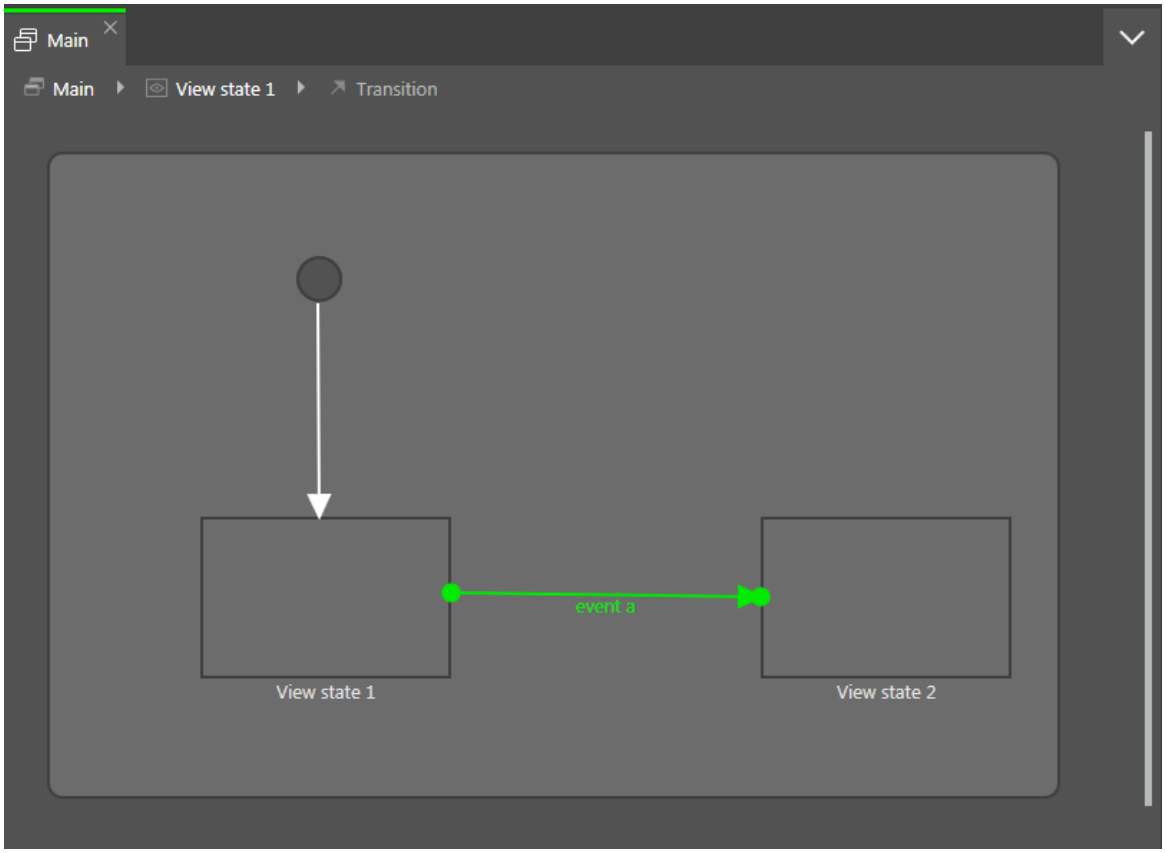

図7.4 トリガー付きの遷移

7.3.4. 遷移への条件の追加

遷移への条件の追加

各遷移に対し、その遷移を実行するために満たすべき条件を定義することができます。

前提条件:

- ステートマシンに少なくとも2つのステートが含まれていること。

- ステート間は遷移によって接続されていること。

ステップ 1
遷移をクリックします。

ステップ 2
遷移に条件を追加するには、[プロパティ]パネルに移動します。`Condition`プロパティの横にある[追加]をクリックします。

ステップ 3
EB GUIDEスクリプトを使用して条件を入力します。

バックグラウンド情報については、6.12「スクリプト言語EB GUIDEスクリプト」をご覧ください。

ステップ 4
[承認]をクリックします。

条件が遷移に追加されます。

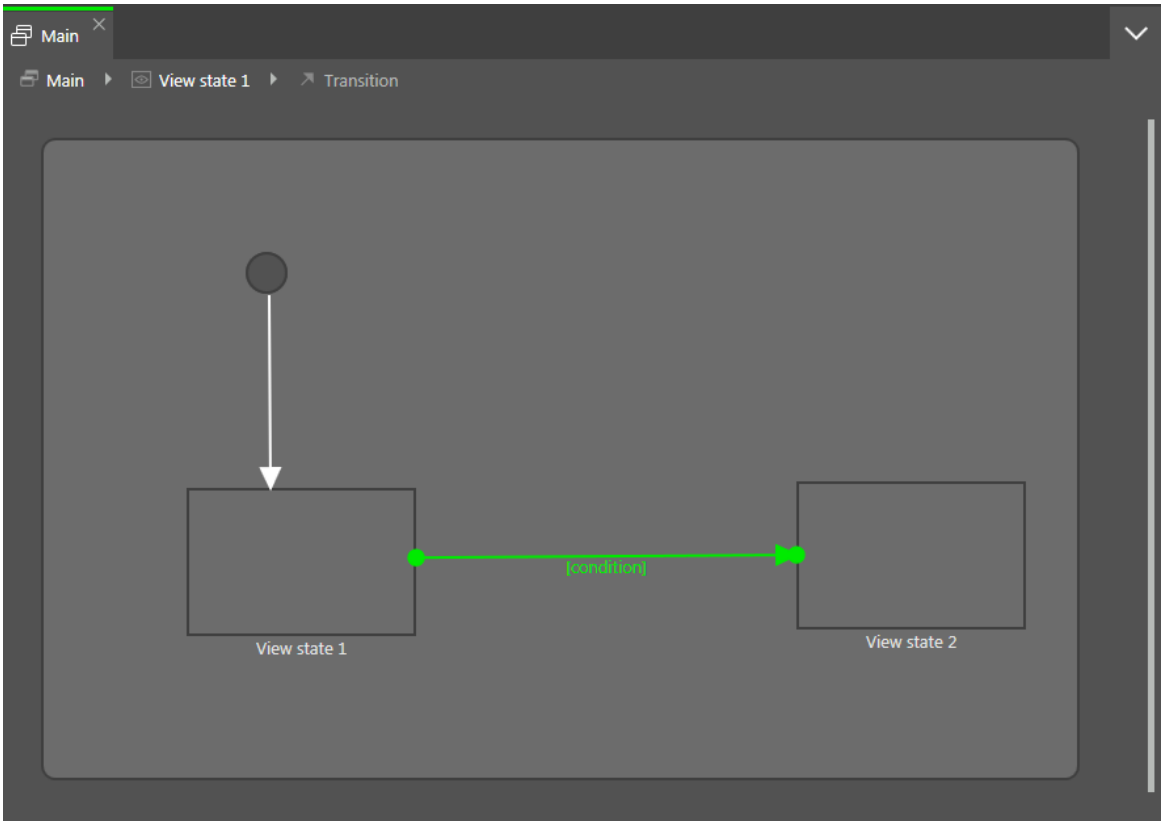

図7.5 条件付きの遷移

7.3.5. 遷移へのアクションの追加

遷移へのアクションの追加

各遷移に対し、その遷移時に実行されるアクションを定義することができます。

前提条件:

- ステートマシンに少なくとも2つのステートが含まれていること。
- ステート間は遷移によって接続されていること。

ステップ 1
遷移をクリックします。

ステップ 2
遷移にアクションを追加するには、[プロパティ]パネルに移動します。`Action`プロパティの横にある[追加]をクリックします。

ステップ 3
EB GUIDEスクリプトを使用してアクションを入力します。

バックグラウンド情報については、6.12「スクリプト言語EB GUIDEスクリプト」をご覧ください。

ステップ 4
[承認]をクリックします。

アクションが遷移に追加されます。

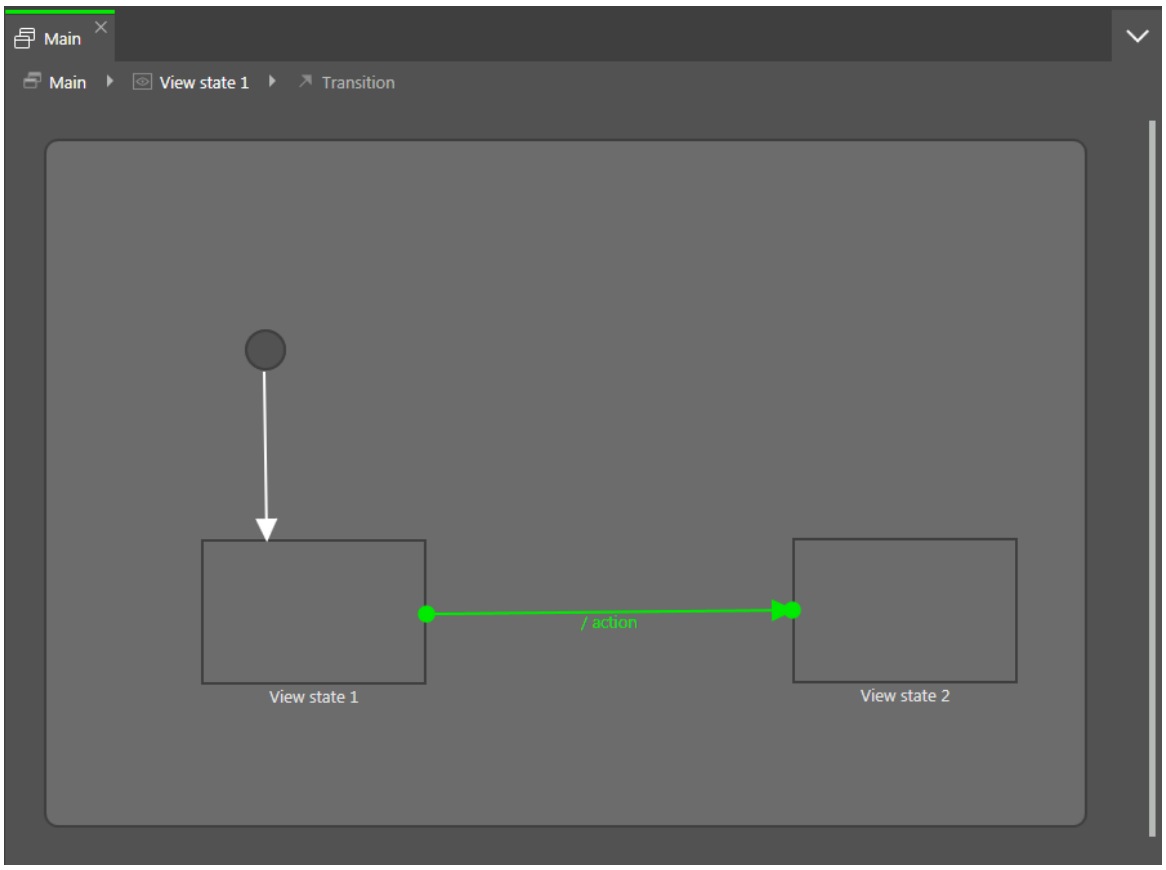

図7.6 アクション付きの遷移

7.3.6. ステートへの内部遷移の追加

ステートへの内部遷移の追加

前提条件:

- ステートマシンにステートが含まれていること。

ステップ 1
ステートを選択します。

ステップ 2
[プロパティ]パネルで、[内部遷移]に移動し、[追加]をクリックします。

内部遷移がステートに追加されます。内部遷移がナビゲーションエリアに表示されます。

8. ヒューマンマシンインターフェイスの外観のモデル化

8.1. ウィジェットの操作

8.1.1. ビューの追加

ビューの追加

前提条件:

- コンテンツエリアにステートマシンが表示されていること。

ステップ 1
[ツールボックス]からビューステートをドラッグし、ステートマシンにドロップします。

ビューステートと共にビューがモデルに追加されます。

ステップ 2
ナビゲーションエリアで、ビューをクリックします。

ステップ 3
F2キーを押し、ビューの名前を変更します。

ステップ 4
コンテンツエリアでビューステートをダブルクリックします。

コンテンツエリアに新しいビューが表示されます。

ティップ

ビューのコピーと検索

既存のビューをコピーして貼り付けるには、コンテキストメニューまたはCtrl+Cキーと Ctrl +Vキーを使用するという方法もあります。

EB GUIDEモデル内で特定のビューを検索するには、ビューの名前を検索ボックスに入力するか、Ctrl +Fキーを使用します。ビューにジャンプするには、ヒットリスト内のビューをダブルクリックします。

8.1.2. ビューへのウィジェットの追加

ビューへのウィジェットの追加

前提条件:

- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]からウィジェットをビューにドラッグしてドロップします。

ウィジェットがビューに追加されます。

ティップ

ウィジェットのコピーと検索

既存のウィジェットをコピーして貼り付けるには、コンテキストメニューまたはCtrl+Cキーと Ctrl +Vキーを使用するという方法もあります。

EB GUIDEモデル内で特定のウィジェットを検索するには、ウィジェットの名前を検索ボックスに入力するか、Ctrl +Fキーを使用します。ウィジェットにジャンプするには、ヒットリスト内のウィジェットをダブルクリックします。

8.1.3. ビューからのウィジェットの削除

ビューからのウィジェットの削除

前提条件:

- EB GUIDEモデルにウィジェットが含まれていること。

ステップ 1
ナビゲーションエリアで、ウィジェットを右クリックします。

ステップ 2
コンテキストメニューの[削除]をクリックします。

ウィジェットが削除されます。

ティップ

コンテンツエリアからのウィジェットの削除
コンテンツエリアでウィジェットを選択してDeleteキーを押すことでも、ウィジェットを削除できます。

8.1.4. ビューへのイメージの追加

ビューへのイメージの追加

前提条件:

- イメージファイルは、`$GUIDE_PROJECT_PATH\resources`ディレクトリに配置します。サポートされているファイルタイプについては、12.9.2.3「イメージ」をご覧ください。
- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]からイメージをビューにドラッグしてドロップします。

ステップ 2
[プロパティ]パネルで、`image`ドロップダウンリストボックスからイメージを選択します。

ビューにイメージが表示されます。

8.1.5. ビューへのシーングラフの追加

ビューへのシーングラフの追加

制限事項と推奨事項については、6.1.2「3Dグラフィックファイルの設定」をご覧ください。

前提条件:

- 3Dグラフィックファイルが使用可能になっていること。サポートされている3Dグラフィックファイル形式については、6.1.1「サポートされている3Dグラフィック形式」をご覧ください。
- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]からシーングラフをビューにドラッグしてドロップします。

ビューに空の矩形が表示されます。

ステップ 2
[プロパティ]パネルで[ファイルのインポート]をクリックします。

ダイアログが開きます。

ステップ 3
3Dグラフィックファイルが格納されているディレクトリに移動します。

ステップ 4
3Dグラフィックファイルを選択します。

ステップ 5
[開く]をクリックします。

インポートが開始します。ダイアログが開きます。

ステップ 6
[OK]をクリックします。

ビューに3Dグラフィックが表示されます。インポートしたウィジェットツリーが、シーングラフを親ノードにしてナビゲーションエリアに表示されます。

ティップ

複数のインポート

複数の3Dグラフィックを1つのシーングラフにインポートすることもできます。

インポート後、複数の3Dグラフィックは重ねてレンダリングされます。3Dオブジェクトを個別に表示するには、`RootNode`の`visible`プロパティを使用します。

8.1.6. ラベルのフォントの変更

ラベルのフォントの変更

前提条件:

- `.ttf`ファイルは、`$GUIDE_PROJECT_PATH\resources`ディレクトリに配置されています。

- EB GUIDEモデルにビューステートが含まれていること。
- ビューにラベルが含まれていること。

ステップ 1
ビューでラベルを選択します。

ステップ 2
[プロパティ]パネルで、`font`ドロップダウンリストボックスからフォントを選択します。

ビューに表示されているラベルのフォントが変更されます。

8.1.7. コンテナーを使用したウィジェットのグループ化

ウィジェットのグループ化

コンテナーを使用するとウィジェットをグループ化できます。

前提条件:

- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]からコンテナーをビューにドラッグしてドロップします。

ステップ 2
コンテンツエリアで、コンテナーの四隅のいずれかをドラッグしてコンテナーを拡大します。

ステップ 3
[ツールボックス]から2つ以上のウィジェットをドラッグし、コンテナーにドロップします。

ウィジェットがコンテナーの子のウィジェットとしてモデリングされます。コンテナーを移動すると、子のウィジェットも一緒に移動します。

8.1.8. ビューへのインスタンシエータの追加

インスタンシエータの追加

前提条件:

- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]からインスタンシエータをビューにドラッグしてドロップします。

ステップ 2
[ツールボックス]からウィジェットをインスタンシエータにドラッグしてドロップします。

ステップ 3
インスタンシエータを選択し、[プロパティ]パネルに移動します。

ステップ 3.1
`numItems`プロパティには、1よりも大きい値を入力してください。

ステップ 3.2
次のいずれかのウィジェット機能を、インスタンシエータに追加します。

- ボックスレイアウト
- フローレイアウト
- グリッドレイアウト
- リストレイアウト

詳細については、8.3.1「ウィジェット機能の追加」をご覧ください。

基本ウィジェットが、`numItems`プロパティで指定された回数だけ、インスタンシエータに指定されたレイアウトで、ビューに表示されます。

インスタンシエータの使い方について詳しい例は、11.4「チュートリアル: 動的コンテンツを使用したリストの作成」をご覧ください。

8.2. ウィジェットプロパティの操作

8.2.1. ウィジェットの配置

ウィジェットの配置

ウィジェットの配置とは、ウィジェットの`x`および`y`プロパティを調整することです。`x`と`y`の値が両方とも`0`になっている原点は、親ウィジェットの左上隅です。

前提条件:

- コンテンツエリアにビューが表示されていること。
- ビューにウィジェットが含まれていること。

ステップ 1
ウィジェットを選択します。

[プロパティ]パネルに、選択したウィジェットのプロパティが表示されます。

ステップ 2
ウィジェットのx座標を定義するには、xテキストボックスに値を入力します。

ステップ 3
ウィジェットのy座標を定義するには、yテキストボックスに値を入力します。

ステップ 4
テキストボックスの外側をクリックします。

コンテンツエリアで、入力した場所にウィジェットが表示されます。

ティップ

別の方法

ウィジェットを目見当で配置するには、コンテンツエリアのウィジェットを選択し、マウスを使って移動します。

8.2.2. ウィジェットのサイズの変更

ウィジェットのサイズの変更

前提条件:

- コンテンツエリアにビューが表示されていること。
- ビューにウィジェットが含まれていること。

ステップ 1
ウィジェットを選択します。

[プロパティ]パネルに、選択したウィジェットのプロパティが表示されます。

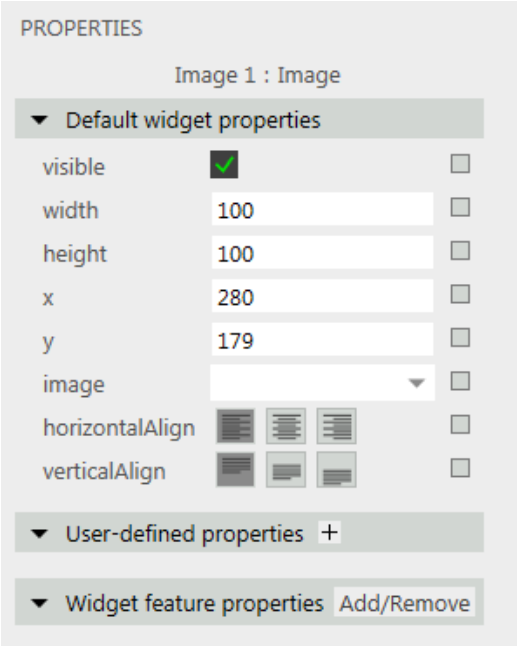

図8.1 イメージのプロパティ

ステップ 2
ウィジェットの高さを定義するには、`height`テキストボックスに値を入力します。

ステップ 3
ウィジェットの幅を定義するには、`width`テキストボックスに値を入力します。

ステップ 4
テキストボックスの外側をクリックします。

コンテンツエリアで、入力したサイズでウィジェットが表示されます。

注記

負の値

`height`および`width`プロパティに負の値を使用しないでください。EB GUIDE Studioは負の値を`0`として扱うため、各ウィジェットが描出されなくなってしまいます。

ティップ

別の方法

ウィジェットのサイズを目見当で変更するには、コンテンツエリアのウィジェットを選択し、角の1つをマウスを使ってドラッグします。

8.2.3. ウィジェットプロパティ間のリンク設定

ウィジェットプロパティ間のリンク設定

2つのウィジェットプロパティが常に同じ値を持つようにするために、2つのウィジェットプロパティをリンクできます。例えば、次の手順は、四角形の`width`プロパティをビューの`width`プロパティとリンクする方法を示しています。

同一ビュー内にあるウィジェットのプロパティのみリンクできます。

インスタンシエータの子ウィジェットのプロパティにリンクすることはできません。

前提条件:

- EB GUIDEモデルにビューステートが含まれていること。
- ビューに四角形が含まれていること。
- 四角形の`width`プロパティがスクリプト値ではないこと。

ステップ 1
四角形をクリックします。

[プロパティ]パネルに、四角形のプロパティが表示されます。

ステップ 2
[プロパティ]パネルで`width`プロパティに移動し、プロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューの[ウィジェットプロパティへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 4
ダイアログ内でビューに移動し、その`width`プロパティをクリックします。

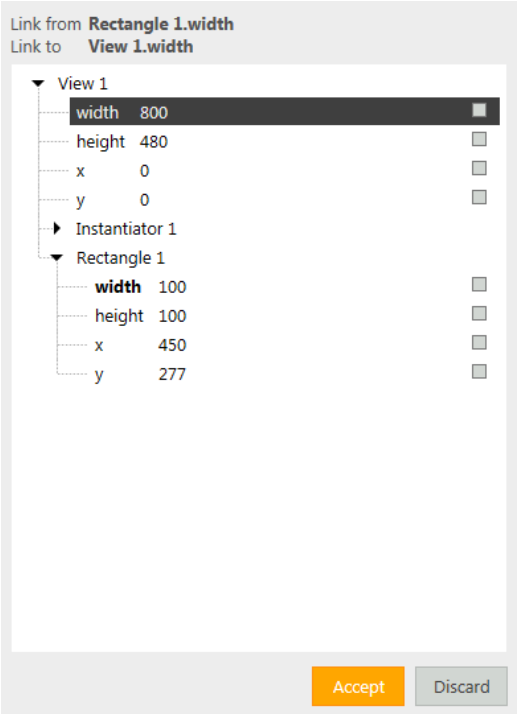

図8.2 ウィジェットプロパティ間のリンク設定

ステップ 5
[承認]をクリックします。

ダイアログが閉じられます。ボタンが`width`プロパティの横に表示されます。これは、四角形の`width`プロパティがビューの`width`プロパティにリンクされたことを示します。ビューの幅を変更するたびに四角形の幅が変更され、逆に四角形の幅を変更するたびにビューの幅が変更されます。

注記

リンクソースとリンクターゲット

ボタンはリンクソースの横にのみ表示されます。リンクターゲットには表示されません。

ティップ

リンクの削除

リンクを削除するには、ボタンをもう一度クリックします。開いたメニューで、[リンクの削除]をクリックします。

8.2.4. ウィジェットプロパティとデータプールアイテムのリンク設定

ウィジェットプロパティとデータプールアイテムのリンク設定

ウィジェットプロパティとデータプールアイテムが常に同じ値を持つようにするために、ウィジェットプロパティとデータプールアイテムをリンクできます。例えば、次の手順は、イメージのimageプロパティと新しいデータプールアイテムをリンクする方法を示しています。

前提条件:

- EB GUIDEモデルにビューステートが含まれていること。
- ビューにイメージが含まれていること。
- イメージのimageプロパティがスクリプト値ではないこと。

ステップ 1
イメージをクリックします。

[プロパティ]パネルに、イメージのプロパティが表示されます。

ステップ 2
[プロパティ]パネルでimageプロパティに移動し、プロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューの[データプールアイテムへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 4
新しいデータプールアイテムを追加するには、コンボボックスに名前を入力します。

ステップ 5
[データプールアイテムname of the datapool itemを追加]をクリックします。

ステップ 6
[承認]をクリックします。

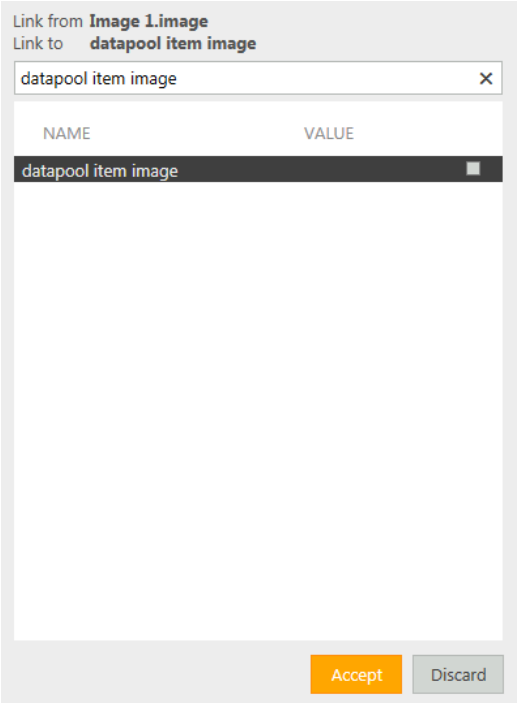

図8.3 データプールアイテムとのリンク設定

新しいデータプールアイテムが追加されます。

ステップ 7
ダイアログが閉じられます。ボタンがimageプロパティの横に表示されます。これは、imageプロパティがデータプールアイテムにリンクされたことを示します。イメージを変更するたびにデータプールアイテムが変更され、逆にデータプールアイテムを変更するたびにイメージが変更されます。

注記

リンクソースとリンクターゲット

ボタンはリンクソースの横にのみ表示されます。リンクターゲットには表示されません。

ティップ

リンクの削除

リンクを削除するには、ボタンをもう一度クリックします。開いたメニューで、[リンクの削除]をクリックします。

8.2.5. ウィジェットへのユーザー定義プロパティの追加

ウィジェットへのユーザー定義プロパティの追加

前提条件:

- EB GUIDEモデルにビューステートが含まれていること。
- ビューにウィジェットが含まれていること。

ステップ 1
ウィジェットを選択します。

[プロパティ]パネルに、選択したウィジェットのプロパティが表示されます。

ステップ 2
[プロパティ]パネルで[ユーザー定義プロパティ]カテゴリに移動し、⊞をクリックします。

メニューが展開されます。

ステップ 3
メニューでユーザー定義プロパティのタイプをクリックします。

選択したタイプの新しいウィジェットプロパティがウィジェットに追加されます。

ステップ 4
プロパティの名前を変更します。

8.2.5.1. `Function (): bool`タイプのユーザー定義プロパティの追加

`Function (): bool`タイプのユーザー定義プロパティの追加

`Function (): bool`タイプのプロパティは、パラメータを持たない、ブール値を返す関数です。この関数をEB GUIDEスクリプトで呼び出すには、ウィジェットプロパティの後に引数リストを指定します。

前提条件:

- EB GUIDEモデルにビューステートが含まれていること。
- ビューにウィジェットが含まれていること。

ステップ 1
ウィジェットを選択します。

[プロパティ]パネルに、選択したウィジェットのプロパティが表示されます。

ステップ 2
[プロパティ]パネルで[ユーザー定義プロパティ]カテゴリに移動し、⊞をクリックします。

メニューが展開されます。

ステップ 3
メニューで`Function (): bool`をクリックします。

`Function (): bool`タイプの新しいウィジェットプロパティがウィジェットに追加されます。

ステップ 4
プロパティの名前を変更します。

ステップ 5
プロパティの横にある[編集]をクリックします。

スクリプトエディターが開きます。

ステップ 6
EB GUIDEスクリプトを使用して新しい関数の動作を定義します。

ステップ 7
[承認]をクリックします。

例**8.1**
`Function (): bool`タイプのプロパティの呼び出し

EB GUIDEモデルには、`Background color`という四角形があります。その四角形に`Function (): bool`タイプのプロパティを追加しました。このプロパティは`change`と呼ばれます。

EB GUIDEモデルのEB GUIDEスクリプトコードでは、次のようにプロパティでスクリプトを呼び出せます。

```
"Background color".change()
```

8.2.6. ユーザー定義プロパティの名前の変更

ユーザー定義プロパティの名前の変更

前提条件:

- EB GUIDEモデルに、ユーザー定義プロパティを持つウィジェットが含まれていること。

ステップ 1
ナビゲーションエリアで、ユーザー定義プロパティを持つウィジェットを選択します。

ステップ 2
[プロパティ]パネルでプロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューで[名前の変更]をクリックします。

ステップ 4
プロパティの名前を入力します。

ステップ 5
Enterキーを押します。

8.3. ウィジェット機能を追加してウィジェットを拡張する

ウィジェット機能は、ウィジェットの外観と動作に関して別の機能を追加します。ウィジェット機能をウィジェットに追加することは、1つ以上のウィジェットプロパティを追加することを意味します。提供されるウィジェット機能は、ウィジェットのタイプによって異なります。

8.3.1. ウィジェット機能の追加

ウィジェット機能の追加

前提条件:

- EB GUIDEモデルにウィジェットが含まれていること。

ステップ 1
ナビゲーションエリアで、ウィジェットをクリックします。

[プロパティ]パネルに、選択したウィジェットのプロパティが表示されます。

ステップ 2
[プロパティ]パネルで[ウィジェット機能プロパティ]カテゴリに移動し、[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

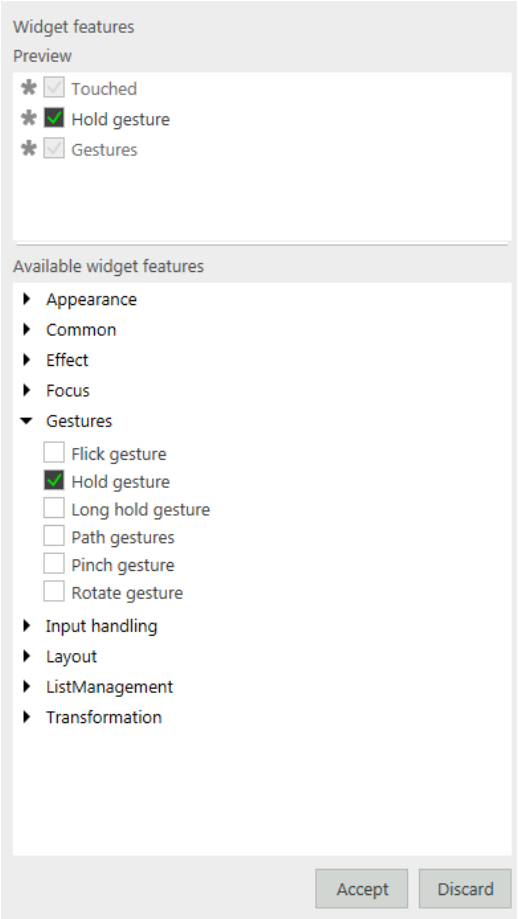

図8.4 ウィジェット機能ダイアログ

ステップ 3
[使用可能なウィジェット機能]からカテゴリを展開し、追加するウィジェット機能を選択します。

選択したウィジェット機能とそれに合わせて自動的に有効化される依存ウィジェット機能の一覧が、[プレビュー]に表示されます。

[承認]をクリックします。

ティップ

ウィジェット機能間の依存関係

ウィジェット機能の中には、他のウィジェット機能を必要とするものがあります。そのため、あるウィジェット機能をオンにすると、他のウィジェット機能が自動的に選択される場合があります。

例えば、[移動可能]というウィジェット機能を追加するとします。その場合、[タッチ]と[タッチ移動]というウィジェット機能も自動的に追加されます。

カテゴリ別に分類されたウィジェット機能の一覧については、12.10「ウィジェット機能」をご覧ください。

チュートリアルについては、以下をご覧ください。

- 11.3「チュートリアル: パスジェスチャーをモデル化する」
- 11.4「チュートリアル: 動的コンテンツを使用したリストの作成」
- 11.2「チュートリアル: EB GUIDEスクリプトを使用したボタン動作のモデル化」

8.3.2. ウィジェット機能の削除

ウィジェット機能の削除

前提条件:

- EB GUIDEモデルにウィジェットが含まれていること。
- 1つ以上のウィジェット機能がウィジェットに追加されていること。

ステップ 1
ナビゲーションエリアで、ウィジェットをクリックします。

[プロパティ]パネルに、選択したウィジェットのプロパティが表示されます。

ステップ 2
[プロパティ]パネルで[ウィジェット機能]カテゴリに移動し、[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

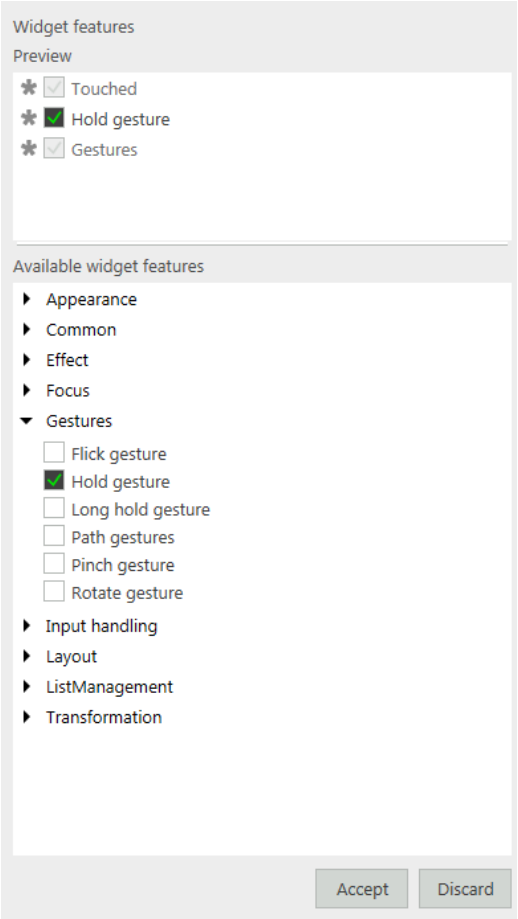

図8.5 ウィジェット機能ダイアログ

ステップ 3
[プレビュー]で、削除するウィジェット機能を解除します。

[承認]をクリックします。

関連するウィジェット機能のプロパティが[プロパティ]パネルから削除されます。

注記

依存関係を持つウィジェット機能の削除

依存関係により自動的に追加されたウィジェット機能は、自動的に削除されません。それらは直接削除できません。子のウィジェット機能を解除する前に、親のウィジェット機能を解除してください。

8.4. EB GUIDEモデルへの言語の追加

ランタイムに言語サポートを有効にするには、EB GUIDEモデルに言語を追加します。

8.4.1. 言語の追加

言語の追加

リスト内の最初の言語は、常にデフォルト言語となり、削除できません。言語を追加した場合、その言語では標準の言語設定が初期値として使用されます。

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[設定] > [言語]の順にクリックします。

使用できる言語が表示されます。

ステップ 3
コンテンツエリアで[追加]をクリックします。

言語が表に追加されます。

ステップ 4
F2キーを押して言語の名前を入力します。

ステップ 5
[言語]ドロップダウンリストボックスから言語を選択します。

ステップ 6
[国]ドロップダウンリストボックスから国を選択します。

言語が追加されました。

ランタイム中の言語変更方法については、11.6「チュートリアル: データプールアイテムへの言語依存テキスト追加」をご覧ください。

8.4.2. 言語の削除

言語の削除

前提条件:

- 少なくとも2つの言語がEB GUIDEモデルに追加されていること。

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[設定] > [言語]の順にクリックします。

使用できる言語が表示されます。

ステップ 3
コンテンツエリアで、言語を選択します。

ステップ 4
コンテンツエリアの[削除]をクリックします。

言語が表から削除されます。

8.5. アニメーションの追加

8.5.1. ウィジェットのアニメーション化

ウィジェットのアニメーション化

前提条件:

- [メイン]ステートマシンに、初期ステートとビューステートが含まれていること。
- 初期ステートにビューステートへの遷移があること。

ステップ 1
ビューステートをダブルクリックします。

ビューがコンテンツエリアに表示されます。

ステップ 2
[ツールボックス]から基本ウィジェットの1つをビューにドラッグします。

ステップ 3
[ツールボックス]からアニメーションを、追加したウィジェットにドラッグします。

ステップ 4
[ツールボックス]から曲線を、追加したウィジェットにドラッグします。

ステップ 5
ナビゲーションエリアで、曲線を階層内でアニメーションの子ウィジェットとなるように移動します。

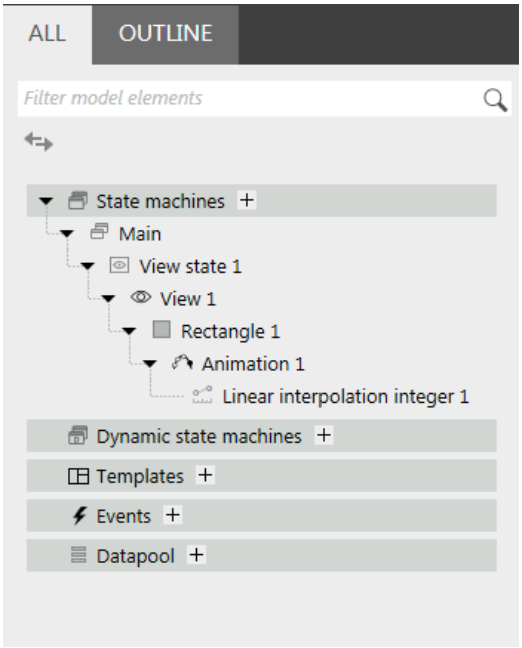

図8.6 アニメーションと曲線(子)を含むウィジェット階層

ステップ 6
基本ウィジェットを選択し、`Conditional script`タイプのユーザー定義プロパティを追加します。詳細については、8.2.5「ウィジェットへのユーザー定義プロパティの追加」をご覧ください。

ステップ 7
プロパティの名前の隣にある[編集]をクリックします。

スクリプトエディターがコンテンツエリアで開きます。

ステップ 8
次のEB GUIDEスクリプトを入力します。

```
function(v:arg0::bool)
{
f:animation_play(v:this->"Animation 1")
}
```

最初に追加されるアニメーションには、デフォルトで[Animation 1]という名前が付きます。Step 2で追加したアニメーションに別の名前が付いている場合は、[On trigger]スクリプトで名前を書き換えてください。

ステップ 9
Step 4で追加した曲線を選択します。

ステップ 10
曲線の`target`プロパティからアニメーション化するプロパティへのリンクを追加します。詳細については、8.2.3「ウィジェットプロパティ間のリンク設定」をご覧ください。

注記

リンクされたプロパティのタイプ

曲線の`target`プロパティと、アニメーション化する基本ウィジェットのプロパティはタイプが同じである必要があります。同じでない場合は、曲線のタイプを変更して基本ウィジェットと同じにしてください。

ステップ 11
シミュレーションを開始します。

リンク先のウィジェットのプロパティが、追加した曲線で指定されたとおりに徐々に変化します。

参考までに、アニメーションや曲線のプロパティを変更して、基本ウィジェットのアニメーションがどう変わるかを確認してください。曲線および曲線プロパティの記述について詳しくは、12.9.3「アニメーション」をご覧ください。アニメーションの具体的な例は、11.5「チュートリアル: 画面内での四角形の移動」をご覧ください。

8.5.2. ビュー遷移のアニメーション化

開始アニメーションの追加

ムーブアニメーションまたはフェードアニメーション付きでビューが表示されるようにするには、ビューテンプレートに開始アニメーションを追加します。

前提条件:

- ビューテンプレートが追加されています。

ステップ 1
ナビゲーションエリアで、ビューテンプレートをクリックします。

ステップ 2
[プロパティ]パネルに移動します。

ステップ 3
ビューが開始されるときに再生されるアニメーションを定義するには、[開始アニメーション]チェックボックスを選択します。

ステップ 4
[遷移タイプ]ドロップダウンリストボックスから、ビュー遷移のタイプを選択します。

ステップ 5
[期間]テキストボックスに時間(ミリ秒)を入力します。

ステップ 6
[終了アニメーション後に再生]チェックボックスを選択します。

結果: このビューテンプレートから派生させたすべてのビューが、定義したアニメーション付きで開始されます。[終了アニメーション後に再生]チェックボックスで、前のビューの終了アニメーションが完了するまで開始アニメーションが待機するように定義しました。

終了アニメーションの追加

ムーブアニメーションまたはフェードアニメーション付きでビューが消えるようにするには、ビューテンプレートに終了アニメーションを追加します。

前提条件:

- ビューテンプレートが追加されています。

ステップ 1
ナビゲーションエリアで、ビューテンプレートをクリックします。

ステップ 2
[プロパティ]パネルに移動します。

ステップ 3
ビューが開始されるときに再生される終了アニメーションを定義するには、[終了アニメーション]チェックボックスを選択します。

ステップ 4
[遷移タイプ]ドロップダウンリストボックスから、ビュー遷移のタイプを選択します。

ステップ 5
[期間]テキストボックスに時間(ミリ秒)を入力します。

ステップ 6
[遅れ]テキストボックスに遅延時間(ミリ秒)を入力します。

結果: このビューテンプレートから派生させたすべてのビューが、定義したアニメーション付きで終了されます。

8.6. ウィジェットの再利用

8.6.1. テンプレートの追加

テンプレートの追加

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで[テンプレート]に移動し、⊞をクリックします。

メニューが展開されます。

ステップ 2
メニューでテンプレートのタイプをクリックします。

選択したタイプの新しいテンプレートが追加されます。コンテンツエリアにテンプレートが表示されます。

ステップ 3
テンプレートの名前を変更します。

ステップ 4
[プロパティ]パネルで、テンプレートのプロパティを編集し、テンプレートインターフェイスを定義します。

ティップ

テンプレートのテンプレート
テンプレートのタイプを既存のテンプレートにすることができます。このため、EB GUIDEではテンプレートからテンプレートを作成できます。

ティップ

テンプレートのコピーと検索
既存のテンプレートをコピーして貼り付けるには、コンテキストメニューまたはCtrl+CキーとCtrl+Vキーを使用するという方法もあります。

EB GUIDEモデル内で特定のテンプレートを検索するには、テンプレートの名前を検索ボックスに入力するか、Ctrl+Fキーを使用します。テンプレートにジャンプするには、ヒットリスト内のテンプレートをダブルクリックします。

8.6.2. テンプレートインターフェイスの定義

テンプレートインターフェイスの定義

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- EB GUIDEモデルにテンプレートが含まれていること。

ステップ 1
テンプレートを選択します。

ステップ 2
テンプレートインターフェースにプロパティを追加するには、プロパティの横にあるボタンをクリックします。メニューの[テンプレートインターフェイスへの追加]をクリックします。

- アイコンがプロパティの横に表示されます。

ステップ 3
テンプレートインターフェースからプロパティを削除するには、プロパティの横にあるボタンをクリックします。メニューの[テンプレートインターフェイスから削除]を選択します。

- アイコンがプロパティの横に表示されなくなります。

注記

インスタンシエータのテンプレート
インスタンシエータのテンプレートでは、インスタンシエータの子ウィジェットのプロパティをテンプレートインターフェイスに追加できません。

8.6.3. テンプレートの使用

テンプレートの使用

前提条件:

- コンテンツエリアにビューが表示されていること。
- [ツールボックス]で、ウィジェットテンプレートが使用可能になっていること。
- ウィジェットテンプレートのテンプレートインターフェースに、プロパティが1つ以上存在すること。

ステップ 1
[ツールボックス]からウィジェットテンプレートをビューにドラッグします。

テンプレートのインスタンスがビューに追加されます。[プロパティ]パネルに、テンプレートインターフェイスのプロパティが表示されます。

ティップ

テンプレートインターフェイスを定義する
[プロパティ]パネルにテンプレートインスタンスのプロパティが一切表示されない場合、テンプレートインターフェイスにはプロパティがまったく追加されていません。この状況を変えるには、テンプレートインターフェイスを定義します。

ステップ 2
[プロパティ]パネルで、テンプレートインスタンスのプロパティを編集します。

プロパティを編集すると、ボタンがボタンに変わります。

ステップ 3
プロパティの値をテンプレートの値にリセットするには、プロパティの横にあるボタンをクリックします。メニューの[テンプレート値へリセット]をクリックします。

8.6.4. テンプレートの削除

テンプレートの削除

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで、テンプレートを右クリックします。

ステップ 2
コンテキストメニューの[削除]をクリックします。

テンプレートが削除されます。

9. データの処理

9.1. イベントの追加

イベントの追加

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで[イベント]に移動し、  をクリックします。

イベントがナビゲーションエリアに追加されます。

ステップ 2
イベントの名前を変更します。

ティップ

イベントのコピーと検索

既存のイベントをコピーして貼り付けるには、コンテキストメニューまたはCtrl+CキーとCtrl+Vキーを使用するという方法もあります。重複を防止するために、貼り付けたイベントはコピーしたイベントとは異なるイベントIDになります。

EB GUIDEモデル内で特定のイベントを検索するには、イベントの名前を検索ボックスに入力するか、Ctrl+Fキーを使用します。イベントにジャンプするには、ヒットリスト内のイベントをダブルクリックします。

9.2. イベントへのパラメータの追加

イベントへのパラメータの追加

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

- イベントが追加されています。

ステップ 1
ナビゲーションエリアで、イベントをクリックします。

[プロパティ]パネルに、選択したイベントのプロパティが表示されます。

ステップ 2
[プロパティ]パネルで、マウスポインターを[パラメータ]に重ねます。

ステップ 3
マウスで ⊞ をクリックします。

ステップ 4
パラメータのタイプを選択します。

選択したタイプのパラメータがイベントに追加されます。

ステップ 5
パラメータの名前を変更します。

9.3. イベントへの対応

イベントに対応するには、イベントIDとイベントグループIDを使用します。EB GUIDE TFでは、ランタイムにイベントを送受信するためにIDが利用されます。

イベントグループの追加

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[設定] > [イベントグループ]の順にクリックします。

ステップ 3
コンテンツエリアで[追加]をクリックします。

イベントグループが表に追加されます。

ステップ 4
イベントグループの名前を変更します。

ステップ 5
イベントグループIDを変更するには、[ID]をダブルクリックし、数字を入力します。

EB GUIDE TFイベントへの対応

前提条件:

- イベントグループが追加されています。
- ナビゲーションエリアに、[すべて]タブが表示されていること。
- イベントが追加されています。

ステップ 1
ナビゲーションエリアで、イベントをクリックします。

[プロパティ]パネルに、選択したイベントのプロパティが表示されます。[プロパティ]パネルに移動します。

ステップ 2
`Event ID`テキストボックスにIDを挿入します。

ステップ 3
`Event group`ドロップダウンリストボックスからイベントを選択します。

9.4. イベントの削除

イベントの削除

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- イベントが追加されています。

ステップ 1
ナビゲーションエリアで、イベントを右クリックします。

ステップ 2
コンテキストメニューの[削除]をクリックします。

イベントが削除されます。

9.5. データプールアイテムの追加

データプールアイテムの追加

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで[データプール]に移動し、⊞をクリックします。

メニューが展開されます。

ステップ 2
メニューでデータプールアイテムのタイプをクリックします。

新しいデータプールアイテムがそのタイプで追加されます。データプールアイテムは内部で使用できるように準備されています。

ステップ 3
データプールアイテムの名前を変更します。

ティップ

データプールアイテムのコピーと検索
既存のデータプールアイテムをコピーして貼り付けるには、コンテキストメニューまたはCtrl +CキーとCtrl +Vキーを使用するという方法もあります。IDの重複を防止するために、貼り付けたデータプールアイテムの`Reader ID`と`Writer ID`は自動的に`-1`に設定されます。

EB GUIDEモデル内で特定のデータプールアイテムを検索するには、データプールアイテムの名前を検索ボックスに入力するか、Ctrl +Fキーを使用します。データプールアイテムにジャンプするには、ヒットリスト内のデータプールアイテムをダブルクリックします。

9.6. リストタイプのデータプールアイテムの編集

リストタイプのデータプールアイテムの編集

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- リストタイプのデータプールアイテムが追加されていること。

ステップ 1
ナビゲーションエリアで、リストタイプのデータプールアイテムをクリックします。

[プロパティ]パネルに、選択したデータプールアイテムのプロパティが表示されます。

ステップ 2

[プロパティ]パネルで`Value`プロパティに移動し、  ボタン(プロパティの横)をクリックします。

エディターが開きます。

ステップ 3
アイテムをリストデータプールアイテムに追加するには、[追加]をクリックします。

新しいエントリーが表に追加されます。

ステップ 4
新しいエントリーの値を`Value`テキストボックスに入力するか、ドロップダウンリストボックスから値を選択します。

ステップ 5
3と4の手順を繰り返し、リストにアイテムをさらに追加します。

ステップ 6
[承認]をクリックします。

リストのコンテンツが`Value`の横に表示されます。

9.7. プロパティのスクリプト値への変換

プロパティのスクリプト値への変換

データプールアイテムやウィジェットのプロパティは、スクリプト値に変換したり、元の値に戻したりできます。データプールアイテムの値を変換するには、以下の操作を行います。ウィジェットプロパティでも同じ手順で変換できます。

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- 1つのデータプールアイテムが追加されています。
- データプールアイテムは言語依存ではありません。
- データプールアイテムはリンクされていません。

ステップ 1
ナビゲーションエリアで、データプールアイテムをクリックします。

[プロパティ]パネルに、選択したデータプールアイテムのプロパティが表示されます。

ステップ 2
[プロパティ]パネルに移動して、Valueプロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューから[スクリプトに変換]をクリックします。

データプールアイテムがスクリプト値に変換されます。

ステップ 4
Valueプロパティの横にある[編集...]をクリックします。

スクリプトエディターがコンテンツエリアで開きます。

ステップ 5
EB GUIDEスクリプトを編集します。

ステップ 6
データプールアイテムをプレーン値に変換するには、Valueプロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 7
メニューで、[プレーン値に変換]をクリックします。

データプールアイテムがプレーン値に変換されます。

9.8. 外部通信の確立

EB GUIDEモデルとアプリケーションの間などに、外部通信を確立するには、通信コンテキストをEB GUIDEモデルに追加します。

通信コンテキストの追加

通信コンテキストを使うと、通信チャネルを開くことができます。

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[設定] > [通信コンテキスト]の順にクリックします。

ステップ 3
コンテンツエリアで[追加]をクリックします。

通信コンテキストが表に追加されます。

ステップ 4
通信コンテキストの名前を、Mediaなどに変更します。

ステップ 5
通信コンテキストIDを変更するには、IDテキストボックスをダブルクリックし、数字を入力します。

ステップ 6
通信コンテキストを独自スレッドで実行するには、[独自スレッドを使用]をクリックします。

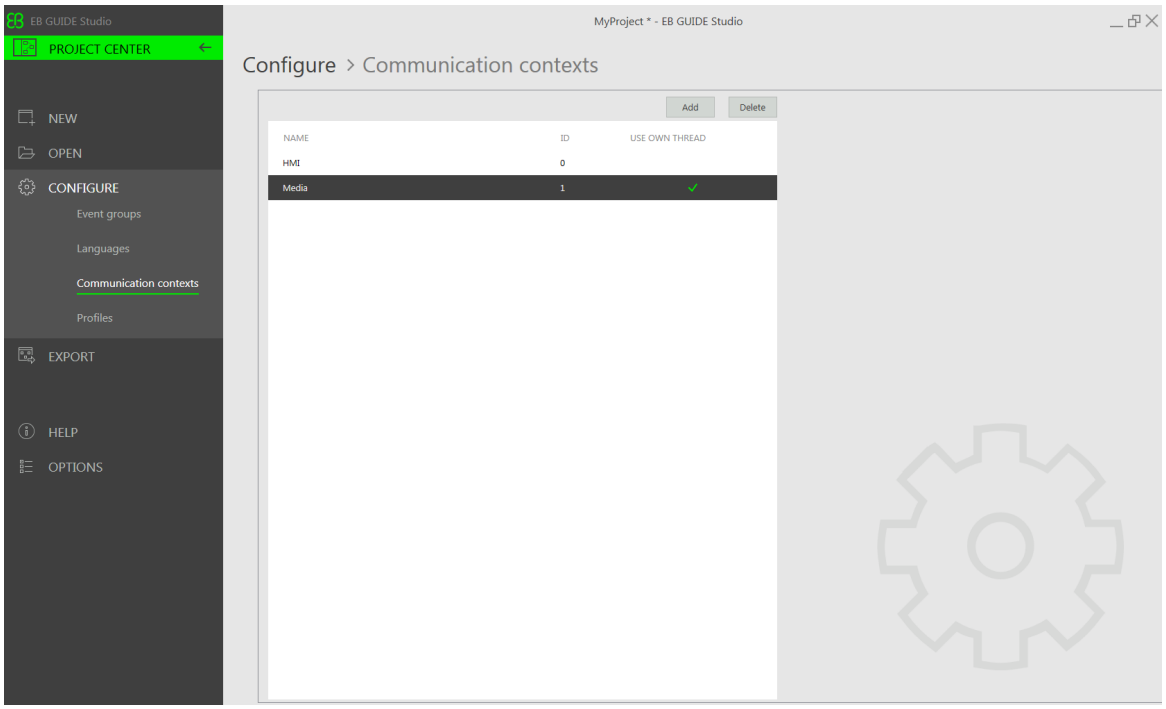

図9.1 通信コンテキストMedia。

データプールアイテムでの外部通信の使用

前提条件:

- 少なくとも2つの通信コンテキストが、EB GUIDEモデルに追加されています。
- 1つのデータプールアイテムが追加されています。

ステップ 1
プロジェクトエディターを開きます。

ステップ 2
ナビゲーションエリアで、データプールアイテムをクリックします。

[プロパティ]パネルに、選択したデータプールアイテムのプロパティが表示されます。

ステップ 3
[プロパティ]パネルで、Reader contextドロップダウンリストボックスから通信コンテキスト(例えばHMI)を選択します。

ステップ 4
[プロパティ]パネルで、Writer contextドロップダウンリストボックスから別の通信コンテキスト(例えばMedia)を選択します。

データプールアイテムに、2つの異なる通信コンテキストが与えられます。EB GUIDEモデルのエクスポート後、データプールアイテムはデータをReader contextからWriter contextに送信します。

上記の手順では、データがHMIからMediaに送信されます。

9.9. データプールアイテム間のリンク設定

データプールアイテム間のリンク設定

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- 1つのデータプールアイテムが追加されています。
- データプールアイテムは言語依存ではありません。
- データプールアイテムはスクリプト値ではありません。

ステップ 1
ナビゲーションエリアで、データプールアイテムをクリックします。

[プロパティ]パネルに、データプールアイテムのプロパティが表示されます。

ステップ 2
[プロパティ]パネルでValueプロパティに移動し、プロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューの[データプールアイテムへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 4
新しいデータプールアイテムを追加するには、コンボボックスに名前を入力します。

ステップ 5
[データプールアイテムname of the datapool itemを追加]をクリックします。

ダイアログが開きます。

ステップ 6
[承認]をクリックします。

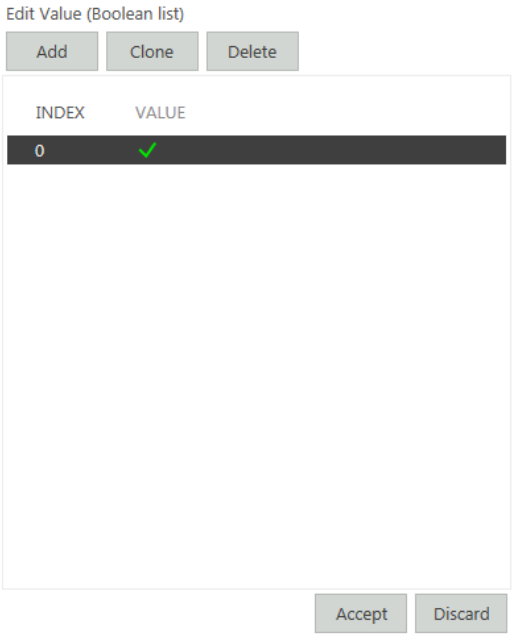

図9.2 データプールアイテム間のリンク設定

ダイアログが閉じられます。Valueプロパティの横に、ボタンが表示されます。これは、Valueプロパティがデータプールアイテムにリンクされたことを示します。一方のデータプールアイテムの値を変更するたびに、他方の値も変更されます。

9.10. データプールアイテムの削除

データプールアイテムの削除

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。
- 1つのデータプールアイテムが追加されています。

ステップ 1
ナビゲーションエリアで、データプールアイテムを右クリックします。

ステップ 2
コンテキストメニューの[削除]をクリックします。

データプールアイテムが削除されます。

10. プロジェクトの処理

10.1. プロジェクトの作成

プロジェクトの作成

ステップ 1
マウスで  をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[新規]をクリックします。

ステップ 3
プロジェクト名を入力し、場所を選択します。

ステップ 4
[作成]をクリックします。

プロジェクトが作成されます。プロジェクトエディターが開かれ、新しいプロジェクトが表示されます。

10.2. プロジェクトを開く

10.2.1. ファイルエクスプローラからプロジェクトを開く

ファイルエクスプローラからプロジェクトを開く

前提条件:

- EB GUIDE Studioプロジェクトが作成されます。

ステップ 1
ファイルエクスプローラを開き、開きたいEB GUIDE Studioプロジェクトファイルを選選択します。EB GUIDE Studioプロジェクトファイルのファイル拡張子は.ebguideです。

ステップ 2
EB GUIDE Studioプロジェクトファイルをダブルクリックします。

プロジェクトがEB GUIDE Studioに開かれます。

10.2.2. プロジェクトをEB GUIDE Studioに開く

プロジェクトをEB GUIDE Studioに開く

前提条件:

- EB GUIDE Studioプロジェクトが作成されます。

ステップ 1
マウスで  をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[開く]タブをクリックします。

ステップ 3
[最近使ったプロジェクト]の一覧にあるプロジェクトを選択するか、[参照]をクリックして、開きたいEB GUIDE Studioプロジェクトファイルを選択します。EB GUIDE Studioプロジェクトファイルのファイル拡張子は `.ebguide` です。

プロジェクトがEB GUIDE Studioに開かれます。

10.3. EB GUIDEモデルのテストと改良

EB GUIDEモデルを対象デバイスにエクスポートする前に、PC上でエラーを解決し、モデルをシミュレートします。

10.3.1. EB GUIDEモデルの検証

EB GUIDEモデルの検証

EB GUIDEは、問題エリアに以下の項目を表示します。

- エラー

▶ ⚠警告

ステップ 1
問題エリアで↻をクリックします。

エラーと警告の数が表示されます。

ステップ 2
[問題]をクリックして、問題エリアを展開します。

エラーと警告の一覧が表示されます。

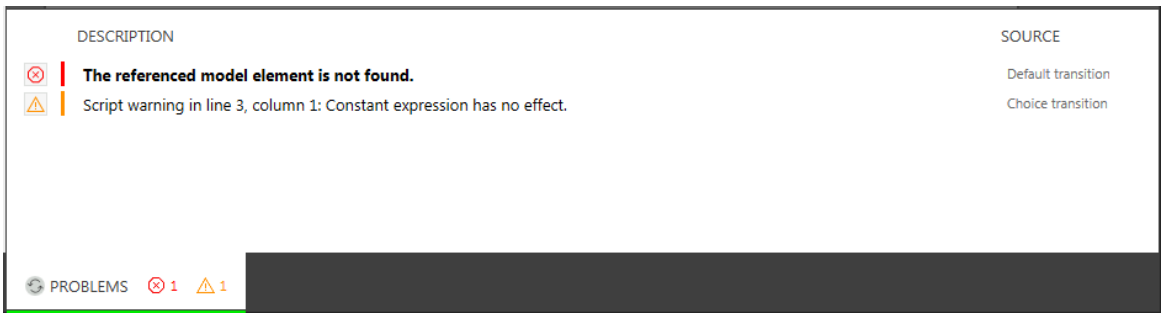

図10.1 問題エリア

ステップ 3
問題の箇所に移動するために、該当行をダブルクリックします。

問題の原因となっている要素が、強調表示されます。

ステップ 4
問題を解決します。

ステップ 5
↻をクリックします。

解決した問題は、問題エリアの一覧から削除されます。

ステップ 6
問題エリアを折りたたむには、[問題]を再度クリックします。

10.3.2. シミュレーションの開始と停止

シミュレーションの開始と停止

ステップ 1
シミュレーションを開始するには、コマンドエリアで▷をクリックします。

シミュレーションとEB GUIDE Monitorが開始されます。シミュレーションは、そのシミュレーション自体の設定で開始されます。

設定を変更するには、プロジェクトセンターに移動し、[設定] > [プロファイル]をクリックします。

ステップ 2
シミュレーションを停止するには、コマンドエリアで□をクリックします。

シミュレーションとEB GUIDE Monitorが停止します。

10.4. EB GUIDEモデルのエクスポート

EB GUIDEモデルのエクスポート

EB GUIDEモデルを対象デバイスにコピーするには、EB GUIDE Studioでそのモデルをエクスポートする必要があります。

EB GUIDEモデルをエクスポートするたびに、必ずプロファイルを選択します。プロファイルにはEB GUIDE TFの起動設定ファイル`gtfStartup.cfg`が記述されています。

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[エクスポート]タブをクリックします。

ステップ 3
`Profile`ドロップダウンリストボックスでプロファイルを選択します。

ステップ 4
[参照]をクリックし、バイナリファイルのエクスポート先となる場所を選択します。

ステップ 5
[フォルダの選択]をクリックします。

ステップ 6
[エクスポート]をクリックします。

選択した場所にバイナリファイルがエクスポートされます。

ティップ

コマンドラインを使用した**EB GUIDE**モデルのエクスポート
コマンドラインオプション`-e <project file, destination dir, profile>`を使用してEB GUIDEモデルをエクスポートすることもできます。

10.5. EB GUIDE Studioの表示言語の変更

EB GUIDE Studioの表示言語の変更

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[オプション]タブをクリックします。

ステップ 3
[表示言語]ドロップダウンリストボックスから言語を選択します。

ステップ 4
EB GUIDE Studioを再スタートします。

再スタート後、グラフィカルユーザーインターフェイスが選択した言語で表示されます。

10.6. プロファイルの設定

EB GUIDE Studioでは、EB GUIDEモデルのさまざまなプロファイルを作成することができます。プロファイルにはEB GUIDE TFの起動設定ファイル`gtfStartup.cfg`が記述されています。

プロファイルを使用して、以下の操作を行えます。

- メッセージの送信
- 読み込む内部ライブラリおよびユーザー定義ライブラリの設定
- シーンの設定
- レンダラーの設定

デフォルトのプロファイルは2つあります。[編集]と[シミュレーション]です。

10.6.1. プロファイルの複製

プロファイルの複製

前提条件:

- EB GUIDE Studioプロジェクトが開いています。
- プロジェクトセンターが表示されます。

ステップ 1
ナビゲーションエリアで、[設定] > [プロファイル]の順にクリックします。

ステップ 2
コンテンツエリアで、[シミュレーション]プロファイルを選択します。

ステップ 3
[複製]をクリックします。

プロファイルが表に追加されます。このプロファイルはデフォルトプロファイルである[シミュレーション]の複製です。

ステップ 4
表でダブルクリックし、プロファイルの名前を`MySimulation`に変更します。

ステップ 5
[シミュレーションに使用]オプションボタンを選択します。

PCでのシミュレーションに、`MySimulation`プロファイルが使用されます。

10.6.2. ライブラリの追加

ライブラリの追加

EB GUIDE TFのデフォルトデリバリは、共有ライブラリをサポートするWindows 10、Linux、QNXなどのオペレーティングシステムで動作します。EB GUIDE TFは実行可能ファイルとライブラリに分かれており、ほとんどの顧客プロジェクトにそのまま適合します。

次のセクションでは、EB GUIDEモデルを操作するユーザー定義ライブラリを追加し、追加機能を提供する方法を示します。

前提条件:

- EB GUIDE Studioプロジェクトが開いています。
- プロジェクトセンターが表示されます。

- ナビゲーションエリアで、[設定] > [プロファイル]タブが選択されています。
- `MySimulation`プロファイルが追加されます。
- `$GUIDE_PROJECT_PATH\resources`に`MyLibraryA`ライブラリがあります。

ステップ 1
コンテンツエリアで、`MySimulation`プロファイルを選択します。

ステップ 2
▶をクリックしてライブラリを展開します。

含まれるすべてのライブラリが[読み込み]表に表示されます。

ステップ 3
[追加]をクリックします。

新しい行が表に追加されます。

ステップ 4
表の[場所]の下にあるドロップダウンリストボックスで`MODEL_PATH`を選択します。

ステップ 5
[名前]テキストボックスに`MyLibraryA`と入力します。

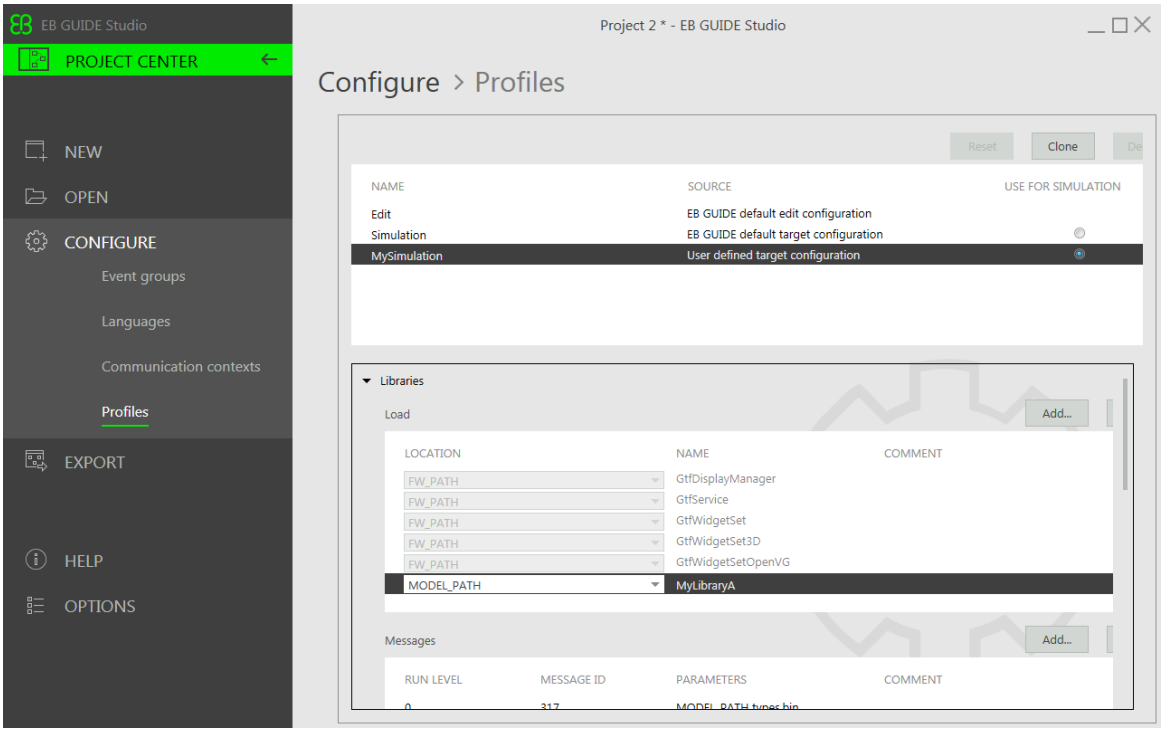

図10.2 ライブラリの表

これで`MyLibraryA`ライブラリが起動コードに追加されました。`MODEL_PATH`は`gtfStartup.cfg`設定ファイルに対する相対ディレクトリを示します。

`FW_PATH`を使用して、`GtfStartup`実行ファイルに対する相対ディレクトリを示します。

10.6.3. メッセージの追加

システムメッセージを送信して、ソフトウェアモジュールを起動または終了することや、ソフトウェアモジュールの動作を変更することができます。システムメッセージには、起動プロセスのどの時点でシステムメッセージが送信されるかを定義するランレベルが含まれます。また、識別IDとオプションパラメータも含まれます。

詳細については、EB GUIDE TFドキュメントの**EB GUIDE TFのソフトウェアモジュール構造**をご覧ください。

注記

EB GUIDE TFの事前定義済みメッセージ
メッセージID範囲0～0xFFFFはEB GUIDE TFおよびEB GUIDE product lineのために予約されています。

メッセージID範囲0x10000～0xFFFFFFFFは管理することができます。

事前定義済みメッセージのメッセージIDとパラメータについては、GtfMessageId.hファイルおよび12.6「gtfStartup.cfg設定ファイル」に記載されています。

メッセージの追加

前提条件:

- EB GUIDE Studioプロジェクトが開いています。
- プロジェクトセンターが表示されます。
- ナビゲーションエリアで、[設定] > [プロファイル]タブが選択されています。

ステップ 1
コンテンツエリアで、プロファイルを選択します。

ステップ 2
をクリックしてライブラリを展開します。

ステップ 3
含まれるすべてのライブラリが[メッセージ]表に表示されます。

ステップ 4
[追加]をクリックします。

新しい行が表に追加されます。

ステップ 5
[ランレベル]テキストボックスに0と入力します。

ステップ 6
[メッセージID]テキストボックスに300と入力します。

ステップ 7
[パラメータ]テキストボックスにUINT32 0xDEADBEAFと入力します。

これでシステムメッセージが追加されました。

メッセージ`GTF_MID_GTF_CORE_CREATE_MODEL`を使用すると、EB GUIDE GTFでIDが0xDEADBEAFの`Gtf-CoreModel`が作成されます。

10.6.4. シーンの設定

EB GUIDE Studioでは、すべてのステートマシンにシーンを設定することができます。

プロジェクトには、次の理由で複数のステートマシンを含めることができます。

- モデルのロジックを異なるステートマシンに分けるため
- 複数のディスプレイまたはレイヤーを使用するため

シーンの設定

前提条件:

- EB GUIDE Studioプロジェクトが開いています。
- プロジェクトセンターが表示されます。
- ナビゲーションエリアで、[設定] > [プロファイル]タブが選択されています。

ステップ 1
コンテンツエリアで▶をクリックして、シーンを展開します。

ステップ 2
[ステートマシン]ドロップダウンリストボックスで、メインディスプレイのステートマシン(例えば、[メイン])を選択します。

ステップ 3
PCデスクトップ上のウィンドウの最初の位置を設定するには、`x`および`y`に値を入力します。

ステップ 4
[レンダラー]ドロップダウンリストボックスからレンダラーを選択します。

ステップ 5
その他のプロパティを調整します。各プロパティの詳細については、12.7「シーン」をご覧ください。

10.7. 言語依存テキストのエクスポートとインポート

10.7.1. 言語依存テキストのエクスポート

ティップ

EB GUIDEモデルの検証

テキストのエクスポートとインポート時のエラーを回避するには、開始前にEB GUIDEモデルを検証します。

言語依存テキストのエクスポート

ユーザーの優先する言語でテキストを指定するには、データプールアイテムの言語依存テキストをすべてエクスポートし、そのテキストを翻訳者に渡します。

前提条件:

- `String`タイプまたは`String list`タイプのデータプールアイテムが追加されます。
- データプールアイテムに言語サポートがあること。言語依存テキストの追加方法については、11.6「チュートリアル: データプールアイテムへの言語依存テキスト追加」をご覧ください。
- 少なくとも2つの言語がEB GUIDEモデルに追加されていること。
- EB GUIDEモデルにエラーおよび警告がないこと。

ステップ 1

 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2

ナビゲーションエリアで、[設定] > [言語]の順にクリックします。

ステップ 3

コンテンツエリアで、翻訳する必要があるターゲット言語を選択します。

複数選択が可能です。

ステップ 4

[エクスポート]をクリックします。

ダイアログが開きます。

ステップ 5

ファイルのエクスポート先となるディレクトリを選択します。

ステップ 6

[フォルダの選択]をクリックします。

結果: エクスポートが開始します。選択したディレクトリにファイルが保存されます。ファイルには言語依存頭字語と.xliff形式が含まれます。ソース言語の値とターゲット言語の値がファイルに格納されます。

注記

言語ごとに1つのファイルがエクスポートされます。
プロジェクトセンターで選択する言語ごとに異なるファイルがエクスポートされます。

10.7.2. 言語依存テキストのインポート

言語依存テキストのインポート

前提条件:

- Stringタイプまたは`String list`タイプのデータプールアイテムが追加されます。
- データプールアイテムに言語サポートがあること。言語依存テキストの追加方法については、11.6「チュートリアル: データプールアイテムへの言語依存テキスト追加」をご覧ください。
- 少なくとも2つの言語がEB GUIDEモデルに追加されていること。
- EB GUIDEモデルにエラーおよび警告がないこと。
- 少なくとも1つの翻訳済み.xliffファイルが利用可能であること。

ステップ 1
をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[設定] > [言語]の順にクリックします。

ステップ 3
[インポート]をクリックします。

ダイアログが開きます。

ステップ 4
翻訳済み.xliffファイルの格納先となるディレクトリを選択します。

ステップ 5
翻訳済み.xliffファイルを選択します。

複数選択が可能です。

ステップ 6
[開く]をクリックします。

インポートが開始します。ダイアログが開きます。

ステップ 7
[閉じる]をクリックします。

11. チュートリアル

この章では、トピックスをアルファベット順で並べています。

11.1. チュートリアル: 動的ステートマシンの追加

動的ステートマシンを使用すると、ランタイム中にポップアップを表示できます。動的ステートマシンは、例えば通常の画面にオーバーレイするエラーメッセージを表示する場合などに使用します。

ここからは、動的ステートマシンを作成する手順について、動的ステートマシンをモデリングし、音量を調節する操作を例に挙げて説明します。ここで説明する順番どおりに操作してください。

所要時間: 20分

イベントとデータプールアイテムを追加する

このセクションでは、イベントとデータプールアイテムを追加する手順について説明します。これらのイベントを使用して、後ほど音量の調節を行います。データプールアイテムは、後のセクションでグラフィック要素の位置を変更するために使用します。

前提条件:

- ナビゲーションエリアに、[すべて]タブが表示されていること。

ステップ 1
ナビゲーションエリアで[イベント]に移動し、⊞をクリックします。

イベントがナビゲーションエリアに追加されます。

ステップ 2
イベントの名前を`Volume up`に変更します。

ステップ 3
イベントを追加し、名前を`Volume down`に変更します。

ステップ 4
イベントを追加し、名前を`Close volume control`に変更します。

ステップ 5
ナビゲーションエリアで[データプール]に移動し、⊞をクリックします。

メニューが展開されます。

ステップ 6
メニューで[整数]をクリックします。

Integerタイプのデータプールアイテムが追加されます。

ステップ 7
データプールアイテムの名前をVolume indicatorに変更します。

これで、3つのイベントとデータプールアイテムが追加されました。

動的ステートマシンを追加して動作をモデリングする

ここからは、動的ステートマシンを追加する手順について説明します。ハプティック動的ステートマシンのモデリングを行い、音量の調節に使用します。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで[動的ステートマシン]に移動し、⊞をクリックします。

メニューが展開されます。

ステップ 2
メニューで[ハプティック動的ステートマシン]をクリックします。

ハプティック動的ステートマシンが追加されます。

ステップ 3
動的ステートマシンの名前をVolume controlに変更します。

ステップ 4
ナビゲーションエリアで、Volume controlをダブルクリックします。

動的ステートマシンがコンテンツエリアに表示されます。

ステップ 5
[ツールボックス]から初期ステートをドラッグし、ステートマシンにドロップします。

ステップ 6
[ツールボックス]からビューステートをドラッグし、ステートマシンにドロップします。

ビューステートと共に、ビューがEB GUIDEモデルに追加されます。

ステップ 7
ナビゲーションエリアで、ビューステートをクリックします。

ステップ 8
F2キーを押し、ビューステートの名前をVolumeに変更します。

ステップ 9
コンテンツエリアで、初期ステートをクリックします。

ステップ 10
緑色のドラッグ点をクリックし、マウスボタンを押したままにします。

ステップ 11
ビューステートまでマウスをドラッグします。

ステップ 12
ビューステートが緑色で強調表示されたら、マウスボタンを離します。

遷移が追加され、緑色の矢印として表示されます。

スライダーをモデリングする

このセクションでは、水平型のスライダーインジケーターをモデリングする手順を説明します。スライダーインジケーターはランタイム中に音量を表示します。

スライダーインジケーターは2つの四角形で構成されます。一方はスライダーの背景、もう一方は音量です。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで、`Volume`ビューステートを展開します。ビューをダブルクリックします。

コンテンツエリアにビューが表示されます。

ステップ 2
[ツールボックス]から四角形をドラッグし、ビューにドロップします。

ステップ 3
ナビゲーションエリアで四角形をクリックし、F2キーを押します。

ステップ 4
四角形の名前を`Slider background`に変更します。

ステップ 5
`Slider background`の外観を変更するには、四角形をクリックし、[プロパティ]パネルに移動します。

ステップ 5.1
`width`テキストボックスに`500`と入力します。

ステップ 5.2
`x`テキストボックスに`125`

ステップ 5.3
`y`テキストボックスに`300`

ステップ 6
[ツールボックス]から四角形をドラッグし、コンテンツエリアの`Slider background`にドロップします。

四角形が、`Slider background`の子ウィジェットとして追加されます。

ステップ 7
ナビゲーションエリアで四角形をクリックし、F2キーを押します。

ステップ 8
四角形の名前をIndicatorに変更します。

ステップ 9
Indicatorの外観を変更するには、四角形をクリックし、[プロパティ]パネルに移動します。

ステップ 9.1
widthテキストボックスに40と入力します。

ステップ 9.2
heightテキストボックスに80

ステップ 9.3
xプロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 9.4
メニューの[データプールアイテムへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 9.5
ドロップダウンリストボックスからVolume indicatorデータプールアイテムを選択します。

ステップ 9.6
[承認]をクリックします。

ダイアログが閉じられます。ボタンがxプロパティの横に表示されます。これで、xの値とVolume indicatorの値がリンクされました。

ステップ 9.7
yテキストボックスに10と入力します。

ステップ 9.8
fillColorプロパティは黒を選択します。

2つの四角形がビューに追加され、四角形の外観が変更されました。

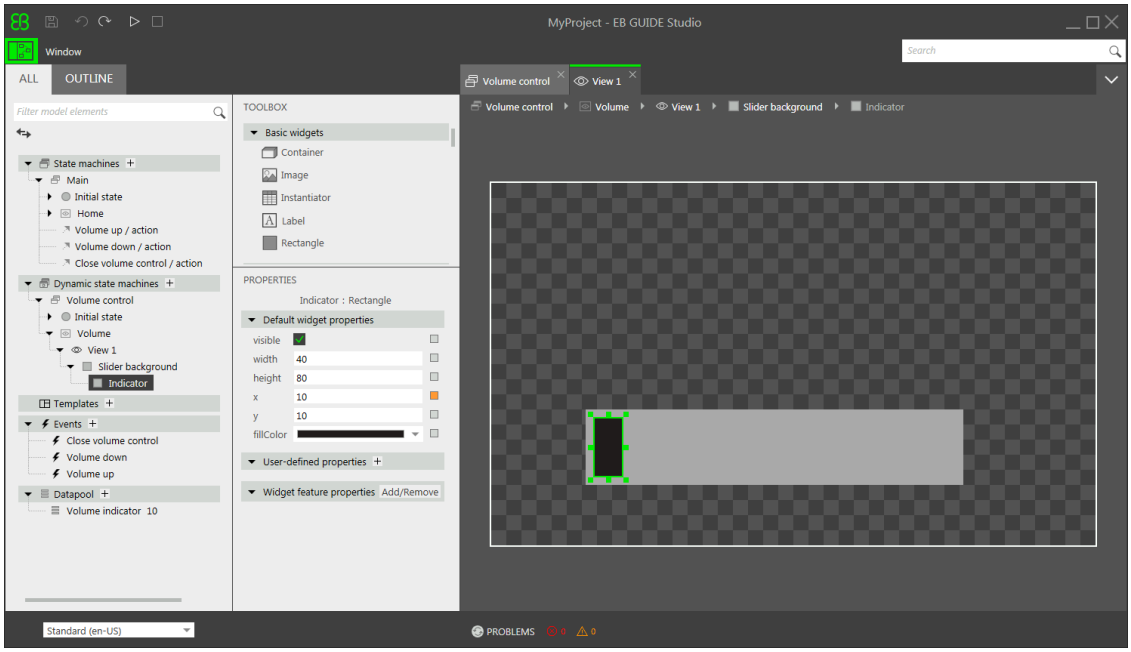

図11.1 2つの四角形を持つ[ビュー1]の外観

ステップ 10
ナビゲーションエリアで、`Volume indicator`データプールアイテムをクリックします。

ステップ 11
[プロパティ]パネルに移動し、`Value`テキストボックスに`10`と入力します。

コンテンツエリアで、四角形`Indicator`の位置が変わります。

`Volume indicator`データプールアイテムで、四角形`Indicator`の`x`の位置を調節できます。

[メイン]ステートマシンにステートを追加する

このセクションでは、[メイン]ステートマシンに初期ステートとビューステートを追加します。ビューステートを使用し、動的ステートマシンが他のステートマシンと平行して動作するようにします。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで、[メイン]をダブルクリックします。

[メイン]ステートマシンがコンテンツエリアに表示されます。

ステップ 2
[ツールボックス]から初期ステートをドラッグし、ステートマシンにドロップします。

ステップ 3
[ツールボックス]からビューステートをドラッグし、ステートマシンにドロップします。

ビューステートと共に、ビューがEB GUIDEモデルに追加されます。

ステップ 4
ビューステートの名前を`Home`に変更します。

ステップ 5
コンテンツエリアで、初期ステートをクリックします。

ステップ 6
遷移を初期ステートから`Home`ビューステートに追加します。

ステップ 7
ナビゲーションエリアで、[メイン]をクリックします。

ステップ 8
[プロパティ]パネルで、`Dynamic state machine list`チェックボックスを選択します。

ここまでの操作が完了すると、動的ステートマシンに関連付けられたEB GUIDEスクリプト関数を使用できます。

初期ステートとビューステートが[メイン]ステートマシンに追加され、ハプティック動的ステートマシンが[メイン]ステートマシンと並行して動作するようになりました。

内部遷移を[メイン]ステートマシンに追加する

このセクションでは、内部遷移の追加を行います。内部遷移を使用すると、動的ステートマシンをランタイム中に開始(プッシュ)および終了(ポップ)できます。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで、[メイン]ステートマシンをクリックします。

ステップ 2
[プロパティ]パネルで、[内部遷移]に移動し、[追加]をクリックします。

内部遷移がステートマシンに追加されます。内部遷移がナビゲーションエリアに表示されます。

ステップ 3
内部遷移をさらに2つ追加します。

ステップ 4
ナビゲーションエリアで、1つ目の内部遷移をクリックします。

ステップ 4.1
[プロパティ]パネルに移動します。

ステップ 4.2
[トリガー]コンボボックスで、`Volume up`を選択します。

ステップ 4.3
[アクション]プロパティの横にある[追加]をクリックします。

ステップ 4.4
次のEB GUIDEスクリプトを入力します。

```
function()
  {
  dp:"Volume indicator" = dp:"Volume indicator" + 20
  f:pushDynamicStateMachine(popup_stack:Main, sm:"Volume control", 0)
  }
```

ステップ 4.5
[承認]をクリックします。

アクションが遷移に追加されます。ナビゲーションエリアで、内部遷移の名前が `Volume up`に変わります。

ステップ 5
ナビゲーションエリアで、2つ目の内部遷移をクリックします。

ステップ 5.1
[プロパティ]パネルに移動します。

ステップ 5.2
[トリガー]コンボボックスで、`Volume down`を選択します。

ステップ 5.3
[アクション]プロパティの横にある[追加]をクリックします。

ステップ 5.4
次のEB GUIDEスクリプトを入力します。

```
function()
  {
  dp:"Volume indicator" = dp:"Volume indicator" - 20
  f:pushDynamicStateMachine(popup_stack:Main, sm:"Volume control", 0)
  }
```

ステップ 5.5
[承認]をクリックします。

アクションが遷移に追加されます。ナビゲーションエリアで、内部遷移の名前が `Volume down`に変わります。

ステップ 6
ナビゲーションエリアで、3つ目の内部遷移をクリックします。

ステップ 6.1
[プロパティ]パネルに移動します。

ステップ 6.2
[トリガー]コンボボックスで、`Close volume control`を選択します。

ステップ 6.3
[アクション]プロパティの横にある[追加]をクリックします。

ステップ 6.4
次のEB GUIDEスクリプトを入力します。

```
function()
 {
 f:popDynamicStateMachine(popup_stack:Main,sm:"Volume control")
 }
```

ステップ 6.5
[承認]をクリックします。

アクションが遷移に追加されます。ナビゲーションエリアで、内部遷移の名前が `Close volume control`に変わります。

3つの内部遷移を追加して、動的ステートマシンの開始と終了を可能にしました。`Volume up`と`Volume down`の内部遷移は、四角形`Indicator`の位置を変更します。

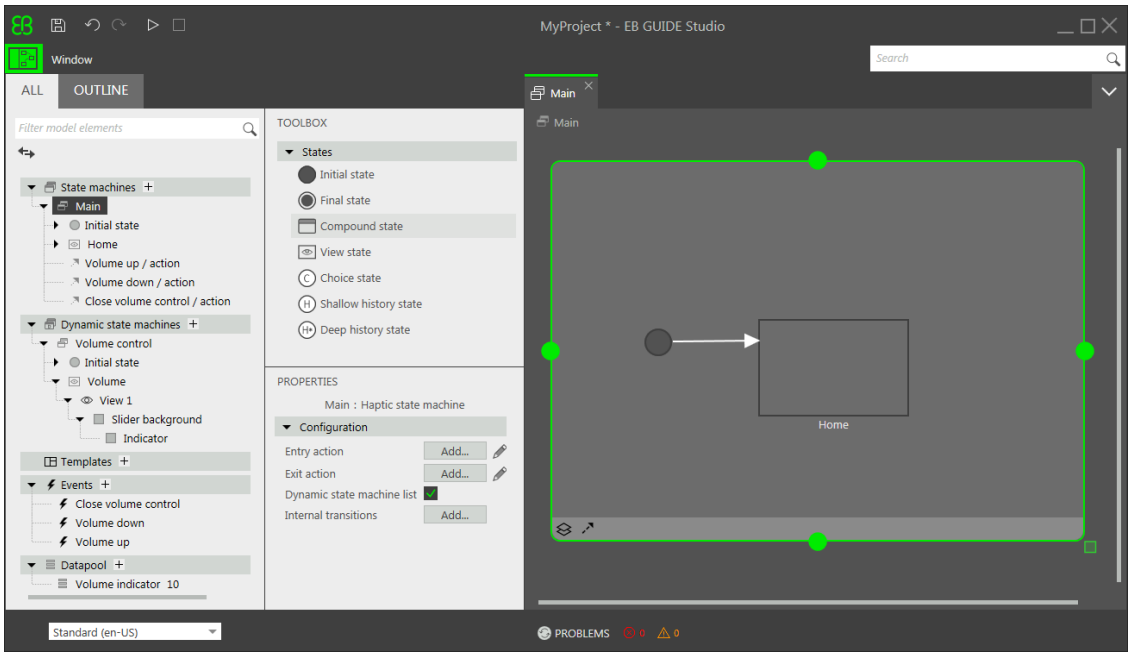

図11.2 すべてのモデル要素が揃ったEB GUIDEモデル

EB GUIDEモデルのシミュレーションとテストを開始する

前提条件:

- 前のセクションの手順を完了していること。

シミュレーションを開始するには、コマンドエリアで▷をクリックします。

シミュレーションとEB GUIDE Monitorが開始されます。EB GUIDEモデルが`Home`ビューステートを表示します。
ステップ 1
EB GUIDE Monitorツールバーで[接続]をクリックします。

ステップ 2
EB GUIDE Monitorで`Volume up`をダブルクリックし、イベントを発火します。

動的ステートマシンが起動し、スライダーインジケーターが表示されます。動的ステートマシンが`Home`ビューステートをオーバーレイします。

`Volume up`または`Volume down`イベントを発火すると、黒い`Indicator`四角形が動きます。`Close volume control`イベントを発火すると、スライダーがビューから消えます。

[メイン]ステートマシンにステートを追加した場合も、`Volume control`動的ステートマシンが他のステートをオーバーレイします。

11.2. チュートリアル: EB GUIDEスクリプトを使用したボタン動作のモデル化

EB　GUIDEスクリプトを使うと、ランタイムの最中にプロパティの値、アクション、または条件を変化させ、それらを評価することができます。

このセクションでは、EB　GUIDEスクリプトを使用してボタンの動作をモデル化する手順を説明します。このボタンは、クリックするとサイズが大きくなり、定義した最大限のサイズに達したら元のサイズに戻ります。失敗を防ぐため、ここで説明する順番どおりに操作してください。

所要時間: 10分

ウィジェットの追加

前提条件:

- [メイン]ステートマシンに、初期ステートとビューステートが含まれていること。
- 初期ステートにビューステートへの遷移があること。
- コンテンツエリアにビューが表示されます。

ステップ 1
[ツールボックス]から四角形をドラッグし、ビューにドロップします。

ステップ 2
ナビゲーションエリアで四角形をクリックし、F2キーを押して四角形の名前を`Background`に変更します。

ステップ 3
[ツールボックス]から四角形をドラッグし、ナビゲーションエリア内でBackground四角形の子ウィジェットとして配置します。

ステップ 4
ナビゲーションエリアで新しい四角形をクリックし、F2キーを押して四角形の名前をButtonに変更します。

ステップ 5
[ツールボックス]からラベルをドラッグし、ナビゲーションエリア内でButton四角形の子ウィジェットとして配置します。

ステップ 6
ナビゲーションエリアでラベルをクリックし、F2キーを押してラベルの名前をButton textに変更します。

ウィジェットの階層が、次のように変わります。

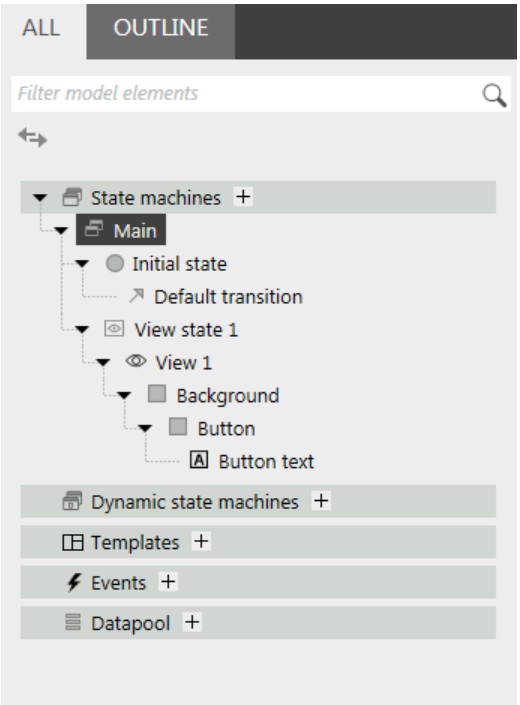

図11.3 ウィジェットの階層

背景を設定する

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアでBackground四角形をクリックし、[プロパティ]パネルに移動します

ステップ 2
widthプロパティの横にあるボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューの[ウィジェットプロパティへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 4
ダイアログ内でビューに移動し、そのwidthプロパティをクリックします。

ステップ 5
[承認]をクリックします。

ダイアログが閉じられます。ボタンがwidthプロパティの横に表示されます。

ステップ 6
Background四角形のheightプロパティを、ビューのheightプロパティにリンクします。

ステップ 7
Background四角形のxプロパティを、ビューのxプロパティにリンクします。

ステップ 8
Background四角形のyプロパティを、ビューのyプロパティにリンクします。

これで、Background四角形がビューのサイズと位置にぴったり重なります。

ボタンの最大幅を定義する

データプールアイテムにボタンの最大幅の値を格納します。この値はランタイムの最中に変更できます。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで[データプール]に移動し、をクリックします。

メニューが展開されます。

ステップ 2
メニューで[整数]をクリックします。

Integerタイプの新しいデータプールアイテムが追加されます。

ステップ 3
データプールアイテムの名前をMaximum widthに変更します。

ステップ 4
[プロパティ]パネルに移動し、Valueテキストボックスに400と入力します。

ボタンを設定する

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアでButton四角形をクリックし、[プロパティ]パネルに移動します

ステップ 1.1
heightテキストボックスに50と入力します。

ステップ 1.2
xテキストボックスに350

ステップ 1.3
yテキストボックスに215

ステップ 1.4
fillColorプロパティに青を選択します。

これで、ボタンが青色になりました。

ステップ 2
[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 3
[使用可能なウィジェット機能]から[入力処理]カテゴリを展開し、[タッチ押下]ウィジェット機能を選択します。

ステップ 4
[承認]をクリックします。

関連するウィジェット機能プロパティがButton四角形に追加され、[プロパティ]パネルに表示されます。

ステップ 5
touchPressedプロパティの横にある[編集...]をクリックします。

ステップ 6
既存のEB GUIDEスクリプトを次のコードに置き換えます。

```
function(v:touchId::int, v:x::int, v:y::int, v:fingerId::int)
     {
         if (v:this.width > dp:"Maximum width") // If the button has grown
                                               // beyond its maximum size...
      {
  // ...reset its dimensions to the default values.
  v:this.height = 50
  v:this.width = 100
  v:this.x = 350
```

```
  v:this.y = 215
      }
      else // Otherwise...
      {

  // ... increase button size...
  v:this.width += 80
  v:this.height += 40

  // ...and move the button to keep it centered.
  v:this.x -= 40
  v:this.y -= 20
      }
         false
     }
```

ステップ 7
[承認]をクリックします。

Button四角形を設定し、ランタイム中にButton四角形のサイズを変更するEB GUIDEスクリプト を作成しました。

ボタンのテキストを設定する

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアでButton textラベルをクリックし、[プロパティ]パネルに移動します。

ステップ 2
textテキストボックスにgrow!と入力します。

ステップ 3
Button textラベルのwidthプロパティを、Button四角形のwidthプロパティにリンクします。

ステップ 4
Button textラベルの<varname>width</varname>プロパティを、Button四角形のheightプロパティにリンクします。

ステップ 5
xテキストボックスに0と入力します。

ステップ 6
yテキストボックスに0

ステップ 7
horizontalAlignプロパティの横にある  をクリックします。

以上の手順で、Button textラベルとButton四角形のサイズと位置が同じになりました。

EB GUIDEモデルの保存およびテスト

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
プロジェクトを保存するには、コマンドエリアで をクリックします。

ステップ 2
シミュレーションを開始するには、コマンドエリアで▷をクリックします。

結果:

シミュレーションを開始すると作成したEB GUIDEモデルが開始され、次のように動作します。

1. 最初に、グレーの画面の中央に青色のボタンが次のように表示されます。

図11.4 結果

2. ボタンをクリックすると、ボタンのサイズが大きくなります。位置は画面中央から変化しません。
3. ボタンの幅が`Maximum width`データプールアイテムの値に達すると、ボタンは再び小さくなり、元のサイズと位置に戻ります。

11.3. チュートリアル: パスジェスチャーをモデル化する

パスジェスチャーとは、タッチスクリーン上に指で描画されたか、その他の入力デバイスによって入力された形状のことです。

このセクションでは、パスジェスチャーをモデル化する手順を説明します。

所要時間: 10分

ウィジェットの追加およびデフォルトウィジェットプロパティの設定

前提条件:

- [メイン]ステートマシンに、初期ステートとビューステートが含まれていること。
- 初期ステートにビューステートへの遷移があること。
- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]から四角形をドラッグし、ビューにドロップします。

ステップ 2
[ツールボックス]からラベルをドラッグし、四角形にドロップします。

ラベルは、四角形の子ウィジェットとして追加されます。

[プロパティ]パネルに、ラベルのプロパティが表示されます。

ステップ 3
[プロパティ]パネルで、`width`テキストボックスに`500`と入力します。

ステップ 4
四角形を選択します。

[プロパティ]パネルに、四角形のプロパティが表示されます。

ステップ 5
`width`テキストボックスに`500`と入力します。

ステップ 6
[プロパティ]パネルで[fillColor]に移動し、赤色を選択します。

2つのウィジェットを追加し、デフォルトウィジェットプロパティを設定しました。

四角形にウィジェット機能を追加する

ユーザーがウィジェット上で開始する形状を入力できるようにするには、[パスジェスチャー]ウィジェット機能を四角形に追加します。入力形状が既知の形状集合とマッチングされ、マッチすればジェスチャーとして認識されます。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
四角形を選択します。

[プロパティ]パネルに、四角形のプロパティが表示されます。

ステップ 2
[プロパティ]パネルで[ウィジェット機能プロパティ]に移動し、[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 3
[使用可能なウィジェット機能]の下で、[ジェスチャー]カテゴリを展開し、`Path gestures`を選択します。

[タッチ]ウィジェット機能が自動的に選択されます。[ジェスチャー]ウィジェット機能でこれが必要なためです。

ステップ 4
[承認]をクリックします。

関連するウィジェット機能プロパティが四角形に追加され、[プロパティ]パネルに表示されます。

ステップ 5
[パスジェスチャー]ウィジェット機能に対し、次のプロパティを編集します。

ステップ 5.1
`onPath`プロパティの横にある[編集...]をクリックします。

ステップ 5.2
次のEB GUIDEスクリプトを入力します。

```
function(v:gestureId::int)
    {
    v:this->"Label 1".text = "recognized path gesture #"
    + f:int2string(v:gestureId);
    }
```

ステップ 5.3
[承認]をクリックします。

ステップ 5.4
`onPathStart`プロパティの横にある[編集...]をクリックします。

ステップ 5.5
次のEB GUIDEスクリプトを入力します。

```
function()
    {
    v:this->"Label 1".text = "path gesture start";
    }
```

ステップ 5.6
[承認]をクリックします。

ステップ 5.7
onPathNotRecognizedプロパティの横にある[編集...]をクリックします。

ステップ 5.8
次のEB GUIDEスクリプトを入力します。

```
function()
    {
    v:this->"Label 1".text = "shape not recognized";
    }
```

ステップ 5.9
[承認]をクリックします。

ステップ 6
シミュレーションを開始するには、コマンドエリアで▷をクリックします。

シミュレーションとEB GUIDE Monitorが開始されます。反応を確認するには、四角形の内部でマウスを使って図形を描画します。

11.4. チュートリアル: 動的コンテンツを使用したリストの作成

インスタンシエータを使うと、ランタイム中にリストを動的に作成することができます。リストタイプのデータプールアイテムに基づき、インスタンシエータが事前定義済みのレイアウトでリストのすべての要素を表示します。データプールアイテムのコンテンツが変更されると、インスタンシエータの表示も変更されます。

このセクションでは、動的コンテンツを使用してリストを作成する手順について説明します。リストの各要素は、ラベルの付いた四角形です。

所要時間: 15分

データプールアイテムの追加

このセクションでは、String listタイプのデータプールアイテムを追加する手順について説明します。データプールアイテムは、インスタンシエータのすべてのリスト要素に値を提供します。データプールアイテムのコンテンツが変更されると、インスタンシエータの表示も変更されます。

前提条件:

- [メイン]ステートマシンに、初期ステートとビューステートが含まれていること。
- 初期ステートにビューステートへの遷移があること。

ステップ 1
リストのコンテンツを表示するために、String listタイプのデータプールアイテムを追加します。

ナビゲーションエリアで[データプール]に移動し、  をクリックします。

メニューが展開されます。

ステップ 2
メニューから[文字列リスト]をクリックします。

String listタイプの新しいデータプールアイテムが追加されます。

ステップ 3
データプールアイテムの名前をMyStringListに変更します。

ステップ 4
MyStringListデータプールアイテムを選択し、[プロパティ]パネルに移動します。

ステップ 5
Valueプロパティの横にある  ボタンをクリックします。

エディターが開きます。

ステップ 5.1
[追加]をクリックします。

新しいエントリーが表に追加されます。

ステップ 5.2
Valueテキストボックスに`One`と入力します。

ステップ 5.3
Two、Three、Four、およびFiveの値をMyStringListデータプールアイテムに追加します。

ステップ 5.4
[承認]をクリックします。

String listタイプのデータプールアイテムが追加されました。データプールアイテムには5つのエントリーが含まれています。

リストのコンテンツは、`Value`プロパティの横に表示されます。

ウィジェットの追加

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ビューにウィジェットを追加するために、コンテンツエリアでビューステートをダブルクリックします。

ビューがコンテンツエリアに表示されます。

ステップ 2
ナビゲーションエリアでビューステートとビューを展開します。

ステップ 3
[ツールボックス]からインスタンシエータをビューにドラッグしてドロップします。インスタンシエータの名前を`MyInstantiator`に変更します。

ステップ 4
[ツールボックス]から四角形をドラッグし、インスタンシエータにドロップします。四角形の名前を`MyRectangle`に変更します。

ステップ 5
[ツールボックス]からラベルをドラッグし、四角形にドロップします。ラベルの名前を`MyLabel`に変更します。

ウィジェットの階層が、次のように変わります。

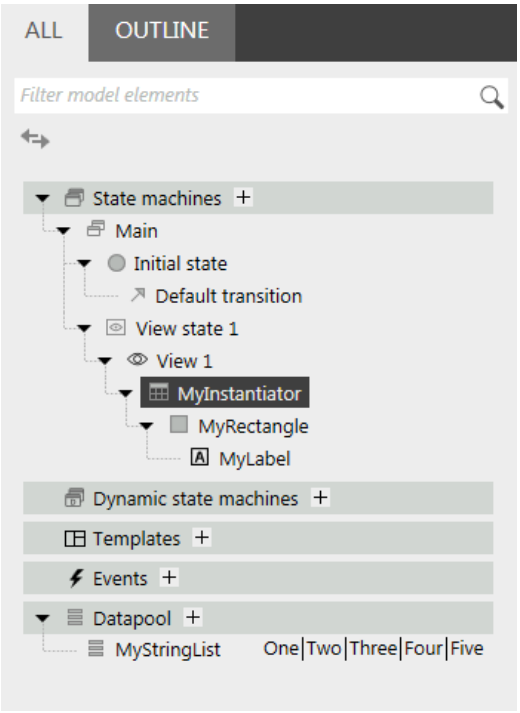

図11.5 インスタンシエータを含むウィジェット階層

インスタンシエータの設定

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
`MyInstantiator`のプロパティを変更するには、インスタンシエータを選択し、[プロパティ]パネルに移動します。

ステップ 2
`width`テキストボックスと`height`テキストボックスに`300`と入力します。

ステップ 3
`x`テキストボックスに`250`と入力します。

ステップ 4
`y`テキストボックスに`150`

ステップ 5
リストの長さを動的に計算するために、条件スクリプトを追加します。

[ユーザー定義プロパティ]カテゴリで、  をクリックします。

メニューが展開されます。

ステップ 5.1
メニューから[条件スクリプト]をクリックします。

ステップ 5.2
プロパティの名前をcalculateNumItemsに変更します。

ステップ 5.3
calculateNumItemsプロパティの横にある[編集]をクリックします。

スクリプトエディターがコンテンツエリアで開きます。

ステップ 5.4
MyStringListデータプールアイテムを[トリガー]リストに追加します。

ステップ 5.5
[On Trigger]スクリプトに次のように入力します。

```
function(v:arg0::bool)
{
 v:this.numItems = length dp:MyStringList;
 false
}
```

MyStringListのコンテンツに応じてリストエントリー数を自動的に変更するスクリプトが追加されました。

ステップ 6
インスタンシエータ内のラベルをすべて整列させるために、レイアウトを追加します。

[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 6.1
[使用可能なウィジェット機能]の下から[レイアウト]カテゴリを展開し、[ボックスレイアウト]ウィジェット機能を選択してラベルを横に整列させます。

関連するウィジェット機能プロパティがインスタンシエータに追加され、[プロパティ]パネルに表示されます。

ステップ 6.2
[承認]をクリックします。

ステップ 6.3
gapテキストボックスに5と入力して、各リスト要素の間隔を5ピクセルに設定します。

ステップ 6.4
[layoutDirection]ドロップダウンリストボックスから[vertical (=1)]を選択して、ラベルを整列させます。

リストの外観を定義し、リストアイテム数を動的に適用するインスタンシエータが設定されました。

リスト要素テキストの設定

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ラベルの外観を変更するために、ラベルを選択して[プロパティ]パネルに移動します。

ステップ 2
`x`テキストボックスと`y`テキストボックスに`0`と入力します。

ステップ 3
ラベルの`width`プロパティから四角形の`width`プロパティへのリンクを追加します。

ステップ 3.1
`width`プロパティの横にある  ボタンをクリックします。

メニューが展開されます。

ステップ 3.2
メニューの[ウィジェットプロパティへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 3.3
ダイアログ内で四角形に移動し、その`width`プロパティを選択します。

ステップ 3.4
[承認]をクリックします。

ダイアログが閉じられます。  ボタンが`width`プロパティの横に表示されます。

ステップ 4
ラベルの<varname>width</varname>プロパティから四角形の`height`プロパティへのリンクを追加します。

ステップ 5
`horizontalAlign`プロパティの横にある  をクリックします。

ラベルの外観が変更され、四角の中心に表示されました。

リスト要素の設定

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
四角形の外観を変更するために、四角形を選択して[プロパティ]パネルに移動します。

ステップ 2
リスト要素が利用可能な幅を必ず使用するよう、四角形のwidthプロパティからインスタンシエータのwidthプロパティへのリンクを追加します。

ステップ 3
heightテキストボックスに50と入力します。

ステップ 4
リストの各ラインに対して一意の位置を定義するために、[ラインインデックス]ウィジェット機能を追加します。

ステップ 4.1
[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 4.2
[使用可能なウィジェット機能]から[リスト管理]カテゴリを展開し、[ラインインデックス]ウィジェット機能を選択します。

lineIndexプロパティが四角形のプロパティに追加されます。

ステップ 5
リストのラベルにMyStringListのコンテンツを入れるため、条件スクリプトを追加します。

ステップ 5.1
[ユーザー定義プロパティ]カテゴリの横にある⊞をクリックします。

メニューが展開されます。

ステップ 5.2
メニューから[条件スクリプト]をクリックします。

ステップ 5.3
プロパティの名前をsetTextに変更します。

ステップ 5.4
setTextプロパティの横にある[編集...]をクリックします。

スクリプトエディターがコンテンツエリアで開きます。

ステップ 5.5
四角のlineIndexプロパティとMyStringListデータプールアイテムを[トリガー]リストに追加します。

ステップ 5.6
[On Trigger]スクリプトに次のように入力します。

```
function(v:arg0::bool)
    {
    v:this->MyLabel.text=dp:MyStringList[v:this.lineIndex];
    false
    }
```

四角形の表示が変更されました。`setText`プロパティで、`MyStringList`のコンテンツが自動的に`MyStringList`のラベルに入るようになりました。

EB GUIDEモデルのテスト

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
シミュレーションを開始するには、コマンドエリアで▷をクリックします。

結果:

`MyStringList`にはデータプールアイテムが5つ含まれるため、OneからFiveのラベルが付いた5つの四角形が、縦方向に並んで表示されます。

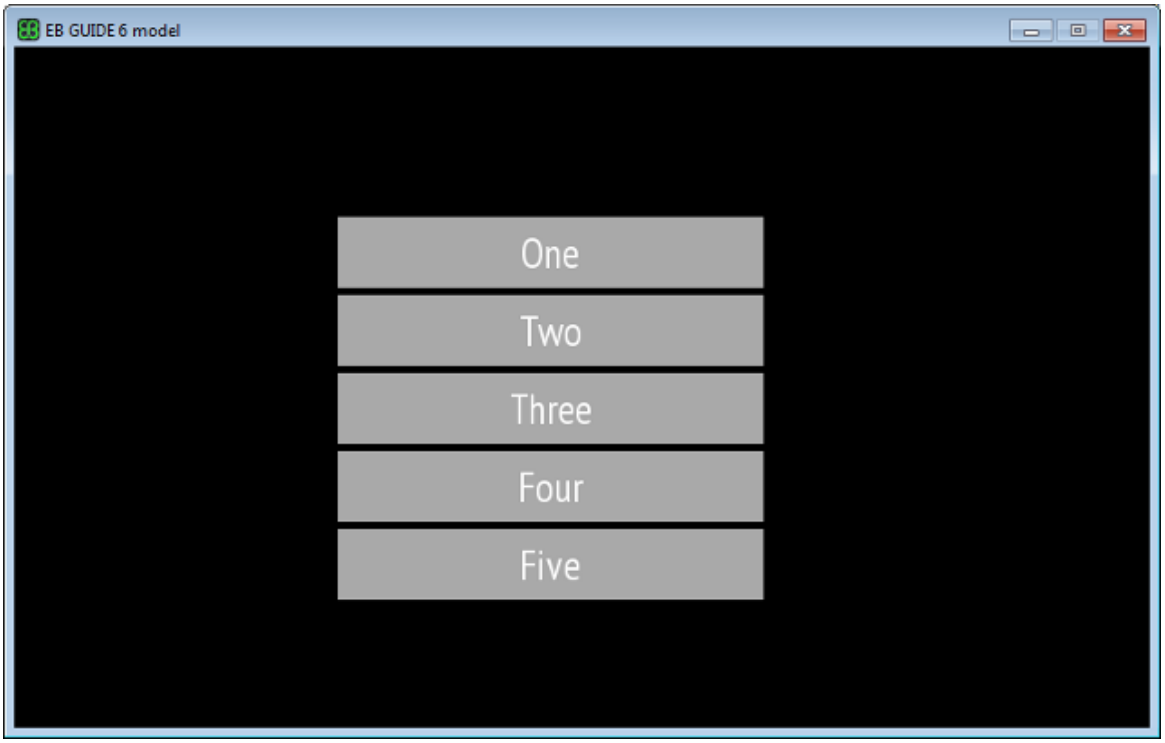

図11.6 インスタンシエータで作成されたリスト

11.5. チュートリアル: 画面内での四角形の移動

このセクションでは、四角形をアニメーション化し、シミュレーションを開始したとき画面内で四角形が移動を繰り返すようにする方法を説明します。

所要時間: 5分

ウィジェットの追加

以下の手順で、ビューに3つのウィジェットを追加し、ウィジェットの階層を編成します。

前提条件:

- コンテンツエリアに、[メイン]ステートマシンが表示されていること。
- [メイン]ステートマシンに、初期ステートとビューステートが含まれていること。
- 初期ステートにビューステートへの遷移があること。

ステップ 1
コンテンツエリアで、ビューステートをダブルクリックします。

ビューがコンテンツエリアに表示されます。

ステップ 2
[ツールボックス]から四角形をドラッグし、ビューにドロップします。

ステップ 3
[ツールボックス]からアニメーションをドラッグし、四角形にドロップします。

ステップ 4
ナビゲーションエリアでアニメーションをクリックし、F2キーを押します。アニメーションの名前を`MyAnimation`に変更します。

ステップ 5
リニア補間整数ウィジェットを[ツールボックス]からドラッグし、四角形にドロップします。

ステップ 6
ナビゲーションエリアで、リニア補間整数の階層を移動させ、アニメーションの子ウィジェットにします。

ここでシミュレーションを開始すると、ビューに四角形が表示されますが、まだ四角形は動きません。

`Conditional script`タイプのユーザー定義プロパティの追加

以下の手順では、四角形にユーザー定義プロパティを追加します。条件スクリプトのプロパティを使用し、シミュレーションの際にアニメーションが始まると四角形を描画するようにします。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
四角形を選択します。

ステップ 2
[プロパティ]パネルで[ユーザー定義プロパティ]カテゴリに移動し、  をクリックします。

メニューが展開されます。

ステップ 3
メニューで`Conditional script`をクリックします。

`Conditional script`タイプのユーザー定義プロパティが四角形に追加されます。

ステップ 4
プロパティの名前を`startAnimation`に変更します。

ステップ 5
`startAnimation`プロパティの横にある[編集...]をクリックします。

スクリプトエディターがコンテンツエリアで開きます。

ステップ 6
次のEB GUIDEスクリプトを入力します。

```
function(v:arg0::bool)
    {
 f:animation_play(v:this->MyAnimation)
    }
```

アニメーションの可視化

このセクションでは、アニメーションを可視化する手順を説明します。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
リニア補間整数ウィジェットを選択します。

ステップ 2
[プロパティ]パネルで `target`プロパティに移動し、プロパティの横にある  ボタンをクリックします。

メニューが展開されます。

ステップ 3
メニューの[ウィジェットプロパティへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 4
ダイアログ内で四角形に移動し、その`x`プロパティを選択します。

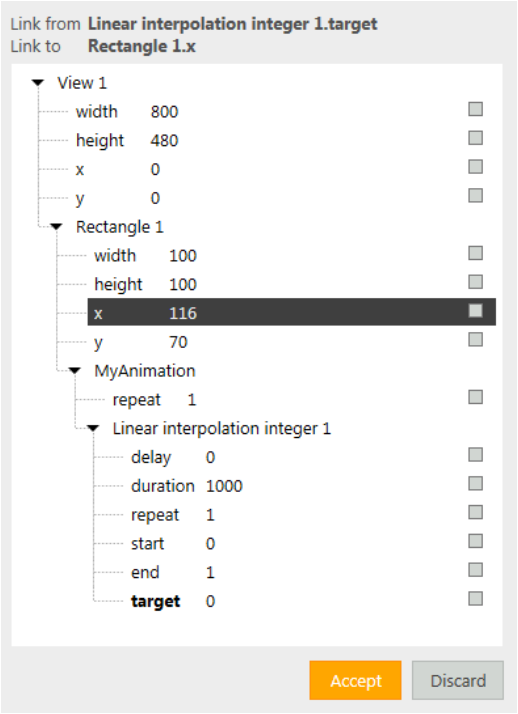

図11.7 ウィジェットプロパティ間のリンク設定

ステップ 5
[承認]をクリックします。

ダイアログが閉じられます。ボタンが`target`プロパティの横に表示されます。

ステップ 6
`end`プロパティを、ビューの`width`プロパティにリンクします。

これらを設定すると、アニメーションの開始時に、四角形の`x`プロパティがゼロからビューの幅の値に変化します。したがって、四角形がビューの左の境界線から右の境界線まで移動します。

ステップ 7
アニメーションを無限に繰り返して実行するには 、`repeat`プロパティに`0`を入力します。

ステップ 8
プロジェクトを保存します。

ステップ 9
シミュレーションを開始するには、コマンドエリアで▷ をクリックします。

結果:

四角形がビューの左端から右端へ 移動を繰り返します。

11.6. チュートリアル: データプールアイテムへの言語依存テキスト追加

EB GUIDEでは、テキストをユーザーの選択した言語で表示することができます。このセクションでは、英語、フランス語、ドイツ語のヒューマンマシンインターフェイスで変化するラベルをモデル化する手順について説明します。

所要時間: 15分

注記

言語依存関係の前提条件

データプールアイテムに言語サポートを追加するには、次の操作を行います。

- `Value`プロパティが別のデータプールアイテムまたはウィジェットプロパティにリンクされている場合は、そのリンクを削除します。
- `Value`プロパティがスクリプト値である場合は、プロパティをプレーン値に変換します。

ウィジェットプロパティとデータプールアイテムのリンク設定

このセクションでは、ラベルの`text`プロパティをデータプールアイテムにリンクする手順について説明します。ランタイム中に、表示テキストがデータプールアイテムから提供されます。

前提条件:

- EB GUIDEモデルに3つの言語、英語、ドイツ語、フランス語が追加されていること。
- コンテンツエリアにビューが表示されていること。
- ビューにラベルが含まれていること。
- ラベルの`text`プロパティがスクリプト値ではないこと。

ステップ 1
ラベルをクリックします。

ステップ 2
[プロパティ]パネルで `text`プロパティに移動し、プロパティの横にあるボタンをクリックします。

ステップ 3
メニューの[データプールアイテムへのリンクを追加]をクリックします。

ダイアログが開きます。

ステップ 4
新しいデータプールアイテムを追加するには、コンボボックスに`Welcome_text`と入力します。

ステップ 5
[データプールアイテムの追加]をクリックします。

ステップ 6
[承認]をクリックします。

データプールアイテムWelcome_textが追加されます。

コンテンツエリアで、ラベルにテキストが何も表示されなくなります。

データプールアイテムに言語依存テキストを入力する

このセクションでは、データプールアイテムに言語依存テキストを追加する手順について説明します。言語ごとに、Valueプロパティに異なるテキストが入ります。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで、Welcome_textデータプールアイテムをクリックします。

ステップ 2
[プロパティ]パネルで、[言語サポート]チェックボックスを選択します。

ステップ 3
Valueテキストボックスに、Welcomeと入力します。

コンテンツエリアで、ラベルにWelcomeと表示されます。

ステップ 4
ナビゲーションエリアの下で、例えばGermanなどの言語をドロップダウンリストボックスから選択します。

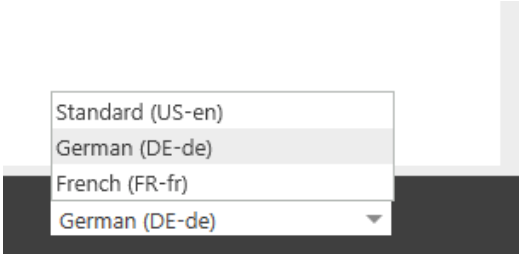

図11.8 Languageドロップダウンリストボックス

ステップ 5
[プロパティ]パネルに移動します。

ステップ 6
Valueテキストボックスに、Willkommenと入力します。

コンテンツエリアで、ラベルにWillkommenと表示されます。

ステップ 7
ナビゲーションエリアの下で、例えばFrenchなどの言語をドロップダウンリストボックスから選択します。

ステップ 8
[プロパティ]パネルに移動します。

ステップ 9
`Value`テキストボックスに、`Bienvenue`と入力します。

コンテンツエリアで、ラベルに`Bienvenue`と表示されます。

ランタイム処理中の言語の変更

このセクションでは、ランタイム中に言語を変更するためのスクリプトを作成する手順について説明します。ユーザーがラベルをクリックするたびに表示言語が変更されます。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
[ナビゲーションエリア]で[データプール]に移動し、  をクリックします。

メニューが展開されます。

ステップ 2
メニューで`Integer`をクリックします。

`Integer`タイプのデータプールアイテムが追加されます。

ステップ 3
データプールアイテムの名前を`SelectedLanguage`に変更します。

ステップ 4
ナビゲーションエリアで、`Label 1`ラベルをクリックします。

ステップ 5
プロパティパネルで[ウィジェット機能プロパティ]に移動し、[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 6
[使用可能なウィジェット機能]から[入力処理]カテゴリを展開し、[タッチ押下]ウィジェット機能を選択します。

ステップ 7
[承認]をクリックします。

関連するウィジェット機能プロパティがラベルに追加され、[プロパティ]パネルに表示されます。

ステップ 8
`touchPressed`プロパティの横にある[編集...]をクリックします。

ステップ 9
既存のEB GUIDEスクリプトを次のコードに置き換えます。

```
function(v:touchId::int, v:x::int, v:y::int, v:fingerId::int)
```

```
{
    if (dp:SelectedLanguage == 0)  // Standard selected
    {
     f:language(l:German)
     dp:SelectedLanguage = 1
    }
    else if (dp:SelectedLanguage == 1)  // German selected
    {
     f:language(l:French)
     dp:SelectedLanguage = 2
    }
    else if (dp:SelectedLanguage == 2)  // French selected
    {
     f:language(l:Standard)
     dp:SelectedLanguage = 0
    }
    false
}
```

ステップ 10
[承認]をクリックします。

ラベルを設定し、ランタイム中にラベルの言語を変更するEB GUIDEスクリプトを作成しました。

結果:

文字列型のデータプールアイテムが、EB GUIDEモデルに追加されました。データプールアイテムには、言語ごとに違う値が入っています。英語の値は`Welcome`、ドイツ語の値は`Willkommen`、フランス語の値は`Bienvenue`です。データプールアイテムは、ラベルの`text`プロパティにリンクされています。EB GUIDEモデルの言語を変更するたびに、ラベルのテキストも変わります。

11.7. チュートリアル: 3Dグラフィックの操作

EB GUIDE Studioでは、EB GUIDEモデルに3Dグラフィックを使用できます。

このセクションでは、EB GUIDEモデルに3Dグラフィックを追加する手順について説明します。扱うのは3Dグラフィックをインポートする方法と、インポートした3Dグラフィックの外観をウィジェット機能で変更する方法です。最高の結果を得られるよう、ここで説明する順番どおりに操作してください。

注記

3Dグラフィック

3Dグラフィックファイルを作成するには、サードパーティの3Dモデリングソフトウェアを使用します。

OpenGL ES 2.0およびDirectX 11レンダラーのみが3Dグラフィックを表示できます。サポートされている3Dグラフィック形式は、COLLADA (`.dae`)およびFilmbox (`.fbx`)です。失敗を防ぐため、Filmbox形式を使用します。

メッシュにテクスチャを適用するには、3Dオブジェクトにテクスチャ座標を設定する必要があります。テクスチャ座標の追加には、サードパーティの3Dモデリングソフトウェアを使用してください。

所要時間: 15分

3Dグラフィックのインポート

このセクションでは、EB GUIDE Studioプロジェクトに3Dグラフィックファイルをインポートする手順について説明します。

前提条件:

- コンテンツエリアに、[メイン]ステートマシンが表示されていること。
- [メイン]ステートマシンに、初期ステートとビューステートが含まれていること。
- 初期ステートにビューステートへの遷移があること。
- 3Dグラフィックファイルが使用可能になっていること。

ステップ 1
コンテンツエリアで、ビューステートをダブルクリックします。

ビューがコンテンツエリアに表示されます。

ステップ 2
[ツールボックス]からシーングラフをビューにドラッグしてドロップします。

ビューに空の矩形が表示されます。

ステップ 3
シーングラフの名前を`My3DGraphic`に変更します。

ステップ 4
[プロパティ]パネルで[ファイルのインポート]をクリックします。

ダイアログが開きます。

ステップ 5
3Dグラフィックファイルが格納されているディレクトリに移動します。

ステップ 6
3Dグラフィックファイルを選択します。

ステップ 7
[開く]をクリックします。

インポートが開始します。[インポートに成功しました]ダイアログが表示されます。ここでインポートログファイルを確認することもできます。

ステップ 8
[OK]をクリックします。

ビューに3Dグラフィックが表示されます。インポートしたウィジェットツリーが、シーングラフを親ノードにしてナビゲーションエリアに表示されます。`My3DGraphic`には、3Dグラフィックファイルのコンテンツに応じて少なくとも材質、カメラ、そして複数の子ウィジェットを持つメッシュ1つ`RootNode`が含まれます。

ウィジェットの追加

このセクションでは、3Dグラフィックに別の光源を追加する手順について説明します。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで`RootNode`を展開します。

ステップ 2
[ツールボックス]から指向性ライトをドラッグし、`RootNode`にドロップします。

`My3DGraphic`に指向性ライトが追加されました。指向性ライトは、3Dシーン内で`RootNode`と同じ変形プロパティを持ちます。

ステップ 3
光源を追加して`RootNode`シーングラフ以外のデフォルトのウィジェットプロパティで配置するには、次のようにします。

ステップ 3.1
[ツールボックス]からグラフノードをドラッグし、`RootNode`にドロップします。

ステップ 3.2
シーングラフノードの名前を`MyLight`に変更します。

ステップ 3.3
[ツールボックス]から指向性ライトをドラッグし、`MyLight`にドロップします。

`My3DGraphic`に指向性ライトが追加されました。指向性ライトの配置を変更するには、`MyLight`のプロパティを変更します。

メッシュの変更

前提条件:

- 前のセクションの手順を完了していること。
- `$GUIDE_PROJECT_PATH/<project name>/resources/<3D graphic name>`ディレクトリに、追加の`.ebmesh`ファイルが含まれていること。

ステップ 1
ナビゲーションエリアで`Mesh 1`をクリックし、[プロパティ]パネルに移動します。

ステップ 2
`mesh`ドロップダウンリストボックスから`.ebmesh`ファイルを選択します。

ビューに表示されているシーングラフのメッシュが変更されます。

テクスチャの変更

ここからは、3Dグラフィックにテクスチャを追加し、変更する手順について説明します。

前提条件:

- 前のセクションの手順を完了していること。
- `$GUIDE_PROJECT_PATH/<project name>/resources/<3D graphic name>`ディレクトリに、`.png`または`.jpg`イメージファイルが含まれていること。

ステップ 1
ナビゲーションエリアで材質をクリックし、[プロパティ]パネルに移動します。

ステップ 2
[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 3
[使用可能なウィジェット機能]で[3D]カテゴリを展開して、例えば[ディフューズテクスチャ]などのテクスチャウィジェット機能を選択します。

ステップ 4
[承認]をクリックします。

関連するウィジェット機能プロパティが材質に追加され、[プロパティ]パネルに表示されます。

ステップ 5
[プロパティ]パネルで、`diffuseTexture`ドロップダウンリストボックスからイメージを選択します。

ビューに表示されているシーングラフのテクスチャが変更されます。

注記

[3D]ウィジェット機能の使用

この説明は、カテゴリ[3D]にある以下のウィジェット機能に該当します。

- [アンビエントテクスチャ]
- [エミッシブテクスチャ]
- [ライトマップテクスチャ]
- [ノーマルマップテクスチャ]
- [不透明テクスチャ]
- [リフレクションテクスチャ]
- [スペキュラテクスチャ]

3Dオブジェクトの複数回表示

このセクションでは、3Dグラフィックの3Dオブジェクトを複数回表示するために別のカメラを追加する手順について説明します。同じオブジェクトを別の視点から移すことが可能になります。

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ナビゲーションエリアで`My3DGraphic`をクリックし、[プロパティ]パネルに移動します。

ステップ 2
`width`テキストボックスに`800`、`height`テキストボックスに`480`と入力します。

`My3DGraphic`シーングラフにビューのサイズが設定されます。

ステップ 3
ナビゲーションエリアで`RootNode`および`Camera001`を展開します。

ステップ 4
`Camera 1`をクリックし、[プロパティ]パネルに移動します。

ステップ 5
[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 6
[使用可能なウィジェット機能]で、[3D]カテゴリを展開して[カメラビューポート]を選択します。

ステップ 7
[承認]をクリックします。

関連するウィジェット機能プロパティが`Camera 1`に追加され、[プロパティ]パネルに表示されます。

ステップ 8
ナビゲーションエリアでRootNodeに移動します。

ステップ 9
[ツールボックス]からカメラをドラッグし、シーングラフノードCamera001にドロップします。

2つ目のカメラが追加されました。

ステップ 10
Camera 2をクリックし、[プロパティ]パネルに移動します。

ステップ 11
nearPlane、farPlane、fieldOfViewテキストボックスに、Camera 1と同じ値を入力します。

Camera 1とCamera 2の視点の位置が同じになりました。

ステップ 12
[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 13
[使用可能なウィジェット機能]で、[3D]カテゴリを展開して[カメラビューポート]を選択します。

ステップ 14
[承認]をクリックします。

関連するウィジェット機能プロパティがCamera 2に追加され、[プロパティ]パネルに表示されます。

ステップ 15
[プロパティ]パネルで、viewportXおよびviewportYテキストボックスに100と入力します。

ビューで3Dオブジェクトが2回、x座標とy座標を変えて表示されます。

12. 参照

この章では、パラメータ、プロパティ、識別子などのリストと表を示します。

12.1. Androidイベント

AndroidイベントはSystemNotificationsイベントグループに属し、イベントグループIDは13です。

表12.1 Androidイベント

イベントID	名前	説明
1	RendererEnabled	Androidライフサイクル管理がレンダラーを停止または起動するとき、アプリケーションによって送信されます パラメータ: ► enabled: trueである場合、レンダラーは有効になります。falseである場合、レンダラーはスリープモードに設定されます。
2	setKeyboardVisibility	仮想キーボードが表示されるように設計されている場合、EB GUIDEモデルによって送信されます パラメータ: ► visibility: trueである場合、仮想キーボードが表示されるようになります。falseである場合、表示されません。
3	onKeyboardVisibilityChanged	仮想キーボードが表示されている場合、アプリケーションによって送信されます パラメータ: ► visibility: trueである場合、仮想キーボードが表示されます。falseである場合、表示されません。
4	onLayoutChanged	画面の表示領域が変更されたとき、アプリケーションによって送信されます

イベントID	名前	説明
		パラメータ(ピクセル単位): ▶ x: 表示画面領域の左上隅のx座標 ▶ y: 表示画面領域の左上隅のy座標 ▶ width: 表示画面領域の幅 ▶ height: 表示画面領域の高さ

12.2. データプールアイテム

表12.2 データプールアイテムのプロパティ

プロパティ名	説明
Value	データプールアイテムの初期値 EB GUIDE Studioから値が提供された場合、エクスポータはプロパティ値をEB GUIDE GTFに提供します。それ以外の場合、EB GUIDE GTFはシステム起動時にプロパティ値を初期化します。
Read-only	trueである場合、内部通信のみを使用できます。Valueはランタイム中は一定で、言語の切り替えで再初期化した場合にのみ変わります。 falseである場合、外部通信を使用できます。Valueはランタイム中に変わることがあります。
Reader ID	リーダーの通信コンテキストがデータプールアイテムにアクセスするために使用するアドレス。Reader IDが定義されていない場合、自動的に計算されます。EB GUIDEモデルを変更すると、データプールアイテムのReader IDが変わることがあります。
Reader context	値の変更に関する通知を受け取り、その変更に対して応答する通信コンテキスト。
Writer ID	ライターの通信コンテキストがデータプールアイテムにアクセスするために使用するアドレス。Writer IDが定義されていない場合、自動的に計算されます。EB GUIDEモデルを変更すると、データプールアイテムのWriter IDが変わることがあります。
Writer context	新しい値を書き込む通信コンテキスト
Windowed	リストでのみ使用可能 trueである場合、EB GUIDE TFはウィンドウ表示リストの操作モードでデータプールアイテムを処理します。初期化でデフォルト値は使用されません。 falseである場合、EB GUIDE TFは標準リストの操作モードでデータプールアイテムを処理します。

12.3. データタイプ

このセクションでは、EB GUIDEのデータタイプについて説明します。以下に示すタイプのユーザー定義プロパティおよびデータプールアイテムを追加できます。

12.3.1. メッシュ

メッシュは3Dオブジェクトの形状を定義します。

使用可能な演算子は次のとおりです。

- 代入(書き込み可能なプロパティ) (=)

メッシュはリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.2. ブール値

ブール値プロパティは値trueおよびfalseを持つことができます。

使用可能な演算子は次のとおりです。

- 等しい(==)
- 等しくない(!=)
- 否定(!)
- AND (&&)
- OR (||)
- 代入(書き込み可能なプロパティ) (=)

ブール値プロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.3. 色

色はRGBA8888フォーマットで格納されます。

使用例: 透過性がない赤色は(255, 0, 0, 255)です。

使用可能な演算子は次のとおりです。

- 等しい(==)
- 等しくない(!=)
- 代入(書き込み可能なプロパティ) (=)

色プロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.4. 条件スクリプト

条件スクリプトは初期化時およびトリガー時の反応に使用されます。条件スクリプトを編集するとき、コンテンツエリアは次のセクションに分割されます。

- [Trigger]ドロップダウンリストボックスには、[On trigger]スクリプトの実行をトリガーするイベントとデータプールアイテムのリストが含まれます。
- [On trigger]スクリプトはイベントをトリガーしたのち、またはデータプールアイテムの値を更新したのち、初期化時に呼び出されます。

  [On trigger]スクリプトのパラメータは、スクリプトを実行する要因を表します。

  [On trigger]スクリプトの戻り値によって、プロパティの変更通知が次のように制御されます。

  trueである場合、変更通知をトリガーします。

  falseである場合、変更通知をトリガーしません。

12.3.5. フロート

浮動小数点数データタイプは単精度32ビットIEEE 754値を表します。

使用可能な演算子は次のとおりです。

- 等しい(==)
- 等しくない(!=)
- より大きい(>)
- 以上(>=)
- 未満(<)
- 以下(<=)
- 加算(+)
- 減算(-)

- 乗算(*)
- 除算(/)
- 代入(書き込み可能なプロパティ) (=)

浮動小数点数プロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.6. フォント

EB GUIDEプロジェクトにフォントを追加するには、フォントファイルを$GUIDE_PROJECT_PATH/<project name>/resources

使用可能な演算子は次のとおりです。

- 代入(書き込み可能なプロパティ) (=)

フォントプロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.7. イメージ

EB GUIDEプロジェクトにイメージを追加するには、イメージファイルを$GUIDE_PROJECT_PATH/<project name>/resourcesディレクトリ内にコピーします。$GUIDE_PROJECT_PATH/<project name>/resources

使用可能な演算子は次のとおりです。

- 代入(書き込み可能なプロパティ) (=)

イメージプロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.8. 整数

EB GUIDEは符号付き32ビット整数をサポートしています。

使用可能な演算子は次のとおりです。

- 等しい(==)
- 等しくない(!=)
- より大きい(>)

- 以上(>=)
- 未満(<)
- 以下(<=)
- 加算(+)
- 減算(-)
- 乗算(*)
- 除算(/)
- 剰余(%)
- 代入(書き込み可能なプロパティ) (=)

整数プロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.3.9. リスト

EB GUIDEは同じデータタイプを持つ値のリストをサポートしています。

以下のリストタイプがあります。

- メッシュリスト
- ブール値リスト
- 色リスト
- 浮動小数点数リスト
- フォントリスト
- イメージリスト
- 整数リスト
- 文字列リスト

以下のタイプはリストで使用できません。

- リスト
- プロパティの参照
- リスト要素の参照

使用可能な演算子は次のとおりです。

- 長さ: (長さ)
- 要素アクセサ: ([])

12.3.10. 文字列

EB GUIDEは、**Hello world**などの文字列をサポートしています。

使用可能な演算子は次のとおりです。

- 等しい（大文字と小文字を区別）(==)
- 等しくない（大文字と小文字を区別）(!=)
- 等しい(大文字と小文字を区別、ASCII範囲内のみ) (=Aa=)
- より大きい(>)
- 以上(>=)
- 未満(<)
- 以下(<=)
- 連結(+)
- 代入(書き込み可能なプロパティ) (=)

文字列プロパティをリストに格納できます。リストの詳細については、12.3.9「リスト」をご覧ください。

12.4. EB GUIDEスクリプト

12.4.1. EB GUIDEスクリプトのキーワード

以下はEB GUIDEスクリプトの予約されているキーワードの一覧です。スクリプト内でこれらの単語を識別子として使用する場合は、単語を引用符で囲む必要があります。

キーワード	説明
`color:`	{0,255,255}などの色パラメータが後に続きます。
`dp:`	データプールアイテムが後に続きます。
`l:`	言語が後に続きます。
`else`	`if`条件が終了します。代わりに後続のブロックが実行されます。
`ev:`	イベントが後に続きます。
`f:`	ユーザー定義関数が後に続きます。
`false`	ブールリテラル値
`fire`	イベントを発行します

キーワード	説明
`if`	ブール式をテストする文が後に続きます。式がtrueである場合、文が実行されます。
`in`	ローカル変数宣言と変数の使用スコープの間のセパレータです `match_event`および`let`とともに使用されます。
`function`	関数を宣言します
`length`	プロパティの長さ
`let`	スコープ内でアクセス可能なローカル変数を宣言します
`list`	整数リストなどのリストタイプを宣言します
`match_event`	現在のイベントが想定されるイベントに対応しているかどうかを確認し、`let`などの変数を宣言します
`popup_stack`	動的ステートマシンの優先順位を定義する動的ステートマシンリスト
`sm:`	ステートマシンが後に続きます
`true`	ブールリテラル値
`unit`	voidタイプの値
`v:`	ローカル変数が後に続きます。
`view:`	ビューが後に続きます。
`while`	条件がtrueである限り、文を繰り返します

12.4.2. EB GUIDEスクリプトの演算子の優先順位

以下は、EB GUIDEスクリプトの演算子とそれらの優先順位および結合性の一覧です。演算子は上から下へ優先順位が高いものから低いものの順に示されています。

表12.3 EB GUIDEスクリプトの演算子の優先順位

演算子	結合性
(())、({})	なし
([])	なし
(->)	左
(.)	なし
(::)	左
長さ	なし
(&)	右
(!)、(-) 単項マイナス	右

演算子	結合性
(*)、(/)、(%)	左
(+)、(-)	左
(<)、(>)、(<=)、(>=)	左
(!=)、(==)、(=Aa=)	左
(&&)	左
(||)	左
(=)、(+=)、(-=)、(=>)	右
(,)	右
(;)	左

12.4.3. EB GUIDEスクリプト標準ライブラリ

この章では、EB GUIDEスクリプトのすべての関数の説明を示します。

12.4.3.1. EB GUIDEスクリプトの関数A

12.4.3.1.1. abs

この関数は整数xの絶対値を返します。

表12.4 absのパラメータ

パラメータ	タイプ	説明
x	整数	絶対値を返す数
<return>	整数	戻り値

12.4.3.1.2. absf

この関数は浮動小数点数xの絶対値を返します。

表12.5 absfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	絶対値を返す数
<return>	浮動小数点数	戻り値

12.4.3.1.3. acosf

この関数はxの逆余弦の主値を返します。

表12.6 acosfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	逆余弦を返す数
<return>	浮動小数点数	戻り値

12.4.3.1.4. animation_before

この関数は、実行中のアニメーションが時間軸上の指定された点を過ぎたかどうかを確認します。

表12.7 animation_beforeのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
time	整数	時間軸上の点
<return>	ブール値	trueである場合、アニメーションはまだ時間軸上の点を過ぎていません。

12.4.3.1.5. animation_beyond

この関数は、実行中のアニメーションが時間軸上の指定された点を過ぎたかどうかを確認します。

表12.8 animation_beyondのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
time	整数	時間軸上の点
<return>	ブール値	trueである場合、アニメーションは時間軸上の点を過ぎています。

12.4.3.1.6. animation_cancel

この関数はアニメーションをキャンセルし、編集したプロパティを現在のステートのままにします。

表12.9 animation_cancelのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
<return>	ブール値	trueである場合、関数が成功しました。

12.4.3.1.7. animation_cancel_end

この関数はアニメーションをキャンセルし、可能であれば、編集したプロパティを終了ステートに設定します。

表12.10 animation_cancel_endのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
<return>	ブール値	trueである場合、関数が成功しました。

12.4.3.1.8. animation_cancel_reset

この関数はアニメーションをキャンセルし、可能であれば、編集したプロパティを初期ステートにリセットします。

表12.11 animation_cancel_resetのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
<return>	ブール値	trueである場合、関数が成功しました。

12.4.3.1.9. animation_pause

この関数はアニメーションを一時停止します。

表12.12 animation_pauseのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
<return>	ブール値	trueである場合、関数が成功しました。

12.4.3.1.10. animation_play

この関数はアニメーションを開始または続行します。

表12.13 animation_playのパラメータ

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション
<return>	ブール値	trueである場合、アニメーションはまだ実行していません。

12.4.3.1.11. `animation_reverse`

この関数はアニメーションを逆方向に再生します。

表12.14 `animation_reverse`のパラメータ

パラメータ	タイプ	説明
`animation`	GtfTypeRecord	操作するアニメーション
`<return>`	ブール値	trueである場合、アニメーションはまだ実行していません。

12.4.3.1.12. `animation_running`

この関数はアニメーションが現在実行中かどうかを確認します。

表12.15 `animation_running`のパラメータ

パラメータ	タイプ	説明
`animation`	GtfTypeRecord	操作するアニメーション
`<return>`	ブール値	trueである場合、アニメーションは実行中です。

12.4.3.1.13. `animation_set_time`

この関数はアニメーションの現在の時間を設定します。アニメーションをスキップまたは再生するために使用できます。

表12.16 `animation_set_time`のパラメータ

パラメータ	タイプ	説明
`animation`	GtfTypeRecord	操作するアニメーション
`time`	整数	時間
`<return>`	ブール値	trueである場合、関数が成功しました。

12.4.3.1.14. `asinf`

この関数はxの逆正弦の主値を計算します。

表12.17 `asinf`のパラメータ

パラメータ	タイプ	説明
`x`	浮動小数点数	逆正弦を返す数
`<return>`	浮動小数点数	戻り値

12.4.3.1.15. atan2f

この関数は2つの引数の符号を使用して結果の象限を決定し、xの逆正接の主値を計算します。

表12.18 atan2fのパラメータ

パラメータ	タイプ	説明
y	浮動小数点数	引数y
x	浮動小数点数	引数x
<return>	浮動小数点数	戻り値

12.4.3.1.16. atan2i

この関数は2つの引数の符号を使用して結果の象限を決定し、xの逆正接の主値を計算します。

表12.19 atan2iのパラメータ

パラメータ	タイプ	説明
y	整数	引数y
x	整数	引数x
<return>	浮動小数点数	戻り値

12.4.3.1.17. atanf

この関数はxの逆正接の主値を計算します。

表12.20 atanfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	逆正接を返す数
<return>	浮動小数点数	戻り値

12.4.3.2. EB GUIDEスクリプトの関数C～H

12.4.3.2.1. ceil

この関数は引数以上の最小整数値を返します。

表12.21 ceilのパラメータ

パラメータ	タイプ	説明
value	浮動小数点数	丸める値
<return>	整数	丸めた値

12.4.3.2.2. changeDynamicStateMachinePriority

この関数は動的ステートマシンの優先順位を変更します。

表12.22 changeDynamicStateMachinePriorityのパラメータ

パラメータ	タイプ	説明
state		動的ステートマシンリストを持つステート
sm	整数	動的ステートマシン
priority	整数	リスト内の動的ステートマシンの優先順位

12.4.3.2.3. character2unicode

この関数は文字列内の最初の文字のUnicode値を返します。

表12.23 character2unicodeのパラメータ

パラメータ	タイプ	説明
str	文字列	入力文字列
<return>	整数	Unicodeとしての文字 エラーの場合は0

12.4.3.2.4. clearAllDynamicStateMachines

この関数は動的ステートマシンリストから動的ステートマシンをすべて削除します。

表12.24 clearAllDynamicStateMachinesのパラメータ

パラメータ	タイプ	説明
state		動的ステートマシンリストを持つステート

12.4.3.2.5. color2string

この関数は色を8桁の16進数値に変換します。

表12.25 color2stringのパラメータ

パラメータ	タイプ	説明
value	色	文字列に変換する色
<return>	文字列	プレフィックスとして#が付いた16進数の文字列としてフォーマットされた色

注記

フォーマットの例
返される文字列のフォーマットは#RRGGBBAAで、赤、緑、青、アルファの各色チャネルを表す2桁の数が含まれます。

例えば、不透明の純赤色は「#ff0000ff」に変換され、半透明の純緑色は「#00ff007f」に変換されます。

12.4.3.2.6. cosf

この関数はxの余弦を返します。このxはラジアン単位で指定します。

表12.26 cosfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	余弦を返す数
<return>	浮動小数点数	戻り値

12.4.3.2.7. deg2rad

この関数は角度を度数からラジアンに変換します。

表12.27 deg2radのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	度数からラジアンに変換する角度
<return>	浮動小数点数	戻り値

12.4.3.2.8. expf

この関数はe(自然対数の底)のx乗の値を返します。

表12.28 expfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	指数

パラメータ	タイプ	説明
<return>	浮動小数点数	戻り値

12.4.3.2.9. float2string

この関数は単純浮動小数点数を文字列に変換します。

表12.29 float2stringのパラメータ

パラメータ	タイプ	説明
value	浮動小数点数	文字列に変換する値
<return>	文字列	文字列としてフォーマットされた浮動小数点値

12.4.3.2.10. floor

この関数はパラメータ値以下の最大整数値を返します。

表12.30 floorのパラメータ

パラメータ	タイプ	説明
value	浮動小数点数	丸める値
<return>	整数	丸めた値

12.4.3.2.11. focusNext

この関数はフォーカスマネージャーのフォーカスを次のフォーカス可能な要素へ強制的に進めます。

表12.31 focusNextのパラメータ

パラメータ	タイプ	説明
<return>	void	

12.4.3.2.12. focusPrevious

この関数はフォーカスマネージャーのフォーカスを前のフォーカス可能な要素へ強制的に戻します。

表12.32 focusPreviousのパラメータ

パラメータ	タイプ	説明
<return>	void	

12.4.3.2.13. format_float

この関数は浮動小数点値をフォーマットします。

表12.33 format_floatのパラメータ

パラメータ	タイプ	説明
format	文字列	次の構造の文字列: %[フラグ] [幅] [.精度]タイプ ▶ フラグ: 出力配置、および符号、空白、先行ゼロ、小数点、8進数と16進数のプレフィックスの出力を制御するオプションの1文字または複数文字。 ▶ 幅: 出力される最小文字数を指定するオプションの10進数。 ▶ 精度: 小数点文字の後の有効桁数または桁数を指定するオプションの10進数。 ▶ タイプ: 関連引数が文字、文字列、整数、または浮動小数点数として解釈されるかどうかを判別する必須の変換指定子文字。
useDotAsDelim-iter	ブール値	区切り記号を定義します。 使用可能な値: ▶ true: ドットを区切り記号として使用します。 ▶ false: カンマを区切り記号として使用します。
value	浮動小数点数	フォーマットする数

警告

C++のprintf仕様を順守してください。
formatパラメータは、C++のprintf仕様に従って定義されます。

この仕様に準拠しない値を使用すると、予期しない動作発生する可能性があります。

例えば、format_floatに許可されるタイプは、f、a、g、およびeであり、1文字のタイプしか許可されません。

12.4.3.2.14. format_int

この関数は整数値をフォーマットします。

表12.34 format_intのパラメータ

パラメータ	タイプ	説明
format	文字列	次の構造の文字列:

パラメータ	タイプ	説明
		%[フラグ] [幅] [.精度]タイプ ▶ フラグ: 出力配置、および符号、空白、先行ゼロ、小数点、8進数と16進数のプレフィックスの出力を制御するオプションの1文字または複数文字。 ▶ 幅: 出力される最小文字数を指定するオプションの10進数。 ▶ 精度: 出力される最小桁数を指定するオプションの10進数。 ▶ タイプ: 関連引数が文字、文字列、整数、または浮動小数点数として解釈されるかどうかを判別する必須の変換指定子文字。
value	整数	フォーマットする数

警告

C++のprintf仕様を順守してください。
formatパラメータは、C++のprintf仕様に従って定義されます。

この仕様に準拠しない値を使用すると、予期しない動作発生する可能性があります。

例えば、format_intに許可されるタイプは、d、i、o、x、およびuであり、1文字のタイプしか許可されません。

12.4.3.2.15. getLineCount

この関数はウィジェットの行数を返します。

表12.35 getLineCountのパラメータ

パラメータ	タイプ	説明
widget	ウィジェット	評価するウィジェット
<return>	整数	行数

12.4.3.2.16. getTextHeight

この関数はテキストのフォントリソースに関する高さを返します。

表12.36 getTextHeightのパラメータ

パラメータ	タイプ	説明
text	文字列	評価するテキスト
font	フォント	評価するフォント
<return>	整数	テキストの高さ

12.4.3.2.17. `getTextLength`

この関数はテキスト内の文字数を返します。

表12.37 `getTextLength`のパラメータ

パラメータ	タイプ	説明
`text`	文字列	評価するテキスト
`<return>`	整数	テキスト内の文字数

12.4.3.2.18. `getTextWidth`

この関数はテキストのフォントリソースに関する幅を返します。

表12.38 `getTextWidth`のパラメータ

パラメータ	タイプ	説明
`text`	文字列	評価するテキスト
`font`	フォント	評価するフォント
`<return>`	整数	テキストの幅

12.4.3.2.19. `has_list_window`

この関数はタイプリストのデータプールアイテムでインデックスが有効かどうかを確認します。ウィンドウ表示リストでは、インデックスが少なくとも1つのウィンドウ内にあるかどうかも確認します。

表12.39 `has_list_window`のパラメータ

パラメータ	タイプ	説明
`itemId`	dp_id	タイプリストのデータプールアイテムのID
`index`	整数	データプールアイテム内のインデックス
`<return>`	ブール値	trueである場合、データプールアイテム内のインデックスは有効であり、少なくとも1つのウィンドウ内にあります。

12.4.3.2.20. `hsba2color`

この関数はHSB/HSV色をGTF色に変換します。

表12.40 `hsba2color`のパラメータ

パラメータ	タイプ	説明
`hue`	整数	0～360の色の値(度単位)

パラメータ	タイプ	説明
saturation	整数	彩度(パーセント単位)
brightness	整数	明度(パーセント単位)
alpha	整数	0 (完全透明)～255 (不透明)の間のアルファ値
<return>	色	アルファ値が適用された結果のGTF色

12.4.3.3. EB GUIDEスクリプトの関数I～R

12.4.3.3.1. int2float

この関数は浮動小数点値に変換された整数値を返します。

表12.41 int2floatのパラメータ

パラメータ	タイプ	説明
value	整数	浮動小数点数に変換する値
<return>	浮動小数点数	浮動小数点数に変換された整数値

12.4.3.3.2. int2string

この関数は単純整数を文字列に変換します。

表12.42 int2stringのパラメータ

パラメータ	タイプ	説明
value	整数	文字列に変換する値
<return>	文字列	文字列に変換された整数値(10進表記)

12.4.3.3.3. isDynamicStateMachineActive

この関数は動的ステートマシンリストを持つステートがアクティブかどうかを確認します。

表12.43 isDynamicStateMachineActiveのパラメータ

パラメータ	タイプ	説明
state		動的ステートマシンリストを持つステート
sm	整数	動的ステートマシン

12.4.3.3.4. `language`

この関数はすべてのデータプールアイテムの言語を切り替えます。

表12.44 `language`のパラメータ

パラメータ	タイプ	説明
`language`	languageType	切り替える言語(`f:language(l:German)`など)
`<return>`	void	

12.4.3.3.5. `localtime_day`

この関数はシステム時刻値からローカル時刻の日[1:31]を抽出します。

表12.45 `localtime_day`のパラメータ

パラメータ	タイプ	説明
`time`	整数	システム時刻が返すタイムスタンプ
`<return>`	整数	抽出された日

12.4.3.3.6. `localtime_hour`

この関数はシステム時刻値のローカル時刻から時を抽出します。

表12.46 `localtime_hour`のパラメータ

パラメータ	タイプ	説明
`time`	整数	システム時刻が返すタイムスタンプ
`<return>`	整数	抽出された時

12.4.3.3.7. `localtime_minute`

この関数はシステム時刻値のローカル時刻から分を抽出します。

表12.47 `localtime_minute`のパラメータ

パラメータ	タイプ	説明
`time`	整数	システム時刻が返すタイムスタンプ
`<return>`	整数	抽出された分

12.4.3.3.8. localtime_month

この関数はシステム時刻値のローカル時刻から月[0:11]を抽出します。

表12.48 localtime_monthのパラメータ

パラメータ	タイプ	説明
time	整数	システム時刻が返すタイムスタンプ
<return>	整数	抽出された月

12.4.3.3.9. localtime_second

システム時刻値からローカル時刻の秒を抽出します。

表12.49 localtime_secondのパラメータ

パラメータ	タイプ	説明
time	整数	システム時刻が返すタイムスタンプ
<return>	整数	抽出された秒

12.4.3.3.10. localtime_weekday

この関数はシステム時刻値のローカル時刻から曜日[0:6]を抽出します。0は日曜日です。

表12.50 localtime_weekdayのパラメータ

パラメータ	タイプ	説明
time	整数	システム時刻が返すタイムスタンプ
<return>	整数	抽出された曜日

12.4.3.3.11. localtime_year

システム時刻値からローカル時刻の年を抽出します。

表12.51 localtime_yearのパラメータ

パラメータ	タイプ	説明
time	整数	システム時刻が返すタイムスタンプ
<return>	整数	抽出された年

12.4.3.3.12. `log10f`

この関数はxの10を底とする対数を返します。

表12.52 `log10f`のパラメータ

パラメータ	タイプ	説明
`x`	浮動小数点数	引数
`<return>`	浮動小数点数	戻り値

12.4.3.3.13. `logf`

この関数はxの自然対数を返します。

表12.53 `logf`のパラメータ

パラメータ	タイプ	説明
`x`	浮動小数点数	引数
`<return>`	浮動小数点数	戻り値

12.4.3.3.14. `nearbyint`

この関数は最も近い整数に丸めます。

表12.54 `nearbyint`のパラメータ

パラメータ	タイプ	説明
`value`	浮動小数点数	丸める値
`<return>`	整数	丸めた値

12.4.3.3.15. `popDynamicStateMachine`

この関数は優先順位キューの最上位の動的ステートマシンを削除します。

表12.55 `popDynamicStateMachine`のパラメータ

パラメータ	タイプ	説明
`state`		動的ステートマシンリストを持つステート
`sm`	整数	動的ステートマシン

12.4.3.3.16. powf

この関数はxのy乗の値を返します。

表12.56 powfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	引数x
y	浮動小数点数	引数y
<return>	浮動小数点数	戻り値

12.4.3.3.17. pushDynamicStateMachine

この関数は優先順位キューに動的ステートマシンを挿入します。

表12.57 pushDynamicStateMachineのパラメータ

パラメータ	タイプ	説明
state		動的ステートマシンリストを持つステート
sm	整数	動的ステートマシン
priority	整数	リスト内の動的ステートマシンの優先順位

12.4.3.3.18. rad2deg

この関数は角度をラジアンから度数に変換します。

表12.58 rad2degのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	引数
<return>	浮動小数点数	戻り値

12.4.3.3.19. rand

この関数は0～2^{31}-1の間の乱数値を取得します。

表12.59 randのパラメータ

パラメータ	タイプ	説明
<return>	整数	0～2^{31}-1の間の乱数

12.4.3.3.20. request_runlevel

この関数はフレームワークに対し、異なるランレベルへの切り替えを要求します。サポートされるランレベルは0のみで、これはプログラムをシャットダウンすることを意味します。

表12.60 request_runlevelのパラメータ

パラメータ	タイプ	説明
runlevel	整数	要求されるランレベル
<return>	void	

12.4.3.3.21. rgba2color

この関数はRGB色空間からGTF色に変換します。

表12.61 rgba2colorのパラメータ

パラメータ	タイプ	説明
red	整数	0～255の範囲の赤色座標
green	整数	0～255の範囲の緑色座標
blue	整数	0～255の範囲の青色座標
alpha	整数	0 (完全透明)～255 (不透明)の範囲のアルファ値
<return>	色	RGB色空間からGTF色に変換され、アルファ値が適用された色

12.4.3.3.22. round

この関数は最も近い整数に丸めます。ただし、中間の場合はゼロから遠ざかるように丸めます。

表12.62 roundのパラメータ

パラメータ	タイプ	説明
value	浮動小数点数	丸める値
<return>	整数	丸めた値

12.4.3.4. EB GUIDEスクリプトの関数S～W

12.4.3.4.1. seed_rand

この関数は乱数生成のシードを設定します。

表12.63 seed_randのパラメータ

パラメータ	タイプ	説明
seed	整数	乱数生成のシードとなる値
<return>	void	

12.4.3.4.2. sinf

この関数はxの正弦を返します。このxはラジアン単位で指定します。

表12.64 sinfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	引数
<return>	浮動小数点数	戻り値

12.4.3.4.3. sqrtf

この関数はxの負ではない平方根を返します。

表12.65 sqrtfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	引数
<return>	浮動小数点数	戻り値

12.4.3.4.4. string2float

この関数は文字列の最初の部分を浮動小数点数に変換します。

文字列の最初の部分の想定される形式は次のとおりです。

1. 先頭の空白(省略可能)
2. プラス(「+」)またはマイナス(「−」)符号(省略可能)
3. 次のいずれか:
   - 10進数
   - 16進数
   - 無限大
   - NAN (非数)

表12.66 string2floatのパラメータ

パラメータ	タイプ	説明
str	文字列	文字列値
<return>	浮動小数点数	戻り値

12.4.3.4.5. string2int

この関数は文字列の最初の部分を整数に変換します。入力が2147483647～-2147483648の範囲を超える場合、結果はこの範囲に切り捨てられます。文字列の先頭が数字ではない場合、関数は0を返します。

表12.67 string2intのパラメータ

パラメータ	タイプ	説明
str	文字列	文字列値
<return>	整数	戻り値

12.4.3.4.6. string2string

この関数は文字列をフォーマットします。

表12.68 string2stringのパラメータ

パラメータ	タイプ	説明
str	文字列	フォーマットする文字列
len	整数	文字列の最大長
<return>	文字列	言語の文字列

12.4.3.4.7. substring

この関数は文字列の部分文字列コピーを作成します。負の終了インデックスを使用できます。

例:

- substring("abc", 0, -1)は「abc」を返します。
- substring("abc", 0, -2)は「ab」を返します。

表12.69 substringのパラメータ

パラメータ	タイプ	説明
str	文字列	入力文字列

パラメータ	タイプ	説明
startIndex	整数	結果文字列の最初の文字インデックス
endIndex	整数	結果に含まれない最初の文字インデックス
<return>	文字列	言語の文字列

12.4.3.4.8. system_time

この関数は現在のシステム時刻を秒数で取得します。結果がlocaltime_*関数に渡されることを意図しています。

表12.70 system_timeのパラメータ

パラメータ	タイプ	説明
<return>	整数	秒数のシステム時刻

12.4.3.4.9. system_time_ms

この関数は現在のシステム時刻をミリ秒数で取得します。

表12.71 system_time_msのパラメータ

パラメータ	タイプ	説明
<return>	整数	ミリ秒数のシステム時刻

12.4.3.4.10. tanf

この関数はxの正接を返します。このxはラジアン単位で指定します。

表12.72 tanfのパラメータ

パラメータ	タイプ	説明
x	浮動小数点数	引数
<return>	浮動小数点数	戻り値

12.4.3.4.11. trace_dp

この関数はデータプールアイテムに関するデバッグ情報を、トレースログと接続ログに書き出します。

表12.73 trace_dpのパラメータ

パラメータ	タイプ	説明
itemId	dp_id	デバッグ情報をトレースするアイテムのデータプールID
<return>	void	

12.4.3.4.12. trace_string

この関数は文字列をトレースログと接続ログに書き出します。

表12.74 trace_stringのパラメータ

パラメータ	タイプ	説明
str	文字列	トレースするテキスト
<return>	void	

12.4.3.4.13. transformToScreenX

この関数はウィジェットとローカル座標を取得し、画面を基準にしたワールド座標系のx位置を返します。

表12.75 transformToScreenXのパラメータ

パラメータ	タイプ	説明
widget	ウィジェット	座標の基準にするウィジェット
localX	整数	ローカル座標のx位置
localY	整数	ローカル座標のy位置
<return>	整数	画面座標のx位置

12.4.3.4.14. transformToScreenY

この関数はウィジェットとローカル座標を取得し、画面を基準にしたワールド座標系の位置のY位置を返します。

表12.76 transformToScreenYのパラメータ

パラメータ	タイプ	説明
widget	ウィジェット	座標の基準にするウィジェット
localX	整数	ローカル座標のx位置
localY	整数	ローカル座標のy位置

パラメータ	タイプ	説明
<return>	整数	画面座標のY位置

12.4.3.4.15. transformToWidgetX

この関数はタッチ反応に提供されるウィジェットと画面座標を取得し、ウィジェットを基準にしたローカル座標系のX位置を返します。

表12.77 transformToWidgetXのパラメータ

パラメータ	タイプ	説明
widget	ウィジェット	座標の基準にするウィジェット
screenX	整数	画面座標のX位置
screenY	整数	画面座標のY位置
<return>	整数	ローカル座標のX位置

12.4.3.4.16. transformToWidgetY

この関数はタッチ反応に提供されるウィジェットと画面座標を取得し、ウィジェットを基準にしたローカル座標系のY位置を返します。

表12.78 transformToWidgetYのパラメータ

パラメータ	タイプ	説明
widget	ウィジェット	座標の基準にするウィジェット
screenX	整数	画面座標のX位置
screenY	整数	画面座標のY位置
<return>	整数	ローカル座標のY位置

12.4.3.4.17. trunc

この関数は常にゼロに近づくように最も近い整数に丸めます。

表12.79 truncのパラメータ

パラメータ	タイプ	説明
value	浮動小数点数	丸める値
<return>	整数	丸めた値

12.4.3.4.18. `widgetGetChildCount`

この関数は指定されたウィジェットの子ウィジェットの数を取得します。

表12.80 `widgetGetChildCount`のパラメータ

パラメータ	タイプ	説明
`widget`	ウィジェット	子ウィジェットの数を取得するウィジェット
`<return>`	整数	子ウィジェットの数

12.5. イベント

表12.81 イベントのプロパティ

プロパティ名	説明
`Name`	イベントの名前
`Event ID`	EB GUIDE TFがイベントを送受信するために使用する数値
`Event group`	イベントグループの名前 イベントグループには、EB GUIDE TFがイベントを送受信するために使用するIDがあります。

12.6. `gtfStartup.cfg`設定ファイル

12.6.1. マッピング規則の構造

EB GUIDE TFのマッピング規則は、信号1つとアクション1つで構成されます。信号とアクションはいずれもトークンで成り立っています。

トークンはスペースまたはタブで区切られます。ダブルスラッシュ(//)と行末の間にあるものはすべて無視されます。空の行は無視されます。二重引用符("")でくくられたテキストは、スペース、タブ、コメントを含んでいても1つのトークンとして解析されます。10進数および16進数形式がサポートされます。

表12.82 特殊文字の入力に使用されるエスケープシーケンス

エスケープシーケンス	特殊文字
\n	改行
\r	キャリッジリターン
\\	\

エスケープシーケンス	特殊文字
\"	"
\t	タブ

12.6.2. 信号

信号の最初のトークンは、信号タイプを定義するキーワードです。それ以降のトークンがタイプ固有のパラメータとなります。

表12.83 信号

キーワード	パラメータ	説明
INIT	なし	`gtfStartup.cfg`のロードが完了しました。
STARTUP	<ランレベル>	起動中に<ランレベル>(0...0xFFFF)に達しました。

12.6.3. アクション

アクションの最初のトークンは、アクションタイプを定義するキーワードです。それ以降のトークンがタイプ固有のパラメータとなります。

表12.84 アクション

キーワード	最初のパラメータ	それ以降のパラメータ	説明
MESSAGE	<MsgID>	メッセージパラメータ	<MsgID> (0...-0xFFFFFFFF)のメッセージと、指定されたメッセージパラメータを送信します。
LOAD (INIT信号のみでサポート)	FW_PATHまたは MODEL_PATH	<ファイルパス>	EB GUIDE GTF拡張ファイルを読み込み、含まれるモジュールを初期化します。FW_PATHとは、`GtfStartup.exe`実行ファイルを基準としたパスです。MODEL_PATHとは、`gtfStartup.cfg`ファイルを基準としたパスです。絶対パスの場合、

キーワード	最初のパラメータ	それ以降のパラメータ	説明
			キーワードFW_PATHおよびMODEL_PATHは同じ結果となります。
LOAD_ALL (INIT信号のみでサポート)	FW_PATHまたは MODEL_PATH	<ディレクトリパス>	ディレクトリ内のEB GUIDE GTF拡張ファイルをすべて読み込み、含まれるモジュールを初期化します。FW_PATHとは、`GtfStartup.exe`実行ファイルを基準としたパスです。MODEL_PATHとは、`gtfStartup.cfg`ファイルを基準としたパスです。絶対パスの場合、キーワードFW_PATHおよびMODEL_PATHは同じ結果となります。

メッセージパラメータは、キーワードトークンとそれに続く値トークンで構成されます。

表12.85 メッセージパラメータ

キーワード	値	説明
UINT8	数値	8ビットの符号なし整数
UINT16	数値	16ビットの符号なし整数
UINT32	数値	32ビットの符号なし整数
INT8	数値	8ビットの符号付き整数
INT16	数値	16ビットの符号付き整数
INT32	数値	32ビットの符号付き整数
STRBUF	文字列	シャットダウンが完了するまで利用できる、文字列を格納しているバッファへのポインター
STRING	文字列	文字列
FW_PATH	文字列	STRBUFと同様ですが、文字列は`GtfStartup.exe`実行ファイルへの相対パスとして解釈されます。
MODEL_PATH	文字列	STRBUFと同様ですが、文字列は`gtfStartup.cfg`ファイルへの相対パスとして解釈されます。

EB GUIDE TFメッセージシステムはランレベルおよびインターフェースの管理や、顧客から提供されるEB GUIDE TFモジュールやアプリケーションなどのフレームワークモジュールの構成に使用されます。事前定義済みメッセージのメッセージIDとパラメータについては、GtfMessageId.hファイルに記載されています。

ティップ

メッセージIDの操作

メッセージIDはメッセージグループに分類されています。つまり、GtfMessageId.hファイル内でメッセージID 401を検索しても何も見つかりません。代わりに次のラインを検索します。

```
#define GTF_MID_RANGE_GTF_DISPLAY 400
```

表示関連のメッセージは、すべてID 400を基準としたものです。文字列GTF_MID_RANGE_-GTF_DISPLAYを検索すると、メッセージID 401について以下のエントリーが表示されます。

```
#define GTF_MID_GTF_DISPLAY_CONNECT
 (uint32_t)(GTF_MID_RANGE_GTF_DISPLAY + 1)
```

注記

EB GUIDE TFの事前定義済みメッセージ

メッセージID範囲0～0xFFFFはEB GUIDE TFおよびEB GUIDE product lineのために予約されています。

メッセージID範囲0x10000～0xFFFFFFFFは管理することができます。

12.6.4. メッセージ

以下のメッセージでは、IDが0xDEADBEAFのGtfCoreModelを例として使用します。

- タイプマネージャーの設定

ランレベル	メッセージ ID	パラメータ
0	317	MODEL_PATH "types.bin"

このメッセージ(GTF_MID_GTF_TYPEMANAGER_CONFIG)は、types.binバイナリファイルの場所を指定します。

- GtfCoreRuntimeの作成

ランレベル	メッセージ ID	パラメータ
0	306	UINT32 0xDEADBEAF UINT8 0

このメッセージ(GTF_MID_GTF_CORE_CREATE_CORE_HOOK_ATF_WORKLOOP)は、通信コンテキストID(この例では0)によって識別されたスレッド内にGtfCoreRuntimeを作成します。異なる通信コンテキストIDのバリエーションについては、GtfMessageId.hをご覧ください。

注記

コンテキストID

ステートマシンの通信コンテキストは、EB GUIDE Studioで確認、設定できます。デフォルトの通信コンテキストIDは0です。

- GtfViewFactoryの作成

ランレベル	メッセージ ID	パラメータ
403	450	UINT32 0xBAADF00D MODEL_PATH "views.bin"

このメッセージ(GTF_MID_GTF_VIEWFACTORY_BINARY)は、views.binバイナリファイルからビューの説明を読み込んでGtfViewFactoryの一意のIDを0xBAADF00Dに定義するGtfViewFactoryを作成します。キーワードMODEL_PATHは、gtfStartup.cfgファイルを基準としたファイルパスを作成します。

- .bdf入力ファイルの設定

ランレベル	メッセージ ID	パラメータ
499	308	UINT32 0xDEADBEAF MODEL_PATH "datapool.bdf"

このメッセージ(GTF_MID_GTF_DATAPOOL_DECLARATIONS)は、IDが0xDEADBEAFのGtfCoreModelに、指定した.bdfファイルを読み込ませます。

- .bvf入力ファイルの設定

ランレベル	メッセージ ID	パラメータ
499	309	UINT32 0xDEADBEAF MODEL_PATH "datapool.bvf"

このメッセージ(GTF_MID_GTF_DATAPOOL_VALUES)は、IDが0xDEADBEAFのGtfCoreModelに、指定した.bvfファイルを読み込ませます。

- ステートマシンファイルの設定

ランレベル	メッセージ ID	パラメータ
499	311	UINT32 0xDEADBEAF MODEL_PATH "model.bin"

このメッセージ(GTF_MID_GTF_STATE_MACHINE_MODEL)は、IDが0xDEADBEAFのGtfCoreModelに、指定したバイナリステートマシンファイルmodel.binを読み込ませます。

- ステートマシンの有効化

ランレベル	メッセージ ID	パラメータ
501	320	UINT32 0xDEADBEAF STRBUF "Main"

このメッセージ(GTF_MID_GTF_ENABLE_STATE_MACHINE)は、IDが0xDEADBEAFのGtfCoreModelにある、Mainという名前のステートマシンを有効にします。

- ディスプレイの設定

ランレベル	メッセージ ID	パラメータ
0	511	STRBUF "Main" STRBUF "windowCaption" STRBUF "My Model"
0	512	STRBUF "Main" STRBUF "hwLayerId" INT32 0
0	513	STRBUF "Main" STRBUF "showWindowFrame" UINT8 1

このメッセージ(GTF_MID_GTF_DISPLAY_CONFIG_STRING、GTF_MID_GTF_DISPLAY_CONFIG_INT、GTF_MID_GTF_DISPLAY_CONFIG_BOOL)は、Mainという名前のステートマシンに属するディスプレイに適用されます。メッセージ511は文字列オプション、メッセージ512は整数オプション、メッセージ513はブール値オプションに使用されます。

- リソース設定ファイルの設定

ランレベル	メッセージ ID	パラメータ
499	312	UINT32 0xDEADBEAF MODEL_PATH "resources.bin"

このメッセージ(GTF_MID_GTF_RESOURCE_MODEL)は、IDが0xDEADBEAFのGtfCoreModelに、バイナリリソース設定ファイルresources.binを読み込ませます。

- デバッグデータベースの設定(オプション)

ランレベル	メッセージ ID	パラメータ
499	318	UINT32 0xDEADBEAF MODEL_PATH "debug.bin"

このメッセージ(GTF_MID_GTF_DEBUGDATABASE_CONFIG)は、デバッグデータベースファイルdebug.binを含みます。

- サービスマッパーTCP/IPポートの設定

ランレベル	メッセージ ID	パラメータ
0	305	UINT16 5456

このメッセージ(GTF_MID_GTF_SERVICE_MAPPER)は、EB GUIDE TFのデバッガーサービスに、TCP/IPポート5456の接続待機を開始させます。

- RomFSコンテナーの読み込み

ランレベル	メッセージ ID	パラメータ
0	701	MODEL_PATH "container.romfs"

このメッセージ(GTF_MID_GTF_FILESYSTEM_LOAD_ROMFS)は、container.romfsによって指定されたRomFSコンテナーをEB GUIDE TFに読み込ませます。

- EB GUIDE TFによるフォントファイルのアクセス方法の設定(オプション)

ランレベル	メッセージ ID	パラメータ
0	510	UINT8 1

このメッセージ(GTF_MID_GTF_FREETYPE_STREAM_TYPE)は、EB GUIDE GTFのフォントアクセス要素を設定します。UINT8パラメータの値が0の場合、ROMマップ済みファイルを使用します。UINT8パラメータの値が1の場合、プレーンファイルI/Oを使用します。

注記

ROMマップ済みファイルアプローチとプレーンファイルI/Oアプローチの比較
一般的に、ROMマップ済みファイルアプローチのほうがパフォーマンスが優れています。ただしQNXなど一部のシステムでは、プレーンファイルI/Oアプローチよりメモリを多く消費してしまいます。プレーンファイルI/Oは、概してROMマップ済みファイルアプローチよりもメモリの消費量が抑えられます。ただし、パフォーマンスは低くなる場合があります。

- EB GUIDEスクリプトトレース関数の出力の無効化

ランレベル	メッセージ ID	パラメータ
0	321	UINT32 0xDEADBEAF

このメッセージは、EB GUIDEスクリプトコードにおけるf:trace_string()とf:trace_dp()の出力を抑制します。

- モニタリング情報のデバッグ

ランタイム中にモニタリング情報を表示する場合、一部のレンダラーはEB GUIDEモデルから独立した追加リソースを必要とします。こうしたリソースは、EB GUIDE GTFランタイムディレクトリ内のmonitoringディレクトリにあります。起動設定でフレーム/秒(FPS)などのモニタリングディスプレイが設定されていない場合、こうしたリソースを安全に削除することができます。FPSモニタリングを有効にするには、レンダラーの操作モード値に適切なビットを設定します。詳細については、EB GUIDE GTFAPIマニュアルをご覧ください。

- FreeTypeキャッシュの設定(オプション)

ランレベル	メッセージ ID	パラメータ
0	515	UINT32 1000 UINT32 2000 UINT32 4000

このメッセージ(GTF_MID_GTF_FREETYPE_CACHE_CONFIGURATION)は、http://www.freetype.org/freetype2/docs/reference/ft2-cache_subsystem.htmlに記載されているとおりFreeTypeキャッシュパラメータを設定します。デフォルト値は以下のとおりです。

- max_faces: 1000
- max_sizes: 2000
- max_bytes: 4000 kB

EB GUIDE GTFのフォントサイズ処理方法が課題となるため、現時点ではft_sizeオブジェクトをft_faceと別にキャッシュすることはできません。max_sizesには、意味がわかる値を使用することを推奨します。

- リソースキャッシュの設定

ランレベル	メッセージ ID	パラメータ
0	520	UINT32 61441 UINT32 1048576

このメッセージ(GTF_MID_GTF_RENDERER_TEXTURE_CACHE)は、サイズが1,048,576バイトのデフォルトディスプレイID(61441)用のリソースキャッシュを作成します。メッセージを複数回追加すると、ディスプレイIDをさらに設定できます。ディスプレイIDとして0を割り当てると、この操作をしなければ設定されなかったすべてのディスプレイIDが、同一のリソースキャッシュを使用するよう設定されます。

12.7. シーン

表12.86 シーンのプロパティ

プロパティ名	説明
height	ハプティックステートマシンのビューがあるエリアの高さは、対象デバイス上でレンダリングされます。
width	ハプティックステートマシンのビューがあるエリアの幅は、対象デバイス上でレンダリングされます。
x	ハプティックステートマシンのビューがあるエリアのxオフセットは、対象デバイス上でレンダリングされます。
y	ハプティックステートマシンのビューがあるエリアのyオフセットは、対象デバイス上でレンダリングされます。
visible	trueである場合、ステートマシンおよび子ウィジェットが表示されます。
projectName	EB GUIDEプロジェクトの名前
windowCaption	ウィンドウフレームに表示されるテキスト

プロパティ名	説明
sceneID	入力処理などに使用できる一意のシーン識別子
maxFPS	再描画率(FPS = フレーム/秒) 再描画率を無制限にするには、0に設定します。
hwLayerID	現在のステートマシンにマップされる対象デバイスの表示上のハードウェアレイヤーのID
colorMode	使用可能な値: ► 1: 32ビット ► 2: 16ビット
multisampling	使用可能な値: ► オフ(= 0): マルチサンプリングなし ► 2x (=1): 2xマルチサンプリング ► 4x (=2): 4xマルチサンプリング
enableRemoteFramebuffer	trueである場合、シミュレーションウィンドウへの画面外のバッファーの転送が有効になります。
showWindowFrame	trueである場合、シミュレーションウィンドウにフレームが表示されます。このフレームを使用すると、ウィンドウをグラブして移動できます。
showWindow	trueである場合、Windowsベースのシステムでシミュレーション用の追加ウィンドウが開きます。
disableVSync	trueである場合、レンダラーの垂直同期が無効になります。
showFPS	使用可能な値: ► 0: FPS を表示しない ► 1: 画面にFPSを表示する ► 2: コンソールにFPSを表示する ► 3: 画面およびコンソールにFPSを表示する
Renderer	シーンのレンダラーを定義します。 使用可能な値: ► DirectX ► OpenGL

ティップ

マルチサンプリングの設定

マルチサンプリングの解像度を高くするほど、レンダリング結果の画質はよくなります。ただし、マルチサンプリングを行うと、特に対象デバイスで、レンダリングパフォーマンスが低下することに注意してください。高解像度の小型ディスプレイでは、マルチサンプリングの効果はほとんどありません。

マルチサンプリングなしで開始し、パフォーマンスが良好であれば、2xまたは4xのマルチサンプリングを試してください。マルチサンプリングを高くしても大きな差がない場合は、低い方の設定を使用してください。

12.8. EB GUIDE GTF がサポートするタッチスクリーンタイプ

サポートされるタイプは、対象デバイスによって異なります。

表12.87 EB GUIDE GTF がサポートするタッチスクリーンタイプ

値	説明	プラットフォーム
0	Galaxy	Linux
1	IMX WVGA	Linux
2	マウスインターフェイスに接続されたタッチスクリーン	すべて
3	一般的なプラットフォーム依存のタッチスクリーンインターフェイス	すべて
4	Lilliput 889GL	QNX
5	一般的なプラットフォーム依存のマルチタッチタッチスクリーンインターフェイス	Linux

12.9. ウィジェット

12.9.1. ビュー

表12.88 ビューのプロパティ

プロパティ名	説明
`name`	ウィジェットの名前

プロパティ名	説明
height	ピクセル単位のウィジェットの高さ
width	ピクセル単位のウィジェットの幅
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
x	ウィジェットのx座標
y	ウィジェットのy座標

ビューテンプレートには、表示遷移アニメーション用に追加のプロパティがあります。ビューが開始されると、開始アニメーションが実行されます。

表12.89 開始アニメーションのプロパティ

プロパティ名	説明
Entry animation	trueである場合、ビューテンプレートのインスタンスには開始アニメーションがあります。
Type	開始アニメーションのタイプ。例えば、[左からムーブイン], [中央からフェードイン]、[ビューをすぐに表示]などです。
Duration	開始アニメーションの時間(ミリ秒)
Delay	開始アニメーションの遅延(ミリ秒)
Play after exit animation	trueである場合、開始アニメーションの開始時間は前の終了アニメーションの時間に依存します。

ビューが終了すると、開始アニメーションが実行されます。

表12.90 終了アニメーションのプロパティ

プロパティ名	説明
Exit animation	trueである場合、ビューテンプレートのインスタンスには終了アニメーションがあります。
Type	終了アニメーションのタイプ。例えば、[上へムーブアウト], [中央へフェードアウト]、[ビューをすぐに非表示]などです。
Duration	終了アニメーションの時間(ミリ秒)
Delay	終了アニメーションの遅延(ミリ秒)

12.9.2. 基本ウィジェット

基本ウィジェットは5つあります。

- ラベル
- イメージ

- 四角形
- コンテナー
- インスタンシエータ

次のセクションでは、基本ウィジェットのプロパティをリストします。

注記 **一意の名前**

同じ親ウィジェットを持つ2つのウィジェットには一意の名前を使用してください。

注記 **負の値**

`height`および`width`プロパティに負の値を使用しないでください。EB GUIDE Studioは負の値を`0`として扱うため、各ウィジェットが描出されなくなってしまいます。

12.9.2.1. ラベル

ラベルは、テキストをビューに配置します。

表12.91 ラベルのプロパティ

プロパティ名	説明
`name`	ウィジェットの名前
`height`	ピクセル単位のウィジェットの高さ
`width`	ピクセル単位のウィジェットの幅
`visible`	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
`x`	親ウィジェットを基準にしたウィジェットのx座標
`y`	親ウィジェットを基準にしたウィジェットのy座標
`text`	ラベルに表示されるテキスト。テキストは、ウィジェットエリアに収まらない場合、デフォルトでは末尾で切り捨てられます。
`textColor`	テキストが表示される色
`font`	テキストが表示されるフォント
`horizontalAlign`	ラベルの境界内のテキストの水平方向の位置揃え
`verticalAlign`	ラベルの境界内のテキストの垂直方向の位置揃え

12.9.2.2. 四角形

四角形は、色の付いた四角形をウィジェットの寸法と座標でビューに描画します。

表12.92 四角形のプロパティ

プロパティ名	説明
name	ウィジェットの名前
height	ピクセル単位のウィジェットの高さ
width	ピクセル単位のウィジェットの幅
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
x	親ウィジェットを基準にしたウィジェットのx座標
y	親ウィジェットを基準にしたウィジェットのy座標
fillColor	四角形を塗りつぶす色

12.9.2.3. イメージ

イメージは、画像をビューに配置します。

表12.93 イメージのプロパティ

プロパティ名	説明
name	ウィジェットの名前
height	ピクセル単位のウィジェットの高さ
width	ピクセル単位のウィジェットの幅
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
x	親ウィジェットを基準にしたウィジェットのx座標
y	親ウィジェットを基準にしたウィジェットのy座標
image	ウィジェットが表示するイメージ
horizontalAlign	ウィジェットの境界内のイメージファイルの水平方向の位置揃え
verticalAlign	ウィジェットの境界内のイメージファイルの垂直方向の位置揃え

注記

サポートされているイメージファイル形式

使用可能なイメージファイル形式は、レンダラーの実装によって異なります。DirectX 11およびOpenGL ES 2.0は、.pngファイルおよび.jpegファイルをサポートしています。

12.9.2.4. コンテナー

コンテナーは、複数のウィジェットを子ウィジェットとして格納し、それらをグループ化します。

表12.94 コンテナーのプロパティ

プロパティ名	説明
name	ウィジェットの名前
height	ピクセル単位のウィジェットの高さ
width	ピクセル単位のウィジェットの幅
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
x	親ウィジェットを基準にしたウィジェットのx座標
y	親ウィジェットを基準にしたウィジェットのy座標

12.9.2.5. インスタンシエータ

インスタンシエータは、ランタイムにウィジェットインスタンスを作成します。インスタンシエータを使用して、リストまたは表をモデル化できます。インスタンシエータの子ウィジェットは、ランタイムに作成されるリストまたは表のラインテンプレートとして利用されます。

表12.95 インスタンシエータのプロパティ

プロパティ名	説明
name	ウィジェットの名前
height	ピクセル単位のウィジェットの高さ
width	ピクセル単位のウィジェットの幅
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
x	親ウィジェットを基準にしたウィジェットのx座標
y	親ウィジェットを基準にしたウィジェットのy座標
numItems	インスタンス化された子要素の数
lineMapping	どの子がどのラインのテンプレートであるかを定義します。

12.9.3. アニメーション

次のセクションでは、[アニメーション]カテゴリ内のウィジェットのプロパティをリストします。

12.9.3.1. アニメーション

アニメーションは、その親ウィジェットに作用します。アニメーションには、子ウィジェットとして最低でも1つの曲線が必要です。

表12.96 アニメーションのプロパティ

プロパティ名	説明
name	アニメーションの名前
alternating	アニメーションを繰り返し実行するかどうかを定義します。
repeat	繰り返しの回数。0は無限数を表します。
enabled	アニメーションを実行するかどうかを定義します。
scale	アニメーション時間を乗じる際の係数
onPause	アニメーションが一時停止した場合に実行される反応。パラメータ: 現在のアニメーション時間。
onPlay	アニメーションが開始または続行された場合に実行される反応。パラメータ: 開始時間および再生の方向(trueの場合は順方向、falseの場合は逆方向)
onTerminate	アニメーションが完了した場合に実行される反応。最初のパラメータ: アニメーション時間。2番目のパラメータ: 終了の理由。次のようにエンコードされています。 ► 0: アニメーションは完了します。 ► 1: アニメーションはキャンセルされます。これは、f:animation_cancelによってトリガーされます。 ► 2: ビュー遷移が原因でウィジェットが破棄されます。 ► 3: アニメーションは最後のステップにジャンプします。これは、f:animation_-cancel_endによってトリガーされます。 ► 4: アニメーションは最後のステップにジャンプしてからキャンセルされます。これは、f:animation_cancel_resetによってトリガーされます。

12.9.3.2. コンスタント曲線

コンスタント曲線は、定義された遅延時間の経過後にターゲット値を設定します。コンスタント曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.97 コンスタント曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。

プロパティ名	説明
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
value	結果のコンスタント値

12.9.3.3. 高速開始曲線

高速開始曲線は、値を定期的に設定します。最初の値は高速ですが、設定のたびに一定のペースで減速します。高速開始曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.98 高速開始曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
startt	初期値
end	最終値

12.9.3.4. 低速開始曲線

低速開始曲線は、値を定期的に設定します。最初の値は低速ですが、設定のたびに一定のペースで加速します。低速開始曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.99 低速開始曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延

プロパティ名	説明
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
start	初期値
end	最終値

12.9.3.5. 二次曲線

二次曲線は、二次関数曲線を使って値を定期的に設定します。二次曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.100 二次曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
velocity	結果を計算するための速度
acceleration	曲線の加速
constant	結果を計算するためのコンスタント値

12.9.3.6. 正弦曲線

正弦曲線は、正弦関数曲線を使って値を定期的に設定します。正弦曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.101 正弦曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
amplitude	正弦曲線の振幅
constant	結果を計算するためのコンスタント値
phase	ラジアン単位の角位変換
frequency	ヘルツ単位の曲線の周波数

12.9.3.7. スクリプト曲線

スクリプト曲線は、EB GUIDEスクリプトで記述された曲線を使って値を設定します。スクリプト曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.102 スクリプト曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
curve	結果の曲線関数

12.9.3.8. リニア曲線

リニア曲線は、リニアプログレッション曲線を使って値を定期的に設定します。リニア曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.103 リニア曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
velocity	結果を計算するための速度

12.9.3.9. リニア補間曲線

リニア補間曲線は、リニア補間曲線を使って値を定期的に設定します。リニア補間曲線は、整数型、ブール型、浮動小数点数型、および色型に対して使用できます。

表12.104 リニア補間曲線のプロパティ

プロパティ名	説明
name	曲線の名前
delay	アニメーションの開始を基準にしたミリ秒単位の遅延
duration	曲線セグメントのミリ秒単位の時間
enabled	アニメーションを実行するかどうかを定義します。
alternating	アニメーションを繰り返し実行するかどうかを定義します。
relative	初期値に更新を適用するかどうかを定義します。
repeat	繰り返しの回数
target	結果値が適用されるターゲットプロパティ
start	初期値
end	最終値

12.9.4. 3Dウィジェット

12.9.4.1. シーングラフ

シーングラフは、3Dオブジェクトをビュー内に配置します。

表12.105 シーングラフのプロパティ

プロパティ名	説明
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
width	ピクセル単位のウィジェットの幅
height	ピクセル単位のウィジェットの高さ
x	親ウィジェットを基準にしたウィジェットのx座標
y	親ウィジェットを基準にしたウィジェットのy座標

12.9.4.2. シーングラフノード

シーングラフノードは子ノードであり、シーングラフや別のシーングラフノードに追加されます。シーングラフノードは、変形プロパティを持つ3Dウィジェットを3Dシーンに配置するために使用します。シーングラフノードには以下の3Dウィジェットを追加できます。

- カメラ
- 指向性ライト
- メッシュ
- 点ライト
- スポットライト

表12.106 シーングラフノードのプロパティ

プロパティ名	説明
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
rotationX	x軸での回転
rotationY	y軸での回転
rotastionZ	z軸での回転
scalingX	x軸方向の拡大縮小
scalingY	y軸方向の拡大縮小
scalingZ	z軸方向の拡大縮小
translationX	x軸での変換
translationY	y軸での変換
translationZ	z軸での変換

12.9.4.3. カメラ

カメラは、特定の視点からのシーンのビューを定義します。複数のカメラを使用すると、複数の視点からシーンを表示できます。

表12.107 カメラのプロパティ

プロパティ名	説明
`enabled`	trueである場合、ウィジェットが有効になります。
`nearPlane`	視線方向からシーンが見え始める、カメラからの最短距離
`farPlane`	視線方向からシーンが消えずにいる、カメラからの最長距離
`fieldOfView`	カメラの縦方向の視野角

12.9.4.4. 指向性ライト

指向性ライトは、シーンを一方向から照らすライトを追加します。

表12.108 指向性ライトのプロパティ

プロパティ名	説明
`enabled`	trueである場合、ウィジェットが有効になります。
`color`	ライトの色
`intensity`	ライトの明度

12.9.4.5. 材質

材質は、メッシュ表面の外観を定義します。

表12.109 材質のプロパティ

プロパティ名	説明
`ambient`	環境光で照らされている場合にオブジェクトが反射する色
`diffuse`	白色光で照らされている場合にオブジェクトが全方向に均一に反射する色
`emissive`	オブジェクトの自己発光色
`shininess`	光沢要素
`specular`	表面に光沢があるオブジェクトが反射する色
`opacity`	オパシティ値 有効なのは`0`と`1`の間の値(`0.3`など)のみであるという点に注意してください。

12.9.4.6. メッシュ

メッシュは3Dオブジェクトの形状を定義します。

表12.110 メッシュのプロパティ

プロパティ名	説明
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
mesh	自動的に作成されるメッシュファイル*.ebmesh
culling	メッシュから三角を集めない(0)か、表向きの三角のみを集める(1)か、裏向きの三角のみを集める(2)かを定義します。

12.9.4.7. 点ライト

点ライトは、電球のように全方向に光を放つライトをシーンに追加します。

表12.111 点ライトのプロパティ

プロパティ名	説明
enabled	trueである場合、ウィジェットが有効になります。
color	ライトの色
intensity	ライトの明度
attenuationConstant	距離が離れるにつれライトを減退させる定数要素
attenuationLinear	距離が離れるにつれライトを減退させる線形要素
attenuationQuadrat-ic	距離が離れるにつれライトを減退させる二次要素

12.9.4.8. スポットライト

スポットライトは、光の影響範囲を円錐状に制限したライトを追加します。

表12.112 スポットライトのプロパティ

プロパティ名	説明
enabled	trueである場合、ウィジェットが有効になります。
color	ライトの色
intensity	ライトの明度
attenuationConstant	距離が離れるにつれライトを減退させる定数要素

プロパティ名	説明
attenuationLinear	距離が離れるにつれライトを減退させる線形要素
attenuationQuadratic	距離が離れるにつれライトを減退させる二次要素
coneAngleInner	ライトの円錐の内側の角度
coneAngleOuter	ライトの円錐の外側の角度

12.10. ウィジェット機能

次のリストには、実装されているすべてのウィジェット機能の説明と、これらをEB GUIDEモデルで使用する方法に関する簡単な説明が含まれます。

12.10.1. 共通

12.10.1.1. 子の可視性の選択

[子の可視性の選択]ウィジェット機能は、子ウィジェットの可視性を処理します。一度に1つの子ウィジェットのコンテンツだけ表示します。

表12.113 [子の可視性の選択]ウィジェット機能のプロパティ

プロパティ名	説明
containerIndex	親ウィジェットの子ウィジェットのインデックス
containerMapping	マッピングが設定されている場合、コンテナーの子はそれぞれcontainerMapping内の適切な値に従って対応されます。 マッピングが設定または定義されていない場合、または長さがコンテナー内の子ウィジェットの数と一致しない場合、マッピングは使用されません。代わりに、ウィジェットツリー内のウィジェットの順序がインデックスとして使用されます。最上位の子のインデックスが0、次のインデックスが1というように続きます。

12.10.1.2. 有効

[有効]ウィジェット機能は、ウィジェットにenabledプロパティを追加します。

表12.114 [有効]ウィジェット機能のプロパティ

プロパティ名	説明
`enabled`	trueである場合、ウィジェットはタッチ入力および押下入力に反応します。

12.10.1.3. フォーカス

[フォーカス]ウィジェット機能は、ウィジェットに入力フォーカスを設定できるようにします。

表12.115 [フォーカス]ウィジェット機能のプロパティ

プロパティ名	説明
`focusable`	ウィジェットにフォーカスするかどうかを定義します。使用可能な値: ▶ `not focusable (=0)` ▶ `only by touch (=1)` ▶ `only by key (=2)` ▶ `focusable (=3)`
`focused`	trueである場合、ウィジェットがフォーカスされます。

12.10.1.4. 押下

[押下]ウィジェット機能は、ウィジェットが押下可能かどうかを定義します。

表12.116 [押下]ウィジェット機能のプロパティ

プロパティ名	説明
`pressed`	trueである場合、ウィジェットにフォーカスがあるときにキーを押せます。

[タッチ]ウィジェット機能を[タッチ押下]ウィジェット機能に組み合わせると、押ボタンをモデル化できます。

12.10.1.5. 選択

[選択]ウィジェット機能は、ウィジェットに`selected`プロパティを追加します。これは通常、アプリケーションまたはHMIモデラーによって設定されます。フレームワークの他のコンポーネントによって変更されることはありません。

表12.117 [選択]ウィジェット機能のプロパティ

プロパティ名	説明
`selected`	trueである場合、ウィジェットが選択されます。

12.10.1.6. 選択グループ

[選択グループ]ウィジェット機能は、オプションボタンの配列をモデル化するために使用されます。配列内では、すべてのオプションボタンが[選択グループ]ウィジェット機能と一意のボタンIDを持ちます。

`buttonValue`プロパティにはデータプールアイテムを使用します。オプションボタン内のすべてのウィジェットにデータプールアイテムを割り当てます。

ボタングループ内のウィジェットの選択および選択解除を行うには、`buttonValue`プロパティを設定する外部アプリケーションを使用します。タッチまたはキー入力や、ボタン値を設定する条件の追加で変更をトリガーすることもできます。

表12.118 [選択グループ]ウィジェット機能のプロパティ

プロパティ名	説明
`buttonId`	ボタングループ内のボタンを識別するID
`buttonValue`	ボタンの現在の値。この値が`buttonId`と一致する場合、ボタンが選択されます。
`selected`	`buttonID`と`buttonValue`が同じであるかどうかを評価します。trueである場合、ボタンが選択されます。

12.10.1.7. スピン

[スピン]ウィジェット機能は、ウィジェットを回転ボタンに変換します。[スピン]ウィジェット機能を持つウィジェットは、内部値を変更することによって増加イベントおよび減少イベントに反応します。[スピン]ウィジェット機能は、プレビュー値を持つスケール、プログレスバー、またはウィジェットの作成に使用できます。

表12.119 [スピン]ウィジェット機能のプロパティ

プロパティ名	説明
`currentValue`	現在の回転値
`maxValue`	`currentValue`プロパティの最大値
`minValue`	`currentValue`プロパティの最小値
`incValueTrigger`	trueである場合、`currentValue`プロパティが1増加します。
`incValueReaction`	`currentValue`プロパティの増加に対する反応
`decValueTrigger`	trueである場合、現在の値が1減少します。
`decValueReaction`	`currentValue`プロパティの減少に対する反応
`steps`	`currentValue`プロパティの増加または減少を計算するステップの数
`valueWrapAround`	使用可能な値: ► true: `minValue`または`maxValue`を超えた場合、`currentValue`プロパティは逆側に進みます。

プロパティ名	説明
	▶ false: `minValue`または`maxValue`を超えた場合、`currentValue`プロパティは増加/減少しません。

12.10.1.8. タッチ

[タッチ]ウィジェット機能は、ウィジェットがタッチ入力に反応できるようにします。

表12.120 [タッチ]ウィジェット機能のプロパティ

プロパティ名	説明
`touchable`	trueである場合、ウィジェットはタッチ入力に反応します。
`touched`	trueである場合、ウィジェットは現在タッチされています。
`touchPolicy`	ウィジェットの境界を超えるタッチおよび動きを処理する方法を定義します。使用可能な値: ▶ `Press then react (=0)` 最初に押下すると、ウィジェットが反応します。移動および解放の通知はウィジェットエリア内でのみアクティブです。 ▶ `Press and grab (=1)` 押すとコンタクトをグラブできます。コンタクトは、ウィジェットエリアの外部に移動しても、グラブされたままです。 ▶ `Press then react on contact (=3)` コンタクトがウィジェットの境界の外部で押された状態になっても、それ以降の移動イベントおよび解放イベントはウィジェットに伝達されます。
`touchBehavior`	タッチ評価を定義します。使用可能な値: ▶ `Whole area (=0)` タッチされたウィジェットを識別するために、レンダラーはウィジェットのクリッピング四角形を評価します。 ▶ `Visible pixels (=1)` タッチされたウィジェットを識別するために、レンダラーはタッチされたピクセルが属するウィジェットを評価します。 アルファ透過性を持つ透過ピクセル、またはOまたはAなどの文字内のピクセルにはタッチできません。 `Visible pixels`値はラベルに対して無効であるという点に注意してください。

[タッチ]ウィジェット機能を[押下]ウィジェット機能に組み合わせると、押ボタンをモデル化できます。

ティップ 	パフォーマンスに関する推奨事項: プロジェクトにおいてパフォーマンスが重要な問題である場合、touchBehaviorプロパティを[エリア全体]に設定します。EB GUIDE GTFでは、[エリア全体]の方が[表示ピクセル]より速く評価されます。

12.10.1.9. テキストの切り捨て

[テキストの切り捨て]ウィジェット機能は、textプロパティのコンテンツがウィジェットエリアに収まらない場合にそのコンテンツを切り捨てます。ウィジェット機能を使用すると、デフォルト設定のtrailingとは異なる切り捨てが可能になります。

表12.121 [テキストの切り捨て]ウィジェット機能のプロパティ

プロパティ名	説明
truncationPolicy	1行のテキストの場合、truncationPolicyプロパティは切り捨ての位置を定義します。使用可能な値: ► leading (=0): テキストはテキストの先頭で置き換えられます。 ► trailing (=1): テキストはテキストの末尾で置き換えられます。 複数行のテキストの場合、truncationPolicyプロパティは、テキストを置き換える位置を定義します。使用可能な値: ► leading (=0): 先頭にある行が置き換えられ、最初に表示される行のテキストがテキストの先頭で切り捨てられます。 ► trailing (=1): 末尾にある行が置き換えられ、最後に表示される行のテキストがテキストの末尾で切り捨てられます。
truncationSymbol	置き換えられたテキスト部分の代わりに表示される文字列

12.10.1.10. 複数行

[複数行]ウィジェット機能を使用すると、ラベルの改行が可能になります。

表12.122 [複数行]ウィジェット機能のプロパティ

プロパティ名	説明
lineGap	行間のサイズ。負の値の場合は行間が狭くなり、正の値の場合は行間が広くなります。
maxLineCount	表示行の最大数。0 = 制限なし

注記

文字置換
'\\' '\¥'のシーケンスは'¥¥'で置き換えられます。'¥\' 'n'のシーケンスは'¥n'で置き換えられます。

ラベルのサイズを大きくして1行でテキストを十分に表示できるようにすると、'¥n'が' 'で置き換えられます。

12.10.2. 効果

12.10.2.1. 枠

[枠]ウィジェット機能は、設定可能な枠をウィジェットに追加します。枠はウィジェットの境界から始まり、ウィジェット内に配置されます。

要件:

- このウィジェット機能は、四角形に対して使用できます。

表12.123 [枠]ウィジェット機能のプロパティ

プロパティ名	説明
`borderThickness`	ピクセル単位の枠の厚さ
`borderColor`	枠のレンダリングに使用する色
`borderStyle`	枠のレンダリングに使用するスタイル

12.10.2.2. 配色

[配色]ウィジェット機能は、ウィジェットおよびウィジェットサブツリーに色を付けます。また、アルファ値が不透明でない場合は透過性にも影響します。

例12.1
配色ウィジェット機能の使用方法

RGBA要素が0.0～1.0であるすべての色について、[配色]ウィジェット機能のアルゴリズムが、ウィジェットの現在の色値に`colorationColor`プロパティ値を掛けます。掛け算はピクセル単位で要素に対して行われます。

半透明のグレーに不透明な青を掛けると、次のように半透明の暗い青になります。

```
(0.5, 0.5, 0.5, 0.5) * (0.0, 0.0, 1.0, 1.0) = (0.0, 0.0, 0.5, 0.5)
```

表12.124 [配色]ウィジェット機能のプロパティ

プロパティ名	説明
`colorationEnabled`	trueである場合、配色が使用されます。
`colorationColor`	使用する配色使用可能な値: ► 純粋 ► 不透明 ► 白

12.10.3. フォーカス

12.10.3.1. ユーザー定義フォーカス

[ユーザー定義フォーカス]ウィジェット機能を使用すると、ウィジェットにフォーカス機能を追加できます。この機能を使用するウィジェットは、ウィジェットサブツリーのローカルフォーカス階層を管理します。

表12.125 [ユーザー定義フォーカス]ウィジェット機能のプロパティ

プロパティ名	説明
`focusNext`	フォーカスを次の子ウィジェットに割り当てるトリガー
`focusOrder`	`focusOrder`プロパティを使用すると、フォーカスを割り当てる際に子ウィジェットをスキップできます。子ウィジェットのIDは、サブツリー内の位置に対応しています。フォーカス不可能な子ウィジェットはデフォルトでスキップされます。子ウィジェットがフォーカスされる順序は、次のとおりです。 ► 定義済み: ユーザー定義のウィジェット順序を使用します。 ► 未定義: デフォルトのウィジェット順序を代わりに使用します。 各子ウィジェットには[フォーカス]ウィジェット機能が必要です。この機能がない場合、ウィジェットはフォーカス処理で無視されます。例: focusOrder=1|0|2は、まず2番目のウィジェットにフォーカスし、次に1番目のウィジェット、最後に3番目のウィジェットにフォーカスすることを意味します。
`focusPrevious`	フォーカスを前の子に割り当てるトリガー
`focusFlow`	フォーカスの動作は階層内で変化します。使用可能な値: ► `stop at hierarchy level (=0)` ► `wrap within hierarchy level (=1)`

プロパティ名	説明
	▶ step up in hierarchy (=2)
focusedIndex	インデックスは、focusOrderリスト内の子ウィジェットの位置を定義します。ウィジェットがフォーカス不可能である場合、リスト内の次の子が使用されます。
initFocus	初期化時にフォーカスされる子ウィジェットのインデックス

12.10.3.2. 自動フォーカス

[自動フォーカス]ウィジェット機能を使用すると、子ウィジェットがフォーカスされる順序が事前定義済みになります。フォーカス可能な子ウィジェットはスキップできません。[自動フォーカス]ウィジェット機能を持つウィジェットは、ウィジェットサブツリーのローカルフォーカス階層を管理します。自動フォーカスウィジェット機能は、focusableプロパティを持つ子ウィジェットのウィジェットサブツリーをチェックします。

フォーカスの順序の計算には、レイアウト内のウィジェットの順序が使用されます。レイアウトの方向に応じて、アルゴリズムは左上または右上から開始されます。

表12.126 [自動フォーカス]ウィジェット機能のプロパティ

プロパティ名	説明
focusNext	フォーカスインデックスが増加する条件
focusPrevious	フォーカスインデックスが減少する条件
focusFlow	フォーカスの動作は階層内で変化します。使用可能な値: ▶ stop at hierarchy (=0) ▶ wrap within hierachy level (=1) ▶ step up in hierarchy (=2)
focusedIndex	フォーカス可能なn番目の子ウィジェットとして現在フォーカスされている子ウィジェットのインデックス
initFocus	このインデックスは、初期化時にフォーカスされる子ウィジェットを定義します。ウィジェットがフォーカス不可能である場合、次にフォーカス可能な子が使用されます。

12.10.4. ジェスチャー

12.10.4.1. フリックジェスチャー

表面を素早くなでるコンタクト

表12.127 [フリックジェスチャー]ウィジェット機能のプロパティ

プロパティ名	説明
onGestureFlick	ジェスチャーが認識されたときにトリガーされる反応 反応引数は、次のとおりです。 ► speed: フリックジェスチャーの相対速度 ピクセル/ミリ秒の速度をflickMinLength/flickMaxTimeで割った値 ► directionX: ジェスチャーの方向ベクトルのx部分 ► directionY: ジェスチャーの方向ベクトルのy部分
flickMaxTime	ジェスチャーがフリックジェスチャーとして認識されるためにコンタクトが所定の位置に留まることが認められるミリ秒単位の最大時間
flickMinLength	コンタクトがフリックジェスチャーとして認識されるために表面上で移動する必要があるピクセル単位の最小距離

12.10.4.2. ホールドジェスチャー

動きのないホールドジェスチャー

注記 [ホールドジェスチャー]ウィジェット機能は、[タッチの喪失]ウィジェット機能をトリガーしません。

表12.128 [ホールドジェスチャー]ウィジェット機能のプロパティ

プロパティ名	説明
onGestureHold	ジェスチャーが認識されたときにトリガーされる反応反応はコンタクトごとに1回のみトリガーされます。つまり、holdDurationが期限切れになったときに、コンタクトがまだ最初のタッチ位置の周囲の小さな境界ボックス内にある場合です。 反応引数は、次のとおりです。 ► x: コンタクト位置のx座標 ► y: コンタクト位置のy座標
holdDuration	ジェスチャーがホールドジェスチャーとして認識されるためにコンタクトが所定の位置に留まる必要があるミリ秒単位の最小時間

12.10.4.3. ロングホールドジェスチャー

動きのないロングホールドジェスチャー

注記

[ロングホールドジェスチャー]ウィジェット機能は、[タッチの喪失]ウィジェット機能をトリガーしません。

表12.129 [ロングホールドジェスチャー]ウィジェット機能のプロパティ

プロパティ名	説明
onGestureLongHold	ジェスチャーが認識されたときにトリガーされる反応反応はコンタクトごとに1回のみトリガーされます。つまり、longHoldDurationが期限切れになったときに、コンタクトがまだ最初のタッチ位置の周囲の小さな境界ボックス内にある場合です。 反応引数は、次のとおりです。 ► x: コンタクト位置のx座標 ► y: コンタクト位置のy座標
longHoldDuration	ジェスチャーがロングホールドジェスチャーとして認識されるためにコンタクトが所定の位置に留まる必要があるミリ秒単位の最小時間

12.10.4.4. パスジェスチャー

1つのコンタクトが描画した形状を、既知の形状集合とマッチングします。

表12.130 [パスジェスチャー]ウィジェット機能のプロパティ

プロパティ名	説明
onPath	入力した形状が一致した場合にトリガーされる反応反応がトリガーされるのは、onPathStartがすでにトリガーされている場合のみです。
onPathStart	コンタクトが最小ボックス(pathMinXBox、pathMinYBox)の外に移動したときにトリガーされる反応。反応引数は、次のとおりです。 ► gestureId: マッチングされたパスのID
onPathNotRecognized	入力した形状が一致しない場合にトリガーされる反応反応がトリガーされるのは、onPathStartがすでにトリガーされている場合のみです。
pathMinXBox	パスジェスチャー認識機能が入力を始めるためにコンタクトが移動する必要があるピクセル単位の最小距離のx座標
pathMinYBox	パスジェスチャー認識機能が入力を始めるためにコンタクトが移動する必要があるピクセル単位の最小距離のy座標

12.10.4.4.1. ジェスチャーID

ジェスチャー識別子は、パスジェスチャー認識機能の設定によって決まります。次の表は、EB GUIDEに含まれる設定の例を示しています。

表12.131 EB GUIDEに含まれるパスジェスチャーの設定の例

ID	形状	説明
0		左から右への屋根形状
1		右から左への屋根形状
2		左から右への水平線
3		右から左への水平線
4		チェックマーク
5		左から右への波形状
6		右から左への波形状

12.10.4.5. ピンチジェスチャー

2つのコンタクトが近づくまたは離れる動き

表12.132 [ピンチジェスチャー]ウィジェット機能のプロパティ

プロパティ名	説明
onGesturePinchStart	ジェスチャーの開始が認識されたときにトリガーされる反応反応引数は、次のとおりです。 ▶ ratio: 最初のコンタクトの距離に対する現在のコンタクトの距離の比率 ▶ centerX: 2つのコンタクト間の現在の中心点のx座標 ▶ centerY: 2つのコンタクト間の現在の中心点のy座標
onGesturePinchUpdate	ピンチ率または中心点が変更されたときにトリガーされる反応。反応引数は、次のとおりです。 ▶ ratio: 最初のコンタクトの距離に対する現在のコンタクトの距離の比率 ▶ centerX: 2つのコンタクト間の現在の中心点のx座標 ▶ centerY: 2つのコンタクト間の現在の中心点のy座標
onGesturePinchEnd	ジェスチャーが終了したときにトリガーされる反応。反応引数は、次のとおりです。 ▶ ratio: 最初のコンタクトの距離に対する現在のコンタクトの距離の比率 ▶ centerX: 2つのコンタクト間の現在の中心点のx座標 ▶ centerY: 2つのコンタクト間の現在の中心点のy座標
pinchThreshold	ジェスチャーが認識されるために各コンタクトが最初の位置から移動する必要があるピクセル単位の最小距離

12.10.4.6. 回転ジェスチャー

2つのコンタクトの円に沿った動き

表12.133 [回転ジェスチャー]ウィジェット機能のプロパティ

プロパティ名	説明
onGestureRotateStart	ジェスチャーの開始が認識されたときにトリガーされる反応
onGestureRotateUpdate	認識された角度または中心点が変更されたときにトリガーされる反応
onGestureRotateEnd	ジェスチャーが終了したときにトリガーされる反応
rotateThreshold	ジェスチャーの開始が認識されるために各コンタクトが最初の位置から移動する必要があるピクセル単位の最小距離

onGestureRotateEnd、onGestureRotateStart、およびonGestureRotateUpdateの反応引数は、次のとおりです。

- angle: 2つの関連コンタクトの最初の位置によって指定される線と、2つのコンタクトの現在の位置によって指定される線の間の角度。角度は反時計回りに測定されます。
- centerX: 2つのコンタクト間の現在の中心点のx座標
- centerY: 2つのコンタクト間の現在の中心点のy座標

12.10.5. 入力処理

12.10.5.1. 内部で移動

[内部で移動]ウィジェット機能を使用すると、ウィジェットが境界内の動きに反応できるようになります。

表12.134 [内部で移動]ウィジェット機能のプロパティ

プロパティ名	説明
`moveOver`	境界内の動きに対するウィジェットの反応 反応引数は、次のとおりです。 - touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID - fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.2. 外部へ移動

[外部へ移動]ウィジェット機能を使用すると、ウィジェットが境界外への動きに反応できるようになります。

表12.135 [外部へ移動]ウィジェット機能のプロパティ

プロパティ名	説明
`moveOut`	境界外への動きに対するウィジェットの反応 反応引数は、次のとおりです。 - touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID - fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.3. 内部へ移動

[内部へ移動]ウィジェット機能を使用すると、ウィジェットが境界内への動きに反応できるようになります。

表12.136 [内部へ移動]ウィジェット機能のプロパティ

プロパティ名	説明
`moveIn`	境界内への動きに対するウィジェットの反応 反応引数は、次のとおりです。 ► touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.4. タッチ押下

[タッチ押下]ウィジェット機能を使用すると、ウィジェットが押下に反応できるようになります。

表12.137 [タッチ押下]ウィジェット機能のプロパティ

プロパティ名	説明
`touchPressed`	押下に対するウィジェットの反応 反応引数は、次のとおりです。 ► touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.5. タッチリリース

[タッチリリース]ウィジェット機能を使用すると、ウィジェットがリリースに反応できるようになります。

表12.138 [タッチリリース]ウィジェット機能のプロパティ

プロパティ名	説明
`touchShortReleased`	リリースに対するウィジェットの反応 反応引数は、次のとおりです。 ► touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.6. タッチの喪失

[タッチの喪失]ウィジェット機能を使用すると、ウィジェットがタッチコンタクトの喪失に反応できるようになります。

コンタクトがジェスチャーの一部である場合や、リリースなしでタッチスクリーンを離れた場合、コンタクトは消える可能性があります。この場合、`touchShortReleased`反応は実行されません。

表12.139 [タッチの喪失]ウィジェット機能のプロパティ

プロパティ名	説明
`onTouchGrabLost`	タッチコンタクトの喪失に対するウィジェットの反応 反応引数は、次のとおりです。 ► touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.7. タッチのステータス変更

[タッチのステータス変更]ウィジェット機能を使用すると、ウィジェットがタッチのステータス変更に反応できるようになります。

表12.140 [タッチのステータス変更]ウィジェット機能のプロパティ

プロパティ名	説明
`touchStatusChanged`	タッチのステータス変更に対するウィジェットの反応 反応引数は、次のとおりです。 ► touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► touchStatus: タッチのタイプのID 使用可能な値: ► `0`: 新規コンタクト ► `1`: タッチ押下 ► `2`: タッチ移動 ► `3`: タッチリリース ► `4`: タッチなしの移動 ► `5`: タッチ終了 ► `6`: あらゆるステータス変更 ► fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.8. タッチ移動

[タッチ移動]ウィジェット機能を使用すると、タッチでの移動に反応できるようになります。

表12.141 [タッチ移動]ウィジェット機能のプロパティ

プロパティ名	説明
`touchMoved`	タッチでの移動に対するウィジェットの反応 反応引数は、次のとおりです。 ► touchId: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► fingerId: ウィジェット内を移動するコンタクトのID

12.10.5.9. ジェスチャー

[ジェスチャー]ウィジェット機能を使用すると、ウィジェットがタッチジェスチャーに反応できるようになります。

[ジェスチャー]ウィジェット機能には、追加プロパティはありません。

12.10.5.10. キー押下

[キー押下]ウィジェット機能を使用すると、ウィジェットがキー押下に反応できるようになります。

表12.142 [キー押下]ウィジェット機能のプロパティ

プロパティ名	説明
`keyPressed`	キー押下に対するウィジェットの反応 反応引数は、次のとおりです。 ► keyId: 処理されるキーのID

12.10.5.11. キーUnicode

[キーUnicode]ウィジェット機能を使用すると、ウィジェットがUnicodeキー入力に反応できるようになります。

表12.143 [キーUnicode]ウィジェット機能のプロパティ

プロパティ名	説明
`keyUnicode`	Unicodeキー入力に対するウィジェットの反応 反応引数は、次のとおりです。 ► keyId: 処理されるキーのID

12.10.5.12. キーリリース

[キーリリース]ウィジェット機能を使用すると、ウィジェットがキーリリースに反応できるようになります。

表12.144 [キーリリース]ウィジェット機能のプロパティ

プロパティ名	説明
`keyShortReleased`	キーリリースに対するウィジェットの反応 反応引数は、次のとおりです。 ► keyId: 処理されるキーのID

12.10.5.13. キーのステータス変更

[キーのステータス変更]ウィジェット機能を使用すると、ウィジェットがキー押下またはキーリリースに反応できるようになります。これは、[ショート押下]、[ロング]、[超ロング]、および[連続]などのキー入力に対する反応を定義します。

表12.145 [キーのステータス変更]ウィジェット機能のプロパティ

プロパティ名	説明
`keyStatusChanged`	キー押下またはキーリリースに対するウィジェットの反応 反応引数は、次のとおりです。 ► keyId: 処理されるキーのID ► ステータス: ステータス変更の数値ID

12.10.5.14. 回転

[回転]ウィジェット機能を使用すると、ウィジェットが回転に反応できるようになります。

表12.146 [回転]ウィジェット機能のプロパティ

プロパティ名	説明
`rotaryReaction`	回転に対するウィジェットの反応を定義します。trueである場合、ウィジェットは入力された回転イベントに反応します。 反応引数は、次のとおりです。 ► rotaryId: 整数ID ► increment: 入力されたイベントが送信されたときに回転入力が移動する単位数

12.10.5.15. 可動

[可動]ウィジェット機能を使用すると、ウィジェットをタッチで移動することができるようになります。

表12.147 [可動]ウィジェット機能のプロパティ

プロパティ名	説明
`moveDirection`	ウィジェットが移動する方向使用可能な値: ▶ `horizontal (=0)` ▶ `vertical (=1)` ▶ `free (=2)`

12.10.6. レイアウト

12.10.6.1. 絶対レイアウト

親ウィジェットの[絶対レイアウト]ウィジェット機能は、子ウィジェットの位置およびサイズを定義します。非表示になっている子ウィジェットは無視されます。追加されたウィジェット機能プロパティは整数リストで構成されます。各リスト要素は1つの子ウィジェットにマップされます。

表12.148 [絶対レイアウト]ウィジェット機能のプロパティ

プロパティ名	説明
`itemLeftOffset`	子ウィジェットの左枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。
`itemTopOffset`	子ウィジェットの上枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。
`itemRightOffset`	子ウィジェットの右枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。
`itemBottomOffset`	子ウィジェットの下枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。

12.10.6.2. ボックスレイアウト

[ボックスレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよびサイズプロパティは、親ウィジェットによって設定されます。非表示になっている子ウィジェットは計算では無視されます。

表12.149 [ボックスレイアウト]ウィジェット機能のプロパティ

プロパティ名	説明
gap	レイアウトの方向に応じた2つの子ウィジェット間のスペース
layoutDirection	リスト要素(子ウィジェット)が配置される方向

12.10.6.3. フローレイアウト

[フローレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよびサイズプロパティは、親ウィジェットによって設定されます。非表示になっている子ウィジェットは計算では無視されます。

表12.150 [フローレイアウト]ウィジェット機能のプロパティ

プロパティ名	説明
horizontalGap	2つの子ウィジェット間の水平方向のスペース
verticalGap	2つの子ウィジェット間の垂直方向のスペース
layoutDirection	リスト要素(子ウィジェット)が配置される方向
horizontalChildAlign	子ウィジェットの水平方向の位置揃え
verticalChildAlign	子ウィジェットの垂直方向の位置揃え ▶ center (=0): 子ウィジェットは中央に配置されます。 ▶ top (=1): 子ウィジェットは上に配置されます。 ▶ bottom (=2): 子ウィジェットは下に配置されます。

12.10.6.4. グリッドレイアウト

[グリッドレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよびサイズプロパティは、親ウィジェットによって設定されます。非表示になっている子ウィジェットは計算では無視されます。

表12.151 [グリッドレイアウト]ウィジェット機能のプロパティ

プロパティ名	説明
horizontalGap	2つの子ウィジェット間の水平方向のスペース
verticalGap	2つの子ウィジェット間の垂直方向のスペース
numRows	行の数を定義します。
numColumns	列の数を定義します。

12.10.6.5. レイアウト余白

[レイアウト余白]ウィジェット機能は、[フローレイアウト]、[絶対レイアウト]、[ボックスレイアウト]、または[グリッドレイアウト]ウィジェット機能を使用するウィジェットに設定可能な余白を追加します。

表12.152 [レイアウト余白]ウィジェット機能のプロパティ

プロパティ名	説明
leftMargin	左枠の余白
topMargin	上枠の余白
rightMargin	右枠の余白
bottomMargin	下枠の余白

12.10.6.6. リストレイアウト

[リストレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよび[リストインデックス]ウィジェット機能のlistIndexプロパティは、親ウィジェットによって設定されます。

子ウィジェットを作成するインスタンシエータとともに使用するのが最も適しています。

[リストインデックス]ウィジェット機能の詳細については12.10.7.2「リストインデックス」を参照してください。

表12.153 [リストレイアウト]ウィジェット機能のプロパティ

プロパティ名	説明
layoutDirection	リスト要素(子ウィジェット)が配置される方向
scrollOffset	リストをスクロールするピクセルの数
scrollOffsetRebase	scrollOffsetRebaseプロパティが変更されると、現在のscrollOffsetがscrollIndexに変換されます。残りのオフセットはscrollOffsetプロパティに書き込まれます。
firstListIndex	ウィジェット機能によって定義される、最初に表示されるリスト要素のリストインデックス
scrollIndex	scrollOffsetプロパティが適用される基本リストインデックススクロールは、scrollIndexプロパティに指定されているリスト要素から開始されます。
scrollValue	現在のスクロール値
scrollValueMax	リストの終了位置にマップされる最大スクロール値
scrollValueMin	リストの開始位置にマップされる最小スクロール値
bounceValue	scrollOffsetプロパティが有効なスクロール範囲内の位置にある限り、bounceValueプロパティはゼロです。この値は、スクロール位置がリストの開始

プロパティ名	説明
	位置を超える場合は正、スクロール位置がリストの終了位置を超える場合は負になります。bounceValueをscrollOffsetに追加すると、スクロール位置が範囲内に戻ります。
bounceValueMax	scrollOffsetが有効なスクロール範囲外に移動できる最大値。ユーザーがそれ以上スクロールしようとすると、scrollOffsetは切り捨てられます。
segments	水平レイアウト方向の場合: 行の数 垂直レイアウト方向の場合: 列の数
listLength	リスト要素の数
wrapAround	使用可能な値: ► true: scrollValueMinまたはscrollValueMaxを超えた場合、scroll-Valueプロパティは逆側に進みます。 ► false: scrollValueMinまたはscrollValueMaxを超えた場合、scroll-Valueプロパティは増加/減少しません。

12.10.6.7. 拡大縮小モード

[拡大縮小モード]ウィジェット機能は、イメージのサイズがウィジェットのサイズと異なる場合にイメージを表示する方法を定義します。

表12.154 [拡大縮小モード]ウィジェット機能のプロパティ

プロパティ名	説明
scaleMode	イメージの拡大縮小モード。使用可能な値: ► 0 = 元のサイズ ► 1 = サイズに合わせる ► 2 = 縦横比を維持する

12.10.7. リスト管理

12.10.7.1. ラインインデックス

[ラインインデックス]ウィジェット機能は、ウィジェットにラインインデックスプロパティを追加します。これは、表とともに使用するよう設計されています。

表12.155 [ラインインデックス]ウィジェット機能のプロパティ

プロパティ名	説明
lineIndex	表内の現在のラインのインデックス

12.10.7.2. リストインデックス

[リストインデックス]ウィジェット機能は、ウィジェットにlistIndexプロパティを追加します。これは、[リストレイアウト]ウィジェット機能とともに使用するよう設計されています。

表12.156 [リストインデックス]ウィジェット機能のプロパティ

プロパティ名	説明
listIndex	リスト内の現在のウィジェットのインデックス

12.10.7.3. テンプレートインデックス

[テンプレートインデックス]ウィジェット機能は、ウィジェットにラインテンプレートインデックスプロパティを追加します。これは、インスタンシエータとともに使用するよう設計されています。

表12.157 [テンプレートインデックス]ウィジェット機能のプロパティ

プロパティ名	説明
lineTemplateIndex	使用されているラインテンプレートのインデックス

12.10.7.4. ポートの表示

[ポートの表示]ウィジェット機能は、ウィジェットの境界にあるサイズ超過の要素をクリップします。これは、コンテナーまたはリストとともに使用するよう設計されています。

[ポートの表示]ウィジェット機能は、次のモデル要素に対して有効です。

- [ポートの表示]を追加したウィジェットの子ウィジェットは、そのウィジェットの寸法内でクリップされます。
- [ポートの表示]を追加したウィジェットは、その親ビューの寸法内でクリップされます。

表12.158 [ポートの表示]ウィジェット機能のプロパティ

プロパティ名	説明
xOffset	子ウィジェットの描画エリアにおける表示クリッピングの水平方向のオフセット
yOffset	子ウィジェットの描画エリアにおける表示クリッピングの垂直方向のオフセット

12.10.8. 3D

[3D]カテゴリのウィジェット機能は、3Dウィジェットに対してのみ使用することができます。

12.10.8.1. カメラビューポート

[カメラビューポート]ウィジェット機能は、シーングラフ内でのカメラの描画領域を定義します。

要件:

- [カメラビューポート]ウィジェット機能は、カメラに対して使用できます。

表12.159 [カメラビューポート]ウィジェット機能のプロパティ

プロパティ名	説明
viewportX	シーングラフ内のビューポートのx原点
viewportY	シーングラフ内のビューポートのy原点
viewportWidth	ビューポートの幅
viewportHeight	ビューポートの高さ

12.10.8.2. アンビエントテクスチャ

[アンビエントテクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [アンビエントテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.160 [アンビエントテクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
ambientTexture	テクスチャのファイル名
ambientTextureAddressModeU	u方向のテクスチャのアドレスモード
ambientTextureAddressModeV	v方向のテクスチャのアドレスモード
ambientFilterMode	テクスチャのフィルタモード

12.10.8.3. ディフューズテクスチャ

[ディフューズテクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [ディフューズテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.161 [ディフューズテクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
diffuseTexture	テクスチャのファイル名
diffuseTextureAddressModeU	u方向のテクスチャのアドレスモード
diffuseTextureAddressModeV	v方向のテクスチャのアドレスモード
diffuseFilterMode	テクスチャのフィルタモード

12.10.8.4. エミッシブテクスチャ

[エミッシブテクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [エミッシブテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.162 [エミッシブテクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
emissiveTexture	テクスチャのファイル名
emissiveTextureAddress-ModeU	u方向のテクスチャのアドレスモード
emissiveTextureAddressMod-eV	v方向のテクスチャのアドレスモード
emissiveFilterMode	テクスチャのフィルタモード

12.10.8.5. ライトマップテクスチャ

[ライトマップテクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [ライトマップテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.163 [ライトマップテクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
lightMapTexture	テクスチャのファイル名

プロパティ名	説明
lightMapTextureAddress-ModeU	u方向のテクスチャのアドレスモード
lightMapTextureAddressMod-eV	v方向のテクスチャのアドレスモード
lightMapFilterMode	テクスチャのフィルタモード

12.10.8.6. ノーマルマップテクスチャ

[ノーマルマップ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [ノーマルマップテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.164 [ノーマルマップ]ウィジェット機能のプロパティ

プロパティ名	説明
normalMapTexture	テクスチャのファイル名
normalMapTextureAddress-ModeU	u方向のテクスチャのアドレスモード
normalMapTextureAddress-ModeV	v方向のテクスチャのアドレスモード
normalMapFilterMode	テクスチャのフィルタモード

12.10.8.7. 不透明テクスチャ

[不透明テクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [不透明テクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.165 [不透明テクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
opaqueTexture	テクスチャのファイル名
opaqueTextureAddressModeU	u方向のテクスチャのアドレスモード
opaqueTextureAddressModeV	v方向のテクスチャのアドレスモード

プロパティ名	説明
opaqueFilterMode	テクスチャのフィルタモード

12.10.8.8. リフレクションテクスチャ

[リフレクションテクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [リフレクションテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.166 [リフレクションテクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
reflectionTopTexture	テクスチャのファイル名
reflectionBottomTexture	テクスチャのファイル名
reflectionLeftTexture	テクスチャのファイル名
reflectionRightTexture	テクスチャのファイル名
reflectionFrontTexture	テクスチャのファイル名
reflectionBackTexture	テクスチャのファイル名
reflectionFilterMode	テクスチャのフィルタモード

注記

[リフレクションテクスチャ]ウィジェット機能
EB GUIDE Studioは、イメージファイルが以下のプロパティすべてで選択されている場合にのみ、[リフレクションテクスチャ]ウィジェット機能を表示します。

- reflectionTopTexture
- reflectionBottomTexture
- reflectionLeftTexture
- reflectionRightTexture
- reflectionFrontTexture
- reflectionBackTexture

イメージファイルのサイズは同一でなければなりません。

12.10.8.9. スペキュラテクスチャ

[スペキュラテクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。

要件:

- [スペキュラテクスチャ]ウィジェット機能は、材質に対して使用できます。

表12.167 [スペキュラテクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明
specularTexture	テクスチャのファイル名
specularTextureAddress-ModeU	u方向のテクスチャのアドレスモード
specularTextureAddressMod-eV	v方向のテクスチャのアドレスモード
specularFilterMode	テクスチャのフィルタモード

12.10.9. 変形

変形では、ウィジェットの位置、形式、およびサイズを変更します。

変形が実行される順序は、ウィジェットツリー内の順序と同じです。同じウィジェットツリーの階層レベルで複数の変形が1つのウィジェットに適用される場合、その順序は次のようになります。

1. 変換
2. せん断
3. 拡大縮小
4. z軸中心の回転
5. y軸中心の回転
6. x軸中心の回転

12.10.9.1. ピボット

[ピボット]ウィジェット機能は、ウィジェットに適用される変形のピボット点を定義します。ピボット点が設定されていない場合、デフォルトのピボット点は(0.0, 0.0, 0.0)になります。

表12.168 [ピボット]ウィジェット機能のプロパティ

プロパティ名	説明
pivotX	親ウィジェットを基準にしたx軸のピボット点
pivotY	親ウィジェットを基準にしたy軸のピボット点
pivotZ	ウィジェットがシーングラフである場合、親ウィジェットを基準にしたz軸のピボット点

12.10.9.2. 回転

[回転]ウィジェット機能は、ウィジェットおよびサブツリーを回転させるために使用します。

表12.169 [回転]ウィジェット機能のプロパティ

プロパティ名	説明
rotationEnabled	回転を使用するかどうかを定義します。
rotationAngleX	x軸での回転角度。このプロパティはシーングラフにのみ影響します。
rotationAngleY	y軸での回転角度。このプロパティはシーングラフにのみ影響します。
rotationAngleZ	z軸での回転角度。

12.10.9.3. 拡大縮小

[拡大縮小]ウィジェット機能は、ウィジェットおよびサブツリーを拡大縮小するために使用します。

表12.170 [拡大縮小]ウィジェット機能のプロパティ

プロパティ名	説明
scalingEnabled	拡大縮小を使用するかどうかを定義します。
scalingX	x軸でのパーセント単位の拡大縮小
scalingY	y軸でのパーセント単位の拡大縮小
scalingZ	ウィジェットがシーングラフである場合、z軸でのパーセント単位の拡大縮小

12.10.9.4. せん断

[せん断]ウィジェット機能は、ウィジェットサブツリー内のウィジェットをゆがめるために使用します。

表12.171 [せん断]ウィジェット機能のプロパティ

プロパティ名	説明
shearingEnabled	せん断を使用するかどうかを定義します。
shearingXbyY	x軸のy軸に対するせん断量
shearingXbyZ	ウィジェットがシーングラフである場合、x軸のz軸に対するせん断量
shearingYbyX	y軸のx軸に対するせん断量
shearingYbyZ	ウィジェットがシーングラフである場合、y軸のz軸に対するせん断量
shearingZbyX	ウィジェットがシーングラフである場合、z軸のx軸に対するせん断量
shearingZbyY	ウィジェットがシーングラフである場合、z軸のy軸に対するせん断量

12.10.9.5. 変換

[変換]ウィジェット機能は、ウィジェットおよびサブツリーを変換するために使用します。これにより、ウィジェットはx、y、およびz方向に移動します。

表12.172 [変換]ウィジェット機能のプロパティ

プロパティ名	説明
`translationEnabled`	変換を使用するかどうかを定義します。
`translationX`	x軸での変換
`translationY`	y軸での変換
`translationZ`	ウィジェットがシーングラフである場合、z軸での変換

13. インストール

13.1. バックグラウンド情報

13.1.1. 制限

注記

互換性

EB GUIDE product line 6は、以前のメジャーバージョンと一切互換性がありません。

注記

EB GUIDE Speech Extension

EB GUIDE Speech Extensionは、アドオン製品であり、購入しなければ使用できません。

注記

ユーザーの権限

EB GUIDEをWindows 7 またはWindows 10システムにインストールするには、管理者権限が必要です。

13.1.2. システム要件

以下の構成を考慮してください。

表13.1 EB GUIDE Studioの推奨構成

ハードウェア	クアッドコアCPU(最低2 GHz)および8 GB RAMを搭載するPC
オペレーティングシステム	Windows 7、Windows 10
ディスプレイ解像度	2台のモニターを1920 x 1080ピクセルで使用
ソフトウェア	Microsoft .NET Framework 4.5.1 DirectX 11

表13.2 EB GUIDE SDKの推奨構成

ソフトウェア開発キット	Microsoft Visual Studio 2013以降
ファイル統合	CMake

13.2. EB Commandからのダウンロード

EB Commandは、EB GUIDE product lineソフトウェアのダウンロード元となるサーバーです。

注記

アカウントをアクティブにする
製品を注文すると、営業担当者からメールが送られてきます。電子メールに埋め込まれたリンクをクリックします。電子メールとブラウザに記載された手順に従ってアカウントを作成してから、ログインに進みます。

EB Commandからのダウンロード

前提条件:

- ユーザーアカウントがアクティブ化されます。

ステップ 1
ブラウザを開き、https://command.elektrobit.com/command/mod_perl/login.plに移動します。

EB Commandのフロントページが表示されます。

ステップ 2
言語を変更するには、画面の左下隅で言語を切り替えます。

ステップ 3
エイリアスを入力します。これはユーザー名となります。

ステップ 4
パスワードを入力し、[Login]ボタンをクリックします。

メインページが表示されます。

ステップ 5
プロジェクト(`EB GUIDE Studio`など)を選択します。プロジェクトの概要が表示されます。

ステップ 6
ダウンロードするバージョンの配布コンテナー(`EB GUIDE Studio Core 6.x`など)を選択します。ダウンロード可能なすべての項目の概要が表示されます。

ステップ 7
ダウンロードするファイルの横にある`Actions`チェックボックスを選択します。

ステップ 8
[Download Selection]をクリックします。

ティップ

複数のファイルのダウンロード

複数のファイルをダウンロードするために選択した場合は、ダウンロードパッケージが生成されます。ファイル`CommandDownload<date>.zip`をローカルシステムに保存するためのダイアログが表示されます。

ダウンロードを開始します。EB Commandからログアウトするために、[Logout]ボタンをクリックします。

13.3. EB GUIDEのインストール

EB GUIDEのインストール

前提条件:

- セットアップファイル`studio_setup.exe`のダウンロードが完了しています。
- オペレーティングシステムの管理者権限を持っていること。

ステップ 1
セットアップファイル`studio_setup.exe`をダブルクリックします。

ダイアログが開きます。

ステップ 2
[Yes]をクリックします。

[Setup - EB GUIDE Studio]ダイアログが開きます。

ステップ 3
ライセンス使用許諾契約書に同意し、[Next]をクリックします。

ステップ 4
インストール先のディレクトリを選択します。

デフォルトのインストールディレクトリは、`C:\Program Files (x86)\Elektrobit\EB GUIDE <version>`です。

ステップ 5
[Next]をクリックします。

要約ダイアログにすべての選択したインストール設定が表示されます。

ステップ 6
表示されている設定でインストールを実行するには、[Install]をクリックします。

インストールが開始されます。

ステップ 7
セットアップを終了するには、[Finish]をクリックします。

EB GUIDEのインストールはこれで完了です。

ティップ

複数のインストール
複数のEB GUIDEバージョンをインストールすることが可能です。

13.4. EB GUIDEのアンインストール

EB GUIDEのアンインストール

注記

EB GUIDEの永久的な削除
以下の手順に従うと、EB GUIDEがPCから永久的に削除されます。

前提条件:

- EB GUIDEのインストールが完了していること。
- オペレーティングシステムの管理者権限を持っていること。

ステップ 1
Windowsの[スタート]メニューで、[すべてのプログラム]をクリックします。

ステップ 2
[Elektrobit]メニューで、アンインストールするバージョンをクリックします。

ステップ 3
サブメニューで、[Uninstall]をクリックします。

用語集

#

3Dグラフィック	3Dグラフィックは、3Dシーンの仮想画像です。3Dシーンは、3Dモデル(メッシュや形状)、材質、光源、カメラをまとめたものです。材質は色やテクスチャ、仮想ライト効果下での動作などで3Dモデルの外観を定義します。カメラは、3Dシーンの仮想画像を撮影する視点となります。

A

アプリケーションプログラミングインターフェイス	アプリケーションプログラミングインターフェイス

C

通信コンテキスト	通信コンテキストは、通信が行われる環境を記述します。各通信コンテキストは、一意の数値IDによって識別されます。

D

データプール	データプールは、データプールアイテムへのアクセスをランタイムに提供するEB GUIDEモデル内のデータキャッシュです。アプリケーションとヒューマンマシンインターフェイスとの間のデータ交換のために使用されます。
データプールアイテム	データプールアイテムはデータを格納し、やり取りします。データプール内の各アイテムには通信方向があります。

E

EB GUIDE GTF	EB GUIDE GTFは、EB GUIDE product lineのグラフィックターゲットフレームワークであり、EB GUIDE TFの一部です。EB GUIDE GTFは、ターゲットデバイスでEB GUIDEモデルを実行するためのランタイム環境を表しています。
EB GUIDE GTF SDK	EB GUIDE GTF SDKはEB GUIDE GTFに含まれている開発環境です。EB GUIDE SDKのサブセットにあたります。もう1つのサブセットがEB GUIDE Studio SDKです。
EB GUIDEモデル	EB GUIDEモデルは、EB GUIDE Studioで作成されたヒューマンマシンインターフェイスの記述です。

EB GUIDE product line	EB GUIDE product lineは、ヒューマンマシンインターフェイスモデルを記述したり、そのモデルを組み込み環境のシステムで動作するグラフィカルユーザーインターフェイスに変換したりするために必要なソフトウェアライブラリおよびツールの集まりです。
EB GUIDEスクリプト	EB GUIDEスクリプトはEB GUIDE product lineのスクリプト記述言語です。EB GUIDEスクリプトを使用すると、データプール、モデル要素(ウィジェット、ステートマシンなど)、システムイベントにアクセスできます。
EB GUIDE SDK	EB GUIDE SDKは、EB GUIDEの製品コンポーネントであり、EB GUIDE product lineのソフトウェア開発キットです。EB GUIDE Studio SDKとEB GUIDE GTF SDKが含まれています。
EB GUIDE Studio	EB GUIDE Studioは、グラフィカルユーザーインターフェイスによってヒューマンマシンインターフェイスのモデリングや記述を行うためのツールです。
EB GUIDE Studio SDK	EB GUIDE Studio SDKは、EB GUIDE Studioと通信するためのアプリケーションプログラミングインターフェイス(API)です。EB GUIDE SDKのサブセットにあたります。もう1つのサブセットがEB GUIDE GTF SDKです。
EB GUIDE TF	EB GUIDE TFは、EB GUIDE product lineのランタイム環境であり、EB GUIDE GTFとEB GUIDE STFが含まれています。EB GUIDEモデルを実行するために必要です。

G

GL	グラフィカルライブラリ
GUI	グラフィカルユーザーインターフェイス

H

HMI	ヒューマンマシンインターフェイス

L

ライブラリ	ライブラリは、EB GUIDE Studioで使用される一連のリソースです。EB GUIDEプロジェクトに必要なライブラリは、プロジェクトセンターで定義されます。

M

モデル要素	モデル要素とは、EB GUIDEモデル内のオブジェクト(例えば、ステート、ウィジェット、データプールアイテム)です。 EB GUIDEモデル参照

O

OS	オペレーティングシステム

P

プロファイル	プロジェクトセンターにおいて、プロファイルは仕様の集まりです。プロファイルでプロジェクトのライブラリ、メッセージ、シーンを定義します。EB GUIDEモデルをエクスポートする際、プロファイルのデータは`gtfStartup.cfg`設定ファイルに書き込まれます。
プロジェクトセンター	プロジェクトセンターには、プロファイルや言語など、すべてのプロジェクト関連機能があります。
プロジェクトエディター	プロジェクトエディターでは、ヒューマンマシンインターフェイスの動作と外観をモデリングします。

R

リソース	リソースとは、EB GUIDEプロジェクトの一部となるデータパッケージです。例えば、フォント、イメージ、メッシュなどがあります。オペレーティングシステムによっては、リソースがEB GUIDEモデルの外部(例えば、ファイル)に保存されています。

S

共有ライブラリ	共有ライブラリは静的ライブラリと異なり、実行用プログラムの準備中に読み込むことができます。Windowsプラットフォームでは、共有ライブラリはダイナミックリンクライブラリと呼ばれ、ファイル拡張子は`.dll`となります。Unixシステムでは、共有ライブラリは共有オブジェクトと呼ばれ、ファイル拡張子は`.so`となります。
ステート	ステートは、ステートマシンのステータスを定義したものです。ステートやステート遷移は、ステートチャートでモデリングされます。
ステートマシン	ステートマシンとは、ステート、ステート間の遷移、動作の集合です。ステートマシンは、システムのダイナミックな動作を記述します。

T

遷移	遷移は、あるステートから別のステートへの変化を定義したものです。通常、遷移はイベントによってトリガーされます。

U

UI	ヒューマンマシンインターフェイス

V

ビュー	ビューは、プロジェクト特有のヒューマンマシンインターフェイス画面のグラフィカル表現であり、ステートマシンの特定のステートに関連しています。ビューは、ウィジェットのツリーで構成されます。

W

ウィジェット	ウィジェットは基本的なグラフィカル要素です。ウィジェットは、グラフィカルユーザーインターフェイスとの対話処理のために使用されます。

インデックス

シンボル

3Dウィジェット, 33, 52, 92
  リファレンス, 242
3Dオブジェクト, 33
3Dグラフィック, 33, 52, 188, 279
  インポート, 188
  サポートされている形式, 33, 52
  メッシュ, 52
  追加, 112
せん断
  リファレンス, 273
アイコン
  ヒューマンマシンインターフェイス, 68
アクション, 225
  エントリーアクション, 97
  終了アクション, 98
  遷移, 107
アニメーション, 35, 92, 129, 131, 181
  リファレンス, 237
  終了アニメーション, 36, 131
  開始アニメーション, 36, 131
アプリケーションプログラミングインターフェイス, 36, 279
(参照 アプリケーションプログラミングインターフェイス)
アンビエントテクスチャ
  リファレンス, 268
イベント, 48, 64
  コピー, 136
  リファレンス, 224
  貼り付け, 136
  追加, 136
イベントシステム, 48
イメージ
  9-patch, 51
  サポートされている形式, 51
  データタイプ, 198
  リファレンス, 236
  追加, 112
インスタンシエータ, 174
  リファレンス, 237
  追加, 114
インポート
  言語依存テキスト, 156
ウィジェット, 91, 282
  3Dウィジェット, 92
  アニメーション, 92
  グループ, 114
  サイズの変更, 116
  削除, 111
  基本, 92
  追加, 111
  配置, 115
ウィジェットテンプレート, 94, 132, 135
ウィジェットテンプレートインターフェイス, 94
ウィジェットプロパティ, 93
  EB GUIDEスクリプト, 62
  ウィジェットテンプレート, 94
  ウィジェットプロパティへのリンク, 117
  ウィジェット機能プロパティ, 93
  デフォルトプロパティ, 93
  データプールアイテムへのリンク, 119
  ユーザー定義プロパティ, 93, 121
  追加, 121
ウィジェット機能
  パスジェスチャー, 172
  削除, 126
  追加, 124
ウィンドウ表示リスト
  データプールアイテム, 46
エクスポート
  言語依存テキスト, 154
エミッシブテクスチャ
  リファレンス, 269
エントリーアクション, 102
  ステートマシン, 97
カメラ
  リファレンス, 244
カメラビューポート
  リファレンス, 268
キーUnicode
  リファレンス, 261
キーのステータス変更

リファレンス, 262
キーリリース
リファレンス, 262
キー押下
リファレンス, 261
グリッドレイアウト
リファレンス, 264
コピー
イベント, 136
データプールアイテム, 139
コマンドエリア
プロジェクトエディター, 42
コマンドライン, 68
コンスタント曲線
リファレンス, 238
コンソール (参照 コマンドライン)
コンテナー
リファレンス, 236
追加, 114
コンテンツエリア
プロジェクトエディター, 41
プロジェクトセンター, 38
システムメッセージ, 153
シミュレーション, 148
ショートカット
ヒューマンマシンインターフェイス, 68
シーングラフ, 33, 52, 112, 188
テクスチャ, 188
リファレンス, 243
追加, 112
シーングラフノード
リファレンス, 243
シーン設定
リファレンス, 231
ジェスチャー, 89
パスジェスチャー, 89
リファレンス, 261
非パスジェスチャー, 89
ジェスチャーID
リファレンス, 256
スクリプト値, 66, 140
スクリプト曲線, 241
ステータスバー
プロジェクトエディター, 45
ステート, 71, 98, 99, 162, 281
エントリーアクション, 101
ビューステート, 73
初期ステート, 73
履歴ステート, 76
最終ステート, 74
混合ステート, 71
終了アクション, 102
遷移, 103
選択ステート, 75
ステートマシン, 70, 281
UML 2.5記法, 87
UMLとの比較, 87
インクルードステートマシン, 70, 88
ステート, 71
ステートマシンの実行, 83
ハプティックステートマシン, 70
ロジックステートマシン, 70
削除, 98
動的ステートマシン, 70
追加, 96
遷移, 79
スピン
リファレンス, 248
スペキュラテクスチャ
リファレンス, 271
スポットライト
リファレンス, 245
タッチ
リファレンス, 249
タッチのステータス変更
リファレンス, 260
タッチの喪失
リファレンス, 259
タッチジェスチャー (参照 ジェスチャー)
タッチリリース
リファレンス, 259
タッチ入力 (参照 ジェスチャー)
タッチ押下
リファレンス, 259

タッチ移動
リファレンス, 260
ツールボックス
プロジェクトエディター, 43
テキストの切り捨て
リファレンス, 250
テンプレート
作成, 132
使用, 134
削除, 135
テンプレートインターフェイス, 133
プロパティの削除, 133
プロパティの追加, 133
テンプレートインデックス
リファレンス, 267
ディスプレイ
設定, 154
ディフューズテクスチャ
リファレンス, 268
データタイプ
イメージ, 198
フォント, 198
ブール値, 196
ブール値リスト, 196
メッシュ, 196
メッシュリスト, 196
リスト, 199
整数, 198
文字列, 200
条件スクリプト, 197
浮動小数点数, 197
色, 196
データプール, 45, 279
データプールアイテム, 46, 140, 279
インポート, 156
ウィンドウ表示リスト, 46
エクスポート, 154
コピー, 139
リスト, 139
リファレンス, 195
リンク, 143
言語サポート, 185
貼り付け, 139
追加, 138
トリガー
遷移, 105
トークン, 225
ナビゲーションエリア
プロジェクトエディター, 39
プロジェクトセンター, 38
ノーマルマップテクスチャ
リファレンス, 270
パスジェスチャー, 172
リファレンス, 255, 256
ビュー, 91, 282
リファレンス, 233
追加, 110
ビューテンプレート
リファレンス, 233, 234
ピボット
リファレンス, 272
ピンチジェスチャー
リファレンス, 257
フォント, 51
データタイプ, 198
フォーカス
リファレンス, 247
フリックジェスチャー
リファレンス, 253
フローレイアウト
リファレンス, 264
ブール値
データタイプ, 196
ブール値リスト
データタイプ, 196
プロジェクトエディター, 39, 281
コマンドエリア, 42, 45
コンテンツエリア, 41
ツールボックス, 43
ナビゲーションエリア, 39
問題エリア, 45
プロジェクトセンター, 37, 281
コンテンツエリア, 38
ナビゲーションエリア, 38

プロパティパネル
コマンドエリア, 44
プロジェクトエディター, 44
プロファイル, 281
gtfStartup.cfg, 150
複製, 151
ホールドジェスチャー
リファレンス, 254
ボタン
ヒューマンマシンインターフェイス, 68
ボックスレイアウト
リファレンス, 263
ポートの表示
リファレンス, 267
マルチサンプリング, 233
マルチタッチ入力, 90
メッシュ, 52
データタイプ, 196
リファレンス, 245
メッシュリスト
データタイプ, 196
メッセージ, 225
追加, 153
モデル要素, 47, 280
削除, 103
ユーザー定義フォーカス
リファレンス, 252
ユーザー定義プロパティ, 121
ライター通信コンテキスト, 37, 142
ライトマップテクスチャ
リファレンス, 269
ライブラリ, 280
追加, 151
ラインインデックス
リファレンス, 266
ラベル
フォント, 113
リファレンス, 235
リスト, 139
データタイプ, 199
作成, 174
リストインデックス
リファレンス, 267
リストレイアウト
リファレンス, 265
リソース, 281
3Dグラフィック, 52
イメージ, 51
フォント, 51
メッシュ, 52
リソース管理, 50
リニア曲線, 241
リニア補間整数, 181
リニア補間曲線, 242
リフレクションテクスチャ
リファレンス, 271
リンク
ウィジェットプロパティ, 117, 119
データプールアイテム, 143
リーダー通信コンテキスト, 37, 142
レイアウト余白
リファレンス, 265
レンダラー
設定, 154
ロングホールドジェスチャー
リファレンス, 254
不透明テクスチャ
リファレンス, 270
二次曲線
リファレンス, 240
低速開始曲線
リファレンス, 239
信号, 225
共有ライブラリ, 281
内部で移動
リファレンス, 258
内部へ移動
リファレンス, 258
内部遷移, 109
効果
ウィジェット機能, 251
動的ステートマシン
追加, 96, 158
可動

リファレンス, 263
問題エリア, 147
プロジェクトエディター, 45
四角形
リファレンス, 235
回転
リファレンス, 262, 273
回転ジェスチャー
リファレンス, 257
基本ウィジェット, 92
リファレンス, 234
変換
リファレンス, 274
外部へ移動
リファレンス, 258
子の可視性の選択
リファレンス, 246
押下
リファレンス, 247
拡大縮小
リファレンス, 273
拡大縮小モード
リファレンス, 266
指向性ライト
リファレンス, 244
整数
データタイプ, 198
文字列
データタイプ, 200
有効
リファレンス, 246
材質
リファレンス, 244
条件
遷移, 106
条件スクリプト
データタイプ, 197
枠
リファレンス, 251
正弦曲線
リファレンス, 240
浮動小数点数
データタイプ, 197
混合ステート, 99
点ライト
リファレンス, 245
終了アクション, 102
ステートマシン, 98
終了アニメーション, 131
リファレンス, 234
絶対レイアウト
リファレンス, 263
自動フォーカス
リファレンス, 253
色
データタイプ, 196
複数行
リファレンス, 250
言語
変更, 185
言語依存テキスト, 185
インポート, 156
エクスポート, 154
設定
ディスプレイ, 154
設定ファイル, 224
貼り付け
イベント, 136
データプールアイテム, 139
通信コンテキスト, 37, 141, 279
遷移, 79, 103, 281
アクション, 107
トリガー, 105
内部, 109
条件, 106
移動, 104
追加, 103
選択
リファレンス, 247
選択グループ
リファレンス, 248
選択ステート, 100
配色
リファレンス, 251

開始アニメーション, 131
リファレンス, 234
高速開始曲線
リファレンス, 239

E

EB GUIDE GTF, 279
EB GUIDE GTF SDK, 279
EB GUIDE Monitor, 148
EB GUIDE product line, 279
EB GUIDE SDK, 279
EB GUIDE Studio, 279
EB GUIDE Studio SDK;, 279
EB GUIDE TF, 279
EB GUIDEスクリプト, 53, 140, 279
if-then-else, 59
L値, 57
R値, 57
Whileループ, 59
イベント, 64
ウィジェットプロパティ, 62
コメント, 54
スクリプト値, 66
チュートリアル, 166
データプールアクセス, 61
データ型, 54
ネームスペース, 53
リスト, 63
ローカル変数, 57
外部関数呼び出し, 60
式, 55
文字列の書式設定, 65
標準ライブラリ, 66
識別子, 53
EB GUIDEプロジェクト, 47
EB GUIDEモデル, 47, 279
モデル要素, 47
EB GUIDE拡張機能, 49

F

finger ID, 90

G

GL, 280
gtfStartup.cfg, 224
プロファイル, 150
GUI, 280

H

HMI, 280

O

OS, 281

T

todo
EB GUIDEスクリプト, 54

U

UI, 282