



Elektrobit

EB GUIDE Studio

ユーザーマニュアル

バージョン6.6.0.142803



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

Confidential and proprietary information.

ALL RIGHTS RESERVED. No part of this publication may be copied in any form, by photocopy, microfilm, retrieval system, or by any other means now known or hereafter invented without the prior written permission of Elektrobit Automotive GmbH.

ProOSEK®, tresos®, and street director® are registered trademarks of Elektrobit Automotive GmbH.

All brand names, trademarks and registered trademarks are property of their rightful owners and are used only for description.

Copyright 2018, Elektrobit Automotive GmbH.

目次

1. 本書について	15
1.1. 対象者: モデラー	15
1.2. ユーザーマニュアルの構成	15
1.3. 本書で使用する表記スタイル	16
1.4. ネーミングルール	18
1.5. パスに関する規則	18
2. 安全で正しい使い方	20
2.1. 使用目的	20
2.2. 考えられる誤用	20
3. サポート	21
4. の基本 EB GUIDE	22
4.1. ボタン EB GUIDE product line	22
4.2. EB GUIDE Studio	22
4.2.1. ヒューマンマシンインターフェースの動作のモデル化	22
4.2.2. ヒューマンマシンインターフェースの外観のモデル化	23
4.2.3. データの処理	23
4.2.4. EB GUIDEモデルのシミュレート	23
4.2.5. EB GUIDEモデルのエクスポート	24
4.3. EB GUIDE TF	24
5. チュートリアル: はじめに	26
5.1. 起動中 EB GUIDE	26
5.2. プロジェクトの作成	27
5.3. ヒューマンマシンインターフェースの動作のモデル化	28
5.4. ヒューマンマシンインターフェースの外観のモデル化	31
5.5. シミュレーションを開始する	34
6. バックグラウンド情報	35
6.1. 3Dグラフィック	35
6.1.1. サポートされている3Dグラフィック形式	35
6.1.2. 3Dグラフィックファイルの設定	35
6.1.3. 3Dグラフィックファイルのインポート	36
6.2. アニメーション	37
6.2.1. ウィジェットのアニメーション	38
6.2.2. ビュー遷移のアニメーション	38
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	39
6.4. 通信コンテキスト	39
6.5. グラフィカルユーザーインターフェースのコンポーネント	39
6.5.1. プロジェクトセンター	40
6.5.1.1. ナビゲーションエリア	40
6.5.1.2. コンテンツエリア	40

6.5.2. プロジェクトエディター	41
6.5.2.1. ナビゲーションコンポーネント	42
6.5.2.2. [概要]コンポーネント	43
6.5.2.3. ツールボックスコンポーネント	43
6.5.2.4. プロパティコンポーネント	44
6.5.2.5. コンテンツエリア	45
6.5.2.6. [イベント]コンポーネント	47
6.5.2.7. [データプール]コンポーネント	47
6.5.2.8. [アセット]コンポーネント	47
6.5.2.9. コマンドエリア	47
6.5.2.10. 問題検出コンポーネント	49
6.5.3. ドッキング可能なコンポーネント	49
6.5.4. EB GUIDE Monitor	50
6.6. データプール	52
6.6.1. 概念	52
6.6.2. データプールアイテム	52
6.6.3. ウィンドウ表示リスト	52
6.7. EB GUIDE モデルとEB GUIDEプロジェクト	53
6.8. イベント処理	54
6.8.1. イベントシステム	54
6.8.2. イベント	54
6.9. 拡張機能	55
6.9.1. EB GUIDE Studio 拡張機能	55
6.9.2. EB GUIDE GTF 拡張機能	55
6.10. 言語	55
6.10.1. での表示言語 EB GUIDE Studio	55
6.10.2. EB GUIDEモデルの言語	56
6.10.3. 言語依存テキストのエクスポートとインポート	56
6.11. スキン	56
6.12. リソース管理	57
6.12.1. フォント	57
6.12.1.1. ビットマップフォント	58
6.12.2. イメージ	58
6.12.2.1. 9-patchイメージ	58
6.12.3. 3Dグラフィック用メッシュ	59
6.12.4. .psd ファイル形式	60
6.13. スクリプト言語 EB GUIDEスクリプト	60
6.13.1. アプリケーションの機能とエリア	60
6.13.2. ネームスペースと識別子	61
6.13.3. コメント	61
6.13.4. データ型	62
6.13.5. 式	62

6.13.6. 定数と参照	63
6.13.7. 算術式と論理式	64
6.13.8. L値とR値	64
6.13.9. ローカル変数	65
6.13.10. Whileループ	66
6.13.11. If-then-else	66
6.13.12. 外部関数呼び出し	67
6.13.13. データプールアクセス	68
6.13.14. ウィジェットプロパティ	69
6.13.15. リスト	70
6.13.16. イベント	71
6.13.17. 文字列の書式設定	72
6.13.18. 標準ライブラリ	73
6.14. スクリプト値	73
6.15. ショートカット、ボタン、アイコン	75
6.15.1. ショートカット	75
6.15.2. コマンドラインオプション	76
6.15.2.1. Studio.Console.exeのコマンドラインオプション	76
6.15.2.2. Monitor.Console.exeのコマンドラインオプション	77
6.15.3. ボタン	77
6.15.4. アイコン	78
6.16. ステートマシンとステート	79
6.16.1. ステートマシン	79
6.16.1.1. ハプティックステートマシン	79
6.16.1.2. ロジックステートマシン	79
6.16.1.3. 動的ステートマシン	79
6.16.2. ステート	80
6.16.2.1. 混合ステート	80
6.16.2.2. ビューステート	81
6.16.2.3. 初期ステート	81
6.16.2.4. 最終ステート	82
6.16.2.5. 選択ステート	83
6.16.2.6. 履歴ステート	84
6.16.3. 遷移	87
6.16.4. ステートマシンの実行	91
6.16.5. EB GUIDE の記法とUML記法の比較	95
6.16.5.1. サポートされている要素	96
6.16.5.2. サポートされない要素	96
6.16.5.3. 偏差	96
6.17. タッチ入力	97
6.17.1. 非パスジェスチャー	97
6.17.2. パスジェスチャー	97

6.17.3. 入力処理とジェスチャー	98
6.17.4. マルチタッチ入力	98
6.18. ウィジェット	99
6.18.1. ビュー	99
6.18.2. ウィジェットのカテゴリ	100
6.18.3. ウィジェットプロパティ	101
6.18.4. ウィジェットテンプレート	103
6.18.5. ウィジェット機能	103
6.18.5.1. フォーカスウィジェット機能カテゴリ	105
6.18.5.2. リスト管理ウィジェット機能カテゴリ	106
7. ヒューマンマシンインターフェースの動作のモデル化	108
7.1. ステートマシンのモデリング	108
7.1.1. ステートマシンの追加	108
7.1.2. 動的ステートマシンの追加	108
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	109
7.1.4. ステートマシンに対する終了アクションの定義	110
7.1.5. ステートマシンの削除	110
7.2. モデリングステート	110
7.2.1. ステートの追加	110
7.2.2. 混合ステートへのステートの追加	111
7.2.3. 選択ステートの追加	112
7.2.4. ステートに対するエントリーアクションの定義	113
7.2.5. ステートに対する終了アクションの定義	114
7.2.6. ステートマシンからのモデル要素の削除	115
7.3. ステート間を遷移で接続	115
7.3.1. 2つのステート間に遷移を追加	115
7.3.2. 遷移の移動	116
7.3.3. 遷移に対するトリガーの定義	117
7.3.4. 遷移への条件の追加	118
7.3.5. 遷移へのアクションの追加	119
7.3.6. ステートへの内部遷移の追加	121
8. ヒューマンマシンインターフェースの外観のモデル化	122
8.1. ウィジェットの操作	122
8.1.1. ビューの追加	122
8.1.2. ビューへの基本ウィジェットの追加	123
8.1.2.1. 四角形を追加する	123
8.1.2.2. 楕円を追加する	123
8.1.2.2.1. 楕円を編集する	124
8.1.2.3. イメージを追加する	124
8.1.2.4. ラベルを追加する	126
8.1.2.4.1. ラベルのフォントの変更	127
8.1.2.5. コンテナを追加する	128

8.1.2.6. インスタンスエータの追加	129
8.1.2.7. アニメーションの追加	130
8.1.2.8. アルファマスクの追加	132
8.1.3. ビューへの3Dウィジェットの追加	132
8.1.3.1. ビューへのシーングラフの追加	132
8.1.4. ビューに.psdファイルを追加する	134
8.1.5. ビューからのウィジェットの削除	135
8.2. ウィジェットプロパティの操作	135
8.2.1. ウィジェットの配置	135
8.2.2. ウィジェットのサイズの変更	136
8.2.3. ウィジェットプロパティ間のリンク設定	138
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	139
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	141
8.2.5.1. タイプのユーザー定義プロパティの追加 <code>Function () : bool</code>	142
8.2.6. ユーザー定義プロパティの名前の変更	143
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	144
8.3.1. ウィジェット機能の追加	144
8.3.2. ウィジェット機能の削除	146
8.4. EB GUIDEモデルへの言語の追加	147
8.4.1. 言語の追加	148
8.4.2. 言語の削除	149
8.5. スキンのサポートの操作	149
8.5.1. EB GUIDEモデルへのスキンの追加	150
8.5.2. データプールアイテムにスキンのサポートを追加する	150
8.5.3. スキンを切り替える	151
8.5.4. スキンを削除する	152
8.6. ビュー遷移のアニメーション化	152
8.6.1. 開始アニメーションの追加	152
8.6.2. 終了アニメーションの追加	153
8.7. ウィジェットの再利用	154
8.7.1. テンプレートの追加	154
8.7.2. テンプレートインターフェースの定義	154
8.7.3. テンプレートの使用	155
8.7.4. テンプレートの削除	156
9. データの処理	157
9.1. イベントの追加	157
9.2. イベントへのパラメータの追加	157
9.3. イベントへの対応	158
9.4. イベントの削除	159
9.5. データプールアイテムの追加	159
9.6. リストタイプのデータプールアイテムの編集	160
9.7. プロパティのスクリプト値への変換	161

9.8. 外部通信の確立	162
9.9. データプールアイテム間のリンク設定	163
9.10. データプールアイテムの削除	164
10. プロジェクトの処理	166
10.1. プロジェクトの作成	166
10.2. プロジェクトを開く	167
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	167
10.2.2. プロジェクトをに開く EB GUIDE Studio	167
10.3. モデル要素の名前を変更する	168
10.4. EB GUIDEモデルの検証およびモデル実行	168
10.4.1. EB GUIDEモデルの検証	169
10.4.1.1. を使用したEB GUIDEモデルの検証 EB GUIDE Studio	169
10.4.1.2. コマンドラインを使用したEB GUIDEモデルの検証	170
10.4.2. シミュレーションの開始と停止	170
10.5. EB GUIDEモデルのエクスポート	170
10.5.1. を使用したEB GUIDEモデルのエクスポート EB GUIDE Studio	171
10.5.2. コマンドラインを使用したEB GUIDEモデルのエクスポート	171
10.6. EB GUIDE Studioの表示言語の変更	172
10.7. プロファイルの設定	172
10.7.1. プロファイルの追加	173
10.7.2. ライブラリの追加	173
10.7.3. シーンの設定	175
10.8. 言語依存テキストのエクスポートとインポート	176
10.8.1. 言語依存テキストのエクスポート	176
10.8.2. 言語依存テキストのインポート	177
10.8.2.1. EB GUIDE Studioを使用した言語依存テキストのインポート	178
10.8.2.2. コマンドラインを使用した言語依存テキストのインポート	178
10.9. EB GUIDE Monitorを操作する	179
10.9.1. EB GUIDE Monitorでイベントを発行する	179
10.9.2. でのデータプールアイテムの値の変更 EB GUIDE Monitor	180
10.9.3. EB GUIDE Monitorでスクリプトを開始する	181
10.9.3.1. でのスクリプトファイルの記述 EB GUIDE Monitor	182
10.9.4. スタンドアロンアプリケーションとしてEB GUIDE Monitorを起動する	185
11. チュートリアル	187
11.1. チュートリアル: 動的ステートマシンの追加	187
11.2. チュートリアル: EB GUIDEスクリプトを使用してボタン動作をモデリングする	195
11.3. チュートリアル: パスジェスチャーをモデル化する	202
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	205
11.5. チュートリアル: 画面内で楕円を移動する	212
11.6. チュートリアル: データプールアイテムに言語依存テキストを追加する	215
11.7. チュートリアル: 3Dグラフィックの操作	219
12. リファレンス	225

12.1. Androidイベント	225
12.2. データプールアイテム	226
12.3. データタイプ	226
12.3.1. メッシュ	226
12.3.2. ブール値	226
12.3.3. 色	227
12.3.4. 条件スクリプト	227
12.3.5. フLOAT	228
12.3.6. フォント	228
12.3.7. イメージ	228
12.3.8. 整数	229
12.3.9. リスト	229
12.3.10. 文字列	230
12.4. EB GUIDEスクリプト	230
12.4.1. EB GUIDEスクリプト キーワード	230
12.4.2. EB GUIDEスクリプト 演算子の優先順位	231
12.4.3. EB GUIDEスクリプト 標準ライブラリ	232
12.4.3.1. EB GUIDEスクリプト 関数A	232
12.4.3.1.1. abs	232
12.4.3.1.2. absf	233
12.4.3.1.3. acosf	233
12.4.3.1.4. animation_before	233
12.4.3.1.5. animation_beyond	233
12.4.3.1.6. animation_cancel	234
12.4.3.1.7. animation_cancel_end	234
12.4.3.1.8. animation_cancel_reset	234
12.4.3.1.9. animation_pause	234
12.4.3.1.10. animation_play	235
12.4.3.1.11. animation_reverse	235
12.4.3.1.12. animation_running	235
12.4.3.1.13. animation_set_time	235
12.4.3.1.14. asinf	236
12.4.3.1.15. atan2f	236
12.4.3.1.16. atan2i	236
12.4.3.1.17. atanf	237
12.4.3.2. EB GUIDEスクリプト の関数C~H	237
12.4.3.2.1. ceil	237
12.4.3.2.2. changeDynamicStateMachinePriority	237
12.4.3.2.3. character2unicode	237
12.4.3.2.4. clearAllDynamicStateMachines	238
12.4.3.2.5. color2string	238
12.4.3.2.6. cosf	238

12.4.3.2.7. deg2rad	239
12.4.3.2.8. expf	239
12.4.3.2.9. float2string	239
12.4.3.2.10. floor	239
12.4.3.2.11. focusNext	240
12.4.3.2.12. focusPrevious	240
12.4.3.2.13. format_float	240
12.4.3.2.14. format_int	241
12.4.3.2.15. getLineCount	241
12.4.3.2.16. getTextHeight	242
12.4.3.2.17. getTextLength	242
12.4.3.2.18. getTextWidth	242
12.4.3.2.19. has_list_window	243
12.4.3.2.20. hsba2color	243
12.4.3.3. EB GUIDEスクリプト の関数I〜R	243
12.4.3.3.1. int2float	243
12.4.3.3.2. int2string	244
12.4.3.3.3. isDynamicStateMachineActive	244
12.4.3.3.4. language	244
12.4.3.3.5. localtime_day	244
12.4.3.3.6. localtime_hour	245
12.4.3.3.7. localtime_minute	245
12.4.3.3.8. localtime_month	245
12.4.3.3.9. localtime_second	245
12.4.3.3.10. localtime_weekday	246
12.4.3.3.11. localtime_year	246
12.4.3.3.12. log10f	246
12.4.3.3.13. logf	246
12.4.3.3.14. nearbyint	247
12.4.3.3.15. popDynamicStateMachine	247
12.4.3.3.16. powf	247
12.4.3.3.17. pushDynamicStateMachine	247
12.4.3.3.18. rad2deg	248
12.4.3.3.19. rand	248
12.4.3.3.20. shutdown	248
12.4.3.3.21. rgba2color	248
12.4.3.3.22. round	249
12.4.3.4. EB GUIDEスクリプト の関数S〜W	249
12.4.3.4.1. seed_rand	249
12.4.3.4.2. sinf	249
12.4.3.4.3. skin	250
12.4.3.4.4. sqrtf	250

12.4.3.4.5. string2float	250
12.4.3.4.6. string2int	251
12.4.3.4.7. string2string	251
12.4.3.4.8. substring	251
12.4.3.4.9. system_time	252
12.4.3.4.10. system_time_ms	252
12.4.3.4.11. tanf	252
12.4.3.4.12. trace_dp	252
12.4.3.4.13. trace_string	253
12.4.3.4.14. transformToScreenX	253
12.4.3.4.15. transformToScreenY	253
12.4.3.4.16. transformToWidgetX	253
12.4.3.4.17. transformToWidgetY	254
12.4.3.4.18. trunc	254
12.4.3.4.19. widgetGetChildCount	254
12.5. イベント	255
12.6. model.json 設定ファイル	255
12.6.1. でのサンプルmodel.json	262
12.7. platform.json 設定ファイル	264
12.7.1. EB GUIDE Studioでのサンプルplatform.json	266
12.8. シーン	268
12.9. EB GUIDE GTFがサポートするタッチスクリーンタイプ	270
12.10. ウィジェット	271
12.10.1. ビュー	271
12.10.2. 基本ウィジェット	272
12.10.2.1. アルファマスク	272
12.10.2.2. アニメーション	273
12.10.2.2.1. コンスタント曲線	274
12.10.2.2.2. 高速開始曲線	274
12.10.2.2.3. 低速開始曲線	275
12.10.2.2.4. 二次曲線	275
12.10.2.2.5. 正弦曲線	276
12.10.2.2.6. スクリプト曲線	276
12.10.2.2.7. リニア曲線	277
12.10.2.2.8. リニア補間曲線	277
12.10.2.3. コンテナ	278
12.10.2.4. 楕円	278
12.10.2.5. イメージ	279
12.10.2.6. インスタンスエータ	279
12.10.2.7. ラベル	280
12.10.2.8. 四角形	280
12.10.3. 3Dウィジェット	281

12.10.3.1. 環境光	281
12.10.3.2. カメラ	281
12.10.3.3. 指向性ライト	282
12.10.3.4. 材質	282
12.10.3.5. メッシュ	282
12.10.3.6. PBR GGX材質	283
12.10.3.7. PBR Phong材質	284
12.10.3.8. 点ライト	285
12.10.3.9. シーングラフ	285
12.10.3.10. シーングラフノード	286
12.10.3.11. スポットライト	286
12.11. ウィジェット機能	287
12.11.1. 共通	287
12.11.1.1. 子の可視性の選択	287
12.11.1.2. 有効	287
12.11.1.3. フォーカス	288
12.11.1.4. 複数行	288
12.11.1.5. 押下	289
12.11.1.6. 選択	289
12.11.1.7. 選択グループ	289
12.11.1.8. スピン	290
12.11.1.9. テキストの切り捨て	291
12.11.1.10. タッチ	291
12.11.2. 効果	292
12.11.2.1. 枠	292
12.11.2.2. 配色	293
12.11.2.3. ストローク	293
12.11.3. フォーカス	294
12.11.3.1. 自動フォーカス	294
12.11.3.2. ユーザー定義フォーカス	294
12.11.4. ジェスチャー	295
12.11.4.1. フリックジェスチャー	295
12.11.4.2. ホールドジェスチャー	296
12.11.4.3. ロングホールドジェスチャー	296
12.11.4.4. パスジェスチャー	297
12.11.4.4.1. ジェスチャーID	298
12.11.4.5. ピンチジェスチャー	299
12.11.4.6. 回転ジェスチャー	300
12.11.5. 入力処理	300
12.11.5.1. ジェスチャー	300
12.11.5.2. キー押下	301
12.11.5.3. キーリリース	301

12.11.5.4. キーのステータス変更	301
12.11.5.5. キーUnicode	302
12.11.5.6. 内部へ移動	302
12.11.5.7. 外部へ移動	303
12.11.5.8. 内部で移動	303
12.11.5.9. 可動	304
12.11.5.10. 回転	304
12.11.5.11. タッチの喪失	304
12.11.5.12. タッチ移動	305
12.11.5.13. タッチ押下	305
12.11.5.14. タッチリリース	306
12.11.5.15. タッチのステータス変更	306
12.11.6. レイアウト	307
12.11.6.1. 絶対レイアウト	307
12.11.6.2. ボックスレイアウト	308
12.11.6.3. フローレイアウト	309
12.11.6.4. グリッドレイアウト	309
12.11.6.5. レイアウト余白	310
12.11.6.6. リストレイアウト	310
12.11.6.7. 拡大縮小モード	312
12.11.7. リスト管理	312
12.11.7.1. ラインインデックス	312
12.11.7.2. リストインデックス	313
12.11.7.3. テンプレートインデックス	313
12.11.7.4. ビューポート	313
12.11.8. 3D	314
12.11.8.1. カメラビューポート	314
12.11.8.2. アンビエントテクスチャ	314
12.11.8.3. ディフューズテクスチャ	315
12.11.8.4. エミッシブテクスチャ	316
12.11.8.5. ライトマップテクスチャ	317
12.11.8.6. ノーマルマップテクスチャ	318
12.11.8.7. 不透明テクスチャ	319
12.11.8.8. リフレクションテクスチャ	319
12.11.8.9. スペキュラテクスチャ	320
12.11.8.10. トーンマッピング	321
12.11.9. 変換	322
12.11.9.1. ピボット	322
12.11.9.2. 回転	323
12.11.9.3. 拡大縮小	323
12.11.9.4. せん断	323
12.11.9.5. 変換	324

13. のインストール EB GUIDE Studio	325
13.1. バックグラウンド情報	325
13.1.1. 制限	325
13.1.2. システム要件	325
13.2. ダウンロードする EB GUIDE	326
13.3. インストールする EB GUIDE	326
13.4. をアンインストールする EB GUIDE	327
用語集	329
インデックス	333

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ース(HMI)を作成するために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ー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ースを定義します。

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、重要な情報を示すために次の絵文字およびシグナル語が使用されます。

シグナル語**警告**は、正常に設定するための必須の情報を示します。

警告



ソースと問題の種類

ソフトウェアに発生する可能性があること

その問題によってどのような状態になるか

問題を回避する方法

シグナル語**注意**は、対象事項に関する重要な情報を示します。

注記



重要な情報

対象事項に関する重要な情報を示します。

シグナル語**ヒント**は、役に立つヒント、コツ、およびショートカットを提供します。

ティップ



役立つヒント

役立つヒントを示します

このドキュメントでは、一部の単語や語句を[太字]、**斜体**、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、次の例をご覧ください。

デフォルトのテキストは、Arial RegularフォントまたはMS PGothicフォントで表記します。

フォント	説明	使用例
太字	新しい用語または重要な用語を強調する	設定の 基本構成要素 は、モジュール設定です。
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: <code>function_name/abcdirectory</code> 。
等幅フォント (Courier)	ユーザーの入力、コード、およびファイルのディレクトリ	<pre>CC_FILES_TO_BUILD = (PROJECT_PATH) / source/network/can_node.c CC_- FILES_TO_BUILD += \$(PROJECT_PATH) / source/network/can_config.c</pre> <p>このモジュールはBswM_Dcm_ RequestSessionMode() 関数を呼び出します。</p> <p>プロジェクト名は、Project_Testと入力します。</p>
角括弧[]	オプションのパラメータがあるコマンド構文、GUI要素でオプションパラメータを示す	<ol style="list-style-type: none">insertBefore [<opt>][Enter]キーを押します。
波括弧{ }	必須のパラメータがあるコマンド構文で必須パラメータを示す	insertBefore {<file>}
三点リーダー	複数のパラメータを持つコマンド構文の場合に、パラメータがさらに続くことを示す	insertBefore [<opt>...]
破断線	使用可能なパラメータのいずれかを選択するコマンド構文で、使用可能なすべてのパラメータを示す	allowinvalidmarkup {on off}



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 6.6
```

- ▶ 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 6.6/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 6.6/projects/
```

- ▶ EB GUIDEモデルのエクスポート先ディレクトリは、\$EXPORT_PATHと表されます。

1.5. パスに関する規則

EB GUIDE Studio は、Windows 10で260文字を超えるパス名を扱えます。フルパス名は260文字を超えることができても、パス内の単一のファイルまたはディレクトリの名前には依然として248文字の制限があります。

注記



長いパス名 **Windows 7**

Windows 7 では、長いパス名は扱えません。長いパス名を使用するには、Windows 10でEB GUIDE Studioを実行します。Windows 10で長いパス名を有効にする方法については、Windows 10のドキュメントをご覧ください。

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、インフォテイン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ー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、一切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。

- ▶ Elektrobit Automotive GmbH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ー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Elektrobit Automotive GmbH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. の基本 EB GUIDE

EB GUIDE は、ヒューマンマシンインターフェース(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. ボタン 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ー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や各種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ー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ー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ー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ー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ースとアプリケーションとの間の通信は、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詳細な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ー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。

EB GUIDE Monitorを使用してEB GUIDEモデルを制御し、次の操作を実行することもできます。

- ▶ データプールアイテムの値を変更することによって、表示されるデータを変更する
- ▶ イベントを発行することによって、ユーザー入力をモデル実行する
- ▶ ログのすべての変更を追跡する
- ▶ スクリプトを開始する

EB GUIDE Monitorは、スタンドアロンアプリケーションとして使用することもできます。

4.2.5. EB GUIDEモデルのエクスポート

EB GUIDEモデルを対象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、対象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一連の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 Target Frameworkは、グラフィカルヒューマンマシンインターフェース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、ヒューマンマシンインターフェース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ー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)で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

音声ユーザーインターフェースは、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、音声認識と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にはサードパーティの音声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

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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ー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。

所要時間: 20分

5.1. 起動中 EB GUIDE



起動中 EB GUIDE

前提条件:

- EB GUIDE がインストールされていること。

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、起動するバージョンをクリックします。

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

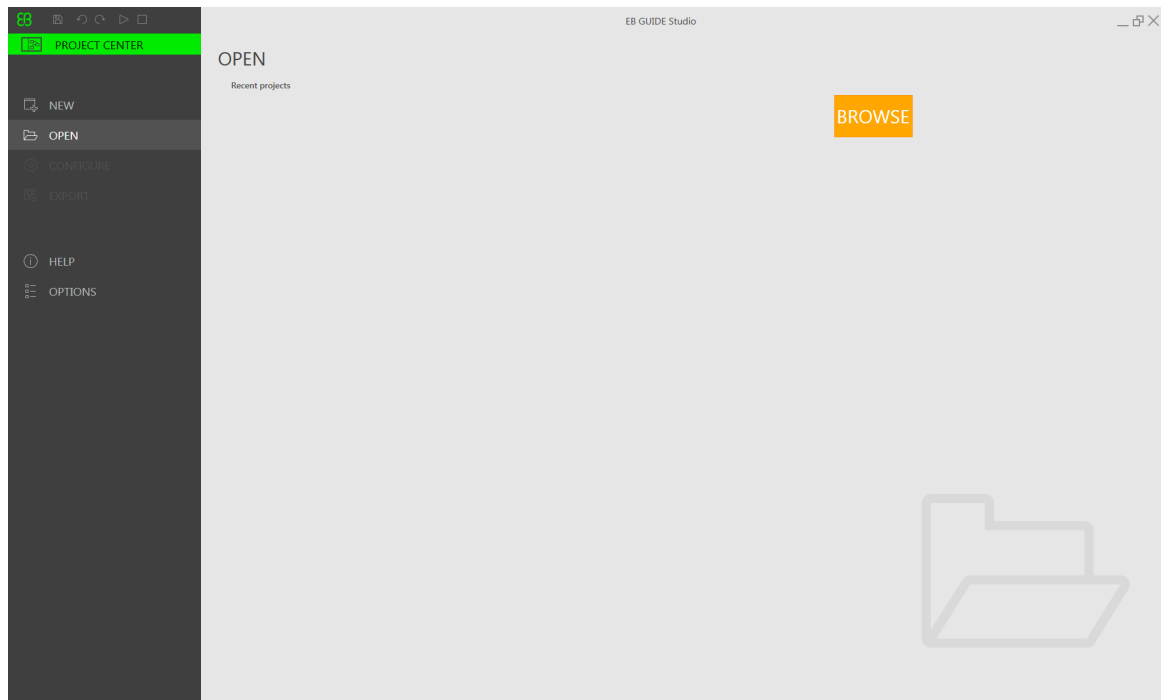


図5.1 プロジェクトセンター

5.2. プロジェクトの作成



プロジェクトの作成

前提条件:

- 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。

ステップ 1

プロジェクトセンターの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 2

コンテンツエリアで、[場所]としてC:/tempディレクトリを選択します。

ステップ 3

MyProjectというプロジェクト名を入力します。

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。

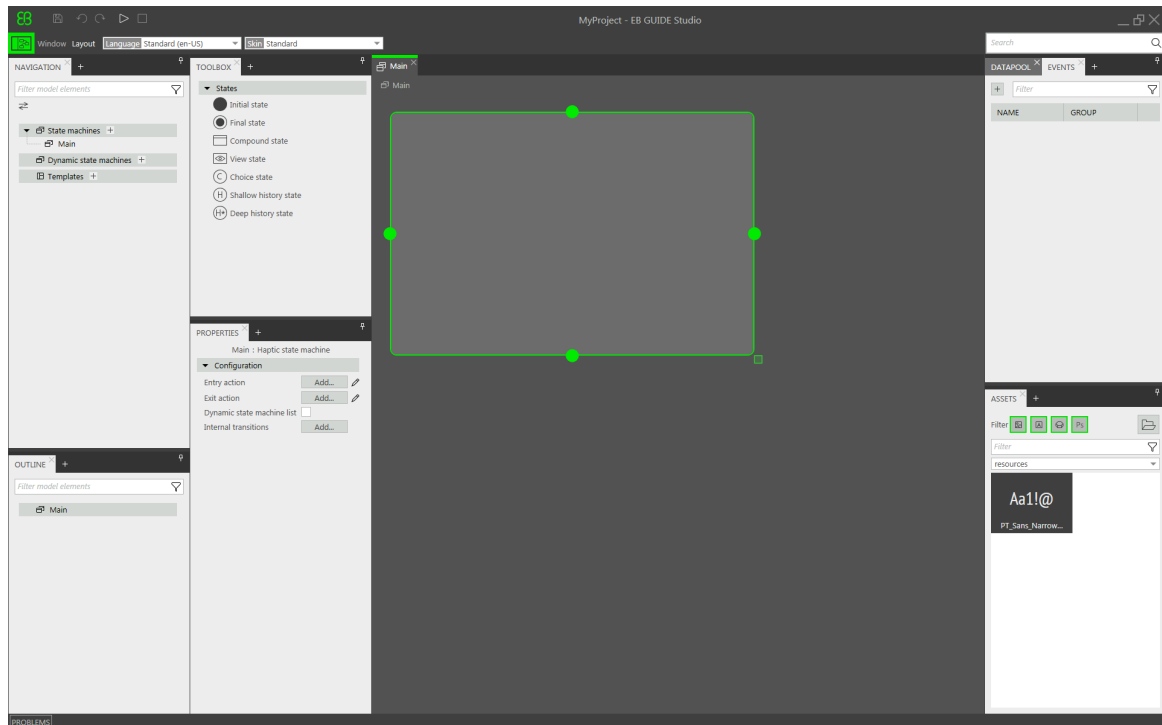


図5.2 [メイン]ステートマシンが表示されたプロジェクトエディター

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。

前提条件:

- 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。

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1]と共に、ビューがEB GUIDEモデルに追加されます。

ステップ 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。

ステップ 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。

ステップ 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]コンポーネントの両方に、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。

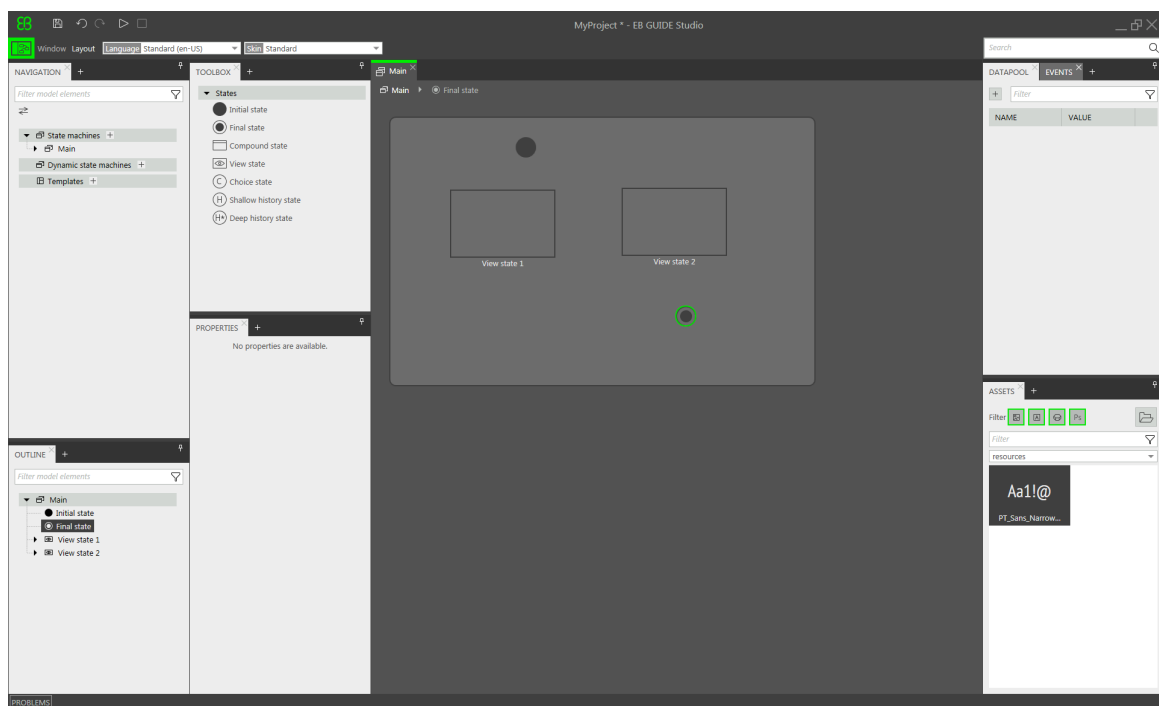


図5.3 ステートが表示されたプロジェクトエディター



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。

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、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。

ステップ 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。

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である[ビューステート1]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。

遷移が作成され、緑色の矢印として表示されます。

ステップ 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。

ステップ 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。

ステップ 7

[プロパティ]コンポーネントに移動し、[トリガー]コンボボックスにEvent 1と入力して、[イベントの追加]をクリックします。

[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]が発行されたときには、必ずこの遷移が実行されます。

ステップ 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。

[イベント2]をトリガーとして新たに追加します。

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。

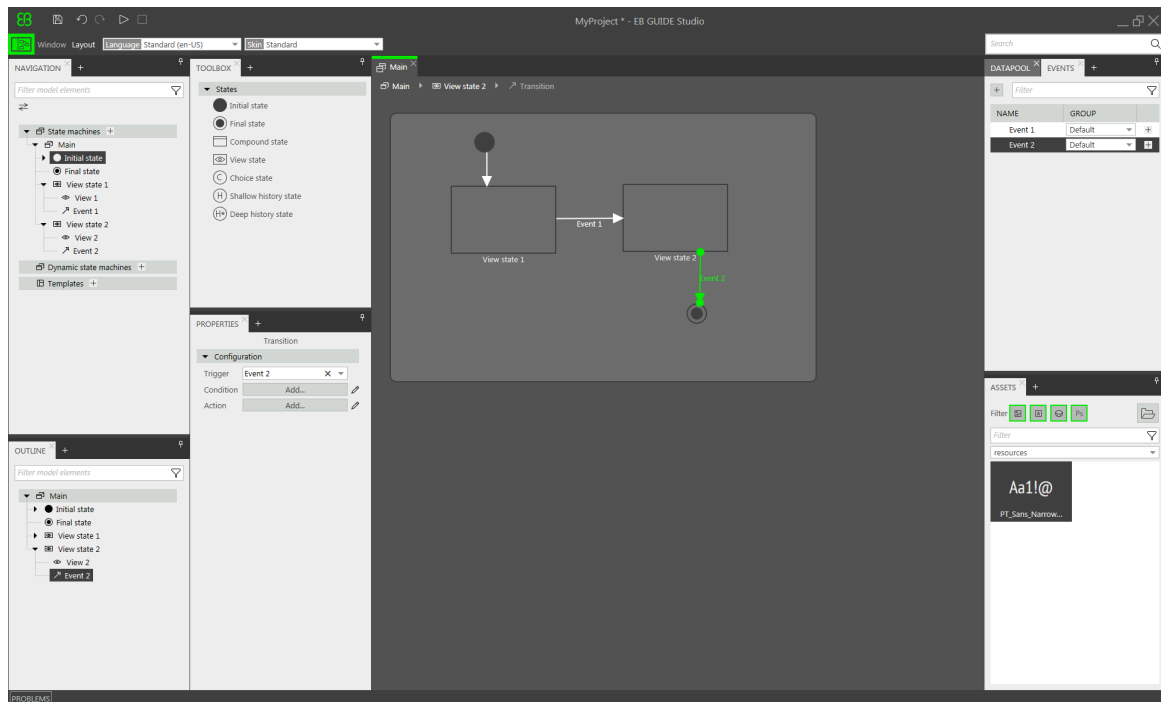


図5.4 イベントを伴う遷移でリンクされたステート

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ー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

前提条件:

- [ビューステート1]がモデルに追加されていること。

ステップ 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。

前提条件:

- コンテンツエリアに、[ビュー1]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

[プロパティ]コンポーネントで[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。

ステップ 3

[使用可能なウィジェット機能]の下で、[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]コンポーネントに追加されます。

ステップ 4

[プロパティ]コンポーネントで、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。

ステップ 5

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

ステップ 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]が発行されます。

ステップ 7

[承認]をクリックします。

ステップ 8

[プロパティ]コンポーネントで、fillColorプロパティとして赤を選択します。

ステップ 9

[ナビゲーション]コンポーネントで、[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。

ステップ 10

ステップ1~5を繰り返します。

ステップ 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
}
```

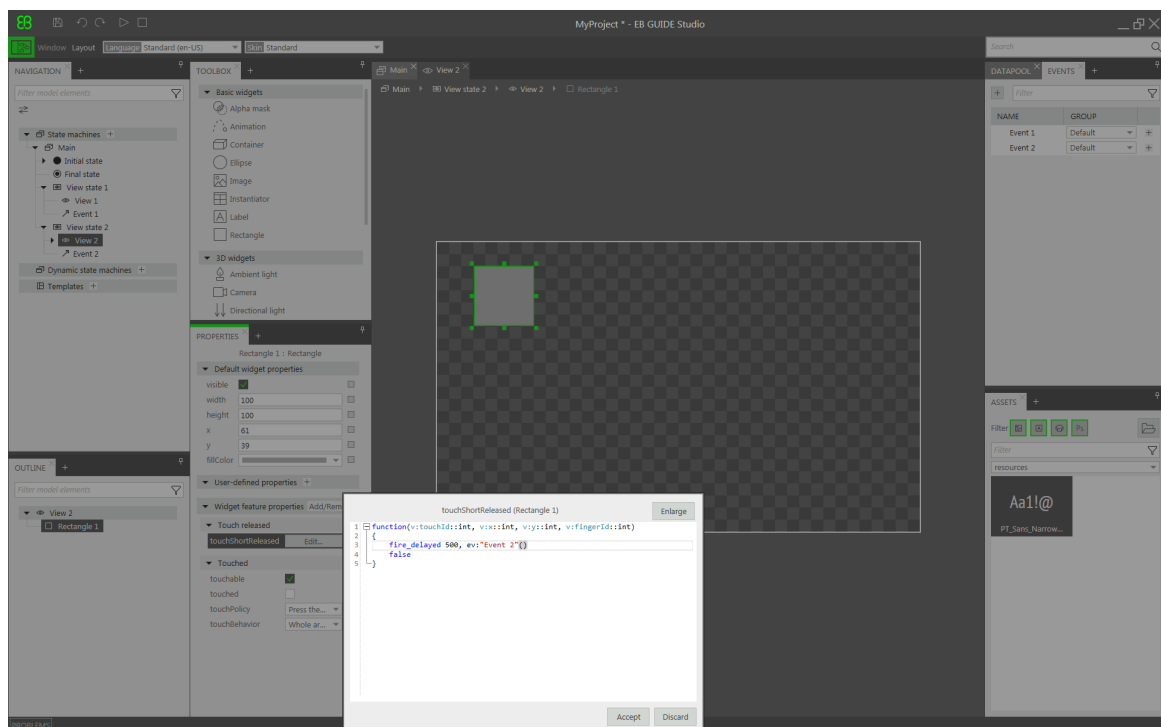


図5.5 を持つウィジェットプロパティ EB GUIDEスクリプト

ステップ 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]が発行されます。

ステップ 13

[プロパティ]コンポーネントで、fillColorプロパティとして青を選択します。


5.5. シミュレーションを開始する

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。




シミュレーションを開始する

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

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]が発行され、[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]が発行され、[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。

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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バージョン2.0以上および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
- ▶ メッシュおよび少なくとも1つの材質を含む少なくとも1つのオブジェクト
- ▶ 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ンネルを含む3Dオブジェクト
- ▶ 指向性光源

- ▶ 環境光源
- ▶ 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。インポートする3Dグラフィックファイルの3Dシーンにアニメーションが含まれている場合、EB GUIDE Studioはリニアキー値補間整数曲線およびリニアキー値補間浮動小数点数曲線を使用してそれらのアニメーションをインポートします。

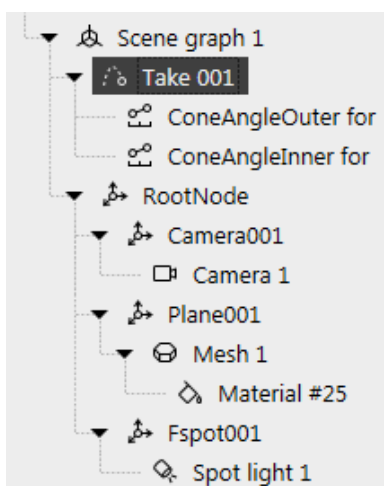


図6.1 [ナビゲーション]コンポーネントに表示されるシーングラフの例

注記



制限

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。

.fbxファイルのインポート時にはデフォルトの材質ウィジェットのみが作成されます。3Dモデルにその他のタイプの材質がある場合は、EB GUIDE Studioによってデフォルトの材質のみが追加され、そのプロパティはデフォルト値に設定されます。EB GUIDE Studioでは、PBR Phong材質およびPBR GGX材質ウィジェットを使用して、その他のタイプの材質を追加できます。

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、[6.18「ウィジェット」](#)、[12.10.3「3Dウィジェット」](#)、および[12.11.8「3D」](#)をご覧ください。

手順については、[8.1.3.1「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[基本ウィジェット]カテゴリには、アニメーションと呼ばれるウィジェットが含まれています。どのアニメーションに対しても、コンスタント曲線、リニア補間曲線、正弦曲線など、一連の曲線を追加できます。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、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を意味します。

アニメーションと曲線のプロパティについては、[12.10.2.2「アニメーション」](#)をご覧ください。

手順については、[8.1.2.7「アニメーションの追加」](#)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プロパティについては、[12.10.1「ビュー」](#)をご覧ください。

手順については、[8.6「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との間のすべての通信データがアプリケーションプログラミングインターフェイス(API)で抽象化されます。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、ライターアプリケーションで発行され、リーダーアプリケーションで更新されるまで、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。

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ースのコンポーネント

EB GUIDE Studioのグラフィカルユーザーインターフェースは、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つのコンポーネント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ー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。

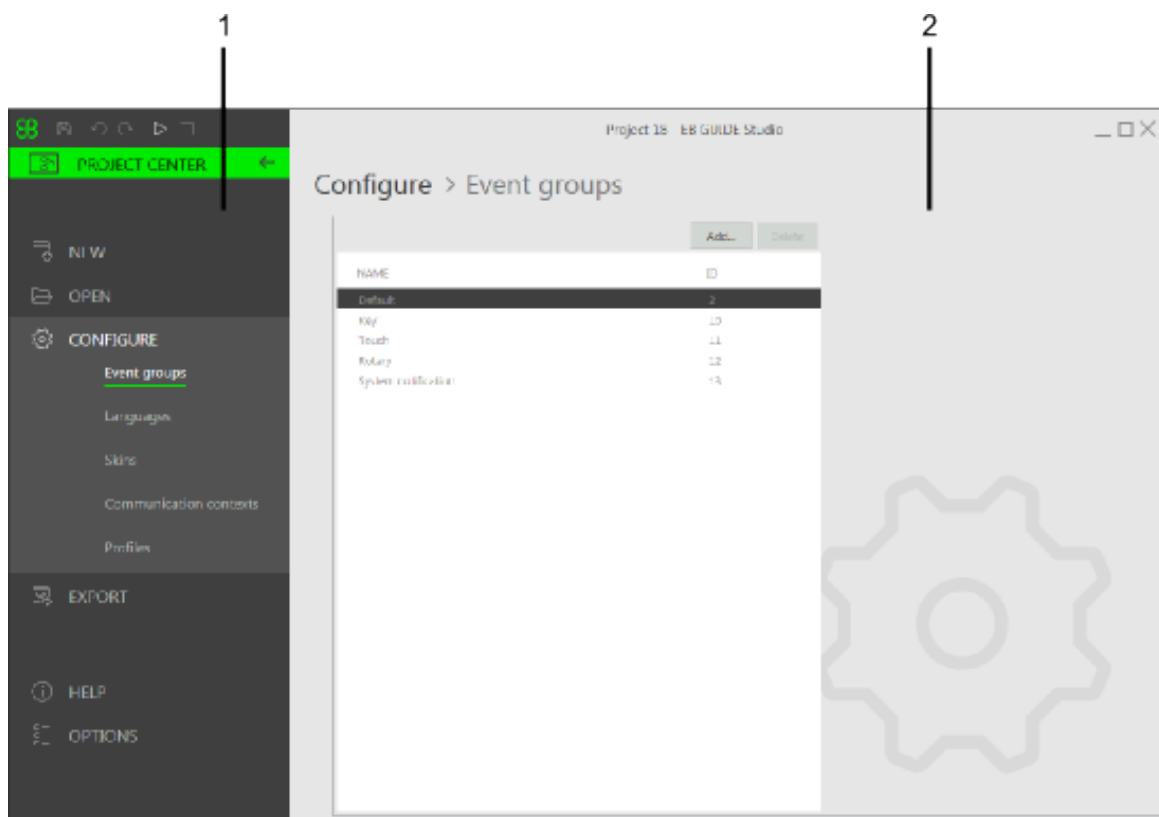


図6.2 ナビゲーションエリア(1)とコンテンツエリア(2)で構成されるプロジェクトセンター

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ています。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ー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、以下のエリアおよびコンポーネントで構成されています。プロジェクトのすべてのコンポーネントはドッキングまたはフローティングにすることができ、コンテンツエリアを除くプロジェクトエディターの任意の位置に配置できます。

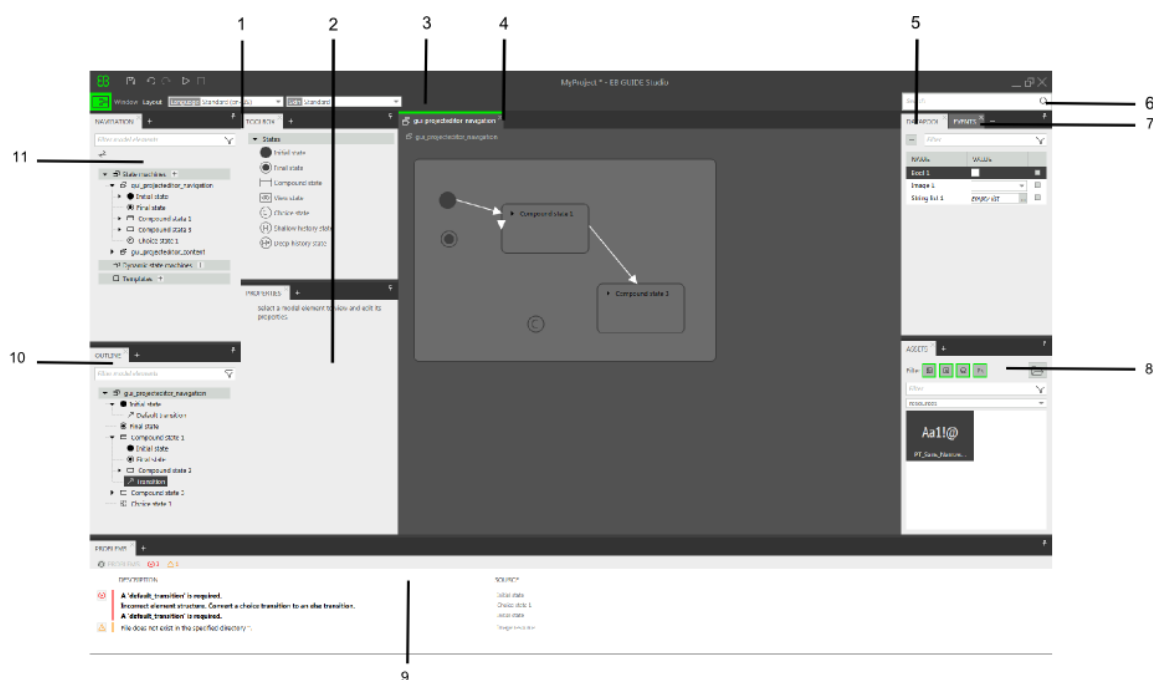


図6.3 プロジェクトエディターとそのエリアおよびコンポーネント

1 [ツールボックス]コンポーネント

2 [プロパティ]コンポーネント

3 コマンドエリア

4 コンテンツエリア

5 [データプール]コンポーネント

6 検索ボックス

7 [イベント]コンポーネント

8 [アセット]コンポーネント

9 [問題検出]コンポーネント

10 [概要]コンポーネント

11 [ナビゲーション]コンポーネント

6.5.2.1. ナビゲーションコンポーネント

[ナビゲーション]コンポーネントには、EB GUIDEモデルのステート、ビュー、アニメーション、遷移などのモデル要素が階層構造として表示され、任意の要素に移動できます。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]コンポーネントには、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。

EB GUIDEモデルに要素(ステートマシン、動的ステートマシン、テンプレートなど)を追加する場所でもあります。ウィジェットやアニメーションなど、[ツールボックス]にある要素をドラッグアンドドロップ操作によって追加できます。

最上部には、コンポーネント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]コンポーネント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。

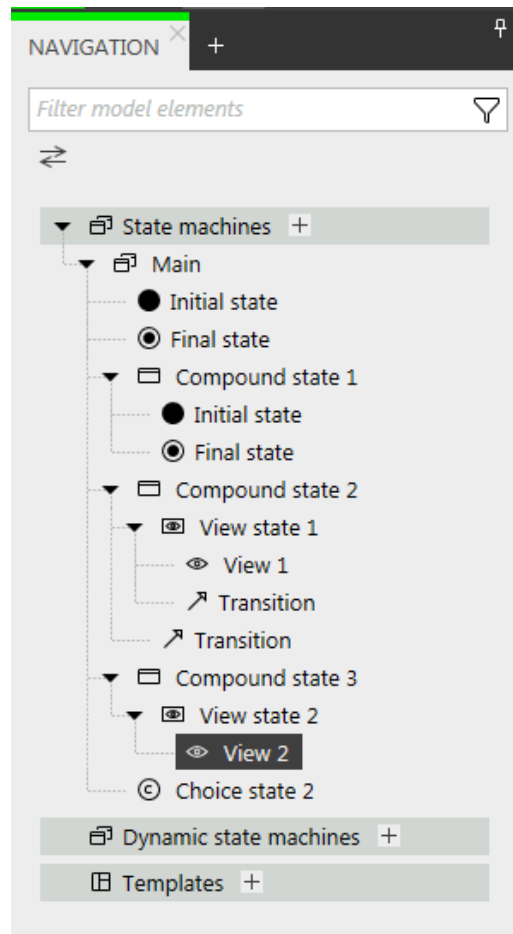


図6.4 プロジェクトエディターの[ナビゲーション]コンポーネント

6.5.2.2. [概要]コンポーネント

[ナビゲーション]コンポーネント、または現在コンテンツエリアに表示されているエディターコンポーネントで選択されたツリー部分に含まれている構造およびモデル要素のみが表示されます。

注記



フィルタボックス

コンポーネントの最上部には、コンポーネント内の要素を検索するためのフィルタボックスがあります。

コンポーネント内の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。

6.5.2.3. ツールボックスコンポーネント

モデリングに必要なツールはすべて、[ツールボックス]コンポーネント([ツールボックス]とも呼ばれます)にあります。コンテンツエリアに表示されている要素に応じて、[ツールボックス]では異なる一連のツールが提供され、それらはコンテンツエリアまたは[ナビゲーション]コンポーネントにドラッグできます。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。

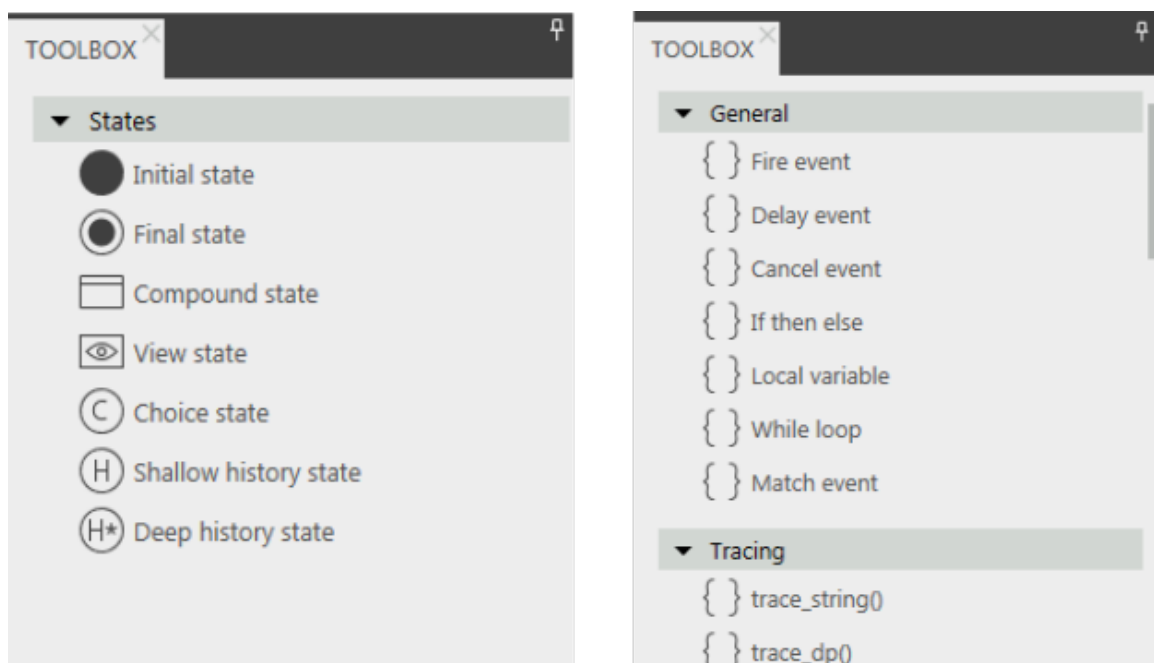


図6.5 プロジェクトエディターのツールボックス

6.5.2.4. プロパティコンポーネント

[プロパティ]コンポーネントには、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。プロパティはカテゴリでグループ化され、[プロパティ]コンポーネントで編集できます。

プロパティをクリックしてF3キーを押すと、参照検索が開始されます。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。

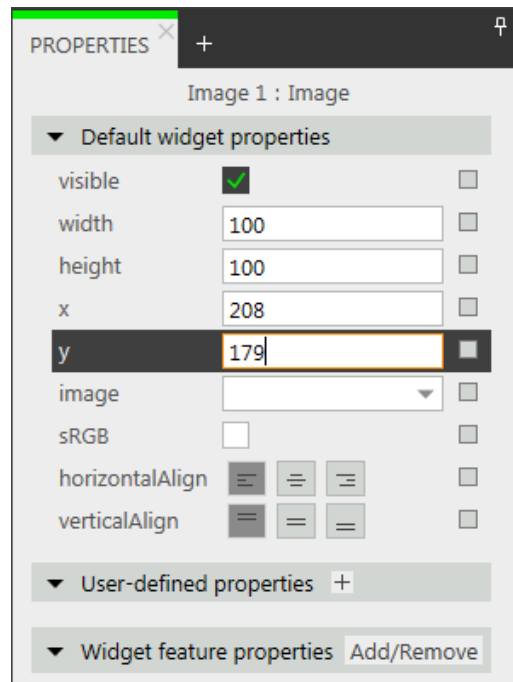


図6.6 ウィジェットのプロパティが表示された[プロパティ]コンポーネント

6.5.2.5. 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]コンポーネント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]コンポーネント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。

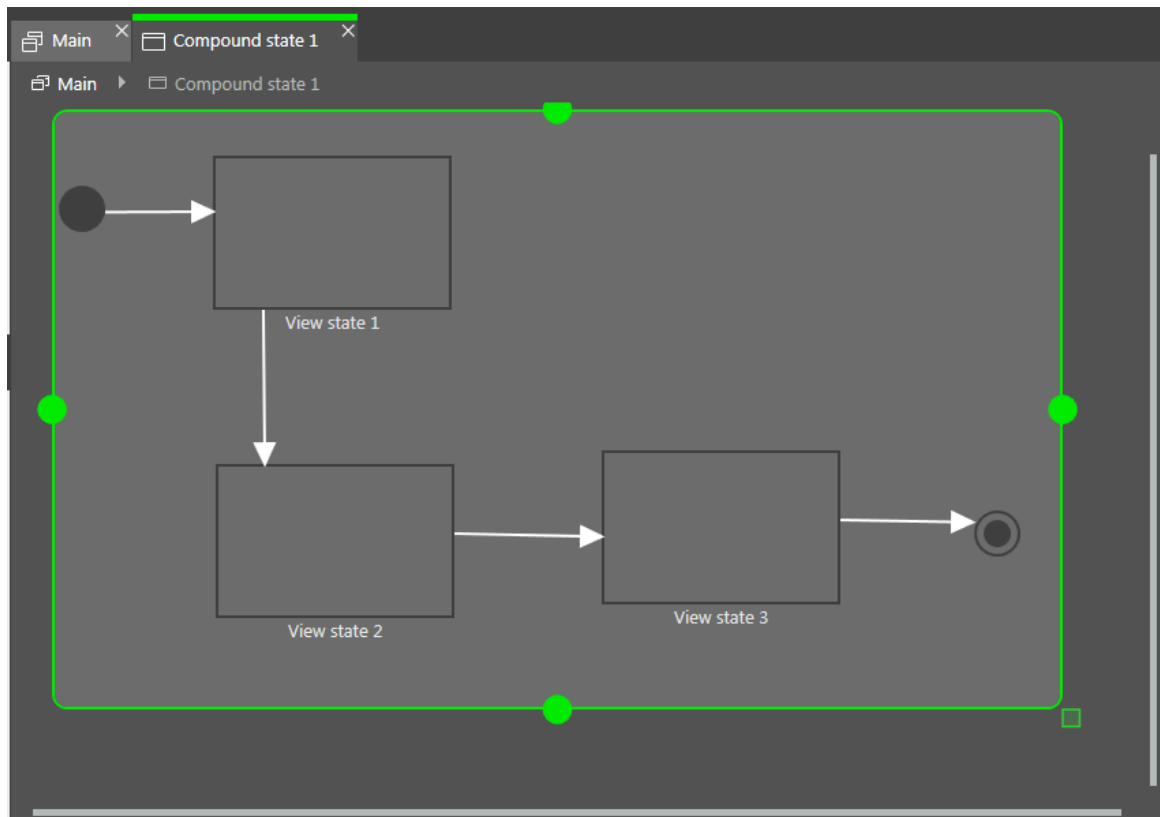


図6.7 プロジェクトエディターのコンテンツエリア

コンテンツエリアに開いているビューがあり、そのビューにアニメーションが含まれている場合は、[アニメーション]エディターが開かれます。[アニメーション]エディターでは、曲線をウィジェットプロパティに追加できます。また、プレビュー内のハンドルを移動することで、曲線の`delay`および`duration`プロパティを編集することもできます。

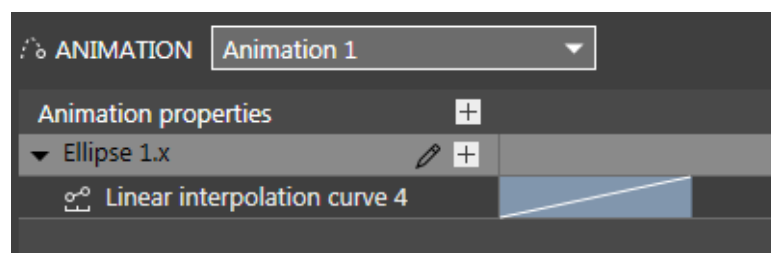


図6.8 アニメーションエディター

リファレンス検索を開始するには、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キーを押します。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モデル内でのすべての出現箇所のリストが表示されます。

6.5.2.6. [イベント]コンポーネント

ここでは、モデルにイベントを追加したり、プロパティ(イベント表の[Name]、[グループ]、[タイプ]、[Parameter name]など)を編集できます。

注記



フィルタボックス

コンポーネントの最上部には、コンポーネント内の要素を検索するためのフィルタボックスがあります。

コンポーネント内の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。

6.5.2.7. [データプール]コンポーネント

ここでは、[データプール]アイテムを追加したり、プロパティ([Name]、[値]など)を編集したりできます。データプールアイテムへのリンクの追加、値からスクリプトへの変換、および言語サポートやスキンのサポートの追加を行うこともできます。

注記



フィルタボックス

コンポーネントの最上部には、コンポーネント内の要素を検索するためのフィルタボックスがあります。

コンポーネント内の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。

6.5.2.8. [アセット]コンポーネント

ここでは、イメージ、フォント、.ebmeshファイル、.psdファイルなどのリソースを追加できます。`$GUIDE_PROJECT_PATH/<project name>/resources`ディレクトリとそのサブディレクトリにあるすべてのリソースファイルがコンポーネントのプレビューエリアに表示されます。ドラッグアンドドロップ操作によってモデルにリソースを追加できます。

注記




フィルタボックス

コンポーネントの最上部には、コンポーネント内の要素を検索するためのフィルタボックスがあります。

コンポーネント内の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。

6.5.2.9. コマンドエリア

コマンドエリアには、次のものがあります。

- ▶ プロジェクトセンターを開くための  ボタン
- ▶ モデルの要素を検索してそれらにジャンプするための検索ボックス
- ▶ 詳細なメニュー

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。

カテゴリ	説明
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。

6.5.2.10. 問題検出コンポーネント

[問題検出]コンポーネントでは、モデルが有効であるかどうかをチェックできます。現在開いているEB GUIDEモデルの考えられるエラーと警告が表示されます。問題が発生する部分に直接ジャンプするには、説明をダブルクリックします。

6.5.3. ドッキング可能なコンポーネント

プロジェクトのすべてのコンポーネントは、タブとしてドッキングしたり、ドッキング解除によってフローティングコンポーネントにしたりすることができます。コンポーネントはフローティングコンポーネントとして、コンテンツエリアを除くプロジェクトセンターの任意の部分にドラッグできます。

ドッキングコントロールの矢印はドッキングの位置を選択するときに役立ち、ライブプレビューにはレイアウトがどのようになるかが表示されます。

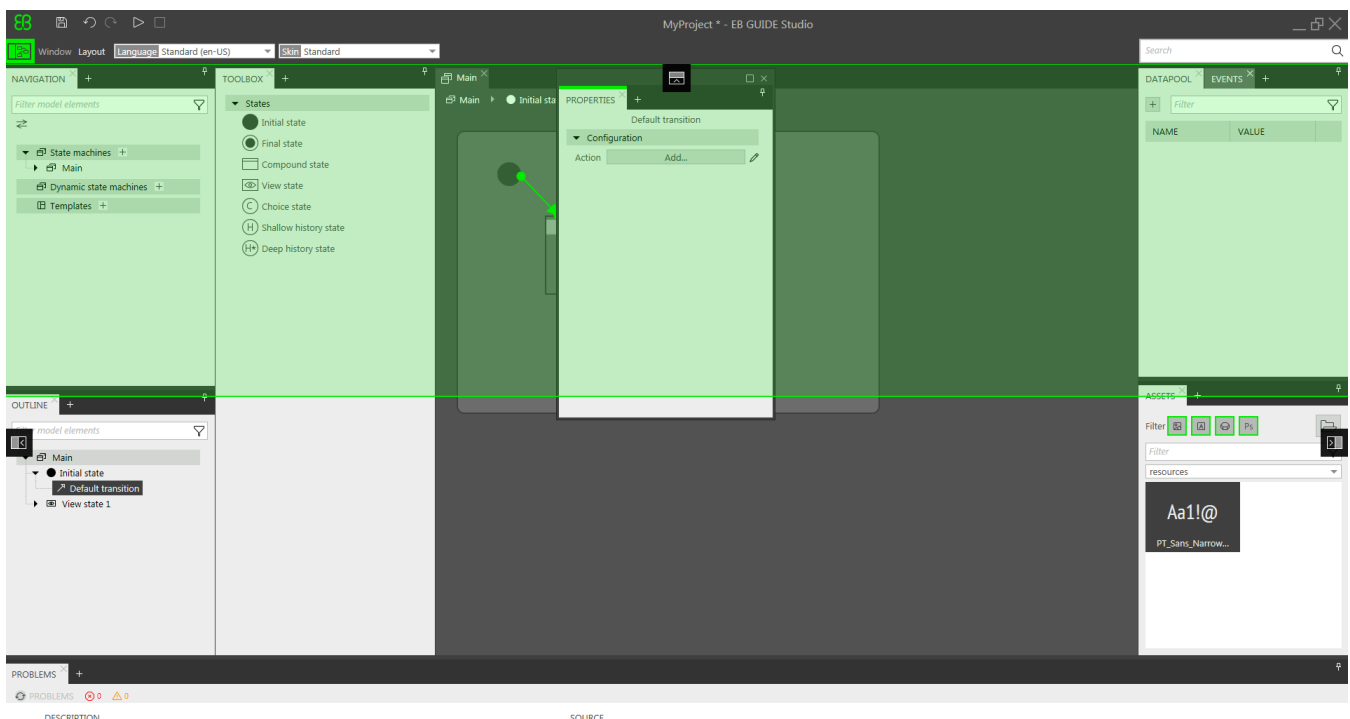


図6.9 ドッキングコントロールとライブプレビュー

注記



デフォルトレイアウト

デフォルトレイアウトに復帰するには、コマンドエリアに移動して、[レイアウト] > [Reset to default layout]を選択します。

注記



自動非表示

プロジェクトエディターで使用可能な領域を広げるには、コンポーネントを非表示にすることができます。

- ▶ コンポーネントまたはコンポーネントグループを非表示にするには、ピン記号をクリックします。
- ▶ 非表示のコンポーネントを表示するには、マウスをタブの上に合わせて、ピン記号を再びクリックします。

6.5.4. EB GUIDE Monitor

EB GUIDE は、モデル実行時にEB GUIDEモデルを監視および制御するためのEB GUIDE Monitorツールを提供しています。EB GUIDE Monitor には、EB GUIDEモデルのデータプール、イベントシステム、およびステートマシンと通信するためのメカニズムが含まれています。

EB GUIDE Monitor は、EB GUIDEモデルのモデル実行時にEB GUIDE Studioで自動的に起動されます。EB GUIDE Monitorは、スタンドアロンアプリケーションとして使用することもできます。

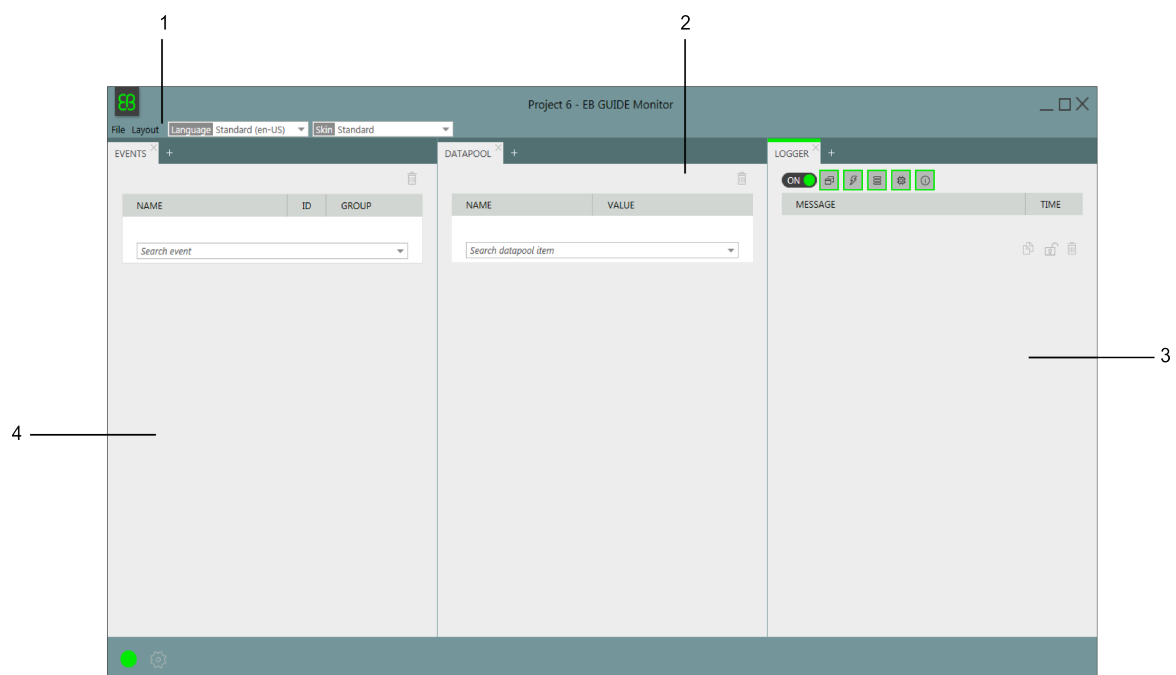


図6.10 EB GUIDE Monitor のデフォルトレイアウト

1 [レイアウト]メニュー

2 [データプール]コンポーネント

3 [ロガー]コンポーネント



4 [イベント]コンポーネント

EB GUIDE Monitor には、次のコンポーネントが含まれています。

- ▶ [イベント]コンポーネントでは、イベントを検索および発行できます。イベントにパラメータがある場合は、そのパラメータを変更したうえでこのイベントを発行することができます。
- ▶ [データプール]コンポーネントでは、データプールアイテムを検索して、それらの値を変更できます。
- ▶ [ロガー]コンポーネントでは、すべての変更、情報メッセージ、エラー、および警告が追跡されます。コンポーネントの最上部には、コンポーネント内のエントリーにフィルタをかけるためのフィルタボックスがあります。自動スクロール機能を有効または無効にするには、コンポーネントの最下部にある自動スクロールのチェックボックスをオンまたはオフにします。
- ▶ [スクリプト]コンポーネントでは、スクリプトを開始したり、出力スクリプトメッセージを確認したりできます。[スクリプト]コンポーネントは、デフォルトレイアウトには表示されません。このコンポーネントを追加するには、[レイアウト] > [スクリプト]をクリックします。
- ▶ [ステートマシン]コンポーネントには、現在アクティブなステートとステートマシンが表示されます。[ステートマシン]コンポーネントは、デフォルトレイアウトに含まれていません。このコンポーネントを追加するには、[レイアウト] > [ステートマシン]をクリックします。

コンポーネントを再配置したり、プロジェクトのニーズに応じて新しいコンポーネントを追加したりできます。EB GUIDE Monitorウィンドウ内でコンポーネントをドッキングおよびドッキング解除することもできます。

EB GUIDE Monitorウィンドウの左下隅には、接続ステータスを示す次のボタンがあります。

ボタン	ステータス
	EB GUIDE Monitor が接続されます。 ボタンをクリックすると、EB GUIDE Monitorが切断されます。
	EB GUIDE Monitor は切断されています。 ボタンをクリックすると、EB GUIDE Monitorが接続されます。
	EB GUIDE Monitor は切断されています。 ボタンをクリックすると、EB GUIDE Monitorの接続設定を設定できます。

コマンドエリアのドロップダウンボックスを使用すると、言語およびスキンを変更することもできます。

手順については、[10.9「EB GUIDE Monitorを操作する」](#)をご覧ください。

EB GUIDE Monitor APIについては、`$GUIDE_INSTALL_PATH/doc/monitor/monitor_api.chm`をご覧ください。

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、以下の目的に使用するモデル要素です。

- ▶ 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。

通信チャンネルを開くには、ライターおよびリーダーアプリケーションを使用します。

内部通信は、データを格納するために使用されます。2つのアプリケーションを使うと、外部通信が確立されます。

手順については、[9.8「外部通信の確立」](#)をご覧ください。

6.6.3. ウィンドウ表示リスト

EB GUIDE product lineは、ウィンドウ表示リストの概念をサポートします。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つのアプリケーション(例えば、メディアアプリケーション)によって提供され、別のアプリケーション(例えば、ヒューマンマシンインターフェース)によってその一部のみが表示されます。

ライターアプリケーションは、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。リーダーアプリケーションは、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アプリケーションがライターアプリケーション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いたときに、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。ウィンドウの更新は、カーソルが移動した後に少し遅延します。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上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、アプリケーション内部での通信、またはアプリケーション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、異なるサブスクリ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るモデル要素です。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拡張機能は、EB GUIDE Studioに対する補足であり、すべてのEB GUIDEモデルで有効です。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拡張機能はEB GUIDE GTFに対する補足であり、EB GUIDE Studioに追加の機能を提供しますが、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.new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. での表示言語 EB GUIDE Studio

EB GUIDE Studio にはグラフィカルユーザーインターフェー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、[10.6「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ースでは、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。

注記



スキンのサポートは使用できません

データプールアイテムに言語サポートを定義した場合、同じアイテムにスキンのサポートを追加することはできません。

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に言語依存テキストを追加する」](#)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者に転送します。.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、[10.8「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. スキン

スキンを使用すると、同じEB GUIDEモデルに異なるデータプール値を定義することによって、異なるヒューマンマシンインターフェースを定義できます。このようにして、同じヒューマンマシンインターフェースにさまざまな外観(夜用モードと昼用モードのスキンなど)を定義できます。

ランタイム中にスキンを切り替えると、異なるデータプール値の効果を確認できます。

スキンのサポートは標準のデータプール値にのみ使用でき、スクリプト値またはリンクされたデータプールアイテムには使用できません。

注記



言語サポートは使用できません

データプールアイテムにスキンのサポートを定義した場合、同じアイテムに言語サポートを追加することはできません。

手順については、[8.5「スキンのサポートの操作」](#)をご覧ください。

6.12. 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 は次のタイプのリソースファイルをサポートします。

1. フォント
2. イメージ
3. 3Dグラフィック用メッシュ
4. .psd ファイル形式

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ディレクトリ`$GUIDE_PROJECT_PATH/<project name>/resources`に追加する必要があります。

6.12.1. 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)、OpenTypeフォント(*.otf)、およびビットマップフォント(*.fnt)です。

手順については、[8.1.2.4.1「ラベルのフォントの変更」](#)をご覧ください。

6.12.1.1. ビットマップフォント

EB GUIDE Studio のバージョン3.0では、Angelcodeによる*.fntビットマップフォントをサポートしています。ビットマップフォントを作成するには、サードパーティのフォントジェネレータ(Angelcodeビットマップフォントジェネレータなど)を使用します。詳しくは、<http://www.angelcode.com>をご覧ください。

生成されたフォントの以下の設定項目について確認します。

- ▶ 適切なフォントサイズが定義されている。
- ▶ 文字セットがUnicodeである。
- ▶ フォント記述子がバイナリである。
- ▶ テクスチャが8ビットの.pngファイルとして提供されている。

次のことに注意してください。

- ▶ EB GUIDE Studioでは、ラベルのfontプロパティを使用してビットマップフォントのフォントサイズを変更することができません。つまり、.fntフォントを生成する際にサイズを定義する必要があります。
- ▶ [ストローク]ウィジェット機能は、ビットマップフォントに適用されません。フォント用に特定のアウトラインが必要な場合は、.fntフォントの生成時に定義します。
- ▶ \$GUIDE_PROJECT_PATH/resourcesディレクトリ内に、サードパーティツールで生成した.fntビットマップフォントおよび.pngテクスチャファイル用のサブディレクトリを1つ作成します。EB GUIDE Studio では、.png ファイルが.fntファイルと同じディレクトリ内にあるものと想定しています。

複数のビットマップフォントがある場合は、そうしたフォントごとにサブディレクトリを作成します。

6.12.2. 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)、JPEG (*.jpg)、および9-patchイメージ(*.9.png)です。

手順については、[8.1.2.3「イメージを追加する」](#)をご覧ください。

6.12.2.1. 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。

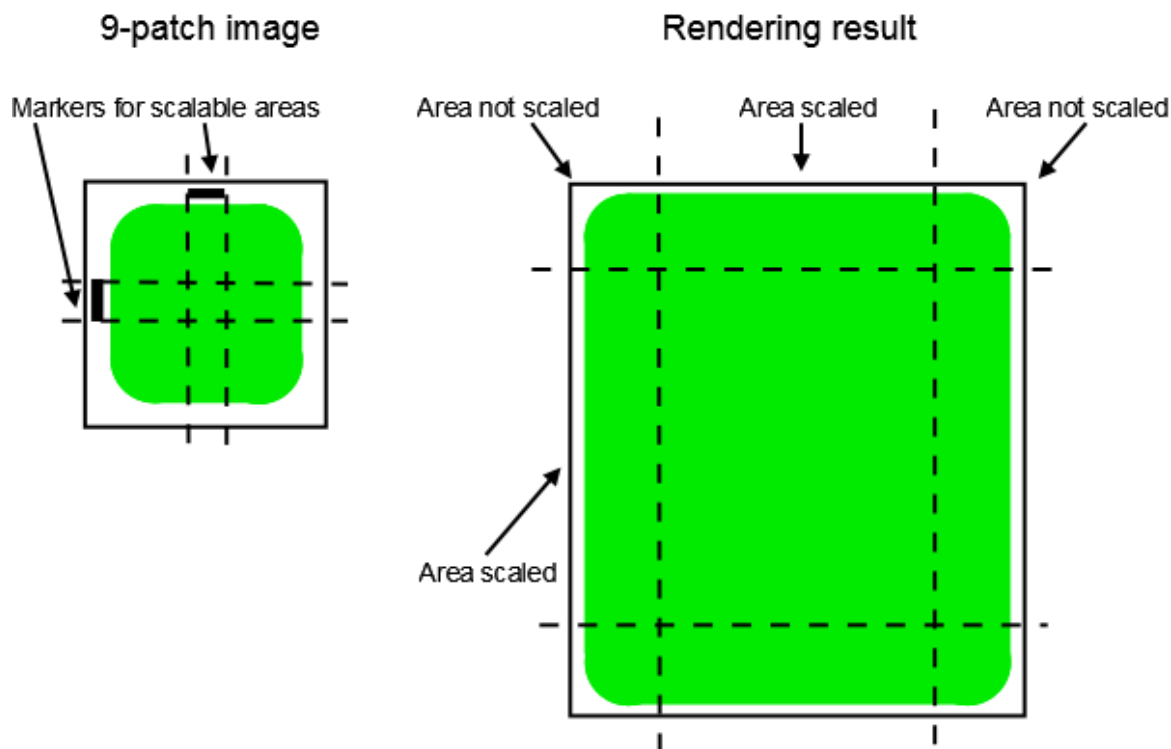


図6.11 9-patchの例

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バージョン2.0以上およびDirectXレンダラーを使う必要があります。
- ▶ 9-patch処理は、.pngイメージにのみ適用されます。
- ▶ 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。

手順については、[8.1.2.3「イメージを追加する」](#)をご覧ください。

6.12.3. 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、[8.1.3.1「ビューへのシーングラフの追加」](#)をご覧ください。

6.12.4. .psd ファイル形式

EB GUIDE Studio は .psd ファイル形式をサポートします。EB GUIDE Studio に .psd ファイルをインポートすると、ウィジェットツリーが作成されます。ウィジェットツリーは、インポート中に .psd ファイルのレイヤーから作成されたコンテナとイメージで構成されます。次のことに注意してください。

- ▶ .psd ファイルのレイヤーが非表示に設定されている場合、対応するコンテナまたはイメージの `visible` プロパティの横にあるチェックボックスがクリアされます。
- ▶ .psd ファイルのレイヤーに透過性値が設定されている場合は、インポート後に、[配色]ウィジェット機能が対応するイメージに追加されます。`colorationColor` プロパティのアルファチャネルは、.psd ファイルと同じ透過性値に設定されます。

手順については、[8.1.4「ビューに .psd ファイルを追加する」](#)をご覧ください。

6.13. 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。

6.13.1. 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利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

6.13.2. 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!";
```

6.13.3. コメント

EB GUIDEスクリプト には2種類のコメントがあります。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[問題検出]コンポーネント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。

6.13.4. 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dot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。

6.13.5. 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
```

6.13.6. 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 {}
- ▶ キーワードを使用 unit



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
```

6.13.7. 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、等価演算子の等しい(==)または等しくない(!=)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。

- ▶ 文字列どうしは(+)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
```

6.13.8. L値とR値

EB GUIDEスクリプトの式には、l-values およびr-valuesという2種類のものがあります。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。

6.13.9. 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式の構文は、次のとおりです。

```
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;
    }
}
```

6.13.10. Whileループ

while EB GUIDEスクリプトのループ構文は、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bool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();
}
```

6.13.11. If-then-else

if-then-else EB GUIDEスクリプトのは、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 の使用方法 `if-then-else`

```
// 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 {}
```

6.13.12. 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);
```

6.13.13. 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datapool item of type X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います。
- ▶ リダイレクトリンク演算子(=>)は、データプールアイテムのリンクターゲットを変更します。リンクソースにすることができるのは、すでにリンクされているデータプールアイテムのみです。

6.13.14. 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`という変数が作成されます。

スクリプトは、ウィジェットの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への反応として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;
}
```

6.13.15. 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。

6.13.16. 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;
}
```

6.13.17. 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);
```

6.13.18. 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。

6.14. 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6.12 EB GUIDE StudioのEB GUIDEスクリプトエディター

- ▶ [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の書き込み時に呼び出されます。

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、イベントのトリガー後またはプロパティの更新後、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、スクリプト実行の原因を表します。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できます。
- [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。

6.15. ショートカット、ボタン、アイコン

6.15.1. ショートカット

次の表に、EB GUIDE Studioで利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	&tf;のユーザーマニュアルを開く EB GUIDE TF
F1キー	のユーザーマニュアルを開く EB GUIDE Studio
Shift+F2	[データプール]または[イベント]コンポーネント、および選択した要素が使用されているすべての場所(例えば、EB GUIDEスクリプト)の選択した要素の名前を変更します。データプールアイテムとイベントに使用できます。
F2キー	選択された要素の名前変更
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]エリアまたはコンポーネントから削除
-	[ナビゲーション]または[概要]コンポーネント内の選択した要素を折りたたむ
*と+	[ナビゲーション]または[概要]コンポーネント内の選択したモデル要素を展開

ショートカット	説明
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

6.15.2. コマンドラインオプション

6.15.2.1. Studio.Console.exeのコマンドラインオプション

次の表に、EB GUIDE StudioのStudio.Console.exeで利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。未定義のコマンドラインオプションは無視されます。

コマンドラインの一般的な構文は次のとおりです。

```
Studio.Console.exe <option> "project_name.ebguide"
```

表6.4 Studio.Console.exeのコマンドラインオプション

オプション	説明
-c <logfile dir>	EB GUIDEモデルを検証し、logfile dirで指定されたディレクトリにログファイルを書き込む
-e <destination dir>	EB GUIDEモデルを出力先ディレクトリにエクスポート destination dir コマンドラインオプション-pとともに使用。以下の例をご覧ください。
-h	ヘルプメッセージを表示
-l <language file>	language file (.xliff)として保存された1つの言語ファイルをEB GUIDEモデルにインポートし、ログファイルを作成します
-m	プロジェクトの移行を許可
-o	プロジェクトファイルを開く
-p <profile>	エクスポート中にprofileを指定されたプロファイルとして使用



例6.23 コマンドラインオプション

コマンドラインStudio.Console.exe -e "C:/temp/exported_project" -p "target_profile" -o "project_name.ebguide"は、プロファイルtarget_profileを使用してproject_name.ebguideを指定された出力先ディレクトリC:/temp/exported_projectにエクスポートします。

手順については、以下をご覧ください。

- ▶ [10.4.1.2「コマンドラインを使用したEB GUIDEモデルの検証」](#)
- ▶ [10.5.2「コマンドラインを使用したEB GUIDEモデルのエクスポート」](#)
- ▶ [10.8.2.2「コマンドラインを使用した言語依存テキストのインポート」](#)

6.15.2.2. Monitor.Console.exeのコマンドラインオプション

次の表に、EB GUIDE MonitorのMonitor.Console.exeで利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。未定義のコマンドラインオプションは無視されます。

コマンドラインの一般的な構文は次のとおりです。

```
Monitor.Console.exe <option> "monitor.cfg"
```

表6.5 Monitor.Console.exeのコマンドラインオプション

オプション	説明
-c <host:port>	EB GUIDEモデルを実行中のEB GUIDE GTFプロセスに接続
-h	ヘルプメッセージを表示
-l <language>	EB GUIDE Monitorの言語を en (英語)、ja(日本語)、ko(韓国語)、zh-cn(中国語)のいずれかに設定
-o	設定ファイルを開く monitor.cfg
-s	定義済みスクリプトのすべてのメソッドを実行



例6.24 コマンドラインオプション




コマンドラインMonitor.Console.exe -l koは、EB GUIDE Monitorの言語を韓国語に設定します。














EB GUIDE Monitorの使用方法については、[10.9「EB GUIDE Monitorを操作する」](#)をご覧ください。

6.15.3. ボタン

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

表6.6 EB GUIDE Studioのボタン








ボタン	説明
	元に戻す
	やり直し
	保存




ボタン	説明
	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]コンポーネントの同期を取る
	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。 <div style="display: flex; flex-direction: column; gap: 5px;"> <div> プロパティがローカルである</div> <div> プロパティが別のプロパティにリンクされている</div> <div> 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</div> <div> プロパティ値テンプレート値と等しい</div> </div>
	イベントを発行する

6.15.4. 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6.7 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

アイコン	説明
	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ースに追加されていることを示す

6.16. ステートマシンとステート

6.16.1. 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、ステート間の遷移で構成されます。

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。

6.16.1.1. 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。

6.16.1.2. 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なしでロジックを指定できます。

6.16.1.3. 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。

6.16.2. 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

6.16.2.1. 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6.13 混合ステート

[ナビゲーション]コンポーネントには、ステートの階層がツリー構造で表示されます。

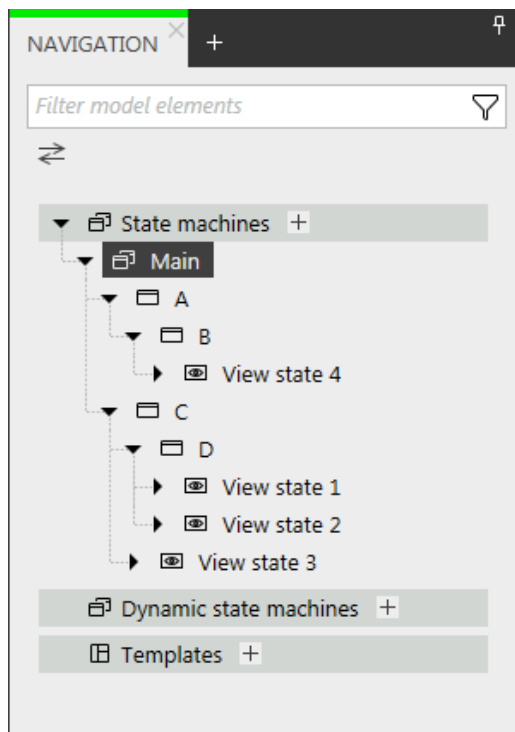


図6.14 ツリー構造で表示されたステート階層

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。

6.16.2.2. 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ー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ースであるウィジェットで構成されます。

6.16.2.3. 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)

▶ 混合状態の子状態への遷移を使う

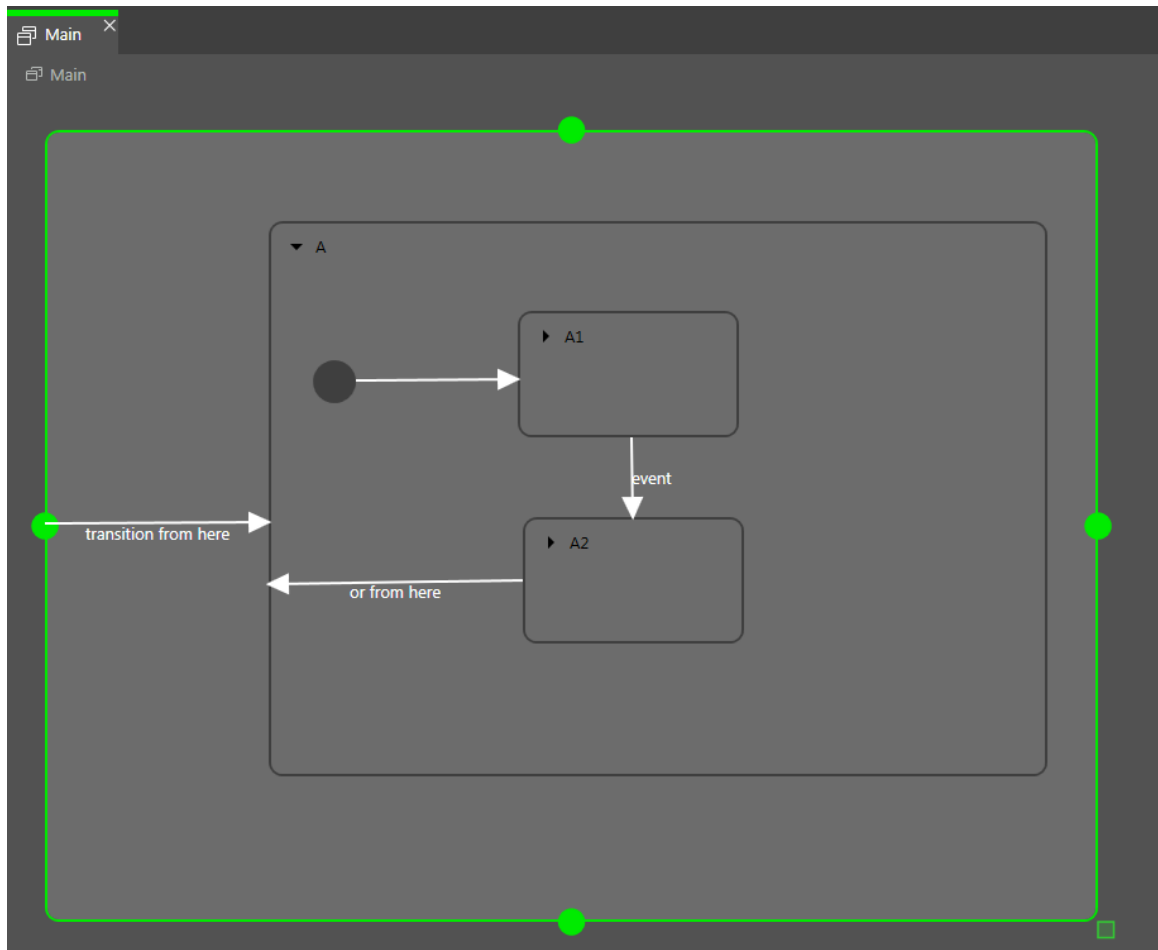


図6.15 初期状態の例

6.16.2.4. 最終状態

最終状態は混合状態を終了するために使用します。状態マシンの最終状態へエントリーすると、状態マシンの動作は終了します。混合状態内の履歴状態はリセットされます。最終状態から他の状態へ向かう遷移はありません。

混合状態には最終状態を1つだけ作成できます。最終状態がトリガーされるのは、以下の状況です。

- ▶ 子状態から混合状態の外部への遷移が発生した(イベントZによる遷移)
- ▶ 混合状態からの出力遷移が発生した(イベントYによる遷移)
- ▶ 混合状態内で最終状態への遷移が発生した(イベントXによる遷移)

最終状態を含む混合状態には、出力遷移が必要です。

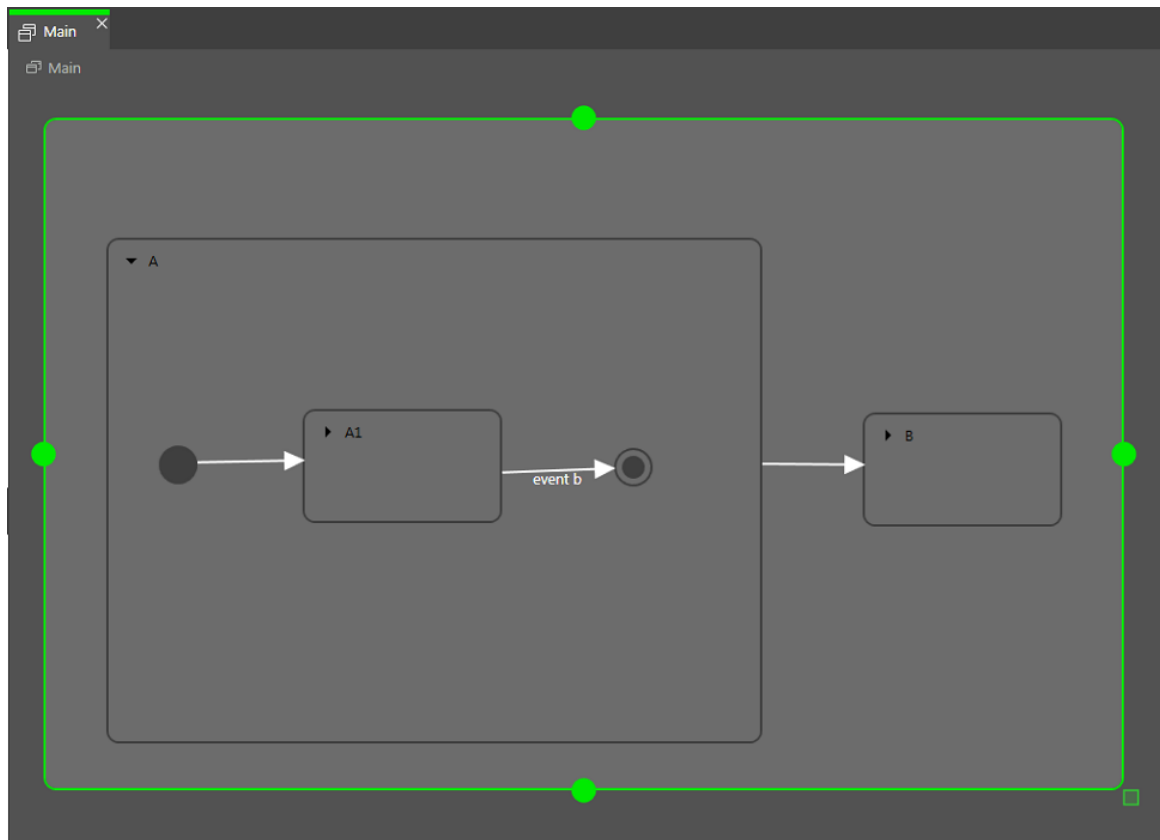


図6.16 混合状態内での最終状態の使用方法

6.16.2.5. 選択状態

選択状態は、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。選択状態は、遷移元の状態と遷移先の状態を接続するものです。1つの選択状態に複数の入出力遷移を設定できます。外へ向かう遷移(出力遷移)のすべてには条件が割り当てられており、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。

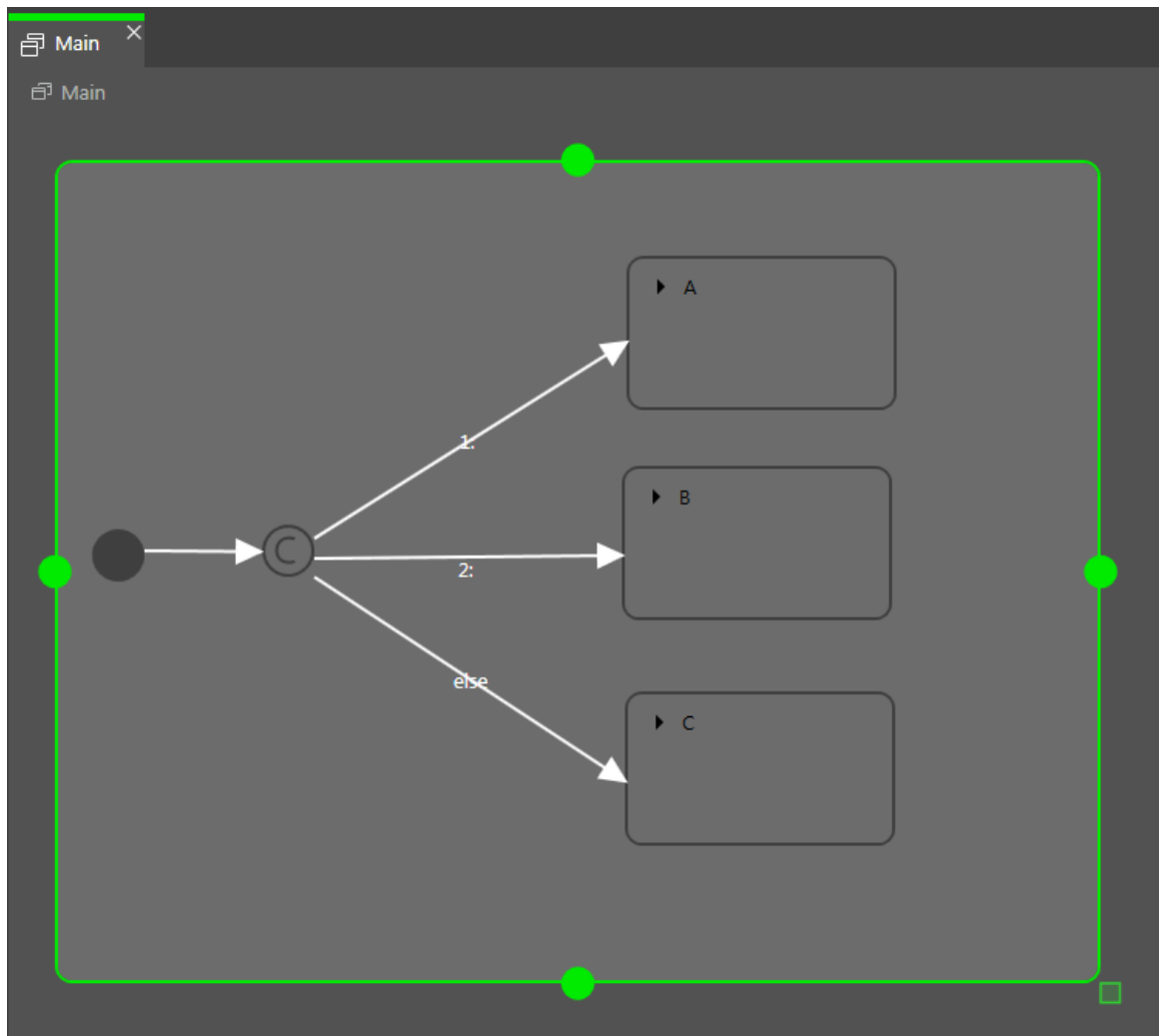


図6.17 入力遷移と出力遷移を持つ選択状態

6.16.2.6. 履歴状態

EB GUIDE には、2種類の履歴状態がサポートされています。

- ▶ 浅い履歴状態には、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。
- ▶ 深い履歴状態には、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。

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

浅い履歴状態は、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6.25 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。

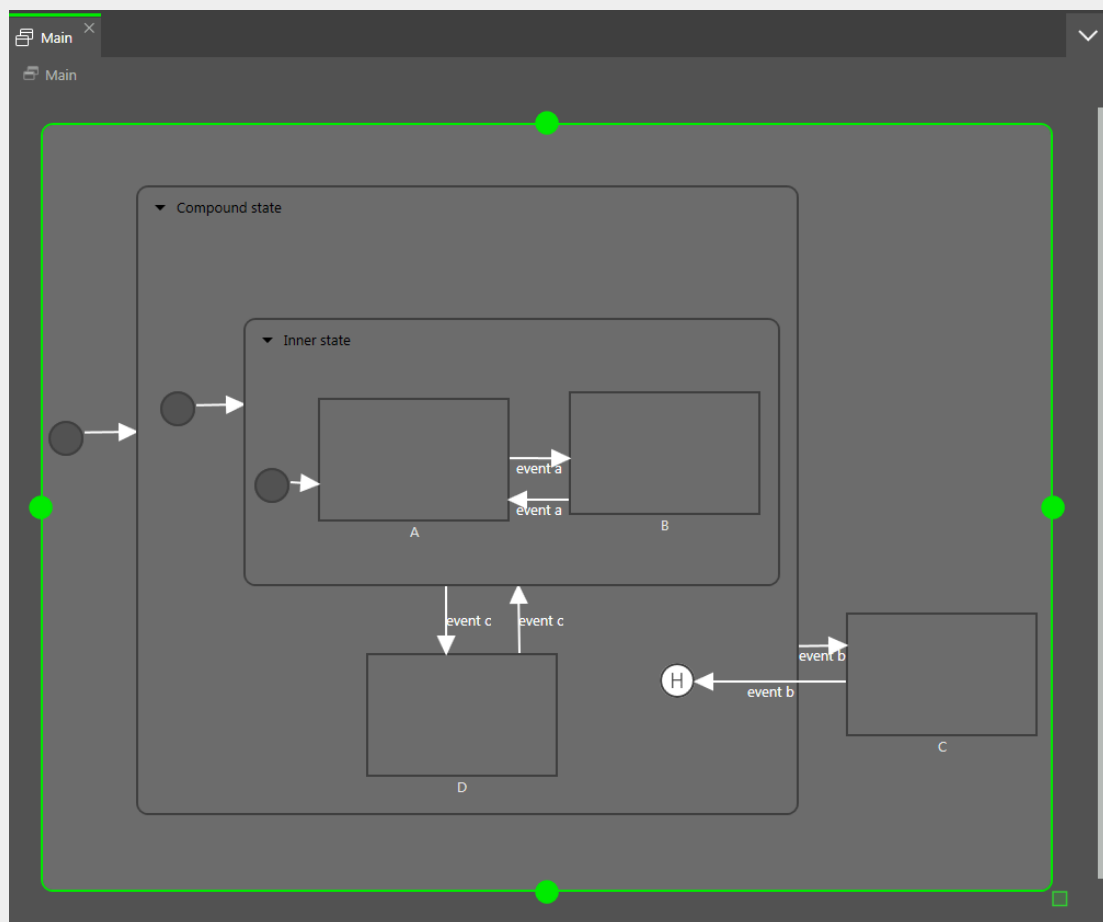


図6.18 浅い履歴ステート

- ▶ 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6.26 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。

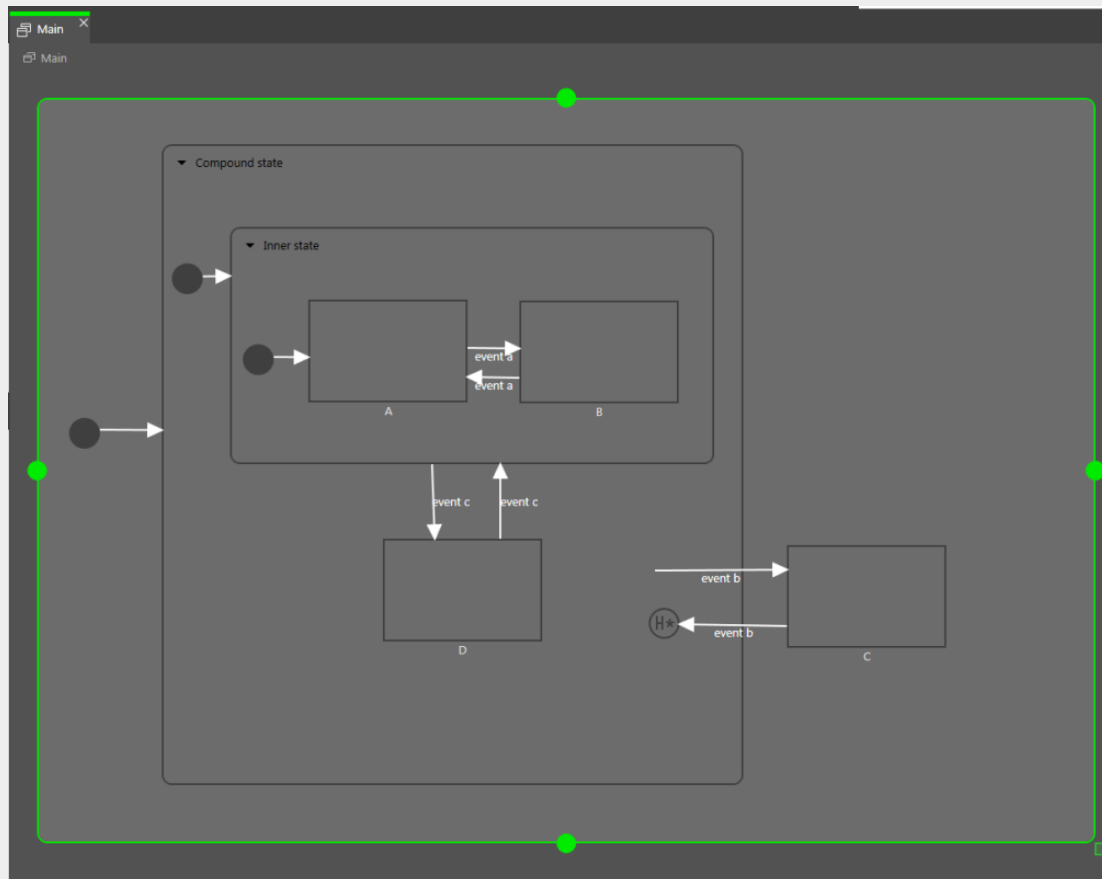


図6.19 深い履歴ステート

- ▶ 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。

6.16.3. 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

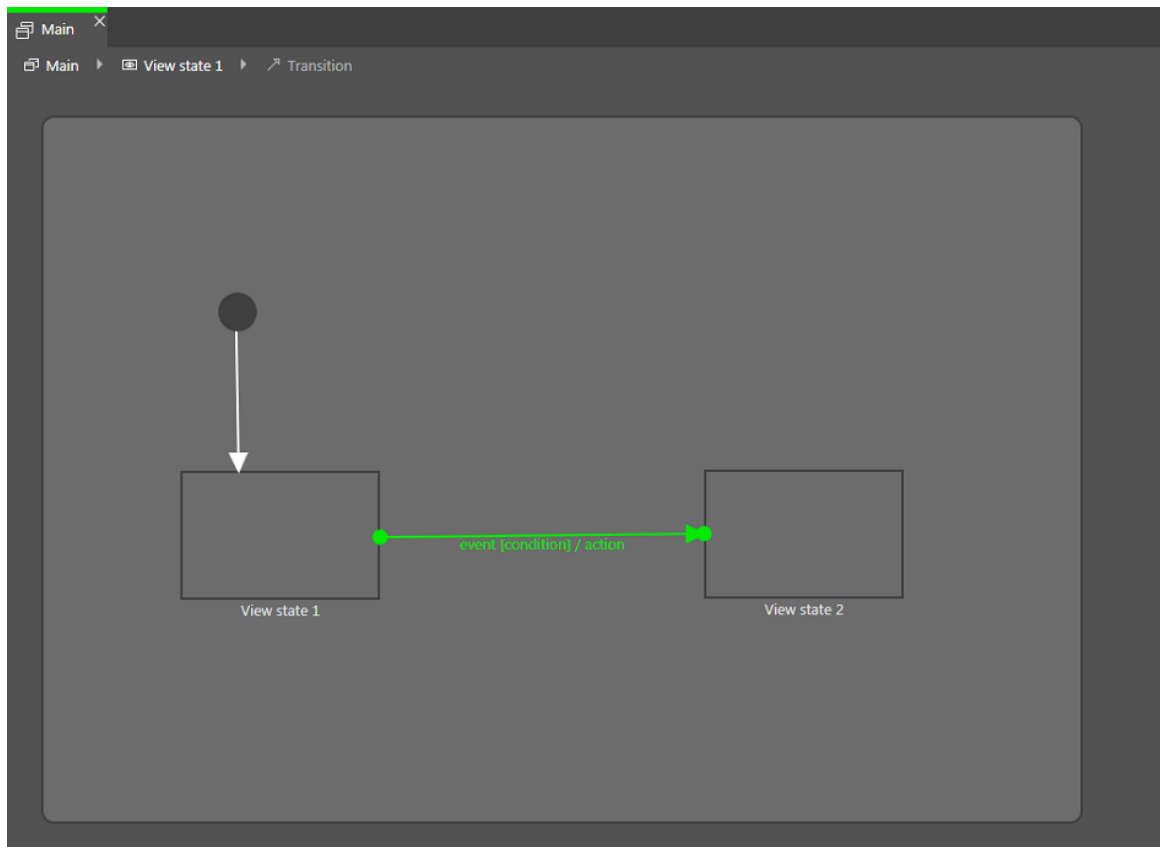


図6.20 遷移

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6.27

遷移の継承

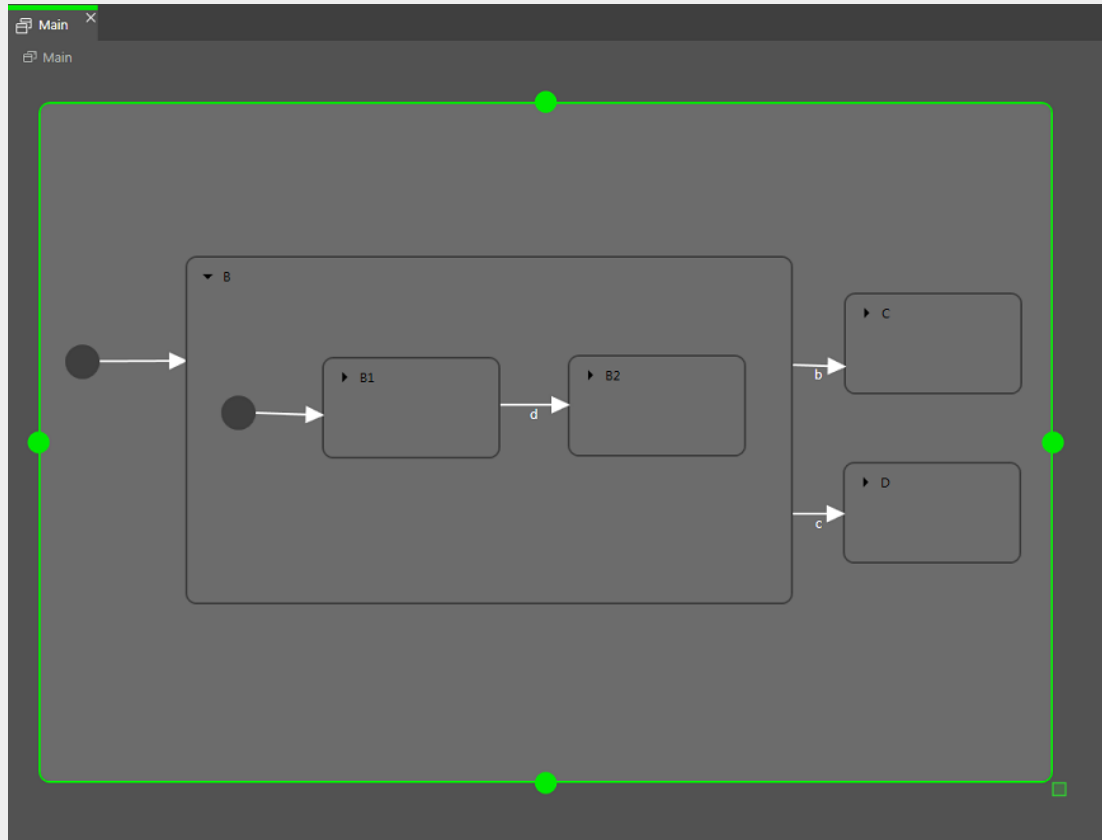


図6.21 遷移の継承

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6.28

遷移のオーバーライド

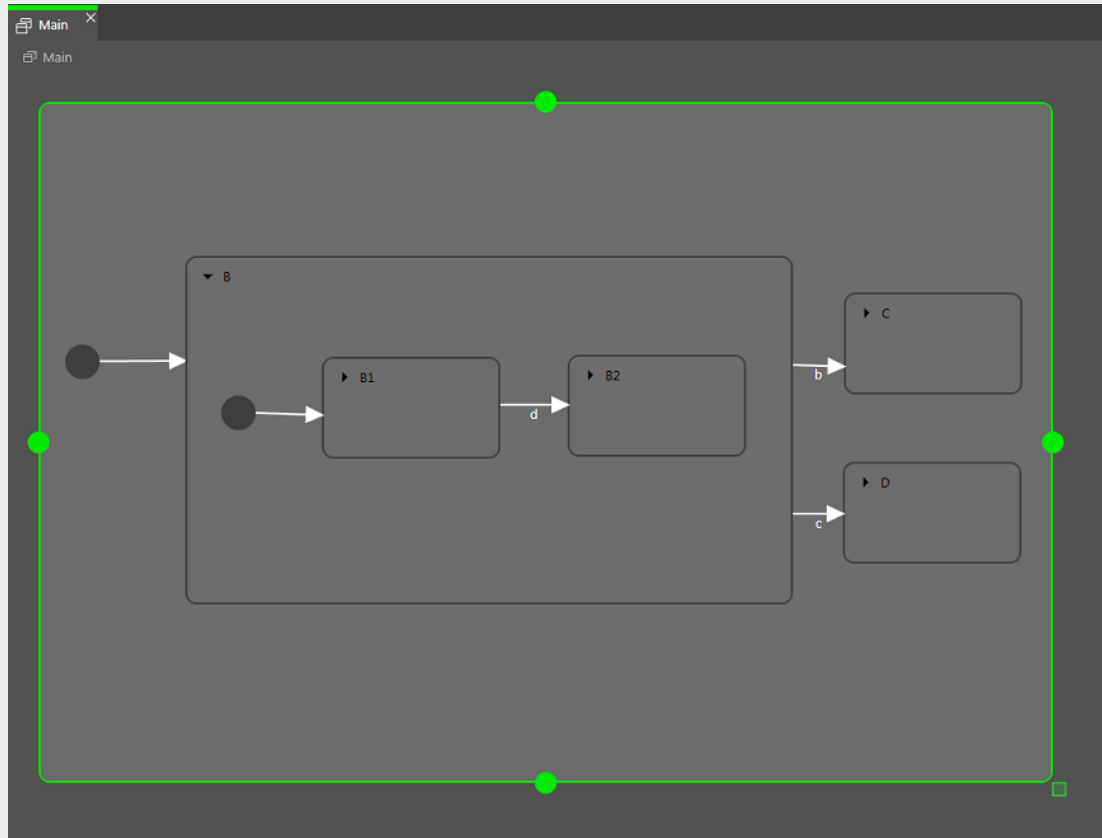


図6.22 遷移のオーバーライド

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。ソース状態は選択状態です。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。どの選択状態にも1つのelse遷移が必要であり、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、ターゲット状態を持たない遷移です。したがって、アクティブ状態を変更しません。内部遷移の目的は、現在の状態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。

状態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、ソース状態とターゲット状態が同じ状態である遷移です。内部遷移とは違い、自己遷移は状態を脱してから同じ状態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。

6.16.4. 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つのアクティブ状態が存在します。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、初期状態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. 現在の状態を出発点に、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、親状態に移動し、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、最上位の状態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。状態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. 現在の状態を出発点に、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、親状態に移動し、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状態から別の状態に変わります。ソース状態が終了し、ターゲット状態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、状態階層内の複数の混合状態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。

状態へエントリーするには、後続の遷移が必要です。例えば、混合状態へエントリーする場合、初期状態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. 最終状態に達した時点で、ステートマシンは停止します。

遷移が状態階層内で複数の状態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6.29

遷移の実行

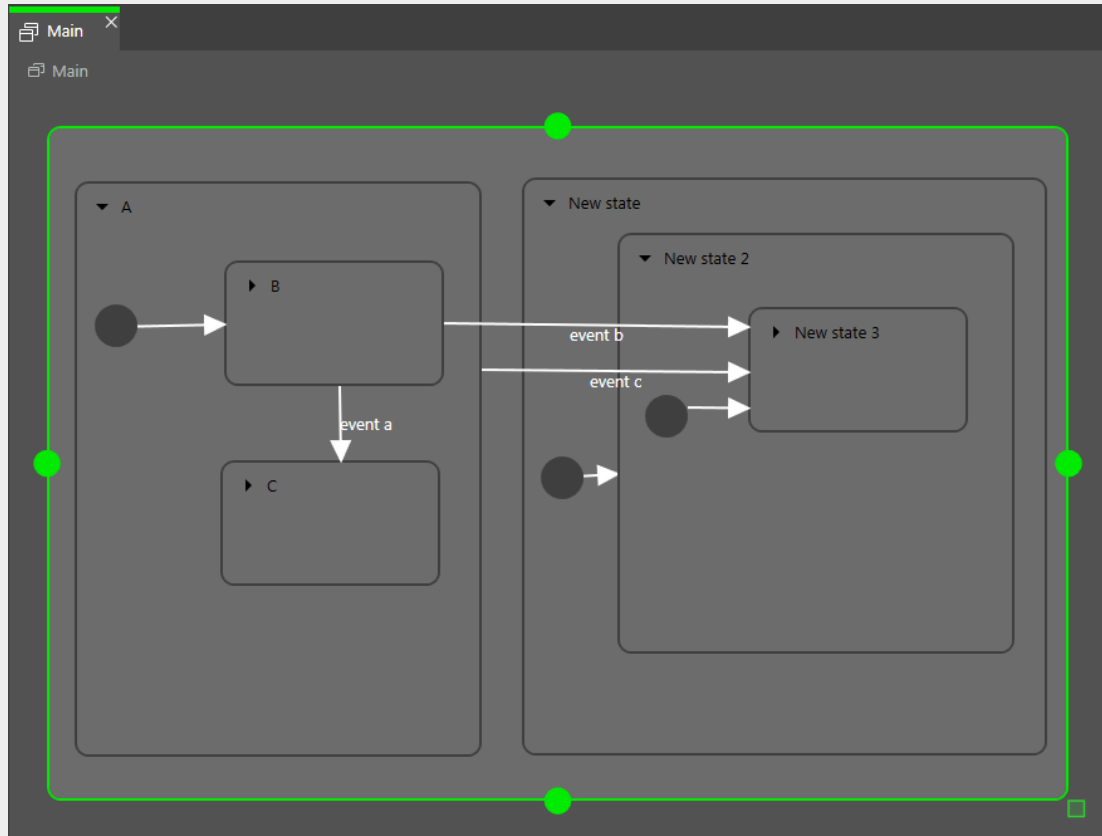


図6.23 遷移の実行

event aが発行されると、以下のことが起こります。

1. ステートBが終了します。
2. ステートCにエントリーします。

event bが発行されると、以下のことが起こります。

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にエントリーします。
4. ステートNew state 2にエントリーします。
5. ステートNew state 3にエントリーします。

event cが発行されると、以下のことが起こります。

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にエントリーします。
4. ステートNew state 2にエントリーします。
5. ステートNew state 3にエントリーします。



例6.30 遷移の実行

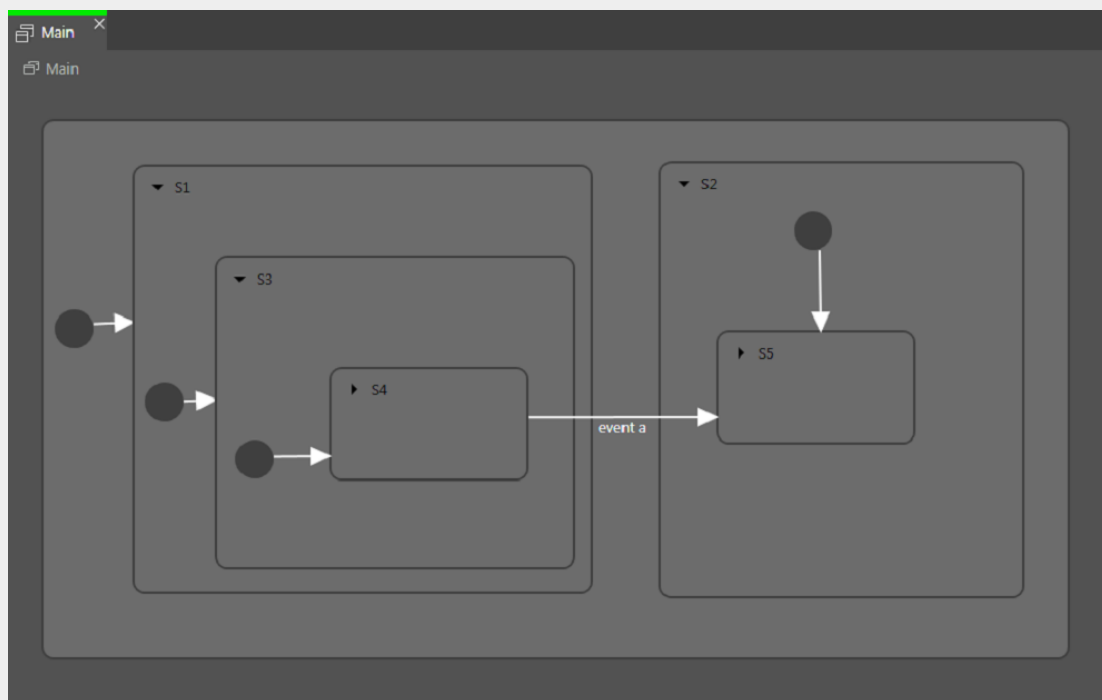


図6.24 遷移の実行

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にエントリーします。
5. ステートS5にエントリーします。



例6.31

遷移の実行

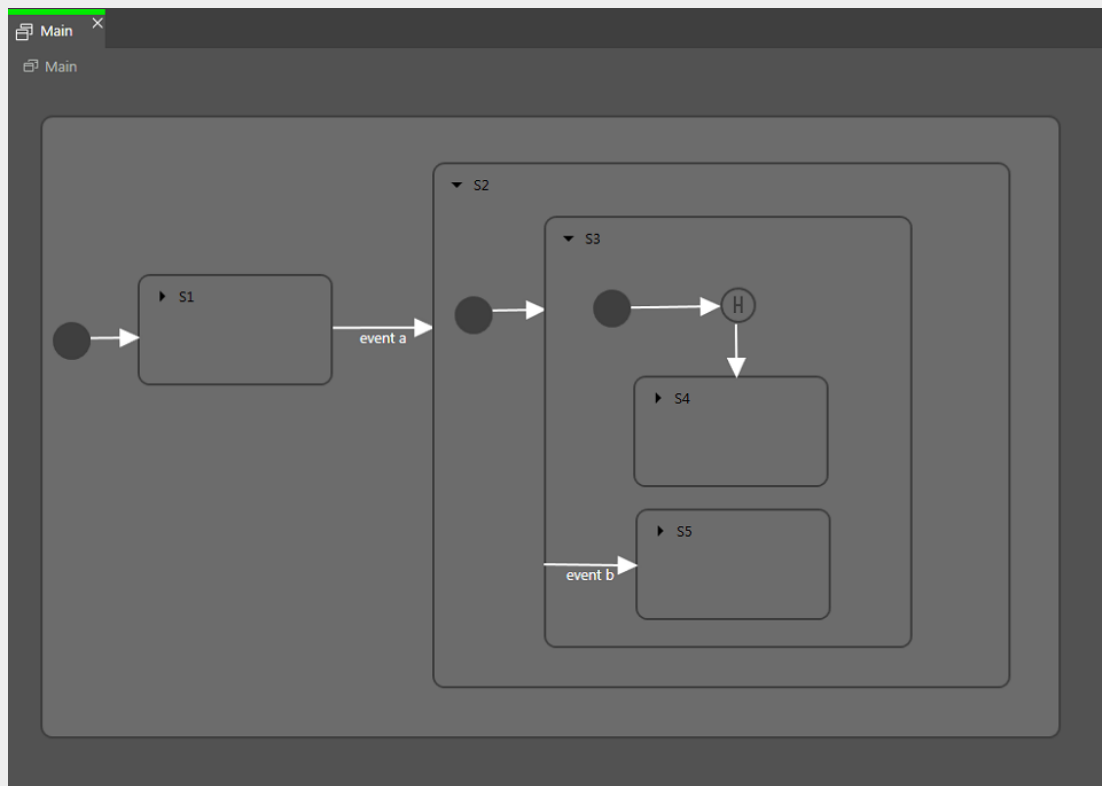


図6.25 遷移の実行

event aによってトリガーされる遷移により、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。

6.16.5. 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の記法と比較して紹介します。

6.16.5.1. 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。

での名前 EB GUIDE	UML 2.5 での名前
初期状態	初期(仮状態)
最終状態	最終状態
混合状態	状態
選択状態	選択(仮状態)
深い履歴状態	深い履歴(仮状態)
浅い履歴状態	浅い履歴(仮状態)
内部遷移	内部遷移
遷移	外部/ローカル遷移 ^a

^aEB GUIDE では、外部遷移とローカル遷移を区別しません。

6.16.5.2. 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

6.16.5.3. 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	を使用した回避策 EB GUIDE
並行状態	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に実装されています。

UML 2.5での概念	を使用した回避策 EB GUIDE
遷移時のタイムトリガー	この概念は、EB GUIDEスクリプト(<code>fire_delayed</code>) を使って、ステートマシン、データプールアイテム、遷移、またはビューに実装されています。

6.17. 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

6.17.1. 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。

6.17.2. 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。

6.17.3. 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。

6.17.4. 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あります。

▶ 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`を使用することで、これらの操作を正しく処理できます。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。

6.18. 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、ボタンは楕円、イメージ、ラベル、および四角形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。

6.18.1. 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。

注記



ビューのサイズを変更する

EB GUIDE Studioでは、ビューのサイズを拡大または縮小することで、一部の領域を詳細に把握したりより広い領域を確認したりできます。拡大または縮小を行うには、スライダーを使うか、ビューの一番下にあるテキストボックスをクリックします。デフォルトのズームレベルは100%になっています。また、Ctrl++キーで拡大、Ctrl+-キーで縮小、Ctrl+0キーでズームレベル100%へのリセットを行うこともできます。

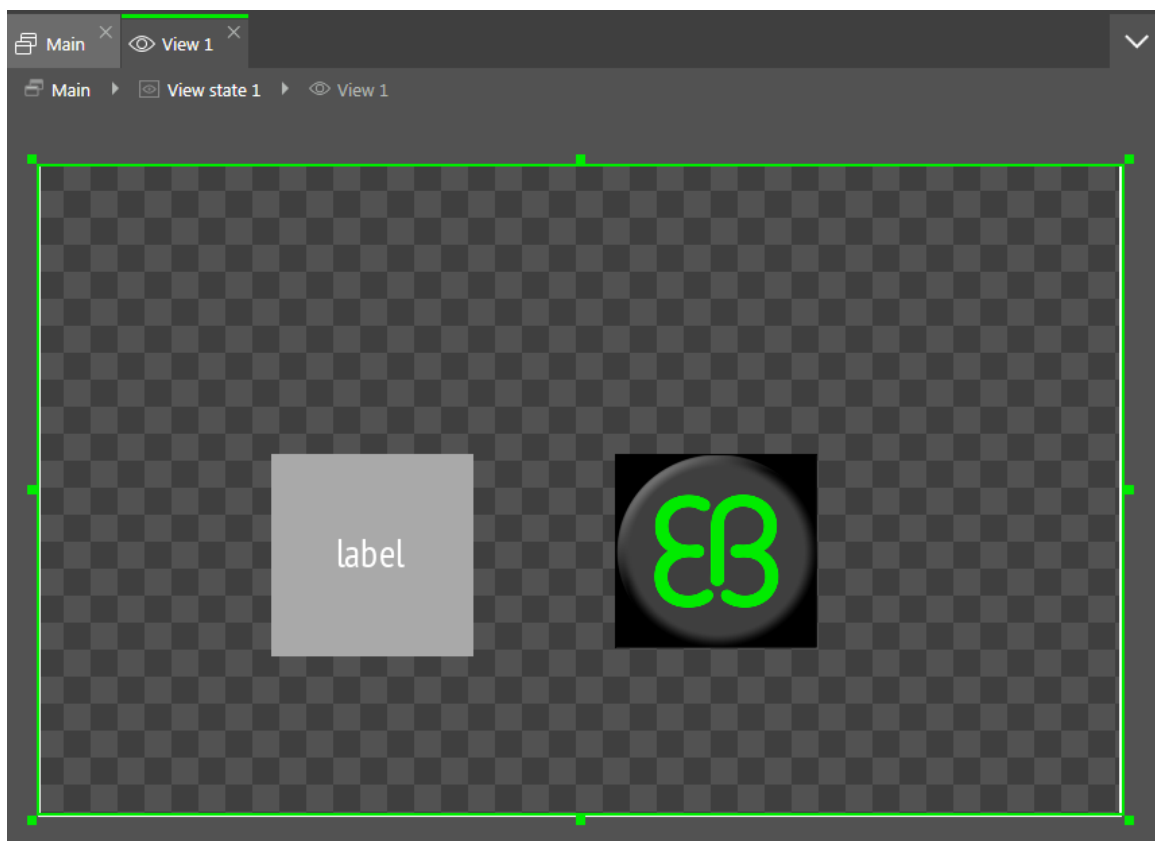


図6.26 四角形、ラベル、イメージを含むビュー

6.18.2. 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

基本ウィジェットとしては、アルファマスク、アニメーション、コンテナー、楕円、イメージ、インスタシエータ、ラベル、および四角形があります。

▶ 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、PBR Phong材質、PBR GGX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、および環境光があります。

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バージョン2.0以上またはDirectX 11のレンダラーが必要です。グラフィックドライバがレンダラーのバージョンと互換性があることを確認してください。

▶ ウィジェットテンプレート

この[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。

▶ カスタムウィジェット

[カスタムウィジェット]カテゴリにはカスタマイズされたウィジェットが含まれているため、カスタマイズされたウィジェットがプロジェクトに追加されたときにのみ表示されます。詳細については、弊社のWebサイト<https://www.elektrobit.com/ebguide/learn/resources/>をご覧ください。

手順については、[8.1「ウィジェットの操作」](#)をご覧ください。

6.18.3. 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]コンポーネントには、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。

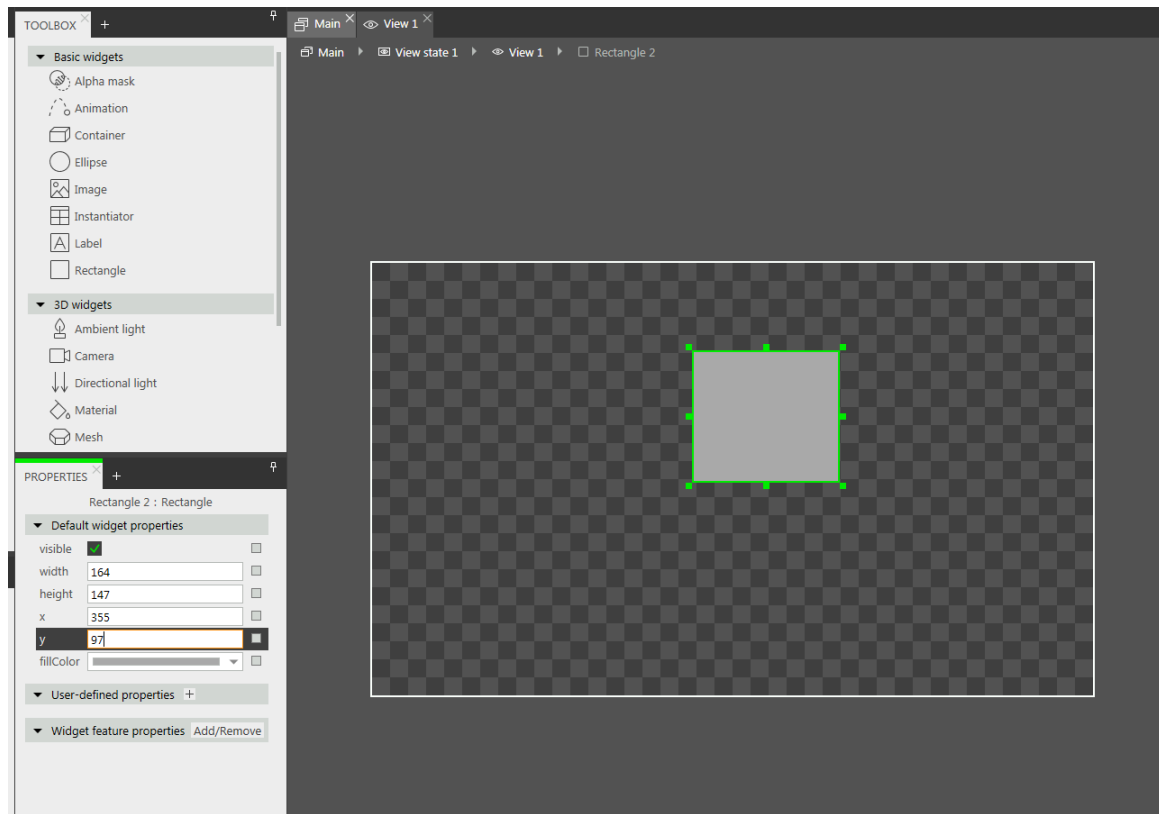


図6.27 四角形とそのプロパティ

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、[12.10「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際に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。



例6.32

[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。

6.18.4. ウィジェットテンプレート

ウィジェットテンプレートでは、EB GUIDEモデルで何度も使用できる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ースを備えています。テンプレートインターフェー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ー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6.33

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 はテンプレートインタフェースに追加されます。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の値はblueのままです。

手順については、[8.7「ウィジェットの再利用」](#)をご覧ください。

6.18.5. ウィジェット機能

ウィジェットおよびウィジェットテンプレートは、ウィジェット機能を使用してその機能を拡張できます。ウィジェット機能には、事前定義済みのウィジェットプロパティがあります。ウィジェット機能は、カテゴリにグループ化されています。

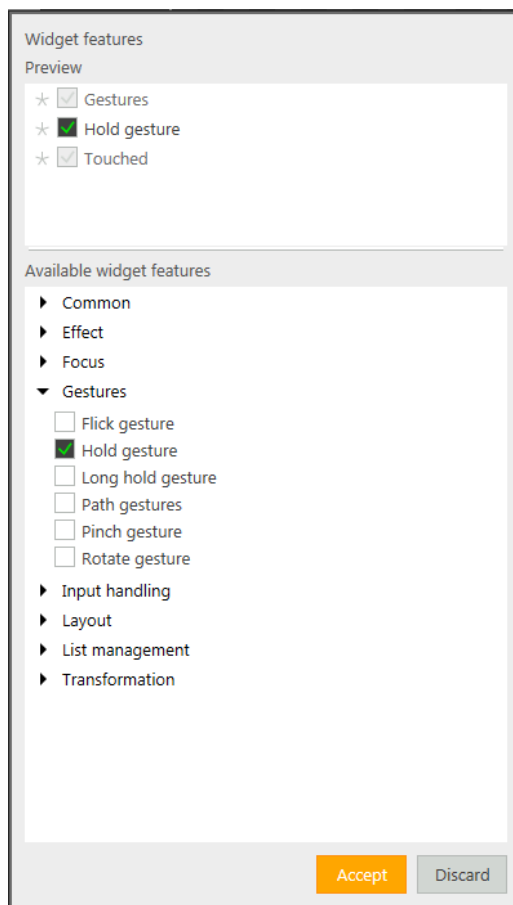


図6.28 ウィジェット機能

ウィジェット機能をウィジェットテンプレートに追加すると、作成されるウィジェットテンプレートのインスタンスに、追加されたウィジェット機能が継承されます。ウィジェットテンプレートのインスタンス、またはテンプレートから作成されたテンプレートには、ウィジェット機能を追加できません。

ウィジェット機能を使用する場合の制限は次のとおりです。

- ▶ ウィジェット機能には継承階層がありません。
- ▶ 1つのウィジェットにウィジェット機能を複数回追加することはできません。
- ▶ 一部のウィジェット機能には相互依存関係があります。これは、あるウィジェット機能を追加するには別のウィジェット機能を追加する必要があること、またはウィジェット機能同士が相互に排他的であることを意味します。
- ▶ ウィジェット機能は特定のウィジェットのタイプに制限されることがあります。
- ▶ ウィジェット機能は、ランタイム中にアクティブ化または非アクティブ化できません。

デフォルトでは、すべてのウィジェット機能は無効にされます。特定のウィジェット機能が必要な場合は、それをウィジェットに追加する必要があります。

手順については、[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](#)」をご覧ください。すべてのウィジェット機能の一覧については、[12.11「ウィジェット機能」](#)をご覧ください。

6.18.5.1. フォーカスウィジェット機能カテゴリ

EB GUIDE Studioでは、ウィジェットのフォーカス管理は[フォーカス]ウィジェット機能([自動フォーカス]と[ユーザー定義フォーカス])を使用してモデリングします。

次の2つのフォーカス方向を使用できます。

1. 順方向: 次のフォーカス可能なウィジェットがフォーカスされます。
2. 逆方向: 前のフォーカス可能なウィジェットがフォーカスされます。

[自動フォーカス]および[ユーザー定義フォーカス]ウィジェット機能は、順方向でフォーカス进行处理する方法の設定を提供します。逆方向の場合は、同じフォーカス順序が逆方向で使用されます。

[フォーカス]ウィジェット機能には次の特性があります。

[自動フォーカス]

このポリシーでは、フォーカスは1番上の行から始まり左から右という順序でフォーカス可能なウィジェットに適用されます。順序はウィジェットツリーの構造によって定義されます。

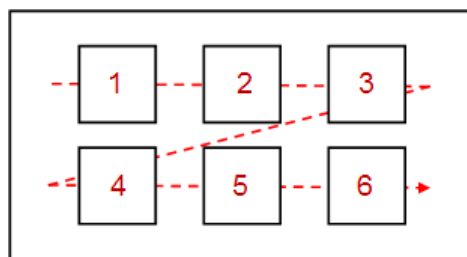


図6.29 [自動フォーカス]ウィジェット機能のポリシー

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。非表示のウィジェット、`focused`プロパティが無効にされているウィジェット、および[フォーカス]ウィジェット機能がないウィジェットは、有効なフォーカス可能なウィジェットとして認識されません。このため、それらは現在フォーカスされるウィジェットを決定するときにスキップされます。

[ユーザー定義フォーカス]

ビューが複雑なために、自動フォーカスポリシーに従ったフォーカス順序がきわめて難しいことがあります。この場合は、ユーザー定義フォーカスの順序を指定しておく役に立ちます。

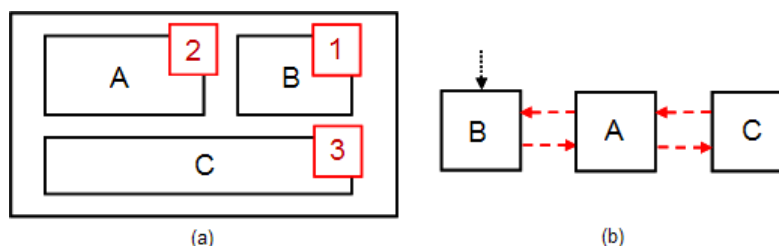


図6.30 [ユーザー定義フォーカス]ウィジェット機能のポリシー

図6.30「[ユーザー定義フォーカス]ウィジェット機能のポリシー」では、(a)はビューを示し、(b)はフォーカス順序を示しています。フォーカスの変更が処理される順序は、ウィジェットツリー構造と異なることがあります。

ウィジェット階層内のウィジェットがフォーカス可能とマークされる場合、それらはフォーカス階層の一部です。このフォーカス階層は、フォーカス可能なウィジェット、フォーカスポリシー、およびその階層内でフォーカスを処理する方法を定義する[自動フォーカス]ウィジェット機能または[ユーザー定義フォーカス]ウィジェット機能で構成されています。フォーカス階層は入れ子にすることができます。

6.18.5.2. リスト管理ウィジェット機能カテゴリ

[ラインインデックス]および[テンプレートインデックス]ウィジェット機能を使用すると、データ(イメージ、曲のタイトルなど)を対応する動的に作成されたインスタシエータのラインテンプレートに接続できます。

[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、インスタシエータウィジェットのラインテンプレートをカスタマイズするために使用されます。[ラインインデックス]ウィジェット機能は、リストまたは表の各ラインの一意の位置を定義します。



例6.34

[ラインインデックス]ウィジェット機能

リストをモデリングする場合は、リストプロパティのエントリーを反映した特定の値がリストの各エントリーにあることが予想されます。リスト内の特定のエントリーにアクセスするには、ラインテンプレートのインスタスとそのエントリーがどのインスタシエータの子であるかを認識している必要があります。[ラインインデックス]ウィジェット機能は、`lineIndex`プロパティを追加します。インスタシエータがラインテンプレートのインスタスを作成するときに、`lineIndex`に値が設定されます。インデックスは最初のインスタスをゼロとして開始されます。インスタシエータに2つの要素がある場合、2番目の要素は`lineIndex`値に1を受け取ります。

手順については、[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。

[テンプレートインデックス]

[テンプレートインデックス]ウィジェット機能では、複雑なデータ抽象化が可能です。非常に複雑なリストまたは表の場合は、1つのエントリーまたは一連のエントリーを視覚化するために、複数のデータリストが必要となります。

例えば、イメージとテキストが混在したコンテンツを持つ表では、イメージのリストと文字列のリストが必要となります。そのような複雑な場合をカバーするために、[テンプレートインデックス]ウィジェット機能は`lineTemplateIndex`プロパティを提供しています。

**例6.35****[テンプレートインデックス]ウィジェット機能**

`lineMapping`プロパティに`0|1`を設定し、`numItems`プロパティに`5`を設定してインスタシエータを使用してリストをモデリングする場合、`lineTemplateIndex`は`0|0|1|1|2`となります。

7. ヒューマンマシンインターフェースの動作のモデル化

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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

ステップ 1

[ナビゲーション]コンポーネントで[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選択します。

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:

- 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]コンポーネントで[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選択します。

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]コンポーネントで、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]コンポーネントで、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]コンポーネントで[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、[6.13「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]コンポーネントで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、[6.13「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

ステップ 1

[ナビゲーション]コンポーネントで、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、最終ステート、および履歴ステートの一意性

初期ステート、最終ステート、および履歴ステートは、混合ステート¹につき一度だけ挿入できます。

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

[ナビゲーション]コンポーネントで混合ステートをダブルクリックします。

混合ステートがコンテンツエリアに展開されます。

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。

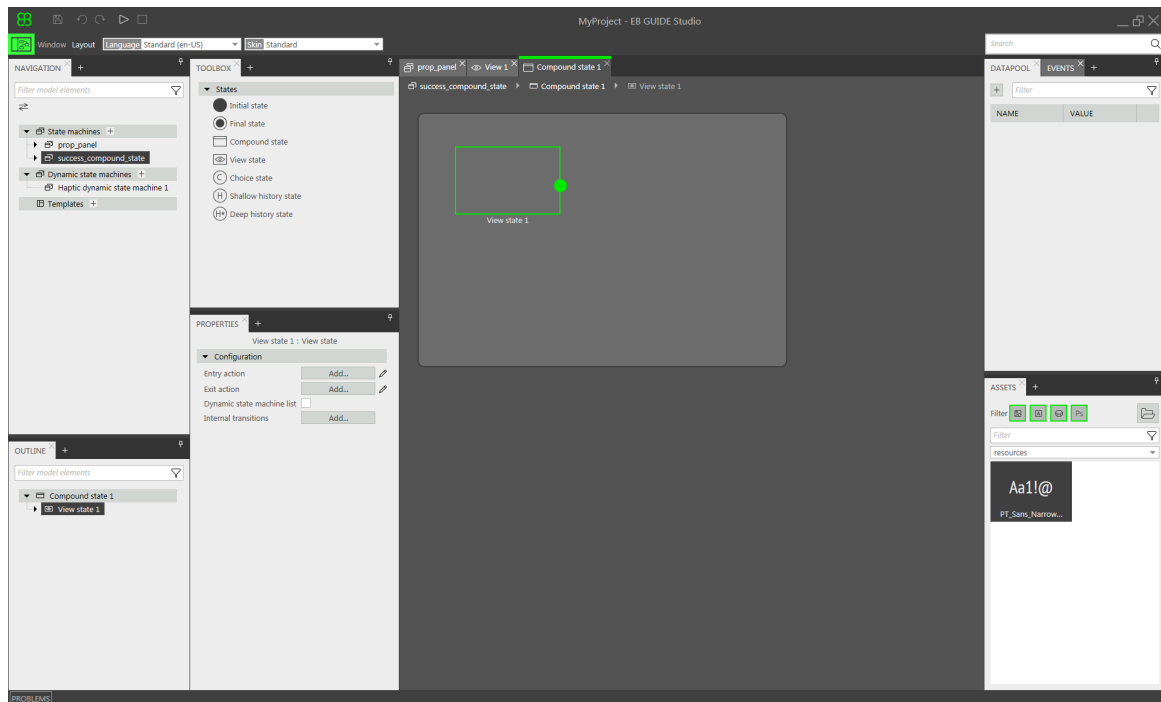


図7.1 入れ子のビューステートを持つ混合ステート

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]コンポーネントで、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。

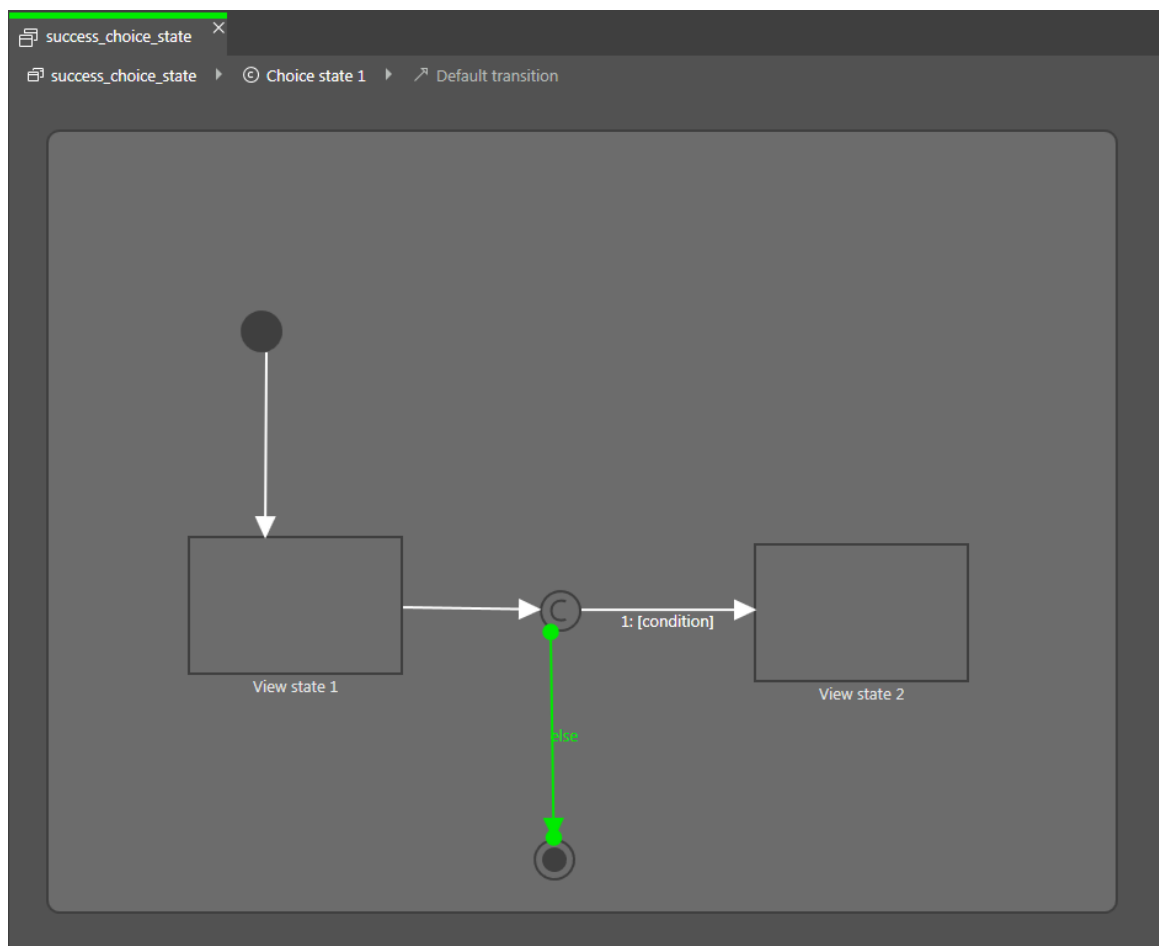


図7.2 選択遷移を持つ選択ステート

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]コンポーネントで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、[6.13「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]コンポーネントで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、[6.13「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]コンポーネントで、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。

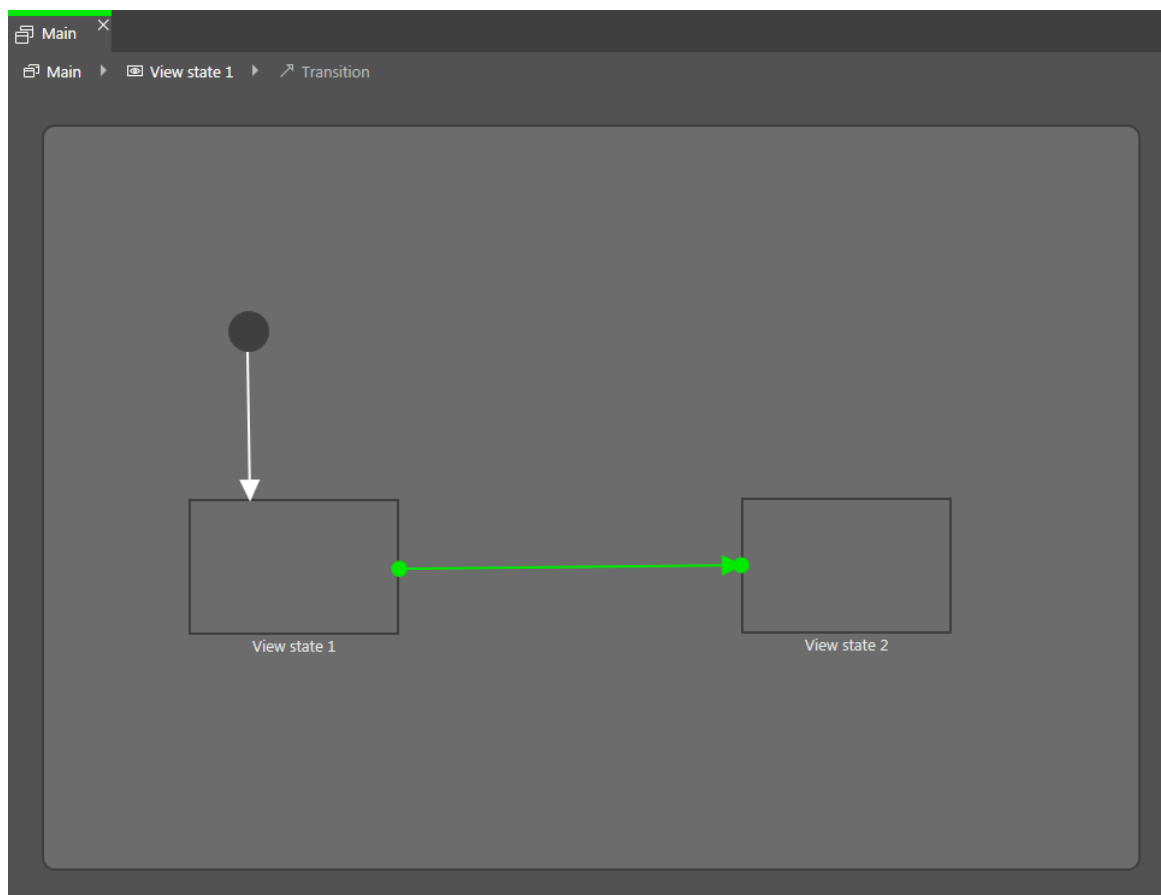


図7.3 遷移

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]コンポーネントで[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。

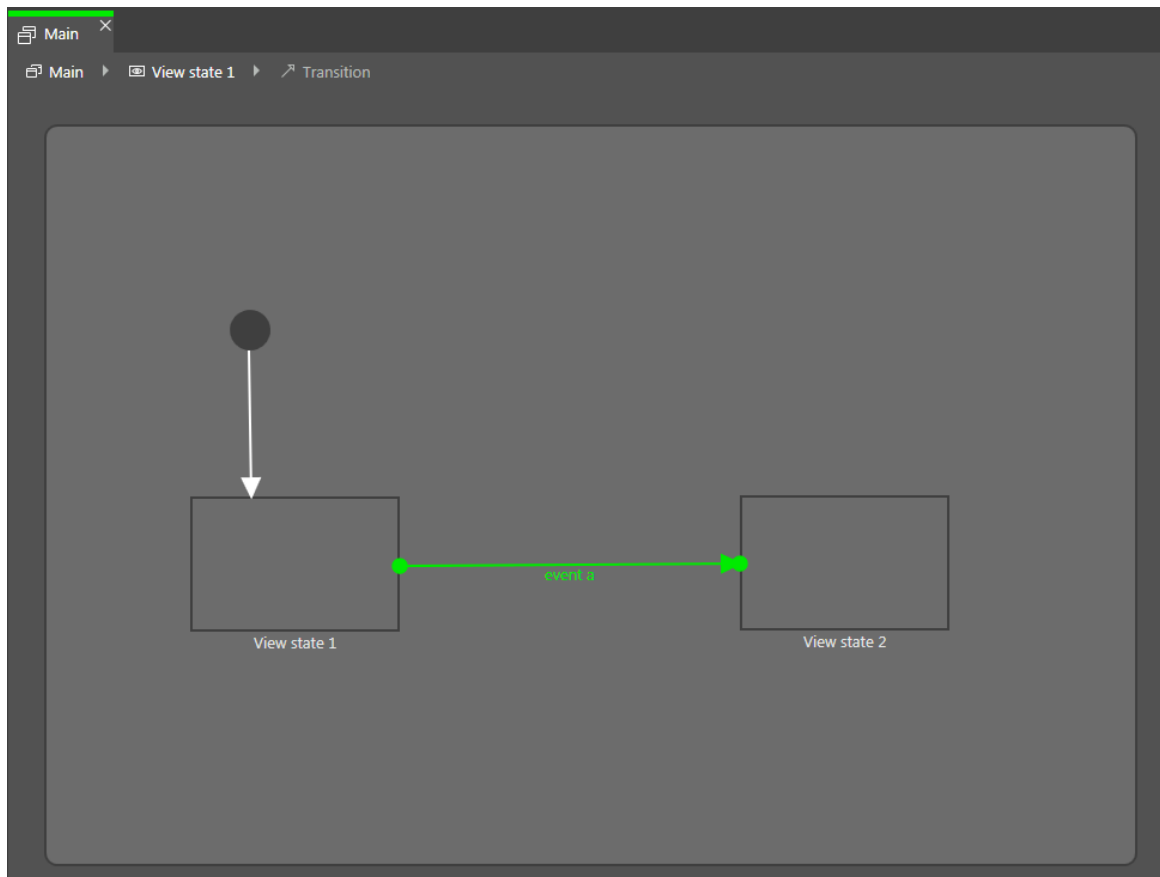


図7.4 トリガー付きの遷移

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]コンポーネントに移動します。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、[6.13「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。

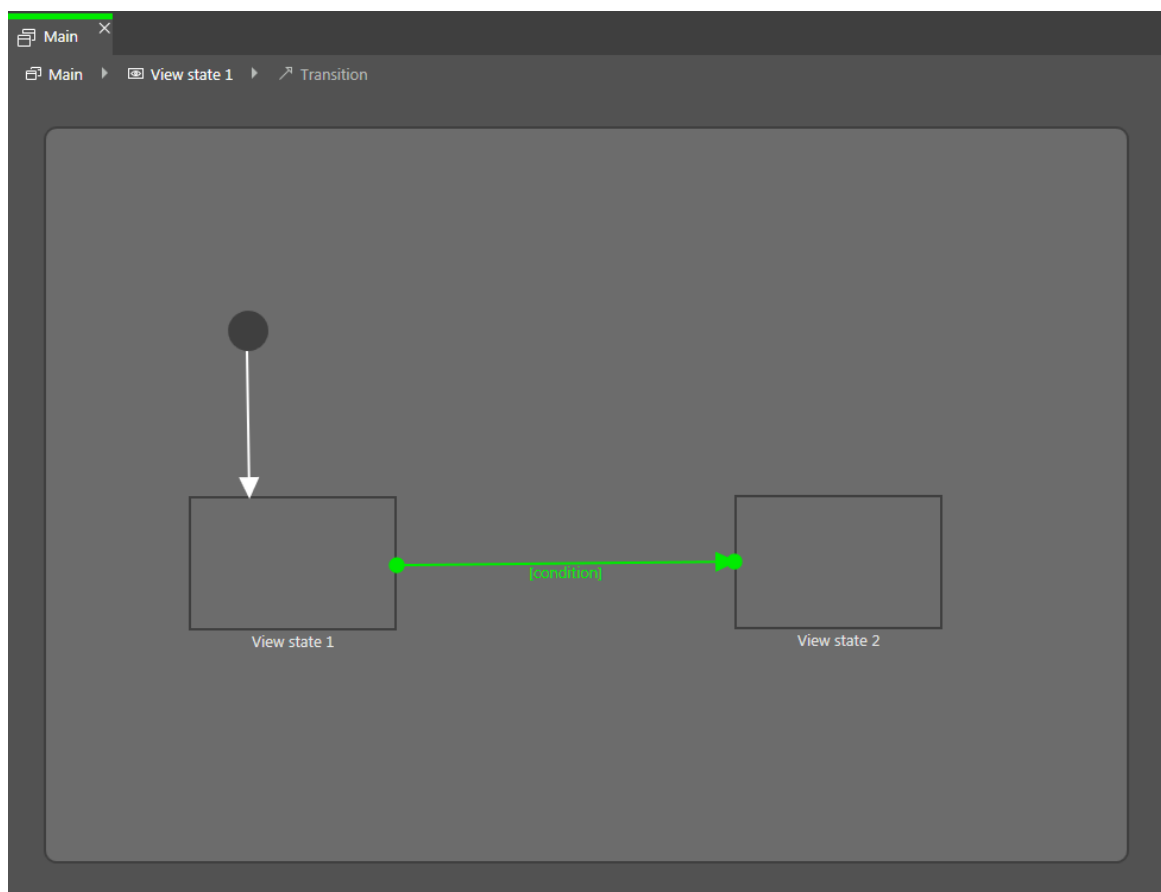


図7.5 条件付きの遷移

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]コンポーネントに移動します。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、[6.13「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。

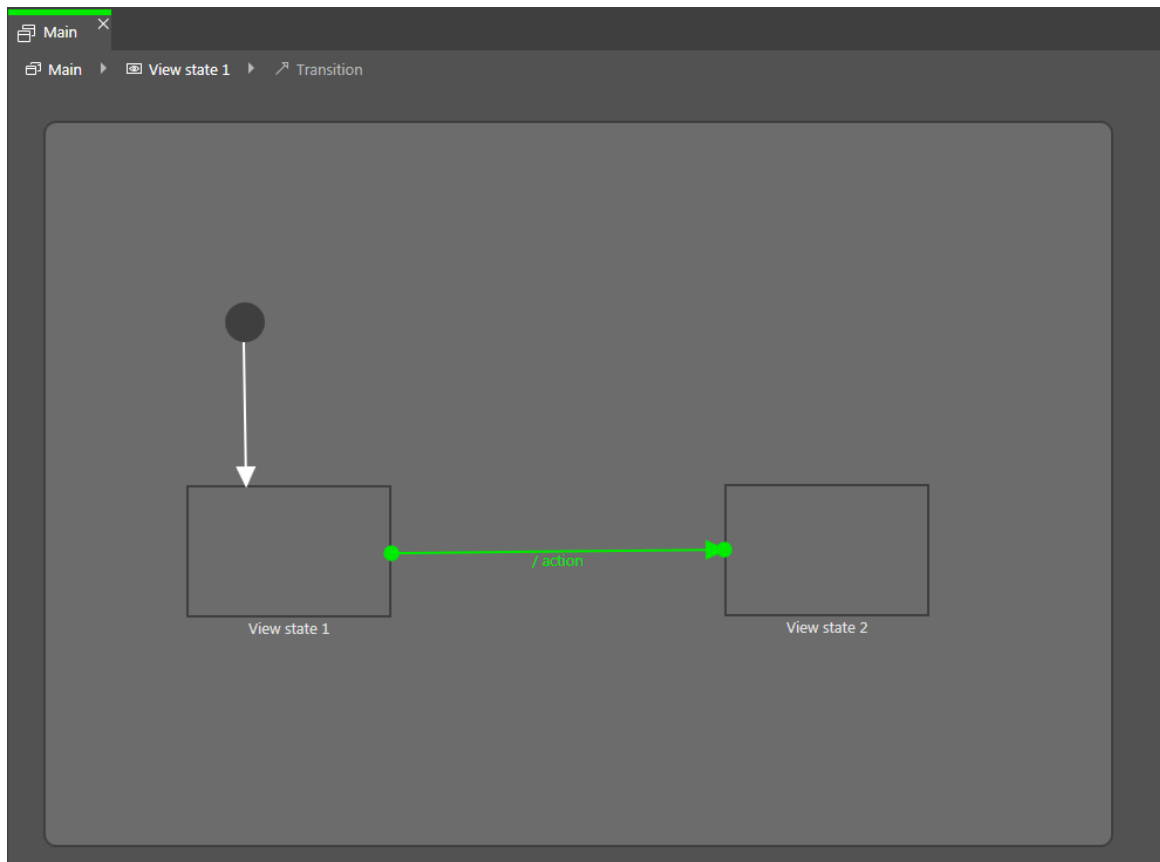


図7.6 アクション付きの遷移

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]コンポーネントで[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]コンポーネントに表示されます。

8. ヒューマンマシンインターフェースの外観のモデル化

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

8.1. ウィジェットの操作

ティップ



ビューおよびウィジェットのコピーと検索

既存のビューまたは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キーを使用できます。

EB GUIDEモデル内で特定のビューまたはウィジェットを検索するには、ビューまた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。ビューまたはウィジェットにジャンプするには、ヒットリスト内のビューまたはウィジェットをダブルクリックします。

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]コンポーネント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。

8.1.2. ビューへの基本ウィジェットの追加

基本ウィジェットの詳細については、[12.10.2「基本ウィジェット」](#)をご覧ください。

8.1.2.1. 四角形を追加する



四角形を追加する

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]から四角形をドラッグし、ビューにドロップします。

四角形がビューに追加されます。

8.1.2.2. 楕円を追加する



楕円を追加する

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]から楕円をドラッグし、ビューにドロップします。

ウィジェットがビューに追加されます。

8.1.2.2.1. 楕円を編集する

楕円の扇型のみを描画したり、楕円の弧を変更したりできます。



扇型を作成する

前提条件:

- ビューに楕円が含まれていること。

ステップ 1

楕円をクリックし、[プロパティ]コンポーネントに移動します。

ステップ 2

扇型の角度をcentralAngleテキストボックスに入力します。

ステップ 3

扇型の方向をsectorRotationテキストボックスに入力します。

扇型を作成しました。



円弧を作成する

前提条件:

- ビューに楕円が含まれていること。

ステップ 1

楕円をクリックし、[プロパティ]コンポーネントに移動します。

ステップ 2

弧の幅をarcWidthテキストボックスに入力します。

円弧を作成しました。

8.1.2.3. イメージを追加する



[ツールボックス]を使用してイメージを追加する

前提条件:

- イメージファイルは、\$GUIDE_PROJECT_PATH/<project name>/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、[6.12.2「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]コンポーネントで、imageコンボボックスからイメージを選択します。または、[アセット]コンポーネントから別のイメージをドラッグして、imageドロップダウンリストボックスにドロップします。

ビューにイメージが表示されます。



[アセット]コンポーネントを使用してイメージを追加する

前提条件:

- イメージファイルは、\$GUIDE_PROJECT_PATH/<project name>/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、[6.12.2「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

[アセット]コンポーネントからイメージファイルをドラッグしてビューにドロップします。

ビューにイメージが表示されます。

ステップ 2

イメージファイルを変更するには、[プロパティ]コンポーネントに移動して、imageコンボボックスからイメージを選択します。または、[アセット]コンポーネントから別のイメージをドラッグして、imageコンボボックスにドロップします。

ビューにイメージが表示されます。



9-patchイメージを追加する

前提条件:

- 9-patchイメージファイルは、\$GUIDE_PROJECT_PATH/<project name>/resourcesディレクトリに配置します。9-patchイメージのバックグラウンド情報については、[6.12.2.1「9-patch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。
- イメージがEB GUIDEモデルに追加されます。

ステップ 1

イメージを選択し、[プロパティ]コンポーネントに移動します。

ステップ 2

imageコンボボックスで9-patchイメージを選択します。

ステップ 3

[ウィジェット機能プロパティ]に移動して、[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

[使用可能なウィジェット機能]で、[レイアウト]カテゴリを展開して[拡大縮小モード]を選択します。

ステップ 5

[承認]をクリックします。

関連するウィジェットプロパティがイメージに追加され、[プロパティ]コンポーネントに表示されます。

ステップ 6

[プロパティ]コンポーネントで、scaleModeプロパティとしてfit to Size (=1)を選択します。

注記



9-patchイメージを追加する

[拡大縮小モード]ウィジェット機能を追加していない場合、またはscaleModeプロパティにoriginal Size (=0)またはkeep aspect ratio (=2)を選択した場合、9-patchイメージは通常の.pngイメージのように拡大縮小されます。

8.1.2.4. ラベルを追加する



[ツールボックス]を使用してラベルを追加する

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]からラベルをドラッグし、ビューにドロップします。

ビューにラベルが追加されます。ラベルのフォントは、デフォルトフォントのPT_Sans_Narrow.ttfです。



[アセット]コンポーネントを使用してラベルを追加する

前提条件:

- フォントファイルは、\$GUIDE_PROJECT_PATH/<project name>/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、[6.12.1「フォント」](#)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

[アセット]コンポーネントからフォントファイルをドラッグしてビューにドロップします。

選択したフォントでビューにラベルが表示されます。

8.1.2.4.1. 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:

- フォントファイルは、`$GUIDE_PROJECT_PATH/<project name>/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、[6.12.1「フォント」](#)をご覧ください。
- 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]コンポーネントで、`font`コンボボックスからフォントを選択します。

または、[アセット]コンポーネントからフォントファイルをドラッグして、`font`コンボボックスにドロップします。

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。`.fnt`ビットマップフォントを選択した場合、フォントのサイズは固定されており、ラベルの`font`プロパティでフォントサイズを変更することはできません。

注記



テキストの高さと行間の計算

次の図は、EB GUIDE Studioでテキストの高さ、行の高さ、および行間が計算される方法を示しています。ラベルのフォントスタイル、サイズ、または行間を変更するときは、これを考慮してください。

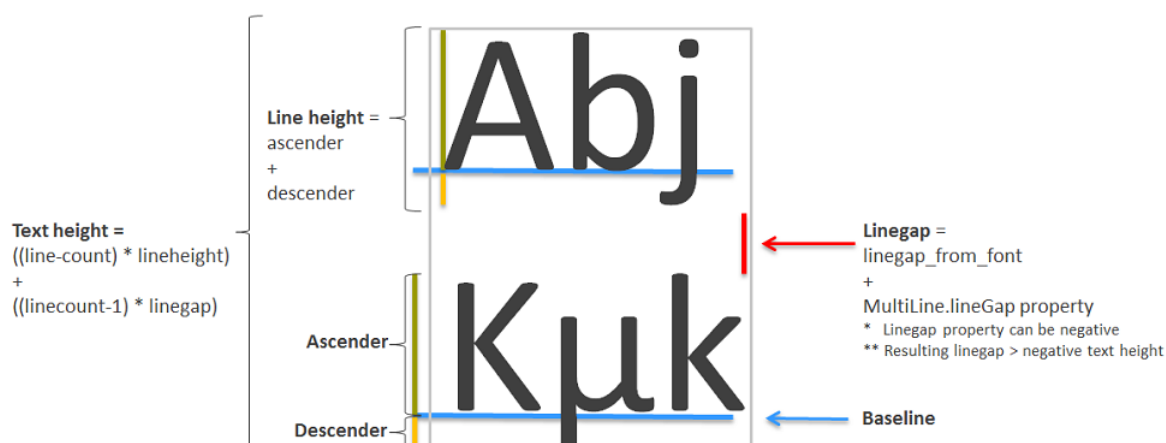


図8.1 テキストの高さ、行の高さ、および行間の計算

8.1.2.5. コンテナを追加する



コンテナを追加する

コンテナ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]からコンテナをドラッグし、ビューにドロップします。

ステップ 2

コンテンツエリアで、コンテナの四隅のいずれかをドラッグしてコンテナを拡大します。

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、コンテナにドロップします。

ウィジェットがコンテナ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。

8.1.2.6. インスタンスエータの追加



インスタンスエータの追加

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]からインスタンスエータをドラッグし、ビューにドロップします。

ステップ 2

[ツールボックス]からウィジェットをドラッグし、インスタンスエータにドロップします。

ウィジェットはラインテンプレートとして機能します。

ステップ 3

インスタンスエータを選択し、[プロパティ]コンポーネントに移動します。

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、インスタンス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。

子ウィジェットが、numItemsプロパティで指定された回数だけ、ウィジェット機能でインスタンス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。

ステップ 4


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

2番目のラインテンプレートとして機能する2番目の子ウィジェットを追加しました。

ステップ 5

インスタンスエータを選択し、[プロパティ]コンポーネントに移動します。

ステップ 5.1

lineMappingの横にあるをクリックします。

ステップ 5.2

[追加]ボタンをクリックします。

新しいエントリーが表に追加されます。

ステップ 5.3

Valueテキストボックスに、0と入力します。

ステップ 5.4

[追加]ボタンをクリックします。

新しいエントリーが表に追加されます。

ステップ 5.5

Valueテキストボックスに、1と入力します。

ラインテンプレートがインスタンス化される順序を定義しました。



例8.1

インスタンス化の順序

lineMappingプロパティはインスタンス化の順序を定義します。例えば、値1|0を入力した場合、インスタシエータはラインテンプレート1を最初の子ウィジェットとしてインスタンス化し、ラインテンプレート0を2番目の子ウィジェットとしてインスタンス化します。

lineMappingプロパティは、反復して適用されます。これは、numItemsプロパティに10と入力した場合、結果は1|0|1|0|1|0|1|0|1|0という順序になることを意味します。

インスタシエータの使い方を示す詳細なサンプルについては、[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。

注記



ラインテンプレートのプロパティをリンクする
リンクの規則を次に示します。

- ▶ ラインテンプレート同士のプロパティをリンクすることはできません。
- ▶ インスタシエータの外部からそのラインテンプレートにリンクすることはできません。
- ▶ ラインテンプレートから対応するインスタシエータにリンクできます。

8.1.2.7. アニメーションの追加



アニメーションの追加

曲線の詳細や曲線プロパティの記述については、[12.10.2.2「アニメーション」](#)をご覧ください。

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 2

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

ステップ 3

[アニメーション]エディターに移動し、[アニメーションプロパティ]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 4

[アニメーションプロパティ]の下でアニメーション化するプロパティを選択し、[アニメーション曲線]の下で該当する曲線を選択します。

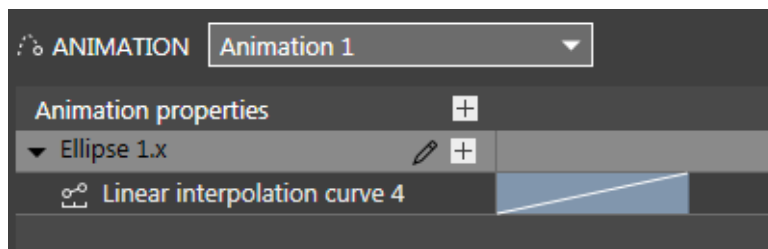


図8.2 サンプル曲線が表示された[アニメーション]エディター

ステップ 5

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 6

プロパティの名前の横にある[プロパティ]コンポーネントで、[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 7

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 8

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は変更できます。

アニメーションの具体的なサンプルについては、[11.5「チュートリアル: 画面内で楕円を移動する」](#)をご覧ください。

8.1.2.8. アルファマスクの追加



アルファマスクの追加

アルファマスクの詳細については、[12.10.2.1「アルファマスク」](#)をご覧ください。

前提条件:

- `$GUIDE_PROJECT_PATH/<project name>/resources`ディレクトリにイメージが含まれていること。
- コンテンツエリアにビューが表示されていること。

ステップ 1

[ツールボックス]からアルファマスクをドラッグし、ビューにドロップします。

ステップ 2

[プロパティ]コンポーネントに移動し、`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。

注記



サポートされているアルファマスク用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バージョン2.0以上では、`.png`ファイルおよび `.jpg`ファイルがサポートされます。RGBイメージは、グレースケールイメージに変換されてから アルファマスクとして使用されます。グレースケールイメージはそのまま使用されます。イメージのアルファチャネルは 無視されます。

アルファマスクを9-patchイメージと一緒に使用することはできません。

ステップ 3

[ツールボックス]にある基本ウィジェットのいずれかを子ウィジェットとしてアルファマスクに追加します。

アルファチャネル(子ウィジェットのオパシティ)は、アルファマスクを使って制御されます。

8.1.3. ビューへの3Dウィジェットの追加

8.1.3.1. 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。ファイルに、カメラ、光源、およびメッシュと少なくとも1つの材質を含む1つのオブジェクトが含まれていること。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]コンポーネントで[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]コンポーネントに表示されます。インポートされる3Dシーンにアニメーションがある場合は、リニアキー値補間整数曲線またはリニアキー値補間浮動小数点数曲線が追加されます。EB GUIDE Studioでそれらの曲線の基礎となるキー値ペアを変更することはできません。

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。

8.1.4. ビューに.psdファイルを追加する



ビューに.psdファイルを追加する

前提条件:

- .psdファイルが\$GUIDE_PROJECT_PATH/<project name>/resourcesにあること。バックグラウンド情報については、[6.12.4「.psdファイル形式」](#)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

[アセット]コンポーネントで、resourceフォルダーを選択します。

ステップ 2

プレビューエリアで、.psdファイルをドラッグして、コンテンツエリアにドロップします。

インポートのステータスメッセージが表示されます。

ステップ 3

[OK]をクリックします。

インポートが成功した場合は、.psdファイルから作成されたウィジェットツリーが[ナビゲーション]コンポーネントに表示されます。ウィジェットツリーはコンテナとイメージで構成されており、.psdファイルの構造が反映されます。抽出されたすべてのイメージが含まれるサブディレクトリが\$GUIDE_PROJECT_PATH/<project name>/resourcesディレクトリに作成されます。



.psdファイルからイメージを抽出する

.psdファイルからイメージを抽出することができます。ファイルのインポートは必要ありません。これはウィジェットツリーが作成されないことを意味します。

前提条件:

- .psdファイルが\$GUIDE_PROJECT_PATH/<project name>/resourcesにあること。バックグラウンド情報については、[6.12.4「.psdファイル形式」](#)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

[アセット]コンポーネントで、.psdファイルを右クリックし、[.psdファイルからイメージを生成]を選択します。

抽出されたすべてのイメージが含まれるサブディレクトリが\$GUIDE_PROJECT_PATH/<project name>/resourcesディレクトリに作成されます。

8.1.5. 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]コンポーネントで、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、ウィジェットを削除することもできます。

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]コンポーネントに、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]コンポーネントに、選択したウィジェットのプロパティが表示されます。

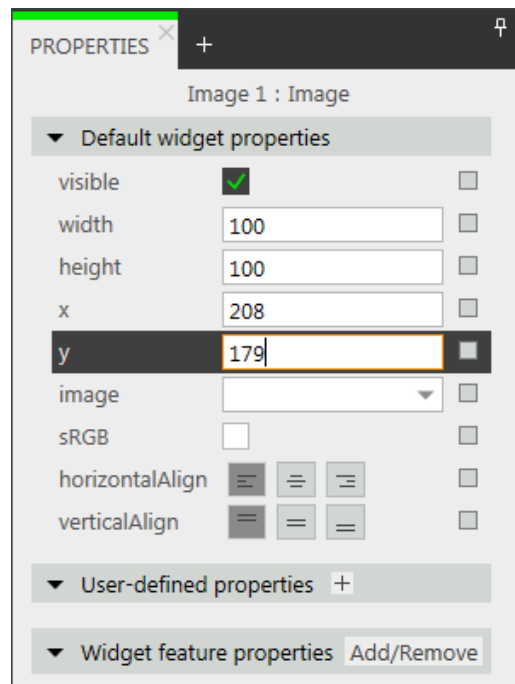


図8.3 イメージのプロパティ

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うです。

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。

インスタンス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]コンポーネントに、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]コンポーネントで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。

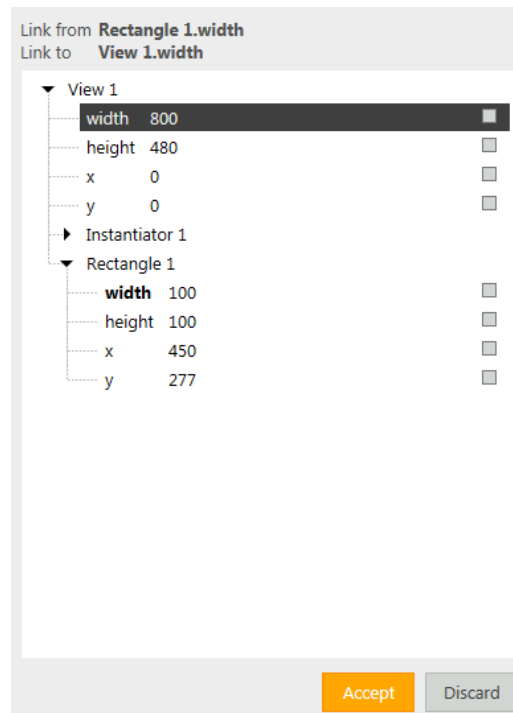



図8.4 ウィジェットプロパティ間のリンク設定

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]コンポーネントに、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]コンポーネントで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、テキスト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。

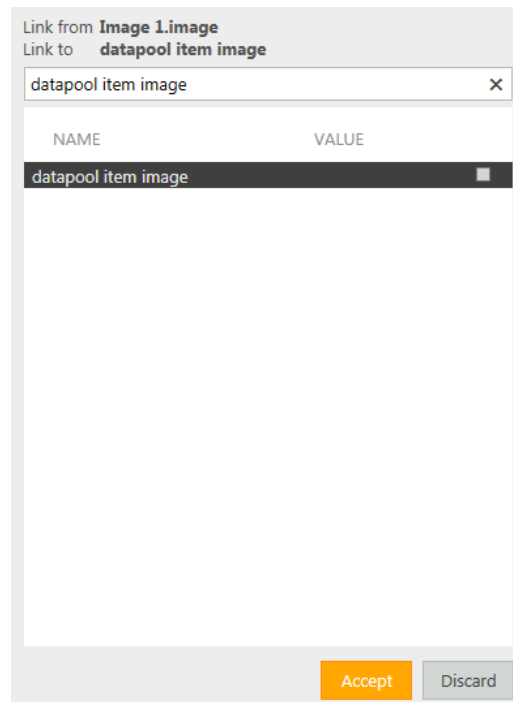


図8.5 データプールアイテムとのリンク設定

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]コンポーネントに、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]コンポーネントで[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. 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`

`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]コンポーネントに、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]コンポーネントで[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8.2

タイプのプロパティの呼び出し `Function () : bool`

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]コンポーネントで、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]コンポーネント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]コンポーネントでウィジェットをクリックします。

[プロパティ]コンポーネントに、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]コンポーネントで[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。

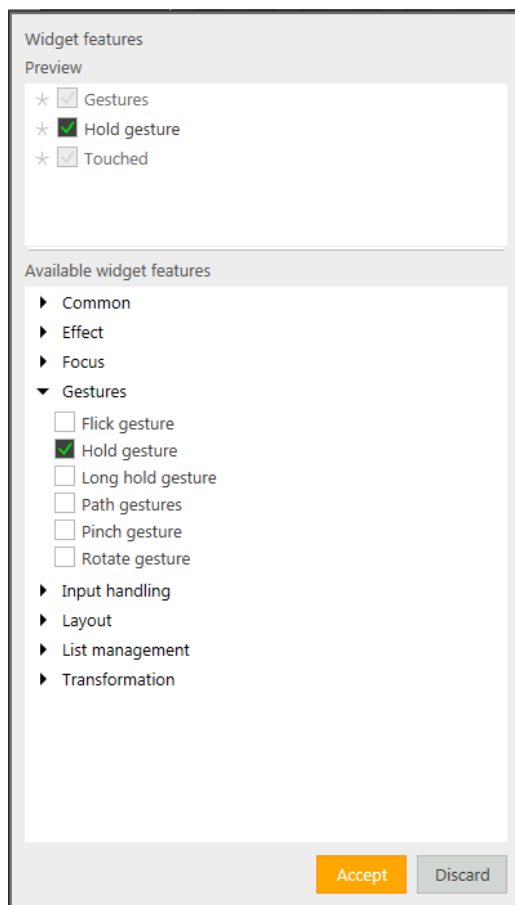


図8.6 ウィジェット機能ダイアログ

ステップ 3

[使用可能なウィジェット機能]の下でカテゴリを展開し、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]の下に表示されます。

[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、[12.11「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てボタン動作をモデリングする」](#)

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]コンポーネントでウィジェットをクリックします。

[プロパティ]コンポーネントに、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]コンポーネントで[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。

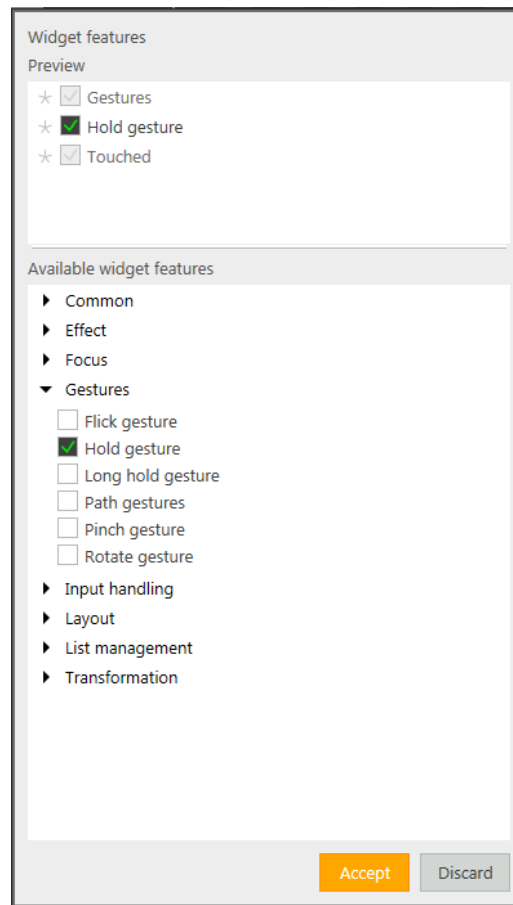


図8.7 ウィジェット機能ダイアログ

ステップ 3

[プレビュー]の下で、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]コンポーネントから削除されます。

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

注記



スキンのサポートは使用できません

データプールアイテムに言語サポートを定義した場合、同じアイテムにスキンのサポートを追加することはできません。



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、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に言語依存テキストを追加する」](#)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. スキンのサポートの操作

スキンのサポートを使用すると、モデルに異なるデータプール値を定義できます。このようにして、同じモデルに異なる外観(夜用モードと昼用モードなど)を定義できます。

8.5.1. EB GUIDEモデルへのスキンの追加

注記




言語サポートは使用できません

データプールアイテムにスキンのサポートを定義した場合、同じアイテムに言語サポートを追加することはできません。



EB GUIDEモデルへのスキンの追加

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] > [スキン]の順にクリックします。

標準スキンが各モデルにデフォルトで追加されます。

ステップ 3

コンテンツエリアで[追加]をクリックします。

スキンが表に追加されます。

ステップ 4

F2キーを押し、スキンの名前を変更します。

新しいスキンがモデルに追加されます。プロジェクトエディターのコマンドエリアのドロップダウンリストボックスで、新しいスキンを選択できます。

8.5.2. データプールアイテムにスキンのサポートを追加する



データプールアイテムにスキンのサポートを追加する

異なるデータプール値を定義することによってEB GUIDEモデルにさまざまな外観を定義するには、最初にデータプールアイテムにスキンのサポートを追加する必要があります。


前提条件:

- EB GUIDEモデルにデータプールアイテムが含まれていること。
- スキンがモデルに追加されていること。

ステップ 1

プロジェクトエディターで、[データプール]コンポーネントに移動します。


ステップ 2

データプールアイテムの[値]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[Add skin support]をクリックします。

ダイアログが閉じられます。[値]プロパティの横に  ボタンが表示されます。これは、スキンのサポートがこのデータプールアイテムに追加され、各スキんに異なる値を定義できるようになったことを示します。

ステップ 4

データプールアイテムに異なる値を定義するには、[データプール]コンポーネントでデータプールを選択します。

[プロパティ]コンポーネントに、EB GUIDEモデルで使用可能なすべてのスキンが含まれている表が表示されます。

ステップ 5

表内の各スキんに値を定義します。

8.5.3. スキンを切り替える



スキンを切り替える

前提条件:

- EB GUIDEモデルにデータプールアイテムが含まれていること。
- スキンがモデルに追加されていること。

ステップ 1

プロジェクトエディターで、コマンドエリアに移動します。

ステップ 2

ドロップダウンリストボックスでスキンを選択します。

コンテンツエリアに、このスキンで有効なデータプール値を持つモデルが表示されます。また、モデル実行モードでは特定のスキン値を持つモデルが表示されます。

8.5.4. スキンを削除する




スキンを削除する

前提条件:

- スキンが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] > [スキン]の順にクリックします。

現在のプロジェクトのすべてのスキンが一覧表示されます。

ステップ 3

削除するスキンを選択して、[削除]をクリックします。

スキンが表から削除されます。

8.6. ビュー遷移のアニメーション化

8.6.1. 開始アニメーションの追加



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]コンポーネント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]コンポーネントに移動します。

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(ミリ秒単位)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]チェックボックスにより、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。

8.6.2. 終了アニメーションの追加



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]コンポーネント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]コンポーネントに移動します。

ステップ 3

ビューの終了時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(ミリ秒単位)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(ミリ秒単位)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。

8.7. ウィジェットの再利用

8.7.1. テンプレートの追加



テンプレートの追加

ステップ 1

[ナビゲーション]コンポーネントで[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]コンポーネント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ースを定義します。

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。EB GUIDE このため、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。

8.7.2. テンプレートインターフェースの定義



テンプレートインターフェースの定義

前提条件:

- 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]コンポーネントでプロパティの横にある■ボタンをクリックします。メニューの[テンプレートインターフェースへの追加]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ースから削除]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



インスタンスエータのテンプレート

インスタンスエータのテンプレートでは、インスタンス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ースに追加できません。

8.7.3. 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]コンポーネントに、テンプレートインターフェー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ースを定義する


[プロパティ]コンポーネント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ー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ースを定義します。

ステップ 2

[プロパティ]コンポーネント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。

8.7.4. テンプレートの削除



テンプレートの削除

ステップ 1

[ナビゲーション]コンポーネント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

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

9.1. イベントの追加



イベントの追加

ステップ 1

[イベント]コンポーネントで、+をクリックします。

イベントが表に追加されます。

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:

- イベントがEB GUIDEモデルに追加されます。

ステップ 1

[イベント]コンポーネントでイベントをクリックします。

ステップ 2

イベント表で、イベントの横にある \pm をクリックします。

ステップ 3

ドロップダウンリストボックスから、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 4

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、数値を入力します。



イベントへの対応 EB GUIDE TF

前提条件:

- イベントグループが追加されています。
- イベントがEB GUIDEモデルに追加されます。

ステップ 1

[イベント]コンポーネントでイベントをクリックします。

[プロパティ]コンポーネントに、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

Event IDテキストボックスにIDを挿入します。

ステップ 3

[イベント]コンポーネントに移動し、Groupドロップダウンリストボックスからイベントグループを選択します。

9.4. イベントの削除



イベントの削除

前提条件:

- イベントがEB GUIDEモデルに追加されます。

ステップ 1

[イベント]コンポーネントで、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

ステップ 1

[データプール]コンポーネントで、+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。

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:

- 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

[データプール]コンポーネントで、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。

ステップ 2

Value列で、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、コンボ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は、Value列に表示されます。

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:

- 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

[データプール]コンポーネントで、データプールアイテムをクリックし、 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 2

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 3


Value列で、[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 4

EB GUIDEスクリプトを編集します。

ステップ 5

データプールアイテムをプレーン値に変換するには、 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 6

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。

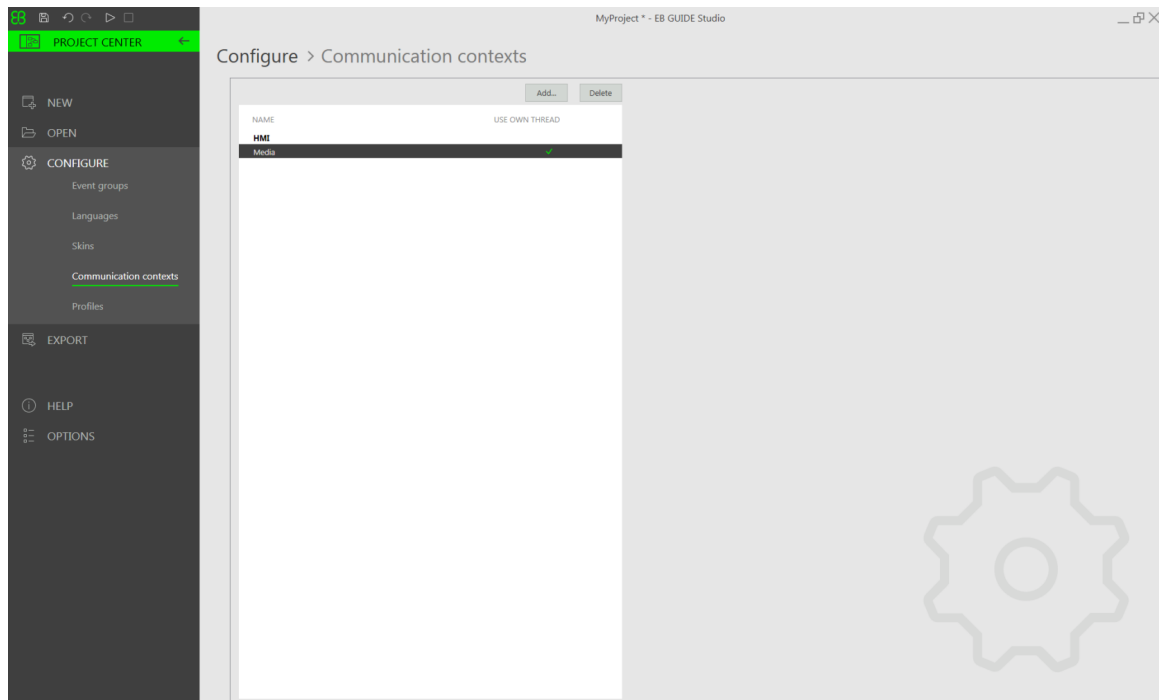


図9.1 通信コンテキストMedia。

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:

- 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

[データプール]コンポーネントでデータプールアイテムをクリックします。

ステップ 2

■ 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、テキスト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。

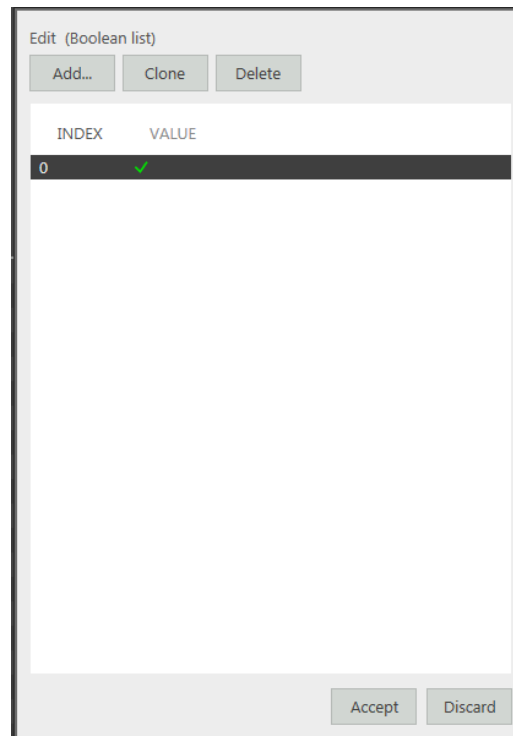



図9.2 データプールアイテム間のリンク設定

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、他方のデータプールアイテム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:

- 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

[データプール]コンポーネントで、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

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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



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. プロジェクトをに開く EB GUIDE Studio




プロジェクトをに開く EB GUIDE Studio

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

 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. モデル要素の名前を変更する



モデル要素の名前を変更する

次の手順は、モデル要素(ステート、ステートマシン、ウィジェット、遷移、データプールアイテム、イベントなど)の名前を変更する手順を説明しています。

前提条件:

- モデル要素がEB GUIDEモデルに追加されていること。

ステップ 1

モデル要素の名前を変更するには、以下の手順を実行します。

- ▶ ウィジェット、ステート、ステートマシン、遷移などのモデル要素の名前を変更するには、[ナビゲーション]コンポーネントで、そのモデル要素を右クリックします。
- ▶ データプールアイテムの名前を変更するには、[データプール]コンポーネントで、データプールアイテムを右クリックします。
- ▶ イベントの名前を変更するには、[イベント]コンポーネントで、データプールアイテムを右クリックします。

コンテキストメニューが開きます。

ステップ 2

コンテキストメニューで、次のいずれかをクリックします。

- ▶ 選択されているモデル要素のみの名前を変更するには、[名前の変更]をクリックします。
- ▶ 選択されているモデル要素だけでなく、EB GUIDEモデルでのその出現箇所(例えば、EB GUIDEスクリプト内)についても名前を変更するには、[グローバルな名前変更]をクリックします。

10.4. EB GUIDEモデルの検証およびモデル実行

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。

10.4.1. EB GUIDEモデルの検証

10.4.1.1. を使用したEB GUIDEモデルの検証 EB GUIDE Studio




を使用したEB GUIDEモデルの検証 EB GUIDE Studio

EB GUIDEは、[問題検出]コンポーネントに以下の項目を表示します。

- ▶  エラー
- ▶  警告

ステップ 1

[問題検出]コンポーネント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、[問題検出]コンポーネントを展開します。

エラーと警告の一覧が表示されます。

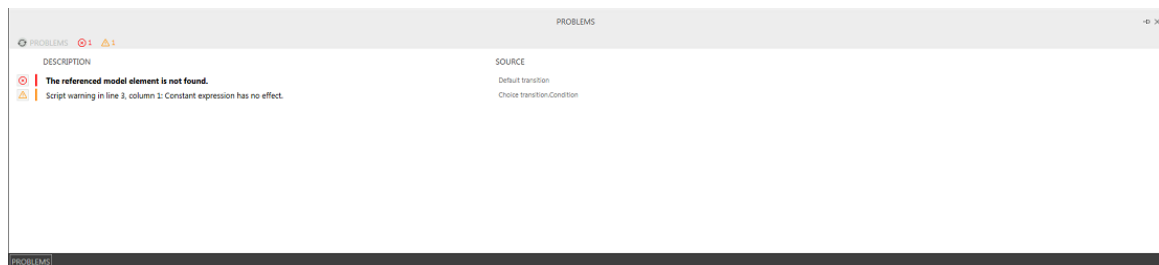


図10.1 問題検出コンポーネント

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、[問題検出]コンポーネントの一覧から削除されます。

ステップ 6

[問題検出]コンポーネントを折りたたむには、[問題]を再度クリックします。

エラーがない場合、EB GUIDEモデルは有効です。警告がいくつかあった場合でも、EB GUIDEモデルは有効です。

10.4.1.2. コマンドラインを使用したEB GUIDEモデルの検証



コマンドラインを使用したEB GUIDEモデルの検証

ステップ 1

コマンドラインで、`$GUIDE_INSTALL_PATH/Studio`に移動します。

ステップ 2

`Studio.Console.exe -c "<logfile dir>/log.txt" -o "$GUIDE_PROJECT_PATH/project_name.ebguide"`と入力します。

EB GUIDEモデルが検証され、結果が指定した場所(<logfile dir>)のログファイルに保存されます。

10.4.2. 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]の順にクリックします。

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。

10.5. EB GUIDEモデルのエクスポート

10.5.1. を使用したEB GUIDEモデルのエクスポート EB GUIDE Studio




を使用したEB GUIDEモデルのエクスポート EB GUIDE Studio

EB GUIDEモデルを対象デバイスにコピーするには、EB GUIDE Studioを使用して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。

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

[フォルダーの選択]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。

10.5.2. コマンドラインを使用したEB GUIDEモデルのエクスポート



コマンドラインを使用したEB GUIDEモデルのエクスポート

前提条件:

- EB GUIDEモデルにエラーおよび警告がないこと。

ステップ 1

コマンドラインで、\$GUIDE_INSTALL_PATH/Studioに移動します。

ステップ 2

Studio.Console.exe -e <destination dir> -p <profile> -o "\$GUIDE_PROJECT_PATH/project_name.ebguide"と入力します。


EB GUIDEモデルが選択した場所<destination dir>に指定したプロファイル<profile>でエクスポートされます。

10.6. EB GUIDE Studioの表示言語の変更



の表示言語の変更 EB GUIDE Studio

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を再起動します。

再起動後、グラフィカルユーザーインターフェースは選択した言語で表示されます。

10.7. 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。

10.7.1. プロファイルの追加



プロファイルの追加

EB GUIDE Studioでプロファイルを追加するには、既存のプロファイルを複製します。

前提条件:

- EB GUIDE Studioプロジェクトが開いていること。
- プロジェクトセンター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[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]をクリックします。

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

10.7.2. 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。

次のタスクでは、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。



ライブラリの追加: プラットフォーム

このタスクは、現在のプラットフォームのすべてのEB GUIDEモデルに使用できるライブラリを追加する方法を示しています。

前提条件:

- EB GUIDE Studioプロジェクトが開いていること。
- プロジェクトセンターが表示されていること。
- 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ること。
- MySimulationプロファイルが追加されていること。
- MyLibraryAおよびMyLibraryBライブラリが\$GTF_INSTALL_PATH/platform/<platform name>で使用可能であること。

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

[プラットフォーム]タブをクリックします。

ステップ 3

次のコードを入力します。

```
{
  "gtf":
  {
    "core":
    {
      "pluginstoload": ["MyLibraryA", "MyLibraryB"]
    }
  }
}
```

MyLibraryAおよびMyLibraryBライブラリが起動コードに追加されました。

注記



JSONオブジェクト表記

EB GUIDE Studio内でplatform.jsonを設定する場合は、JSONオブジェクト表記を使用します。

例えば、[12.7.1「EB GUIDE Studioでのサンプルplatform.json」](#)をご覧ください。

JSON形式の詳細については、<http://www.json.org>をご覧ください。



ライブラリの追加: モデル

このタスクは、現在のEB GUIDEモデルのみが使用できるライブラリを追加する方法を示しています。

前提条件:

- EB GUIDE Studioプロジェクトが開いていること。

- プロジェクトセンターが表示されていること。
- 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ること。
- MySimulationプロファイルが追加されていること。
- MyLibraryAおよびMyLibraryBライブラリが\$GUIDE_PROJECT_PATH/<project name>/resourcesで使用可能であること。

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

[モデル]タブをクリックします。

ステップ 3

次のコードを入力します。

```
{
  "gtf":
  {
    "model":
    {
      "pluginstoload": ["resources/MyLibraryA", "resources/MyLibraryB"]
    }
  }
}
```

MyLibraryAおよびMyLibraryBライブラリが起動コードに追加されました。

注記



JSONオブジェクト表記

EB GUIDE Studioでmodel.jsonを設定する場合は、JSONオブジェクト表記を使用します。

例えば、[12.6.1「でのサンプルmodel.json」](#)をご覧ください。

JSON形式の詳細については、<http://www.json.org>をご覧ください。

10.7.3. 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ること。
- プロジェクトセンターが表示されていること。
- 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ること。

ステップ 1

コンテンツエリアで、[シーン]タブをクリックします。

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、[12.8「シーン」](#)をご覧ください。

10.8. 言語依存テキストのエクスポートとインポート

10.8.1. 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に言語依存テキストを追加する」](#)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

[フォルダーの選択]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。

10.8.2. 言語依存テキストのインポート

10.8.2.1. EB GUIDE Studioを使用した言語依存テキストのインポート




を使用した言語依存テキストのインポート EB GUIDE Studio

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に言語依存テキストを追加する」](#)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。

10.8.2.2. コマンドラインを使用した言語依存テキストのインポート



コマンドラインを使用した言語依存テキストのインポート

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。
- EB GUIDEモデルにエラーおよび警告がないこと。
- 翻訳された1つの.xliff言語ファイルを使用できること。

ステップ 1

コマンドラインで、\$GUIDE_INSTALL_PATH/Studioに移動します。

ステップ 2

Studio.Console.exe -l <language file> -o "\$GUIDE_PROJECT_PATH/project_name.-ebguide"と入力します。

インポートが成功した場合、EB GUIDEモデルが保存されます。インポートが失敗した場合、EB GUIDEモデルは保存されません。どちらの場合もログファイルが生成されます。ログファイルの名前には日付とタイムスタンプが追加されます。

10.9. EB GUIDE Monitorを操作する

10.9.1. EB GUIDE Monitorでイベントを発行する



でイベントを発行する EB GUIDE Monitor

前提条件:

- EB GUIDEモデルにイベントが含まれていること。
- EB GUIDEモデルのシミュレーションが開始されます。
- EB GUIDE Monitorが開始されます。

ステップ 1


EB GUIDE Monitorの[イベント]コンポーネントで、[イベントの検索]検索ボックスを使って発行したいイベントを検索します。

ステップ 2

イベントをクリックします。

イベントがリストに追加されます。

ステップ 3

イベントを発行するには、イベントの横にある[イベント]コンポーネントで  をクリックします。

イベントが発行されます。[ログャー]コンポーネントにログメッセージが表示されます。

ステップ 4


ステップ 4.1

イベントにパラメータがある場合、 をクリックしてパラメータを展開します。

ステップ 4.2

[値]列のパラメータを変更します。

ステップ 4.3

イベントを発行するには、イベントの横にある  をクリックします。

変更後のパラメータでイベントが発行されます。[ログャー]コンポーネントにログメッセージが表示されます。

10.9.2. でのデータプールアイテムの値の変更 EB GUIDE Monitor



EB GUIDE Monitorでのデータプールアイテムの値の変更

前提条件:

- EB GUIDEモデルにデータプールアイテムが含まれていること。
- EB GUIDEモデルのシミュレーションが開始されます。
- EB GUIDE Monitorが開始されます。

ステップ 1

EB GUIDE Monitorの[データプール]コンポーネントで、[データプールアイテムの検索]検索ボックスを使ってデータプールアイテムを検索します。

ステップ 2

データプールアイテムをクリックします。

データプールアイテムがリストに追加されます。

ステップ 3

[値]列でデータプールアイテムの値を変更します。

注記



サポートされるタイプ

以下のデータタイプのデータプールアイテムを変更することができます。f

- ▶ ブール値
- ▶ 色
- ▶ 整数
- ▶ 浮動小数点数
- ▶ 文字列

データプールアイテムの値が変更されます。[ロガー]コンポーネントにログメッセージが表示されます。

10.9.3. EB GUIDE Monitorでスクリプトを開始する



でスクリプトを開始する EB GUIDE Monitor

前提条件:

- EB GUIDEモデルのシミュレーションが開始されます。
- EB GUIDE Monitorが開始されます。
- スクリプトが含まれている .csファイルまたは .dllファイルをコンピューターで使用できること。スクリプトのサンプルについては、[10.9.3.1「でのスクリプトファイルの記述 EB GUIDE Monitor」](#)をご覧ください。

ステップ 1

[スクリプト]コンポーネントを開くには、[レイアウト] > [スクリプト]を選択します。

[スクリプト]コンポーネントがドッキングされたコンポーネントとして開きます。

ステップ 2

[スクリプト]コンポーネントで、[Open]ボタンをクリックします。

ファイルエクスプローラが開きます。

ステップ 3

.csファイルまたは .dllファイルを選択して、[Open]をクリックします。

ファイルに含まれている適用可能なすべてのメソッドおよび対応するクラスが、[スクリプト]表に一覧で示されます。

ステップ 4

メソッドを選択し、開始ボタンをクリックします。

スクリプトが開始されます。[スクリプト出力]エリアにログメッセージが表示されます。

10.9.3.1. でのスクリプトファイルの記述 EB GUIDE Monitor

スクリプトのメソッドの詳細については、\$GUIDE_INSTALL_PATH/doc/monitor/monitor_api.chmのEB GUIDE Monitor APIをご覧ください。

以下に、基本的なEB GUIDE Monitorスクリプト機能のサンプルを示します。

注記



ステートおよびステートマシンでのメソッドの使用

EB GUIDEモデルに同じ名前のステートまたはステートマシンが複数ある場合は、uint ID を使用します。\$EXPORT_ PATH/monitor.cfgで、プロジェクトに関係があるuint IDを検索します。



例10.1

のサンプルスクリプトファイル EB GUIDE Monitor

サンプルスクリプトMonitorScriptSample.csを以下に示します。

```
namespace MyProject
{
    using System.Threading.Tasks;

    using System.Windows.Media; // necessary for Color type!

    using Elektrobit.Guide.Monitor.Scripting.MonitorContext;

    public class Basic
    {
        public async Task PrintMessage(IMonitorContext monitor) //❶
        {
            await monitor.Write("Hello World");
        }

        public async Task FireEvent(IMonitorContext monitor) //❷
        {
            await monitor.FireEvent("nextView");
        }
    }

    public class Events
    {
        public async Task FireEventWithParameter(IMonitorContext monitor)
        {
            await monitor.FireEvent("setBool", true);
        }

        public async Task WaitForEvent(IMonitorContext monitor) //❸
    }
}
```

```
{
    var ev = await monitor.WaitForEvent("nextView");
    await monitor.Write("Even occurred: " + ev.EventModel.Name);
}

public async Task WaitForEventWithParameters(IMonitorContext monitor)
{
    var ev = await monitor.WaitForEvent("setBool");

    bool mv1 = ev["value"]; // read parameter via name
    bool mv2 = ev[0]; // read the parameter via index

    await monitor.Write("Parameter 'value' is: " + mv1);
    await monitor.Write("Parameter [0] is: " + mv2);
}
}

public class Datapool
{
    public async Task WriteDpValue(IMonitorContext monitor) //④
    {
        await monitor.WriteDatapool("Boolean 1", true);
    }

    public async Task ReadDatapoolValue(IMonitorContext monitor) //⑤
    {
        bool boolValue = await monitor.ReadDatapool("Boolean 1");
        string stringValue = await monitor.ReadDatapool("String 1");
        int integerValue = await monitor.ReadDatapool("Integer 1");
        float floatValue = await monitor.ReadDatapool("Float 1");

        await monitor.Write("Boolean: " + boolValue);
        await monitor.Write("String: " + stringValue);
        await monitor.Write("Integer: " + integerValue);
        await monitor.Write("Float: " + floatValue);
    }

    public async Task ReadColor(IMonitorContext monitor)
    {
        Color colorValue = await monitor.ReadDatapool("Color 1");
        await monitor.Write("Boolean: " + colorValue);
    }
}

public class StateMachines
{
    public async Task WaitForStateChanges(IMonitorContext monitor)
```

```
{
    var leftState = await monitor.WaitForStateExit
        ("Main", "State 1"); //⑥
    await monitor.Write(string.Format("State {0} left",
        leftState.Name));

    var enteredState = await monitor.WaitForStateEnter
        ("Main", "State 2"); //⑦
    await monitor.Write(string.Format("State {0} entered",
        enteredState.Name));
}

public async Task WaitForStateMachineChanges(IMonitorContext monitor)
{
    var startedStateMachine = await monitor.WaitForStateMachineStart
        ("Dynamic state machine 1"); //⑧
    await monitor.Write(string.Format("State Machine {0} started",
        startedStateMachine.Name));

    var stoppedStateMachine = await monitor.WaitForStateMachineStop
        ("Dynamic state machine 1"); //⑨
    await monitor.Write(string.Format("State Machine {0} stopped",
        stoppedStateMachine.Name));
}

public class Advanced
{
    public async Task CaptureScreenshot(IMonitorContext monitor) //⑩
    {
        // make sure remote framebuffer is enabled in profile
        uint sceneId = 0;
        await monitor.CaptureScreenshot(sceneId, @"d:/image.png");
    }

    public async Task CountTo10(IMonitorContext monitor)
    {
        for (var i = 0; i < 10; i++)
        {
            await monitor.Write("Hello World: " + i);
            await Task.Delay(1000, monitor.CancellationToken);

            monitor.CancellationToken.ThrowIfCancellationRequested();
        }
    }
}
```

```
public async Task WaitForEventWithTimeout(IMonitorContext monitor) //❧
{
    // Disclaimer:
    // this is just one of many opportunities provided by
    // the .NET's "Task Parallel Library"

    var eventWaitTask = monitor.WaitForEvent("nextView");

    await Task.WhenAny(eventWaitTask, Task.Delay(5000));

    if (!eventWaitTask.IsCompleted || eventWaitTask.IsFaulted)
    {
        return;
    }

    await monitor.Write("event occurred");
}
}
```

- ❶ メッセージを出力するメソッド
- ❷ イベントを発行するメソッド
- ❸ イベントが発行されるまで待機するメソッド
- ❹ データプール値を書き込むメソッド
- ❺ データプール値を読み取るメソッド
- ❻ ステートにエントリーするまで待機したうえでエントリーを報告するメソッド
- ❼ ステートが終了するまで待機したうえで終了を報告するメソッド
- ❽ ステートマシンが開始されるまで待機したうえで開始を報告するメソッド
- ❾ ステートマシンが停止するまで待機したうえで停止を報告するメソッド
- ❿ スクリーンショットをキャプチャするメソッド
- ❧ イベントがタイムアウトするまで待機するメソッド

10.9.4. スタンドアロンアプリケーションとしてEB GUIDE Monitorを起動する

EB GUIDE Monitor は、EB GUIDEモデルのモデル実行中にEB GUIDE Studioで自動的に起動されます。しかし、`$GUIDE_INSTALL_PATH/tools/monitor`のスタンドアロンアプリケーションとして、またはコマンドラインを使って、EB GUIDE Monitorを起動することもできます。



コマンドラインを使用してEB GUIDE Monitorを起動する

前提条件:

- EB GUIDE がインストールされていること。
- EB GUIDEモデルが\$EXPORT_PATHにエクスポートされていること。

ステップ 1

ファイルエクスプローラで、\$GUIDE_INSTALL_PATH/tools/monitorに移動します。

ステップ 2

コマンドラインを開き、次のように入力します。Monitor.exe -c <ip adress>:<port> -o <\$EXPORT_PATH/monitor.cfg>

EB GUIDE Monitor が起動されます。

ティップ



事前設定済みデータプールアイテムおよびイベントの再利用

EB GUIDE Monitorでは、データプールアイテムやイベントを設定できます。設定されている値は、C:/Users/<username>/AppData/Local/Temp/eb_guide_simulation_export/<guide_project>/monitor_settings.xmlに保存されています。事前設定済みの値を再利用するには、monitor_settings.xmlを\$EXPORT_PATHにコピーします。



の表示言語の変更 EB GUIDE Monitor

前提条件:

- EB GUIDE Monitor がスタンドアロンアプリケーションとして起動されていること。

ステップ 1

[ファイル] > [表示言語]メニューから言語を選択します。

ステップ 2

EB GUIDE Monitorを再起動します。

再起動後、グラフィカルユーザーインターフェースが選択した言語で表示されます。

なお、EB GUIDE MonitorがEB GUIDE Studioで起動されている場合、グラフィカルユーザーインターフェースの表示言語は変更できません。つまり、EB GUIDE Monitorの表示言語はEB GUIDE Studioと同じものになります。

11. チュートリアル

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

11.1. チュートリアル: 動的ステートマシンの追加

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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。

ステップ 1

[イベント]コンポーネントに移動して、+をクリックします。

イベントが表に追加されます。

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、その名前をVolume downに変更します。

ステップ 4

イベントを追加し、その名前をClose volume controlに変更します。

ステップ 5

[データプール]コンポーネントに移動して、+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]コンポーネントで[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コンテンツエリアに追加されて表示されます。

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

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

ステップ 5

[ツールボックス]からビューステートをドラッグし、動的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 6

[ナビゲーション]コンポーネントでビューステートをクリックします。

ステップ 7

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 8

遷移を初期ステートからVolumeビューステートに追加します。



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]コンポーネントで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]コンポーネント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]コンポーネントに移動します。

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と入力します。

ステップ 5.3

yテキストボックスに300と入力します。

ステップ 6

[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]コンポーネントの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]コンポーネント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]コンポーネントに移動します。

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と入力します。

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

リストから、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プロパティでは黒を選択します。

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。

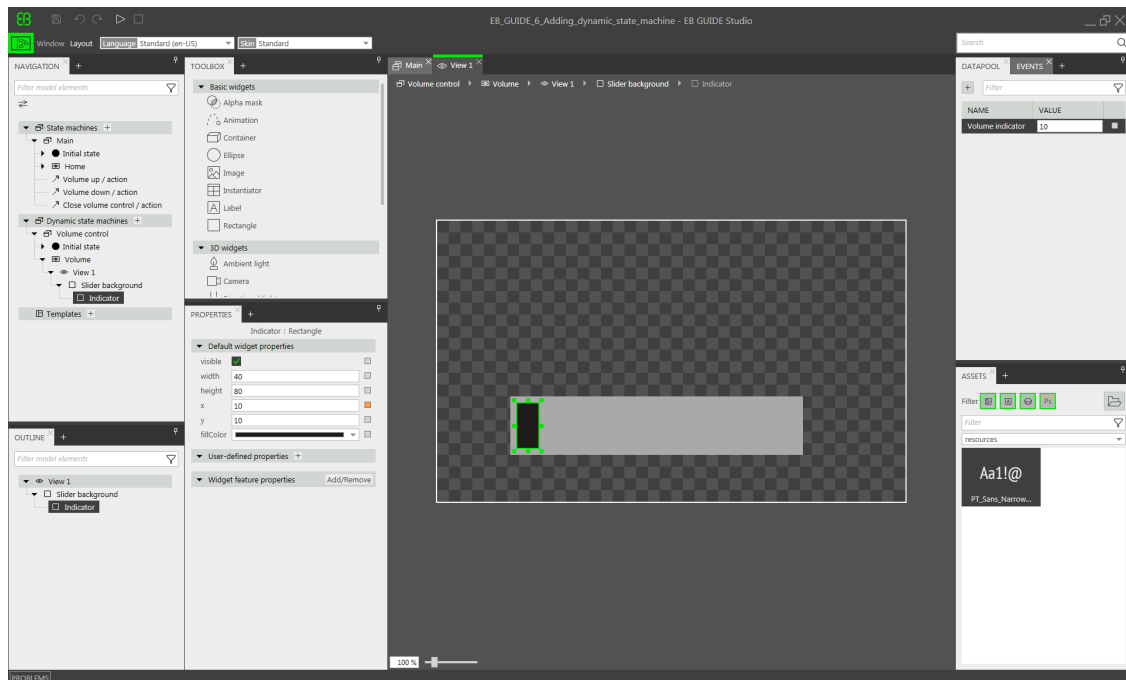


図11.1 2つの四角形を持つ[ビュー1]の外観

ステップ 10

[データプール]コンポーネントで、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

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]コンポーネントで[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

初期ステートからHomeビューステートへの遷移を追加します。

ステップ 7

[ナビゲーション]コンポーネントで[メイン]をクリックします。

ステップ 8

[プロパティ]コンポーネントで、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]コンポーネントで[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]コンポーネントで[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]コンポーネントに表示されます。

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]コンポーネントで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]コンポーネントに移動します。

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]コンポーネント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]コンポーネントで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]コンポーネントに移動します。

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]コンポーネント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]コンポーネントで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]コンポーネントに移動します。

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]コンポーネント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。

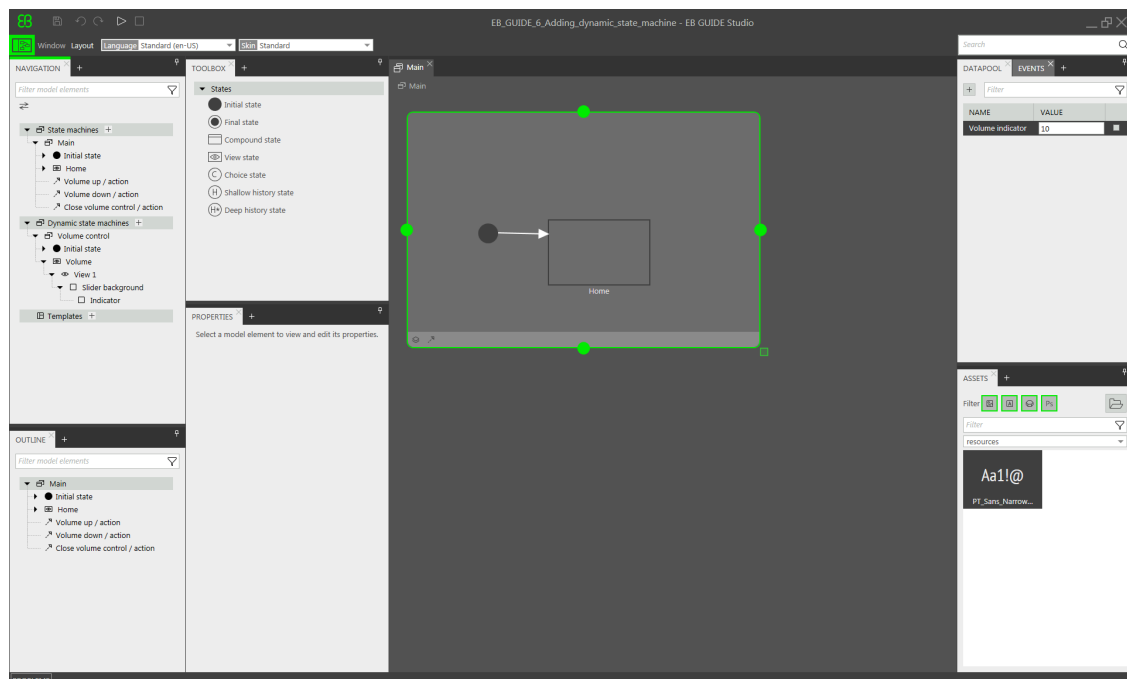


図 11.2 EB GUIDE すべてのモデル要素が揃ったモデル




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モデルにHomeビューステートが表示されます。
[ステップ 1](#)

EB GUIDE Monitorの[イベント]コンポーネントで、Volume upイベントを選択してをクリックし、イベントを発行します。

動的ステートマシンが起動し、スライダーインジケータが表示されます。動的ステートマシンは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イベントが発行されると、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てボタン動作をモデリングする

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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]コンポーネント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]コンポーネント内で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]コンポーネント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]コンポーネント内で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]コンポーネント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。

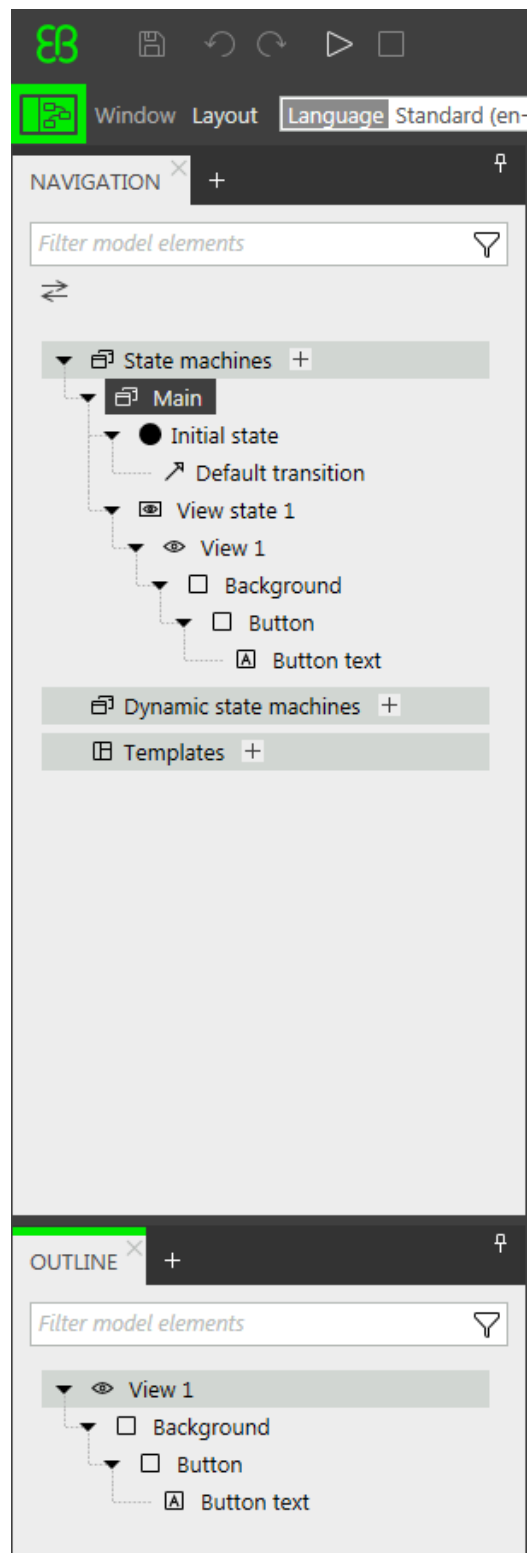


図11.3 ウィジェットの階層



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]コンポーネントでBackground四角形をクリックし、[プロパティ]コンポーネントに移動します。

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

[データプール]コンポーネントで、+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

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]コンポーネントでButton四角形をクリックし、[プロパティ]コンポーネントに移動します。

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と入力します。

ステップ 1.3

yテキストボックスに215と入力します。

ステップ 1.4

fillColorプロパティで青を選択します。

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]で、[入力処理]カテゴリを展開して[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]コンポーネントに表示されます。

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で置き換えます。

```
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]コンポーネントでButton textラベルをクリックし、[プロパティ]コンポーネントに移動します。

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height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と入力します。

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

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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]コンポーネントに、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]コンポーネントで、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]コンポーネントに、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]コンポーネントで[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]コンポーネントに、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]コンポーネントで[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]コンポーネントに表示されます。

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

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを[レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

インスタンスエータを使うと、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、インスタンス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、インスタンス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、インスタンス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、インスタンス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。

[データプール]コンポーネントで、**+**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

 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 4.1

[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 4.2

ValueテキストボックスにOneと入力します。

ステップ 4.3

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 4.4

[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]コンポーネント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。

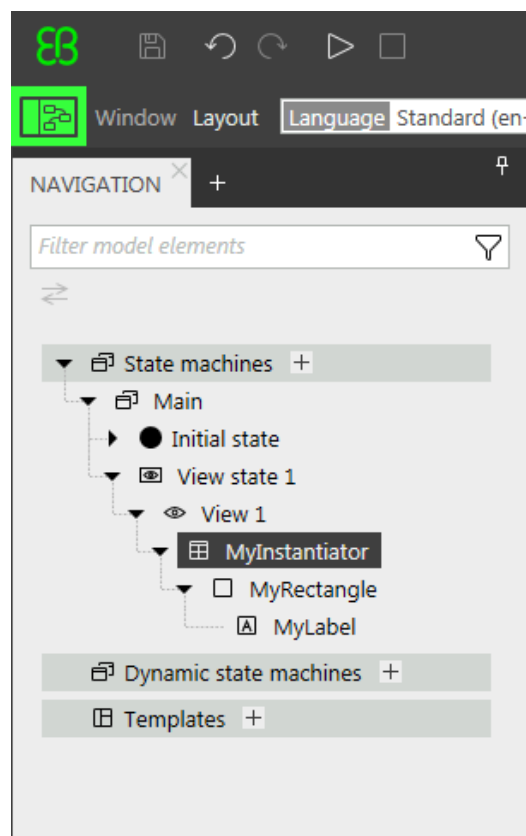


図11.5 インスタシエータを含むウィジェット階層



インスタシエータの設定

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、インスタシエータを選択し、[プロパティ]コンポーネントに移動します。

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と入力します。

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スエータに追加され、[プロパティ]コンポーネントに表示されます。

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ス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]コンポーネントに移動します。


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height 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。

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]コンポーネントに移動します。

ステップ 2

リスト要素で利用可能な幅が必ず使用されるように、四角形のwidthプロパティからインスタンスエータの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

四角形の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プロパティにより、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。

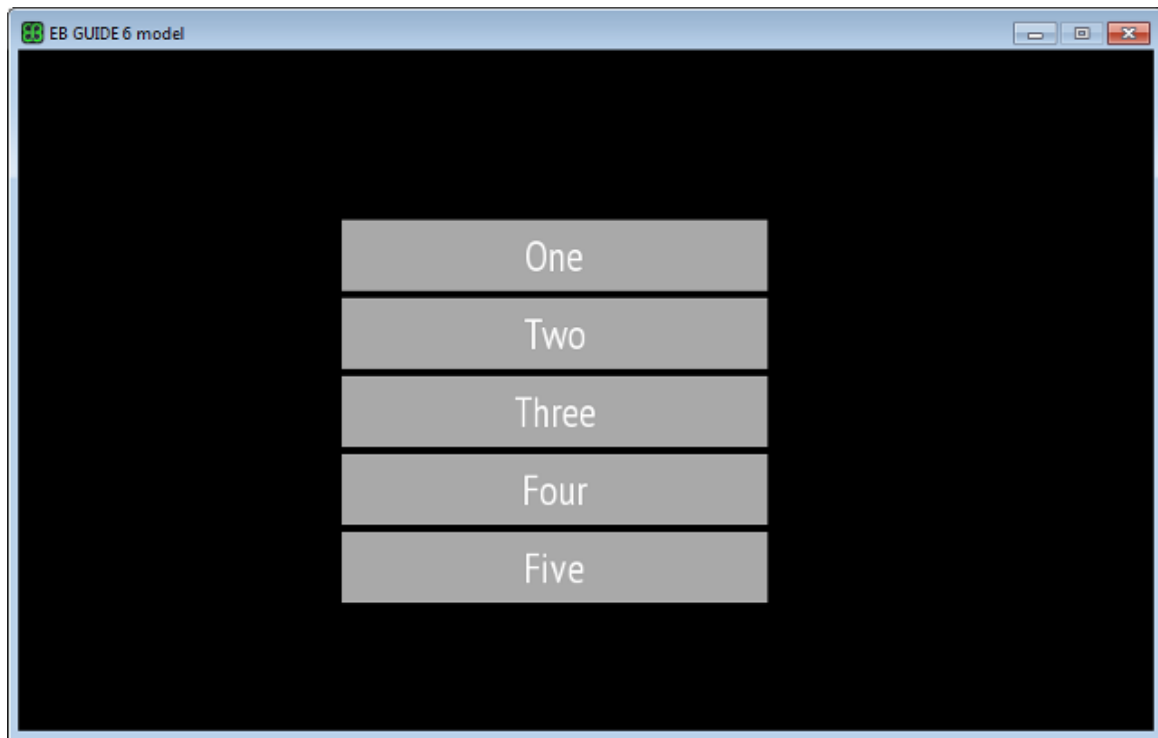


図 11.6 インスタンスエータで作成されたリスト

11.5. チュートリアル: 画面内で楕円を移動する

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

このセクションでは、楕円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楕円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]から楕円をドラッグし、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、楕円にドロップします。

ステップ 4

[ナビゲーション]コンポーネント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

[ツールボックス]からリニア補間整数ウィジェットをドラッグして、[ナビゲーション]コンポーネントにドロップし、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、ビューに楕円が表示されますが、まだ楕円は動きません。



タイプのユーザー定義プロパティの追加 Conditional script

以下の手順では、楕円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楕円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

楕円を選択します。

ステップ 2

[プロパティ]コンポーネントで[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楕円に追加されます。

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

[アニメーション]エディターに移動し、[アニメーションプロパティ]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 2

Ellipse 1の下でxプロパティ、[リニア補間曲線]の順に選択します。

ステップ 3

[承認]をクリックします。

■ ボタンがtargetプロパティの横に表示されます。

ステップ 4

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、楕円の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、楕円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 5

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 6

プロジェクトを保存します。

ステップ 7

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:

楕円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に言語依存テキストを追加する

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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ー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。[言語1]の名前がGermanに、[言語2]の名前がFrenchに設定されていること。
- 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。
- 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。

- コンテンツエリアにビューが表示されます。
- 初期ステートにビューステートへの遷移があること。
- ビューステートにラベルが含まれていること。

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]コンポーネントで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、テキストボックスに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。

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

[データプール]コンポーネントで、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

■ボタンをクリックします。

ステップ 3

メニューの[Add language support]をクリックします。

[プロパティ]コンポーネントに言語プロパティが表示されます。

ステップ 4

[データプール]コンポーネントで、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 5

[プロパティ]コンポーネントに移動します。

ステップ 6

Germanテキストボックスに、Willkommenと入力します。

左上隅のLanguageボックスで、言語をGermanに変更します。

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

Frenchテキストボックスに、Bienvenueと入力します。

左上隅のLanguageボックスで、言語をFrenchに変更します。

コンテンツエリアで、ラベルにBienvenueと表示されます。

英語、ドイツ語、およびフランス語の言語サポートを追加し、言語依存のテキストラベルを定義しました。



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

[データプール]コンポーネントで、+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]コンポーネントで、Label 1ラベルをクリックします。

ステップ 5

[プロパティ]コンポーネントで[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]で、[入力処理]カテゴリを展開して[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]コンポーネントに表示されます。

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で置き換えます。

```
function(v:touchId::int, v:x::int, v:y::int, v:fingerId::int)
{
    if (dp:SelectedLanguage == 0) // Standard selected
    {
        f:language(1:German)
        dp:SelectedLanguage = 1
    }
    else if (dp:SelectedLanguage == 1) // German selected
    {
        f:language(1:French)
        dp:SelectedLanguage = 2
    }
    else if (dp:SelectedLanguage == 2) // French selected
    {
        f:language(1: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:

String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。この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

注記



デフォルトのウィンドウレイアウト

このユーザーマニュアルのすべての手順およびスクリーンショットでは、デフォルトのウィンドウレイアウトが使用されています。手順に従う場合は、EB GUIDE Studioウィンドウを [レイアウト] > [Reset to default layout]を選択することによって、デフォルトレイアウトにリセットすることをお勧めします。

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バージョン2.0以上およびDirectX 11レンダラーのみが3Dグラフィックを表示できます。グラフィックドライバがレンダラーのバージョンと互換性があることを確認してください。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。ファイルに、カメラ、光源、およびメッシュと少なくとも1つの材質を含む1つのオブジェクトが含まれていること。

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]コンポーネントで[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]コンポーネントに表示されます。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]コンポーネントで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。この指向性ライトは、RootNodeの変形プロパティを使用して操作および変換できます。

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]からシーン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]コンポーネントでMesh 1をクリックし、[プロパティ]コンポーネントに移動します。

ステップ 2

meshコンボボックスで、前述のリソースフォルダーから.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。

ステップ 3

または、[アセット]コンポーネントから.ebmeshファイルをドラッグして、meshドロップダウンリストボックスにドロップします。

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]コンポーネントで材質をクリックし、[プロパティ]コンポーネントに移動します。

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]コンポーネントに表示されます。

ステップ 5

[プロパティ]コンポーネントで、diffuseTextureコンボ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]
- ▶ [ディフューズテクスチャ]
- ▶ [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]コンポーネントでMy3DGraphicをクリックし、[プロパティ]コンポーネントに移動します。

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]コンポーネントで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]コンポーネントに移動します。

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]コンポーネントに表示されます。

ステップ 8

[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 9

Camera 2をクリックし、[プロパティ]コンポーネントに移動します。

ステップ 10

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 11

[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 12

[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 13

[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]コンポーネントに表示されます。



ステップ 14

[プロパティ]コンポーネントで、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. リファレンス

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: <ul style="list-style-type: none">▶ <code>enabled</code>: <code>true</code>である場合、レンダラーは有効になります。<code>false</code>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: <ul style="list-style-type: none">▶ <code>visibility</code>: <code>true</code>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。<code>false</code>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: <ul style="list-style-type: none">▶ <code>visibility</code>: <code>true</code>である場合、仮想キーボードが表示されます。<code>false</code>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 パラメータ(ピクセル単位):

イベントID	名前	説明
		<ul style="list-style-type: none">▶ 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

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。

- ▶ [トリガー]コンボボックスには、[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のトリガー後、またはデータプールアイテムの値の更新後、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、スクリプト実行の原因を表します。

[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以下の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) (=)

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以下の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 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条件部が完了しました。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
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、などの変数を宣言します let
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。
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 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 演算子の優先順位

演算子	結合性
(()), ({})	なし

演算子	結合性
([])	なし
(->)	左
(.)	なし
(::)	左
長さ	なし
(&)	右
(!), (-) 単項マイナス	右
(*), (/), (%)	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 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 のパラメータ abs

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 のパラメータ absf

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 のパラメータ acosf

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 のパラメータ animation_before

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 のパラメータ animation_beyond

パラメータ	タイプ	説明
animation	GtfTypeRecord	操作するアニメーション

パラメータ	タイプ	説明
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 のパラメータ animation_cancel

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 のパラメータ animation_cancel_end

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 のパラメータ animation_cancel_reset

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 のパラメータ animation_pause

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 のパラメータ animation_play

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 のパラメータ animation_reverse

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 のパラメータ animation_running

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せん。

表12.16 のパラメータ `animation_set_time`

パラメータ	タイプ	説明
<code>animation</code>	<code>GtfTypeRecord</code>	操作するアニメーション
<code>time</code>	整数	時間
<code><return></code>	ブール値	<code>true</code> 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 のパラメータ `asinf`

パラメータ	タイプ	説明
<code>x</code>	浮動小数点数	逆正弦を返す数
<code><return></code>	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 のパラメータ `atan2f`

パラメータ	タイプ	説明
<code>y</code>	浮動小数点数	引数 y
<code>x</code>	浮動小数点数	引数 x
<code><return></code>	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 のパラメータ `atan2i`

パラメータ	タイプ	説明
<code>y</code>	整数	引数 y
<code>x</code>	整数	引数 x
<code><return></code>	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 のパラメータ atanf

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 のパラメータ ceil

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 のパラメータ changeDynamicStateMachinePriority

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 のパラメータ character2unicode

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
<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 のパラメータ clearAllDynamicStateMachines

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 のパラメータ color2string

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 のパラメータ cosf

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 のパラメータ deg2rad

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 のパラメータ expf

パラメータ	タイプ	説明
x	浮動小数点数	指数
<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 のパラメータ float2string

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 のパラメータ floor

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 のパラメータ focusNext

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 のパラメータ focusPrevious

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 のパラメータ format_float

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。
useDotAs Delimiter	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。

パラメータ	タイプ	説明
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 のパラメータ format_int

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: 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テキストの行数を返します。

表12.35 のパラメータ `getLineCount`

パラメータ	タイプ	説明
<code>widget</code>	ウィジェット	評価するウィジェット
<code><return></code>	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 のパラメータ `getTextHeight`

パラメータ	タイプ	説明
<code>text</code>	文字列	評価するテキスト
<code>font</code>	フォント	評価するフォント
<code><return></code>	整数	テキストの高さ フォントのサイズが 0 または負の値の場合、この関数は 0 を返します。

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 のパラメータ `getTextLength`

パラメータ	タイプ	説明
<code>text</code>	文字列	評価するテキスト
<code><return></code>	整数	テキスト内の文字数

注記



エスケープシーケンス

EB GUIDE スクリプト は、`\n` などのエスケープシーケンスを解決せず、すべての文字をカウントします。例えば、`Label\n` というテキストの場合、`getTextLength` 関数は 7 を返します。

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 のパラメータ `getTextWidth`

パラメータ	タイプ	説明
<code>text</code>	文字列	評価するテキスト

パラメータ	タイプ	説明
font	フォント	評価するフォント
<return>	整数	テキストの幅 フォントのサイズが 0 または負の値の場合、この関数は 0 を返します。

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を確認します。

表 12.39 のパラメータ has_list_window

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 のパラメータ hsba2color

パラメータ	タイプ	説明
hue	整数	0～360の色の値(度単位)
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 のパラメータ `int2float`

パラメータ	タイプ	説明
<code>value</code>	整数	浮動小数点数に変換する値
<code><return></code>	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 のパラメータ `int2string`

パラメータ	タイプ	説明
<code>value</code>	整数	文字列に変換する値
<code><return></code>	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 のパラメータ `isDynamicStateMachineActive`

パラメータ	タイプ	説明
<code>state</code>		動的ステートマシンリストを持つステート
<code>sm</code>	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。この動作は非同期で実行されます。

表12.44 のパラメータ `language`

パラメータ	タイプ	説明
<code>language</code>	<code>languageType</code>	切り替える言語(など) <code>f:language (l:German)</code>
<code><return></code>	<code>void</code>	

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 のパラメータ localtime_day

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 のパラメータ localtime_hour

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 のパラメータ localtime_minute

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 のパラメータ localtime_month

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 のパラメータ localtime_second

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 のパラメータ localtime_weekday

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 のパラメータ localtime_year

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 のパラメータ log10f

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 のパラメータ `logf`

パラメータ	タイプ	説明
<code>x</code>	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 のパラメータ `nearbyint`

パラメータ	タイプ	説明
<code>value</code>	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 のパラメータ `popDynamicStateMachine`

パラメータ	タイプ	説明
<code>state</code>		動的ステートマシンリストを持つステート
<code>sm</code>	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 のパラメータ `powf`

パラメータ	タイプ	説明
<code>x</code>	浮動小数点数	引数 x
<code>y</code>	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 のパラメータ pushDynamicStateMachine

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 のパラメータ rad2deg

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 $0 \sim 2^{31}-1$ の間の乱数値を取得します。

表12.59 のパラメータ rand

パラメータ	タイプ	説明
<return>	整数	$0 \sim 2^{31}-1$ の間の乱数

12.4.3.3.20. shutdown

この関数は、プログラムをシャットダウンするようにフレームワークに要求します。

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12.60 のパラメータ rgba2color

パラメータ	タイプ	説明
red	整数	0～255の範囲の赤色座標

パラメータ	タイプ	説明
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12.61 のパラメータ round

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12.62 のパラメータ seed_rand

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12.63 のパラメータ sinf

パラメータ	タイプ	説明
x	浮動小数点数	引数

パラメータ	タイプ	説明
<return>	浮動小数点数	戻り値

12.4.3.4.3. skin

この関数はすべてのデータプールアイテムのスキンを切り替えます。この動作は非同期で実行されます。

表12.64 のパラメータ skin

パラメータ	タイプ	説明
skin	skinType	切り替えるスキン(など) f:skin(s:Standard)
<return>	void	

12.4.3.4.4. sqrtf

この関数はxの負ではない平方根を返します。

表12.65 のパラメータ sqrtf

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

12.4.3.4.5. 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 のパラメータ string2float

パラメータ	タイプ	説明
str	文字列	文字列値

パラメータ	タイプ	説明
<return>	浮動小数点数	戻り値

12.4.3.4.6. 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 のパラメータ string2int

パラメータ	タイプ	説明
str	文字列	文字列値
<return>	整数	戻り値

12.4.3.4.7. string2string

この関数は文字列をフォーマットします。

表12.68 のパラメータ string2string

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

12.4.3.4.8. 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 のパラメータ substring

パラメータ	タイプ	説明
str	文字列	入力文字列
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

パラメータ	タイプ	説明
<return>	文字列	言語の文字列

12.4.3.4.9. 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。結果は、`localtime_*`関数に渡されることを想定したものになっています。

表12.70 のパラメータ `system_time`

パラメータ	タイプ	説明
<return>	整数	秒数のシステム時刻

12.4.3.4.10. 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 のパラメータ `system_time_ms`

パラメータ	タイプ	説明
<return>	整数	ミリ秒数のシステム時刻

12.4.3.4.11. 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 のパラメータ `tanf`

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

12.4.3.4.12. 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 のパラメータ `trace_dp`

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	

12.4.3.4.13. 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 のパラメータ trace_string

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	

12.4.3.4.14. 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 のパラメータ transformToScreenX

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

12.4.3.4.15. 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 のパラメータ transformToScreenY

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Y位置

12.4.3.4.16. 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 のパラメータ transformToWidgetX

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

12.4.3.4.17. 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 のパラメータ transformToWidgetY

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

12.4.3.4.18. 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 のパラメータ trunc

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

12.4.3.4.19. 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 のパラメータ widgetGetChildCount

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

パラメータ	タイプ	説明
<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. model.json 設定ファイル

model.jsonは、単一のEB GUIDEモデルに関連する設定アイテムが含まれているEB GUIDE TF設定ファイルです。

model.jsonファイルはエクスポートされたEB GUIDEモデルの一部です。

以下の表は、すべてのデフォルト設定パラメータをまとめたものです。

注記



JSONオブジェクト表記

EB GUIDE Studioでmodel.jsonを設定する場合は、JSONオブジェクト表記を使用します。

例えば、[12.6.1「でのサンプルmodel.json」](#)をご覧ください。

JSON形式の詳細については、<http://www.json.org>をご覧ください。

表12.82 共通

設定アイテム	タイプ	説明	デフォルト値
gtf.eventsystem.maxQueue	整数	イベントキューの最大サイズ	0
gtf.model.traces	ブール値	f:trace_stringスクリプト関数のトレースを有効にします	true

設定アイテム	タイプ	説明	デフォルト値
<code>gtf.model.identifier</code>	文字列	EB GUIDEモデルの一意の識別子(EB GUIDE StudioプロジェクトのUUIDと同等)	空
<code>gtf.model.identifier.short</code>	整数	EB GUIDEモデルの短い識別子	<code>0xdeadbeaf</code>

表12.83 ファイルとパス

設定アイテム	タイプ	説明	デフォルト値
<code>gtf.model.path</code>	文字列	EB GUIDEモデルへのパス	None
<code>gtf.model.config</code>	文字列	EB GUIDEモデル設定へのフルパス	<code><gtf.model.path>/model.json</code>
<code>gtf.datapool.descriptionFile</code>	文字列	データプール記述ファイルの名前	<code>datapool.gtf</code>
<code>gtf.model.files.sm</code>	文字列	ステートマシン記述ファイルの名前	<code>model.bin</code>
<code>gtf.model.files.rm</code>	文字列	リソース記述ファイルの名前	<code>resources.bin</code>
<code>gtf.model.files.views</code>	文字列	ビュー記述ファイルの名前	<code>views.bin</code>
<code>gtf.model.files.types</code>	文字列	タイプ記述ファイルの名前	<code>types.bin</code>
<code>gtf.model.pluginstoload</code>	文字列リスト	読み込むEB GUIDEモデルプラグインの名前	空の文字列リスト
<code>gtf.eventsystem.mapFile</code>	文字列	イベントシステムマッピングファイルの名前	<code>eventMap.gtf</code>

オプション`gtf.model.coreNames`は、設定されているすべてのコアの名前が含まれている文字列リストです。以下の表には、すべてのコアの設定アイテムが記されています。

表12.84 コア

設定アイテム	タイプ	説明	デフォルト値
<code>gtf.model.cores.<corename>.ownThread</code>	ブール値	独自のスレッドを使用してコアを実行するかどうかを指定します	<code>false</code>
<code>gtf.model.cores.<corename>.id</code>	整数	コアコンテキスト識別子を指定します	0

オプション`gtf.model.sceneNames`は、設定されているすべてのシーンの名前が含まれている文字列リストです。すべてのシーンには、次の表の設定アイテムがあります。

表12.85 シーン

設定アイテム	タイプ	説明	デフォルト値
<code>gtf.model.scenes.<scenename>.-visible</code>	ブール値	シーンの可視性を指定します	<code>true</code>
<code>gtf.model.scenes.<scenename>.width</code>	整数	シーンの幅	800
<code>gtf.model.scenes.<scenename>.-height</code>	整数	シーンの高さ	480
<code>gtf.model.scenes.<scenename>.x</code>	整数	シーンの始点の座標	0
<code>gtf.model.scenes.<scenename>.y</code>	整数	シーンの始点の座標	0
<code>gtf.model.scenes.<scenename>.-projectName</code>	文字列	操作しているプロジェクトの名前	
<code>gtf.model.scenes.<scenename>.-windowCaption</code>	文字列	表示されるウィンドウ名のテキスト	
<code>gtf.model.scenes.<scenename>.-sceneId</code>	整数	シーンの識別子	0
<code>gtf.model.scenes.<scenename>.-maxFPS</code>	整数	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。	60
<code>gtf.model.scenes.<scenename>.-hwLayerId</code>	整数	コアコンテキスト識別子を指定します	0
<code>gtf.model.scenes.<scenename>.-colorMode</code>	整数	色モードを指定します。 ▶ 1: 32ビット (RGBA8888) ▶ 2: 16ビット (RGB565) ▶ 3: 24ビット (RGB888) ▶ 4: 32ビットsRGB ▶ 5: 32ビットsRGB(エミュレーション)	1
<code>gtf.model.scenes.<scenename>.-multisampling</code>	整数	シーンのマルチサンプリングを指定します	0

設定アイテム	タイプ	説明	デフォルト値
		<ul style="list-style-type: none"> ▶ 0: マルチサンプリングなし ▶ 1: 2xマルチサンプリング ▶ 2: 4xマルチサンプリング 	
gtf.model.scenes.<scenename>.-enableRemoteFramebuffer	ブール値	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の転送が有効になります	false
gtf.model.scenes.<scenename>.-showWindowFrame	ブール値	レンダラーウィンドウフレームを表示するかどうかを指定します	true
gtf.model.scenes.<scenename>.-showWindow	ブール値	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	true
gtf.model.scenes.<scenename>.-disableVsync	ブール値	trueである場合、レンダラーの垂直同期が無効になります。	false
gtf.model.scenes.<scenename>.-showFPS	整数	使用可能な値: <ul style="list-style-type: none"> ▶ 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 	0
gtf.model.scenes.<scenename>.-renderer	文字列	使用するレンダラーの名前: DirectXRenderer OpenGLRenderer または OpenGL3Renderer	

表12.86 レンダリング共通

設定アイテム	タイプ	説明	デフォルト値
gtf.model.fontCache.width	整数	フォントキャッシュアトラステクスチャの幅	512
gtf.model.fontCache.height	整数	フォントキャッシュアトラステクスチャの高さ	512
gtf.model.fontCache.age	整数	フォントキャッシュのリフレッシュ操作の実行が必要となるまでの最大許容時間	100
gtf.model.traversalStackSize	整数	レンダラートラバーススタックサイズ(バイト単位)	32768

次の表の各設定アイテムは相互に対応しています。これは、レンダラーが同じ数のアイテムが3つのリストすべてにあると予期することを意味します。1つのリストのあるインデックスのエントリは、他のリストの同じインデックスのエントリに対応しています。

表12.87 レンダラー表示拡張機能

設定アイテム	タイプ	説明	デフォルト値
gtf.model.displayId	整数リスト	シーンの識別子	
gtf.model.maxCacheSize	整数リスト	シーンの最大テクスチャキャッシュ	
gtf.model.driverName	文字列リスト	OS シーンの固有のドライバ名。例えば、/dev/fb0	

以下の表の設定アイテムは、TextEngineコンポーネントの設定に使用します。TextEngine は、サードパーティライブラリFreeTypeに基づいています。以下のパラメータはFreeType実装に渡されます。FreeTypeの詳細については、https://www.freetype.org/freetype2/docs/reference/ft2-cache_subsystem.htmlをご覧ください。

EB GUIDE TFのフォントサイズ処理方法により、ft_sizeオブジェクトがft_faceオブジェクトと別にキャッシュされることはありません。max_sizesの値がターゲットプラットフォームのハードウェアによって制限される可能性があることを考慮してください。

表12.88 TextEngine 設定アイテム

設定アイテム	タイプ	説明	デフォルト値
gtf.model.textengine.-replacementGlyph	整数	専用のフォント文字が現在のフォントに見つからない場合に使用されるUnicode文字	0xfffd

設定アイテム	タイプ	説明	デフォルト値
gtf.model.textengine.maxFaces	整数	キャッシュされるフォントフェイスの最大量	0
gtf.model.textengine.maxSizes	整数	キャッシュされるフォントサイズの最大量	0
gtf.model.textengine.maxBytes	整数	キャッシュに使用できるメモリの最大バイト数	0
gtf.model.textengine.-enablePlainFileStream	ブール値	フォントへのアクセスに関する設定を決定します。trueの場合、プレーンファイルへのI/Oアクセスが使用されます。falseの場合、ROMマップ済みファイルへのアクセスが使用されます。	false

注記



ビットマップフォントの設定アイテム

.fntビットマップフォントで可以使用なのは、replacementGlyph設定アイテムのみです。[表 12.88「TextEngine 設定アイテム」](#)のその他の設定アイテムは、ビットマップフォントでは使用できません。

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アプローチは、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。

オプションgtf.model.touchDevicesNamesは、設定されているすべてのタッチデバイスの名前が含まれている文字列リストです。すべてのタッチデバイスで、次の表に示されている設定アイテムを使用できます。

表 12.89 タッチデバイス

設定アイテム	タイプ	説明	デフォルト値
gtf.model.-touchDevices.<deviceName>.-touchscreenType	整数	タッチデバイスタイプを定義します ▶ 0: Galaxy ▶ 1: imx WVGA ▶ 2: Mouse ▶ 3: General	3

設定アイテム	タイプ	説明	デフォルト値
		<ul style="list-style-type: none"> ▶ 4: Lilliput_889GL ▶ 5: General Multitouch ▶ 6: Lilliput with automatic calibration ▶ 7: GenericTouch Configuration 	
gtf.model.- touchDevices.<deviceName>.- displayManagerId	整数	デバイスが有効なシーンIDを指定します	0
gtf.model.- touchDevices.<deviceName>.touchId	整数	デバイスのIDを指定します	0
gtf.model.- touchDevices.<deviceName>.- minimalDistanceToMove	整数	タッチ位置の変更に反応するしきい値	0
gtf.model.- touchDevices.<deviceName>.- touchMoveRepeatTimeout	整数	タッチ位置の変更通知と次の変更通知の間の遅延	0
gtf.model.- touchDevices.<deviceName>.width	整数	タッチ可能なデバイス領域の幅	0
gtf.model.- touchDevices.<deviceName>.height	整数	タッチ可能なデバイス領域の高さ	0
gtf.model.- touchDevices.<deviceName>.x_high	整数	タッチ可能なデバイス領域の最大水平方向解像度の限度	0
gtf.model.- touchDevices.<deviceName>.y_high	整数	タッチ可能なデバイス領域の最大垂直方向解像度の限度	0
gtf.model.- touchDevices.<deviceName>.x_low	整数	タッチ可能なデバイス領域の最小水平方向解像度の限度	0

設定アイテム	タイプ	説明	デフォルト値
gtf.model.- touchDevices.<deviceName>.y_low	整数	タッチ可能なデバイス領域の最小垂直方向解像度の限度	0
gtf.model.- touchDevices.<deviceName>.- devicePath	文字列	タッチに使用されるドライバの名前。例えば、/dev/input0	

12.6.1. でのサンプルmodel.json



例12.1 での EB GUIDE Studio

```
{
  "gtf": {
    "datapool": {
      "descriptionFile": "datapool.gtf"
    },
    "eventsystem": {
      "maxQueue": 0,
      "mapFile": "eventMap.gtf"
    },
    "model": {
      "coreNames": [
        "<core_1>"
      ],
      "cores": {
        "<core_1>": {
          "ownThread": false,
          "id": 0
        }
      },
      "touchDevicesNames": [
        "<device_1>"
      ],
      "touchDevices": {
        "<device_1>": {
          "touchscreenType": 3,
          "displayManagerId": 0,
          "touchId": 0,
          "minimalDistanceToMove": 0,

```

```
        "touchMoveRepeatTimeout":0,
        "width":0,
        "height":0,
        "x_high":0,
        "y_high":0,
        "x_low":0,
        "y_low":0,
        "devicePath":""
    }
},
"displayId":[

],
"driverName":[

],
"fontCache":{
    "width":512,
    "height":512,
    "age":100
},
"maxCacheSize":[

],
"sceneNames":[
    "<scene_1>"
],
"scenes":{
    "<scene_1>":{
        "visible":true,
        "width":800,
        "height":480,
        "x":0,
        "y":0,
        "projectName":"<project_x>",
        "windowCaption":"<Displayed window name text>",
        "sceneId":0,
        "maxFPS":60,
        "hwLayerId":0,
        "colorMode":1,
        "multisampling":0,
        "enableRemoteFramebuffer":false,
        "showWindowFrame":true,
        "showWindow":true,
        "disableVsync":false,
        "showFPS":0,
        "renderer":"DirectXRenderer"
```

```
    }  
  },  
  "traces":true,  
  "traversalStackSize":32768,  
  "identifier":"",  
  "path":"<binary_folder>",  
  "config":"<gtf.model.path>/model.json",  
  "files":{  
    "sm":"model.bin",  
    "rm":"resources.bin",  
    "views":"views.bin",  
    "types":"types.bin"  
  },  
  "pluginstoload":[  
  
  ]  
}  
}
```

12.7. platform.json 設定ファイル

platform.jsonは、共通アイテムおよびプラットフォーム依存アイテムが含まれているEB GUIDE TF設定ファイルです。

platform.jsonファイルはエクスポートされたEB GUIDEモデルの一部です。

以下の表は、すべてのデフォルト設定パラメータをまとめたものです。

注記



JSONオブジェクト表記

EB GUIDE Studio内で`platform.json`を設定する場合は、JSONオブジェクト表記を使用します。

例については、[12.7.1「EB GUIDE Studioでのサンプルplatform.json」](#)をご覧ください。

JSON形式の詳細については、<http://www.json.org>をご覧ください。

表12.90 プラットフォームの設定

設定アイテム	タイプ	説明	デフォルト値
<code>gtf.servicemapper.port</code>	整数	サービスの接続ポート(例: EB GUIDE Monitor)	60000
<code>gtf.core.pluginstoload</code>	文字列リスト	読み込むコアプラグインのリスト(バイナリフォルダーの相対パス、または絶対パス)	None
<code>gtf.launcher.editmode</code>	ブール値	EB GUIDE TFがEB GUIDE Studioで実行されるかどうかを定義します。これは読み取り専用アイテムです。	false
<code>gtf.platform.config</code>	文字列	<code>platform.json</code> ファイルへのフルパス。これは読み取り専用アイテムです。	<binary_folder>/ <code>platform.json</code>
<code>gtf.framework.path</code>	文字列	GtfStartup実行可能ファイルへのパス。これは読み取り専用アイテムです。	<binary_folder>
<code>gtf.diagnostic.memory.interval</code>	整数	メモリー診断の間隔を指定します。値が0の場合、診断は非アクティブ化されます。	0
<code>gtf.ipc.role</code>	文字列	IPCノードの役割。使用可能な値はserverまたはです client	server
<code>gtf.ipc.discovery.network</code>	文字列	サーバークライアント検出メカニズムで使用するIPv4ネットワークアド	255.255.255.255

設定アイテム	タイプ	説明	デフォルト値
		レス。直接接続の場合、これはサーバーのネットワークアドレスを表します。	
gtf.ipc.discovery.port	整数	サーバークライアント検出メカニズムで使われるネットワークポート。直接接続の場合、これはサーバー設定のアイテムgtf.-servicemapper.-portと等しい必要があります。	4711
gtf.ipc.datapool.config	文字列	プロセス間通信の一部となるデータプールアイテムが含まれている設定ファイル	ipc_datapool.gtf
gtf.ipc.discovery.mode	文字列	サーバーとクライアントを接続するために使われる検出モード。使用可能なオプションは、"broadcast"、"multicast"、および"direct"です。	broadcast
gtf.ipc.client.timeout	整数	サーバーへのクライアント接続の再試行時間(ミリ秒単位)。	5000

12.7.1. EB GUIDE Studioでのサンプルplatform.json



例12.2 platform.jsonでの EB GUIDE Studio

```
{  
  "gtf": {  
    "core": {  
      "pluginstoload": [
```

```
        "TfRuntime",
        "TfService",
        "TfGui",
        "TfGUIOpenGLS20",
        "TfGUIOpenGLS3",
        "TfGUIDirectX11"
    ]
},
"servicemapper":{
    "port":60000
},
"launcher":{
    "editmode":true
},
"platform":{
    "config":"<binary_folder>/platform.json"
},
"framework":{
    "path":"<binary_folder>"
},
"diagnostic":{
    "memory":{
        "interval":0
    }
},
"ipc":{
    "role":"server",
    "discovery":{
        "network":"255.255.255.255",
        "port":4711,
        "mode":"broadcast"
    },
    "client":{
        "timeout":5000
    },
    "datapool": {
        "config": "ipc_datapool.gtf"
    }
}
}
```

12.8. シーン

表12.91 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
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: ▶ 32-bit (=1): RGBA8888 ▶ 16-bit (=2): RGB565 ▶ 24-bit (=3): RGB888 ▶ 32-bit sRGB (=4): この値は、グラフィックス処理ユニットのハードウェアサポートを使用します。 イメージウィジェットまたは[ディフューズテクスチャ]ウィジェット機能でsRGBサポートが必要な場合は、この値を使用します。 ▶ 32-bit sRGB (Emulated) (=5): 32-bit sRGBで適切な結果が得られない場合に限り、この値を使用します。
multisampling	使用可能な値:

プロパティ名	説明
	<ul style="list-style-type: none">▶ Off (= 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 <p>「マルチサンプリングの設定」もご覧ください。</p>
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: <ul style="list-style-type: none">▶ Off (=0): FPS を表示しない▶ On screen (=1): 画面にFPSを表示する▶ Console (=2): コンソールにFPSを表示する▶ Console & on screen (=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: <ul style="list-style-type: none">▶ DirectXRenderer▶ OpenGLRenderer▶ OpenGL3Renderer

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、低いほうの設定を使用してください。

ティップ



マルチサンプリングの設定はハードウェア依存

必要なマルチサンプリング設定がハードウェア側で使用できない場合は、それに関する情報をログファイルで参照できます。

12.9. 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12.92 がサポートするタッチスクリーンタイプ EB GUIDE GTF

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

12.10. ウィジェット

12.10.1. ビュー

表12.93 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
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12.94 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。
Transition 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12.95 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。
Transition 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)

プロパティ名	説明
Delay	終了アニメーションの遅延(ミリ秒)

12.10.2. 基本ウィジェット

基本ウィジェットは8つあります。

- ▶ アルファマスク
- ▶ アニメーション
- ▶ コンテナー
- ▶ 楕円
- ▶ イメージ
- ▶ インスタンスエータ
- ▶ ラベル
- ▶ 四角形

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うです。

12.10.2.1. アルファマスク

アルファマスクとは、子ウィジェットのアルファチャネル(オパシティ)をあるイメージによって制御するコンテナーウィジェットです。

表12.96 アルファマスクのプロパティ

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。

プロパティ名	説明
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
enabled	trueの場合、アルファマスクは子ウィジェットに適用されます。
image	アルファチャネル(子ウィジェットのオパシティ)を制御するイメージ
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
scaleMode	イメージの拡大縮小モード。使用可能な値: <ul style="list-style-type: none"> ▶ original size (=0) ▶ fit to size (=1) ▶ keep aspect ratio (=2)

注記



サポートされているアルファマスク用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バージョン2.0以上では、.pngファイルおよび .jpgファイルがサポートされます。RGBイメージは、グレースケールイメージに変換されてから アルファマスクとして使用されます。グレースケールイメージはそのまま使用されます。イメージのアルファチャネルは 無視されます。

アルファマスクを9-patchイメージと一緒に使用することはできません。

12.10.2.2. アニメーション

アニメーションは、ビューに従ってウィジェットの動きを定義します。アニメーションの外観を定義するには、[アニメーション]エディターで曲線を追加します。

表12.97 アニメーションのプロパティ

プロパティ名	説明
enabled	アニメーションを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
alternating	trueの場合、アニメーションは前後/双方向に繰り返し実行されます。 falseの場合、アニメーションは1つの方向のみ/単一方向に繰り返し実行されます。 繰り返し回数はrepeatで定義されます。

プロパティ名	説明
scale	アニメーション時間に乗じる際の係数
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)
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。
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。 <ul style="list-style-type: none"> ▶ 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、によってトリガーされます。 f:animation_cancel ▶ 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、によってトリガーされます。f:animation_cancel_end ▶ 4: アニメーションは最初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、によってトリガーされます。f:animation_cancel_reset

12.10.2.2.1. 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12.98 コンスタント曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
value	結果のコンスタント値
target	結果値が適用されるターゲットプロパティ

12.10.2.2.2. 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12.99 高速開始曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
start	初期値
end	最終値
target	結果値が適用されるターゲットプロパティ

12.10.2.2.3. 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12.100 低速開始曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
start	初期値
end	最終値
target	結果値が適用されるターゲットプロパティ

12.10.2.2.4. 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12.101 二次曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
acceleration	曲線の加速
velocity	結果を計算するための速度
constant	結果を計算するためのコンスタント値
target	結果値が適用されるターゲットプロパティ

12.10.2.2.5. 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12.102 正弦曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
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
frequency	ヘルツ単位の曲線の周波数
phase	ラジアン単位の角位変換
target	結果値が適用されるターゲットプロパティ

12.10.2.2.6. 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12.103 スクリプト曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
curve	結果の曲線関数
target	結果値が適用されるターゲットプロパティ

12.10.2.2.7. 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12.104 リニア曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
velocity	結果を計算するための速度
constant	結果を計算するためのコンスタント値
target	結果値が適用されるターゲットプロパティ

12.10.2.2.8. リニア補間曲線

リニア補間曲線ウィジェットは、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。

注記



リニアキー値補間曲線

3Dグラフィックファイルのインポート中に、インポートされる3Dシーンにアニメーションがある場合、リニアキー値補間整数曲線およびリニアキー値補間浮動小数点数曲線が作成されます。これらの曲線の基礎となるキー値ペアは、EB GUIDE Studioで変更できません。

表12.105 リニア補間曲線のプロパティ

プロパティ名	説明
enabled	アニメーションを実行するかどうかを定義します。
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
repeat	繰り返しの回数。0はいつまでも繰り返されることを表します
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。
target	結果値が適用されるターゲットプロパティ

12.10.2.3. コンテナ

コンテナは、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12.106 コンテナのプロパティ

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	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

12.10.2.4. 楕円

楕円は、色の付いた楕円をウィジェットの寸法と座標でビューに描画します。このウィジェットは、扇形または弧を描出するために使用することもできます。

表12.107 楕円のプロパティ

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

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。
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	楕円を塗りつぶす色
arcWidth	楕円の弧の幅
centralAngle	楕円の扇型を定義する角度
sectorRotation	楕円の扇型の回転を指定する角度

12.10.2.5. イメージ

イメージは、画像をビューに配置します。

表12.108 イメージのプロパティ

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	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
sRGB	このプロパティが有効になっている場合、imageで選択されているイメージはsRGB色空間を使用してレンダリングされます。 sRGB機能を使用するには、プロジェクトセンターの[設定] > [プロファイル]にあるcolorModeプロパティで、32-bit sRGB (=4) または32-bit sRGB (Emulated) (=5) を選択します。
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バージョン2.0以上では、.pngファイルおよび.jpgファイルがサポートされます。

12.10.2.6. インスタシエータ

インスタンスエータは、ランタイムにウィジェットインスタンスを作成します。インスタンスエータを使用して、動的コンテンツまたは静的コンテンツを持つリストまたは表をモデル化できます。インスタンス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。デフォルトでは、インスタンスエータは最初のラインテンプレートのみをインスタンス化します。

表 12.109 インスタンスエータのプロパティ

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	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ウィジェットの数。numItemsが0の場合、子ウィジェットは作成されません。
lineMapping	子ウィジェットとラインテンプレートのラインの関連付けを定義します。つまり、インスタンス化の順序を定義します。

12.10.2.7. ラベル

ラベルは、テキストをビューに配置します。

表 12.110 ラベルのプロパティ

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	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

12.10.2.8. 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12.111 四角形のプロパティ

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	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

12.10.3. 3Dウィジェット

12.10.3.1. 環境光

環境光とは、シーンを均一に照らす光です。環境光は材質ウィジェットのambient 色プロパティに影響を与えます。

表12.112 環境光のプロパティ

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	光の色です。
intensity	光の強度です。0.0は環境光がないことを示します。

12.10.3.2. 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12.113 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

プロパティ名	説明
fieldOfView	カメラの縦方向の視野角
projectionType	カメラの投影タイプを定義します。オブジェクトは、perspective (=0) または orthographic (=1) のどちらかの投影法でレンダリングされます。 注: 投影タイプが直交の場合、ビューボリュームはfieldOfView角を使用して計算されます。

12.10.3.3. 指向性ライト

指向性ライトは、一方向からシーンを照らします。

表12.114 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	ライトの強度です。0.0は指向性ライトがないことを示します。

12.10.3.4. 材質

材質は、Phong反射モデルを使用してメッシュ表面の外観を定義します。

表12.115 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親シーングラフに環境光が追加されない場合、このプロパティに効力はありません。
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。

12.10.3.5. 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12.116 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。

12.10.3.6. PBR GGX材質

PBR GGX材質は、物理的に正しいCook-Torranceモデルを使用してメッシュ表面の外観を定義します。

表12.117 PBR GGX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
specular	表面に光沢があるオブジェクトが反射する色
metallic	表面の質感を金属のようにする値 この値は、ディフューズの影響とスペキュラの影響の間を補間します。 0と1の間の値(0.3など)のみが有効です。
roughness	表面の質感をざらざらにする値 この値は、表面の微細構造を制御します。 0と1の間の値(0.3など)のみが有効です。
opacity	オパシティ値 0と1の間の値(0.3など)のみが有効です。

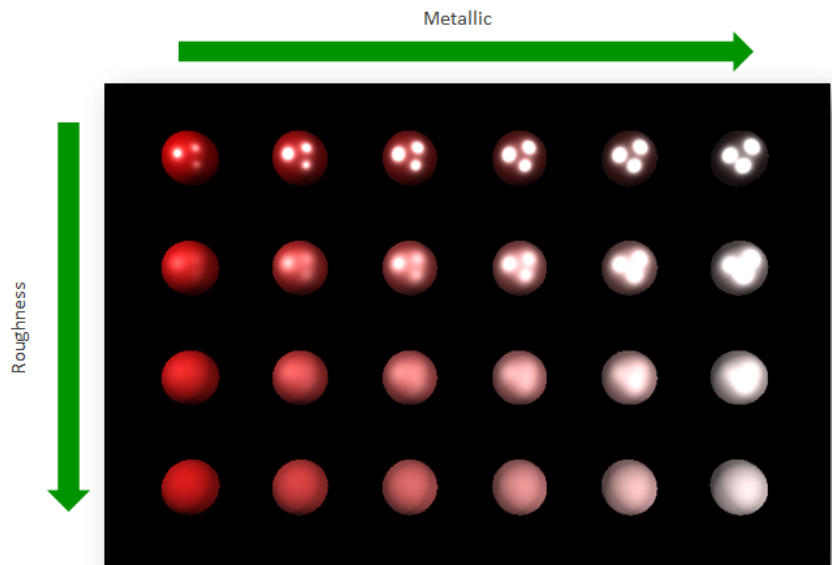


図12.1 物理ベース材質のサンプル

12.10.3.7. PBR Phong材質

PBR Phong材質は、物理的に正しいPhong反射モデルを使用してメッシュ表面の外観を定義します。

表12.118 PBR Phong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
metallic	表面の質感を金属のようにする値 この値は、ディフューズの影響とスペキュラの影響の間を補間します。 0と1の間の値(0.3など)のみが有効です。
opacity	オパシティ値 0と1の間の値(0.3など)のみが有効です。

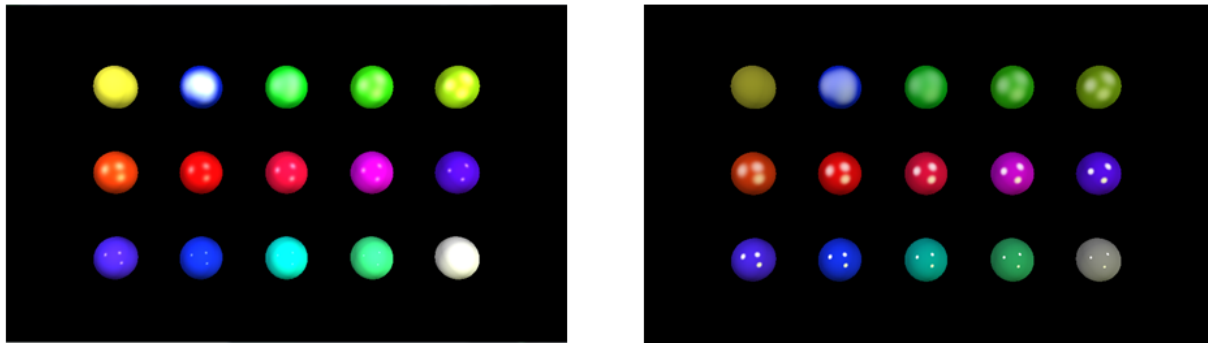


図12.2 正規化されていない材質(左)と正規化された材質(右)のサンプル

12.10.3.8. 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12.119 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	ライトの強度です。0.0は点ライトがないことを示します。
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
attenuationQuadratic	距離が離れるにつれライトを減退させる二次要素

12.10.3.9. 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12.120 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

プロパティ名	説明
y	親ウィジェットを基準にしたウィジェットのY座標

12.10.3.10. 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。次の3Dウィジェットをシーングラフノードに追加できます。

- ▶ 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 12.121 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
rotationZ	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

12.10.3.11. 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 12.122 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。

プロパティ名	説明
color	ライトの色
intensity	ライトの強度です。0.0はスポットライトがないことを示します。
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
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

12.11. 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モデルで使用方法に関する簡単な説明が含まれます。

12.11.1. 共通

12.11.1.1. 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12.123 [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。

12.11.1.2. 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 12.124 [有効]ウィジェット機能のプロパティ

プロパティ名	説明
<code>enabled</code>	<code>true</code> 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。

12.11.1.3. 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 12.125 [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
<code>focusable</code>	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: <ul style="list-style-type: none">▶ <code>not focusable (=0)</code>▶ <code>only by touch (=1)</code>▶ <code>only by key (=2)</code>▶ <code>focusable (=3)</code>
<code>focused</code>	<code>true</code> である場合、ウィジェットがフォーカスされます。

12.11.1.4. 複数行

[複数行]ウィジェット機能を使用すると、改行が可能になります。

制限:

- ▶ [複数行]ウィジェット機能は、ラベルウィジェットにのみ使用できます。

表 12.126 [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
<code>lineGap</code>	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がります。 <code>line gap</code> が小さ過ぎる(大きい負の値)場合、効果はなくなり、テキストは1行でレンダリングされます。これは、例えば、フォントスタイルが <code>PT_Sans_Narrow</code> 、サイズが30に設定されていて、 <code>line gap</code> が-50と定義されている場合に発生します。
<code>maxLineCount</code>	表示行の最大数。0 = 制限なし

ティップ



使用されている行数

スクリプト関数`getLineCount`を使用すると、テキストの行数を取得できます。

詳しくは、[12.4.3.2.15「getLineCount」](#)をご覧ください。

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。

12.11.1.5. 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。

制限:

- ▶ [押下]ウィジェット機能を追加すると、[フォーカス]ウィジェット機能が自動的に追加されます。

表12.127 [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。

12.11.1.6. 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12.128 [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。

12.11.1.7. 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リストをモデル化するために使用されます。リスト内では、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。

制限:

- ▶ [選択グループ]ウィジェット機能を追加すると、[選択]ウィジェット機能が自動的に追加されます。

表 12.129 [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。

12.11.1.8. 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 12.130 [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: <ul style="list-style-type: none">▶ 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。▶ 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。

12.11.1.9. 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です。

制限:

- ▶ [テキストの切り捨て]ウィジェット機能は、ラベルウィジェットにのみ使用できます。

表12.131 [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
<code>truncationPolicy</code>	1行のテキストの場合、 <code>truncationPolicy</code> 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: <ul style="list-style-type: none">▶ <code>leading (=0)</code>: テキストはテキストの先頭で置き換えられます。▶ <code>trailing (=1)</code>: 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、 <code>truncationPolicy</code> 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: <ul style="list-style-type: none">▶ <code>leading (=0)</code>: 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。▶ <code>trailing (=1)</code>: 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。
<code>truncationSymbol</code>	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

12.11.1.10. 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12.132 [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
<code>touchable</code>	<code>true</code> である場合、ウィジェットはタッチ入力に反応します。
<code>touched</code>	<code>true</code> である場合、ウィジェットは現在タッチされています。
<code>touchPolicy</code>	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: <ul style="list-style-type: none">▶ <code>Press then react (=0)</code>: 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。▶ <code>Press and grab (=1)</code>: 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。

プロパティ名	説明
	<ul style="list-style-type: none">▶ <code>Press then react on contact (=3)</code>: 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。
<code>touchBehavior</code>	<p>タッチ評価を定義します。使用可能な値:</p> <ul style="list-style-type: none">▶ <code>Whole area (=0)</code>: 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。▶ <code>Visible pixels (=1)</code>: 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。 <p>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。</p> <p><code>Visible pixels</code>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。</p>

[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`プロパティに`Whole area (=0)`を設定します。EB GUIDE GTF では、`Visible pixels (=1)`より速く`Whole area (=0)`が評価されます。

12.11.2. 効果

12.11.2.1. 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。

制限:

- ▶ 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12.133 [枠]ウィジェット機能のプロパティ

プロパティ名	説明
<code>borderThickness</code>	ピクセル単位の枠の厚さ
<code>borderColor</code>	枠のレンダリングに使用する色

プロパティ名	説明
borderStyle	枠のレンダリングに使用するスタイル

12.11.2.2. 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12.3 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12.134 [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	この配色に使用する色

12.11.2.3. ストローク

[ストローク]ウィジェット機能は、設定可能なテキストの輪郭(ラベル枠)を有効にします。

制限:

- ▶ このウィジェット機能は、ラベルに対して使用できます。

表12.135 [ストローク]ウィジェット機能のプロパティ

プロパティ名	説明
strokeEnabled	trueである場合、ストロークが使用されます。
strokeThickness	ピクセル単位の輪郭の太さ
strokeColor	輪郭のレンダリングに使用する色

12.11.3. フォーカス

[フォーカス]ウィジェット機能カテゴリは、フォーカス管理に関連するウィジェット機能を提供します。

12.11.3.1. 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。[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。

制限:

- ▶ [自動フォーカス]ウィジェット機能では、[フォーカス]ウィジェット機能が自動的に追加されます。

表 12.136 [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
<code>focusNext</code>	フォーカスインデックスが増加する条件
<code>focusPrev</code>	フォーカスインデックスが減少する条件
<code>focusFlow</code>	フォーカスの動作は階層内で変化します。使用可能な値: <ul style="list-style-type: none">▶ <code>stop at hierarchy (=0)</code>▶ <code>wrap within hierarchy level (=1)</code>▶ <code>step up in hierarchy (=2)</code>
<code>focusedIndex</code>	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
<code>initFocus</code>	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。

12.11.3.2. 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。

制限:

- ▶ [ユーザー定義フォーカス]ウィジェット機能では、[フォーカス]ウィジェット機能が自動的に追加されます。

表12.137 [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	<p>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。</p> <ul style="list-style-type: none"> ▶ 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。 <p>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。</p>
focusPrev	フォーカスを前の子に割り当てるトリガー
focusFlow	<p>フォーカスの動作は階層内で変化します。使用可能な値:</p> <ul style="list-style-type: none"> ▶ stop at hierarchy level (=0) ▶ wrap within hierarchy level (=1) ▶ 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、リスト内の次の子を使用されます。
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

12.11.4. ジェスチャー

12.11.4.1. フリックジェスチャー

表面を素早くなでるコンタクト

制限:

- ▶ [フリックジェスチャー]ウィジェット機能を追加すると、[ジェスチャー]および[タッチ]ウィジェット機能が自動的に追加されます。

表12.138 [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

プロパティ名	説明
	反応引数は、次のとおりです。 <ul style="list-style-type: none">▶ 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/で割った値flickMaxTime▶ 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

12.11.4.2. 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

制限:

- ▶ [ホールドジェスチャー]ウィジェット機能を追加すると、[ジェスチャー]および[タッチ]ウィジェット機能が自動的に追加されます。
- ▶ [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 12.139 [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。 <ul style="list-style-type: none">▶ 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

12.11.4.3. 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

制限:

- ▶ [ロングホールドジェスチャー]ウィジェット機能を追加すると、[ジェスチャー]および[タッチ]ウィジェット機能が自動的に追加されます。
- ▶ [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12.140 [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。 <ul style="list-style-type: none">▶ 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

12.11.4.4. 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。

制限:

- ▶ [パスジェスチャー]ウィジェット機能を追加すると、[ジェスチャー]および[タッチ]ウィジェット機能が自動的に追加されます。

表12.141 [パスジェスチャー]ウィジェット機能のプロパティ

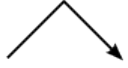
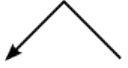




プロパティ名	説明
onPath	入力した形状が一致した場合にトリガーされる反応反応がトリガーされるのは、onPathStartがすでにトリガーされている場合のみです。反応引数は、次のとおりです。 <ul style="list-style-type: none">▶ gestureId: マッチングされたパスのID
onPathStart	コンタクトが最小ボックス(pathMinXBox、pathMinYBox)の外に移動したときにトリガーされる反応。
onPathNotRecognized	入力した形状が一致しない場合にトリガーされる反応反応がトリガーされるのは、onPathStartがす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


プロパティ名	説明
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

12.11.4.4.1. 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に含まれるサンプル設定を示しています。

表 12.142 に含まれるパスジェスチャーの設定の例 EB GUIDE

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

ID	形状	説明
6		右から左への波形状

12.11.4.5. 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

制限:

- ▶ [ピンチジェスチャー]ウィジェット機能を追加すると、[ジェスチャー]および[タッチ]ウィジェット機能が自動的に追加されます。

表 12.143 [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。 <ul style="list-style-type: none"> ▶ 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。 <ul style="list-style-type: none"> ▶ 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。 <ul style="list-style-type: none"> ▶ 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

プロパティ名	説明
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

12.11.4.6. 回転ジェスチャー

2つのコンタクトの円に沿った動き

制限:

- ▶ [回転ジェスチャー]ウィジェット機能を追加すると、[ジェスチャー]および[タッチ]ウィジェット機能が自動的に追加されます。

表 12.144 [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:

- ▶ 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

12.11.5. 入力処理

12.11.5.1. 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。

制限:

- ▶ [ジェスチャー]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。
- ▶ [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。

12.11.5.2. 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。

制限:

- ▶ [キー押下]ウィジェット機能を追加すると、[押下]および[フォーカス]ウィジェット機能が自動的に追加されます。

表 12.145 [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。 ▶ <code>keyId</code> : 処理されるキーのID

12.11.5.3. 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。

制限:

- ▶ [キーリリース]ウィジェット機能を追加すると、[押下]および[フォーカス]ウィジェット機能が自動的に追加されます。

表 12.146 [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。 ▶ <code>keyId</code> : 処理されるキーのID

12.11.5.4. 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。

制限:

- ▶ [キーのステータス変更]ウィジェット機能を追加すると、[押下]および[フォーカス]ウィジェット機能が自動的に追加されます。

表 12.147 [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。 <ul style="list-style-type: none">▶ <code>keyId</code>: 処理されるキーのID▶ <code>status</code>: ステータス変更の数値ID

12.11.5.5. 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。

制限:

- ▶ [キーUnicode]ウィジェット機能を追加すると、[押下]および[フォーカス]ウィジェット機能が自動的に追加されます。

表 12.148 [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。 <ul style="list-style-type: none">▶ <code>keyId</code>: 処理されるキーのID

12.11.5.6. 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。

制限:

- ▶ [内部に移動]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表 12.149 [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。 <ul style="list-style-type: none">▶ <code>touchId</code>: 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▶ <code>x</code>: x座標

プロパティ名	説明
	<ul style="list-style-type: none">▶ y: Y座標▶ fingerId: ウィジェット内を移動するコンタクトのID

12.11.5.7. 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。

制限:

- ▶ [外部に移動]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表 12.150 [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。 <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標▶ fingerId: ウィジェット内を移動するコンタクトのID

12.11.5.8. 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。

制限:

- ▶ [内部で移動]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表 12.151 [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。 <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標

プロパティ名	説明
	▶ <code>fingerId</code> : ウィジェット内を移動するコンタクトのID

12.11.5.9. 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によって移動できるようになります。

制限:

- ▶ [可動]ウィジェット機能を追加すると、[タッチ]および[タッチ移動]ウィジェット機能が自動的に追加されます。

表12.152 [可動]ウィジェット機能のプロパティ

プロパティ名	説明
<code>moveDirection</code>	ウィジェットが移動する方向使用可能な値: <ul style="list-style-type: none">▶ <code>horizontal</code> (=0)▶ <code>vertical</code> (=1)▶ <code>free</code> (=2)

12.11.5.10. 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。

制限:

- ▶ [回転]ウィジェット機能を追加すると、[フォーカス]ウィジェット機能が自動的に追加されます。

表12.153 [回転]ウィジェット機能のプロパティ

プロパティ名	説明
<code>rotaryReaction</code>	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。 <ul style="list-style-type: none">▶ <code>rotaryId</code>: 整数ID▶ <code>increment</code>: 入力されたイベントが送信されたときに回転入力移動する単位数

12.11.5.11. 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。

制限:

- ▶ [タッチの喪失]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表12.154 [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	<p>タッチコンタクトの喪失に対するウィジェットの反応</p> <p>反応引数は、次のとおりです。</p> <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標▶ fingerId: ウィジェット内を移動するコンタクトのID

12.11.5.12. 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。

制限:

- ▶ [タッチ移動]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表12.155 [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	<p>タッチでの移動に対するウィジェットの反応</p> <p>反応引数は、次のとおりです。</p> <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標▶ fingerId: ウィジェット内を移動するコンタクトのID

12.11.5.13. 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。

制限:

- ▶ [タッチ押下]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表 12.156 [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。 <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標▶ fingerId: ウィジェット内を移動するコンタクトのID

12.11.5.14. 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。

制限:

- ▶ [タッチリリース]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表 12.157 [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。 <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標▶ fingerId: ウィジェット内を移動するコンタクトのID

12.11.5.15. 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。

制限:

- ▶ [タッチのステータス変更]ウィジェット機能を追加すると、[タッチ]ウィジェット機能が自動的に追加されます。

表12.158 [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	<p>タッチのステータス変更に対するウィジェットの反応</p> <p>反応引数は、次のとおりです。</p> <ul style="list-style-type: none">▶ 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▶ x: X座標▶ y: Y座標▶ touchStatus: タッチのタイプのID <p>使用可能な値:</p> <ul style="list-style-type: none">▶ 0: 新しいコンタクト▶ 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: 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

12.11.6. レイアウト

12.11.6.1. 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。

制限:

- ▶ [絶対レイアウト]ウィジェット機能では、次のウィジェット機能は除外されます。
 - ▶ [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]

表12.159 [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の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。

12.11.6.2. 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。

制限:

- ▶ [ボックスレイアウト]ウィジェット機能では、次のウィジェット機能は除外されます。
 - ▶ [絶対レイアウト]
 - ▶ [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]

表12.160 [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)が配置される方向使用可能な値: <ul style="list-style-type: none">▶ horizontal (=0)▶ vertical (=1)

12.11.6.3. 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。

制限:

- ▶ [フローレイアウト]ウィジェット機能では、次のウィジェット機能は除外されます。
 - ▶ [絶対レイアウト]
 - ▶ [ボックスレイアウト]
 - ▶ [グリッドレイアウト]
 - ▶ [リストレイアウト]

表12.161 [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)が配置される方向使用可能な値: <ul style="list-style-type: none">▶ horizontal (=0)▶ vertical (=1)
horizontalChildAlign	子ウィジェットの水平方向の位置揃え使用可能な値: <ul style="list-style-type: none">▶ leading (=0): 子ウィジェットは中央に配置されます。▶ center (=1): 子ウィジェットは上に配置されます。▶ trailing (=2): 子ウィジェットは下に配置されます。
verticalChildAlign	子ウィジェットの垂直方向の位置揃え使用可能な値: <ul style="list-style-type: none">▶ 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。

12.11.6.4. 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。

制限:

- ▶ [グリッドレイアウト]ウィジェット機能では、次のウィジェット機能は除外されます。
 - ▶ [絶対レイアウト]
 - ▶ [ボックスレイアウト]
 - ▶ [フローレイアウト]
 - ▶ [リストレイアウト]

表 12.162 [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。

12.11.6.5. 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 12.163 [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

12.11.6.6. 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、[12.11.7.2「リストインデックス」](#)をご覧ください。

制限:

- ▶ [リストレイアウト]ウィジェット機能は、インスタシエータとともに使用するよう設計されています。
- ▶ [リストレイアウト]ウィジェット機能では、次のウィジェット機能は除外されます。
 - ▶ [絶対レイアウト]
 - ▶ [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]

表 12.164 [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)が配置される方向使用可能な値: <ul style="list-style-type: none"> ▶ horizontal (=0) ▶ vertical (=1)
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の外で移動できる最大値。scrollOffsetは、ユーザーがさらにスクロールしようとするとき切り捨てられます。
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:

プロパティ名	説明
	<ul style="list-style-type: none">▶ true: <code>scrollValueMin</code>または<code>scrollValueMax</code>を超えた場合、<code>scrollValue</code>プロパティは逆側に進みます。▶ false: <code>scrollValueMin</code>または<code>scrollValueMax</code>を超えた場合、<code>scrollValue</code>プロパティは増加/減少しません。

12.11.6.7. 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。

制限:

- ▶ [拡大縮小モード]ウィジェット機能は、ウィジェットイメージにのみ使用できます。

表 12.165 [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
<code>scaleMode</code>	イメージの拡大縮小モード。使用可能な値: <ul style="list-style-type: none">▶ <code>0 = original size</code>▶ <code>1 = fit to size</code>▶ <code>2 = keep aspect ratio</code>

12.11.7. リスト管理

12.11.7.1. 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、リストまたは表の各ラインの一意の位置を定義します。

制限:

- ▶ [ラインインデックス]ウィジェット機能は、インスタシエータとともに使用するよう設計されています。

表 12.166 [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
<code>lineIndex</code>	表内の現在のラインのインデックス

12.11.7.2. 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、リスト内のウィジェットの一意の位置を定義します。

制限:

- ▶ [リストインデックス]ウィジェット機能は、[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 12.167 [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

12.11.7.3. 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、使用されるラインテンプレートの一意の位置を定義します。

制限:

- ▶ [テンプレートインデックス]ウィジェット機能は、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 12.168 [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

12.11.7.4. ビューポート

[ビューポート]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。

制限:

- ▶ [ビューポート]は、コンテナまたはリストとともに使用するよう設計されています。
- ▶ [ビューポート]ウィジェット機能は、次のモデル要素に対して有効です。
 - ▶ [ビューポート]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。
 - ▶ [ビューポート]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 12.169 [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
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

12.11.8. 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。

12.11.8.1. 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。

制限:

- ▶ [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 12.170 [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(ピクセル単位)
viewportHeight	ビューポートの高さ(ピクセル単位)

12.11.8.2. 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。

制限:

- ▶ [アンビエントテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表 12.171 [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。使用可能な値: <ul style="list-style-type: none">▶ repeat (=0): テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます▶ clamp (=1): テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
ambientTextureAddressModeV	v方向のテクスチャのアドレスモード。使用可能な値:

プロパティ名	説明
	<ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
<code>ambientFilterMode</code>	<p>テクスチャのフィルタリングモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになります。 ▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。 ▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

12.11.8.3. 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。

制限:

- ▶ [ディフューズテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表12.172 [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
<code>diffuseTexture</code>	テクスチャのファイル名
<code>diffuseTextureAddressModeU</code>	<p>u方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
<code>diffuseTextureAddressModeV</code>	<p>v方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。

プロパティ名	説明
diffuseFilterMode	<p>テクスチャのフィルタリングモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになりやすくなります。 ▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。 ▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。
diffuseSRGB	<p>このプロパティが有効になっている場合、<code>diffuseTexture</code>で選択されているテクスチャはsRGB色空間を使用してレンダリングされます。</p> <p>sRGB機能を使用するには、プロジェクトセンターの[設定] > [プロファイル]にある<code>colorMode</code>プロパティで、<code>32-bit sRGB (=4)</code>または<code>32-bit sRGB (Emulated) (=5)</code>を選択します。</p>

12.11.8.4. 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。

制限:

- ▶ [エミッシブテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表12.173 [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
emissiveTextureAddressModeU	<p>U方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
emissiveTextureAddressModeV	<p>V方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。

プロパティ名	説明
emissiveFilterMode	<p>テクスチャのフィルタリングモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになりやすくなります。 ▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。 ▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

12.11.8.5. 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。

制限:

- ▶ [ライトマップテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表12.174 [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
lightMapTextureAddressModeU	<p>u方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
lightMapTextureAddressModeV	<p>v方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
lightMapFilterMode	<p>テクスチャのフィルタリングモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになりやすくなります。

プロパティ名	説明
	<ul style="list-style-type: none"> ▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。 ▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

12.11.8.6. 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。

制限:

- ▶ [ノーマルマップテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表12.175 [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
<code>normalMapTexture</code>	テクスチャのファイル名
<code>normalMapTextureAddressModeU</code>	<p>U方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
<code>normalMapTextureAddressModeV</code>	<p>V方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
<code>normalMapFilterMode</code>	<p>テクスチャのフィルタリングモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになります。 ▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。 ▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

12.11.8.7. 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。

制限:

- ▶ [不透明テクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表12.176 [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。使用可能な値: <ul style="list-style-type: none">▶ repeat (=0): テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます▶ clamp (=1): テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
opaqueTextureAddressModeV	V方向のテクスチャのアドレスモード。使用可能な値: <ul style="list-style-type: none">▶ repeat (=0): テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます▶ clamp (=1): テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
opaqueFilterMode	テクスチャのフィルタモード。使用可能な値: <ul style="list-style-type: none">▶ point (=0): テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになります。▶ linear (=1): バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。▶ trilinear (=2): 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

12.11.8.8. 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。

制限:

- ▶ [リフレクションテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表12.177 [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リングモード。使用可能な値: <ul style="list-style-type: none">▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになりやすくなります。▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

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。

12.11.8.9. 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。

制限:

- ▶ [スペキュラテクスチャ]ウィジェット機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表 12.178 [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
<code>specularTexture</code>	テクスチャのファイル名
<code>specularTextureAddressModeU</code>	<p>U方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
<code>specularTextureAddressModeV</code>	<p>V方向のテクスチャのアドレスモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>repeat (=0)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャが繰り返されます。ラップまたはタイルとも呼ばれます ▶ <code>clamp (=1)</code>: テクスチャの制限範囲の外側からアクセスされた場合、テクスチャの端のピクセルが使用されます。
<code>specularFilterMode</code>	<p>テクスチャのフィルタリングモード。使用可能な値:</p> <ul style="list-style-type: none"> ▶ <code>point (=0)</code>: テクスチャはまったく平滑化されません。最もコストが低いですが、テクスチャが最小化されたときにアーティファクトがぎざぎざになりやすくなります。 ▶ <code>linear (=1)</code>: バイリニアフィルタリングとも呼ばれます。アーティファクトがぎざぎざにならないように最小化されたときにテクスチャを平滑化します。 ▶ <code>trilinear (=2)</code>: 最もコストが高いが、リニアフィルタリングよりもよい結果となります。

12.11.8.10. トーンマッピング

[トーンマッピング]ウィジェット機能は、トーンマッピングを有効にします。トーンマッピングは、シーングラフで輝度の値を限定された範囲にマッピングする技法です。

制限:

- ▶ [トーンマッピング]ウィジェット機能は、シーングラフで使用できます。

[トーンマッピング]ウィジェット機能は、Erik Reinhard氏らによるグローバルトーンマッピング演算子を実装しています。¹

¹**Photographic tone reproduction for digital images**, Reinhard, Erik et al., "Proceedings of the 29th annual conference on Computer graphics and interactive techniques"(2002年)、267~276ページ

表12.179 [トーンマッピング]ウィジェット機能のプロパティ

プロパティ名	説明
pureWhiteLuminance	純白にマップされる最小の輝度値。ただし、0以上の値のみが有効です。

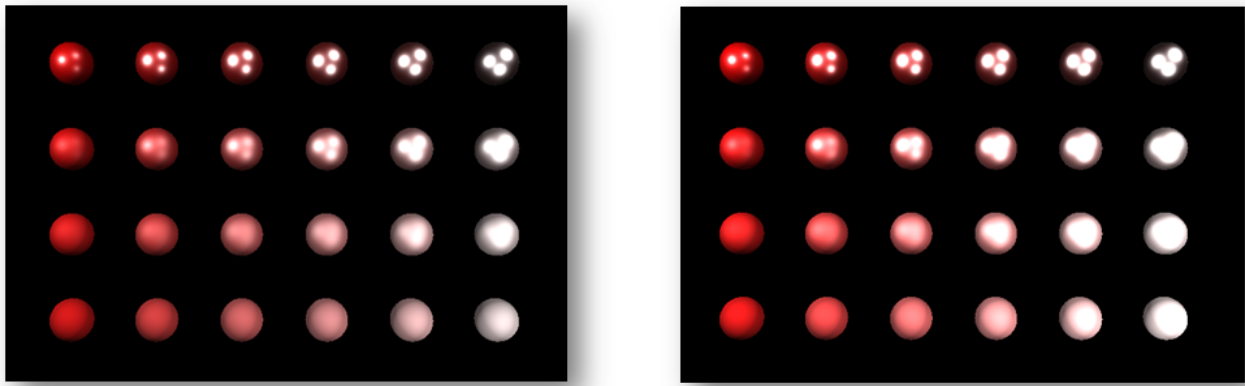


図12.3 トーンマッピングなしイメージ(左)とトーンマッピングありイメージ(右)のサンプル

12.11.9. 変換

[変形]カテゴリのウィジェット機能は、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

12.11.9.1. 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。

制限:

- ▶ [ピボット]ウィジェット機能を追加すると、[回転]、[拡大縮小]、および[せん断]ウィジェット機能が自動的に追加されます。

表12.180 [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

12.11.9.2. 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12.181 [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。

12.11.9.3. 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12.182 [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

12.11.9.4. 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 12.183 [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

12.11.9.5. 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 12.184 [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. のインストール EB GUIDE Studio

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の推奨設定

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	1920x1080ピクセル以上 モニターは2台の使用を推奨
ソフトウェア	Microsoft .NET Framework 4.7

DirectX 11

表13.2 の推奨設定 EB GUIDE SDK

ソフトウェア開発キット	Microsoft Visual Studio 2017以降
ファイル統合	CMake

13.2. ダウンロードする EB GUIDE

EB GUIDEのコミュニティエディションをダウンロードするには、<https://www.elektrobit.com/ebguide/try-eb-guide/>に移動して、手順に従います。

EB GUIDEのエンタープライズエディションをダウンロードするには、EB Commandに移動します。

注記



アカウントをアクティブにする

製品を注文すると、営業担当者から電子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 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。EB Commandからダウンロードする手順については、<https://www.elektrobit.com/support/downloading-from-eb-command/>をご覧ください。

13.3. インストールする EB GUIDE



インストールする EB GUIDE

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. をアンインストールする EB GUIDE



をアンインストールする EB GUIDE

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 がインストールされていること。
- 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ー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、対象デバイスで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ースの記述です。

EB GUIDE product line	EB GUIDE product lineは、ヒューマンマシンインターフェー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ー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ースによってヒューマンマシンインターフェー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ース(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ース

H

HMI	ヒューマンマシンインターフェース
-----	------------------

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

PBR 物理ベースレンダリング

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`model.json`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ー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 S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ース

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ー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ースとの対話処理のために使用されます。

インデックス

Symbols

- .psdファイル形式, 134
- 3Dウィジェット, 35, 59, 100
 - リファレンス, 281
- 3Dオブジェクト, 35
- 3Dグラフィック, 35, 59, 219, 329
 - インポート, 219
 - サポートされている形式, 35, 59
 - メッシュ, 59
 - 追加, 132
- せん断
 - リファレンス, 323
- アイコン
 - ヒューマンマシンインターフェース, 75
- アクション
 - エントリーアクション, 109
 - 終了アクション, 110
 - 遷移, 120
- アニメーション, 38, 100, 130, 152, 212
 - リファレンス, 273
 - 終了アニメーション, 38, 152
 - 開始アニメーション, 38, 152
- アプリケーションプログラミングインターフェイス, 39, 329
(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)
- アルファマスク, 132
 - リファレンス, 272
- アンビエントテクスチャ
 - リファレンス, 314
- イベント, 54, 71
 - コピー, 157
 - リファレンス, 255
 - 発行, 179
 - 貼り付け, 157
 - 追加, 157
- イベントシステム, 54
- イメージ
 - 9-patch, 58
 - サポートされている形式, 58
 - データタイプ, 228
 - リファレンス, 279
 - 追加, 124
- インスタンスエータ, 205
 - ラインテンプレート, 129, 279
 - リファレンス, 279
 - 追加, 129
- インポート
 - 言語依存テキスト, 177
- ウィジェット, 99, 332
 - 3Dウィジェット, 100
 - アニメーション, 100
 - グループ, 128
 - サイズの変更, 136
 - 削除, 135
 - 基本, 100
 - 追加, 123
 - 配置, 135
- ウィジェットテンプレート, 103, 154, 156
- ウィジェットテンプレートインターフェース, 103
- ウィジェットプロパティ, 101
 - EB GUIDEスクリプト, 69
 - ウィジェットテンプレート, 103
 - ウィジェットプロパティへのリンク, 138
 - ウィジェット機能プロパティ, 102
 - デフォルトプロパティ, 102
 - データプールアイテムへのリンク, 139
 - ユーザー定義プロパティ, 102, 141
 - 追加, 141
- ウィジェット機能, 101, 102, 103
 - パスジェスチャー, 202
 - 削除, 146
 - 追加, 144
- ウィンドウ表示リスト
 - データプールアイテム, 52
- エクスポート, 170
 - 言語依存テキスト, 176
- エミッシブテクスチャ
 - リファレンス, 316
- エントリーアクション, 114
 - ステートマシン, 109
- カメラ
 - リファレンス, 281

- カメラビューポート
 - リファレンス, 314
- キーUnicode
 - リファレンス, 302
- キーのステータス変更
 - リファレンス, 301
- キーリリース
 - リファレンス, 301
- キー押下
 - リファレンス, 301
- グリッドレイアウト
 - リファレンス, 309
- グローバルな名前変更, 168
- コピー
 - イベント, 157
 - データプールアイテム, 160
- コマンドエリア
 - プロジェクトエディター, 47
- コマンドライン, 75, 170, 171, 178
- コンスタント曲線
 - リファレンス, 274
- コンソール (参照 コマンドライン)
- コンテナ
 - リファレンス, 278
 - 追加, 128
- コンテンツエリア
 - プロジェクトエディター, 45
 - プロジェクトセンター, 41
- コンポーネント
 - ドッキング, 49
 - ドッキング解除, 49
- シミュレーション, 170
- ショートカット
 - ヒューマンマシンインターフェース, 75
- シーングラフ, 35, 59, 132, 219
 - テクスチャ, 219
 - リファレンス, 285
 - 追加, 132
- シーングラフノード
 - リファレンス, 286
- シーン設定
 - リファレンス, 268
- ジェスチャー, 97
 - パスジェスチャー, 97
 - リファレンス, 295, 300
 - 非パスジェスチャー, 97
- ジェスチャーID
 - リファレンス, 298
- スキン
 - サポート, 56
 - 切り替え, 149
 - 削除, 149
 - 追加, 149
- スクリプト値, 73, 161
- スクリプト曲線, 276
- ステート, 80, 110, 111, 191, 331
 - エントリーアクション, 113
 - ビューステート, 81
 - 初期ステート, 81
 - 履歴ステート, 84
 - 最終ステート, 82
 - 混合ステート, 80
 - 終了アクション, 114
 - 遷移, 115
 - 選択ステート, 83
- ステートマシン, 79, 331
 - UML 2.5記法, 95
 - UMLとの比較, 95
 - インクルードステートマシン, 79, 96
 - ステート, 80
 - ステートマシンの実行, 91
 - ハプティックスステートマシン, 79
 - ロジックスステートマシン, 79
 - 削除, 110
 - 動的ステートマシン, 79
 - 追加, 108
 - 遷移, 87
- ストローク
 - リファレンス, 293
- スピン
 - リファレンス, 290
- スペキュラテクスチャ
 - リファレンス, 320
- スポットライト

- リファレンス, 286
- タッチ
 - リファレンス, 291
- タッチのステータス変更
 - リファレンス, 306
- タッチの喪失
 - リファレンス, 304
- 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, 306
- タッチ入力 (参照 ジェスチャー)
- タッチ押下
 - リファレンス, 305
- タッチ移動
 - リファレンス, 305
- ツールボックス
 - プロジェクトエディター, 43
- ツールボックスコンポーネント
 - 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, 291
- テンプレート
 - 作成, 154
 - 使用, 155
 - 削除, 156
- テンプレートインターフェース, 154
 - プロパティの削除, 154
 - プロパティの追加, 154
- テンプレートインデックス
 - リファレンス, 313
- ディスプレイ
 - 設定, 175
- ディフューズテキストチャ
 - リファレンス, 315
- データタイプ
 - イメージ, 228
 - フォント, 228
 - ブール値, 226
 - ブール値リスト, 227
 - メッシュ, 226
 - メッシュリスト, 226
 - リスト, 229
 - 整数, 229
 - 文字列, 230
 - 条件スクリプト, 227
 - 浮動小数点数, 228
 - 色, 227
- データプール, 52, 329
- データプールアイテム, 52, 161, 329
 - インポート, 177
 - ウィンドウ表示リスト, 52
 - エクスポート, 176
 - コピー, 160
 - リスト, 160
 - リファレンス, 226
 - リンク, 163
 - 変更, 180
 - 言語サポート, 215
 - 貼り付け, 160
 - 追加, 159
- トリガー
 - 遷移, 117
- トーンマッピング
 - リファレンス, 321
- ドッキング
 - コンポーネント, 49
- ドッキング解除
 - コンポーネント, 49
- ナビゲーションエリア
 - プロジェクトセンター, 40
- ナビゲーションコンポーネント
 - プロジェクトエディター, 42
- ノーマルマップテキストチャ
 - リファレンス, 318
- パスジェスチャー, 202
 - リファレンス, 297, 298
- ビュー, 99, 332
 - リファレンス, 271
 - 追加, 122
- ビューテンプレート
 - リファレンス, 271, 271
- ビューポート
 - リファレンス, 313
- ピボット

- リファレンス, 322
- ピンチジェスチャー
 - リファレンス, 299
- フォント, 57
 - OpenTypeフォント, 57
 - TrueTypeフォント, 57
 - データタイプ, 228
 - ビットマップフォント, 57, 58
- フォーカス
 - リファレンス, 288
- フリックジェスチャー
 - リファレンス, 295
- フローレイアウト
 - リファレンス, 309
- ブール値
 - データタイプ, 226
- ブール値リスト
 - データタイプ, 227
- プロジェクトエディター, 41, 331
 - コマンドエリア, 47
 - コンテンツエリア, 45
 - ツールボックス, 43
 - ツールボックスコンポーネント, 43
 - ナビゲーションコンポーネント, 42
 - 問題検出コンポーネント, 49
- プロジェクトセンター, 40, 331
 - コンテンツエリア, 40
 - ナビゲーションエリア, 40
- プロパティコンポーネント
 - 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, 172, 331
 - 複製, 173
 - 追加, 173
- ホールドジェスチャー
 - リファレンス, 296
- ボタン
 - ヒューマンマシンインターフェース, 75
- ボックスレイアウト
 - リファレンス, 308
- マルチサンプリング, 270
- マルチタッチ入力, 98
- メッシュ, 59
 - データタイプ, 226
 - リファレンス, 282
- メッシュリスト
 - データタイプ, 226
- モデル要素, 53, 330
 - 削除, 115
- ユーザー定義フォーカス
 - リファレンス, 294
- ユーザー定義プロパティ, 141
- ライターアプリケーション, 39
- ライトマップテキストチャ
 - リファレンス, 317
- ライブラリ, 330
 - 追加, 173
- ラインインデックス
 - リファレンス, 312
- ラベル, 126
 - フォント, 126, 127
 - リファレンス, 280
 - 追加, 126
- リスト, 160
 - データタイプ, 229
 - 作成, 205
- リストインデックス
 - リファレンス, 313
- リストレイアウト
 - リファレンス, 310
- リソース, 331
 - .psdファイル形式, 60
 - 3Dグラフィック, 59
 - イメージ, 58
 - フォント, 57
 - メッシュ, 59
- リソース管理, 57
- リニア曲線, 277
- リニア補間整数, 212
- リニア補間曲線, 277
- リフレクションテキストチャ
 - リファレンス, 319
- リンク
 - ウィジェットプロパティ, 138, 139

- データプールアイテム, 163
- リーダーアプリケーション, 39
- レイアウト余白
 - リファレンス, 310
- レンダラー
 - 設定, 175
- ロングホールドジェスチャー
 - リファレンス, 296
- 不透明テキストチャ
 - リファレンス, 319
- 二次曲線
 - リファレンス, 275
- 低速開始曲線
 - リファレンス, 275
- 共有ライブラリ, 331
- 内部で移動
 - リファレンス, 303
- 内部へ移動
 - リファレンス, 302
- 内部遷移, 121
- 効果
 - ウィジェット機能, 292
- 動的ステートマシン
 - 追加, 108, 187
- 可動
 - リファレンス, 304
- 問題検出コンポーネント, 168
 - プロジェクトエディター, 49
- 四角形
 - リファレンス, 280
- 回転
 - リファレンス, 304, 323
- 回転ジェスチャー
 - リファレンス, 300
- 基本ウィジェット, 100
 - リファレンス, 272
- 変換
 - リファレンス, 324
- 外部へ移動
 - リファレンス, 303
- 子の可視性の選択
 - リファレンス, 287
- 押下
 - リファレンス, 289
- 拡大縮小
 - リファレンス, 323
- 拡大縮小モード
 - リファレンス, 312
- 指向性ライト
 - リファレンス, 282
- 整数
 - データタイプ, 229
- 文字列
 - データタイプ, 230
- 有効
 - リファレンス, 287
- 材質
 - PBR GGX材質, 283, 331
 - PBR Phong材質, 284, 331
 - リファレンス, 282, 283, 284
- 条件
 - 遷移, 118
- 条件スクリプト
 - データタイプ, 227
- 枠
 - リファレンス, 292
- 楕円
 - リファレンス, 278
- 正弦曲線
 - リファレンス, 276
- 浮動小数点数
 - データタイプ, 228
- 混合ステート, 111
- 点ライト
 - リファレンス, 285
- 環境光
 - リファレンス, 281
- 終了アクション, 114
 - ステートマシン, 110
- 終了アニメーション, 152
 - リファレンス, 271
- 絶対レイアウト
 - リファレンス, 307
- 自動フォーカス

- リファレンス, 294
- 自動非表示, 49
- 色
 - データタイプ, 227
- 複数行
 - リファレンス, 288
- 言語
 - 変更, 215
- 言語依存テキスト, 215
 - インポート, 177
 - エクスポート, 176
- 設定
 - ディスプレイ, 175
- 設定ファイル, 255, 264
- 貼り付け
 - イベント, 157
 - データプールアイテム, 160
- 通信コンテキスト, 39, 162, 329
- 遷移, 87, 115, 331
 - アクション, 119
 - トリガー, 117
 - 内部, 121
 - 条件, 118
 - 移動, 116
 - 追加, 115
- 選択
 - リファレンス, 289
- 選択グループ
 - リファレンス, 289
- 選択ステート, 112
- 配色
 - リファレンス, 293
- 開始アニメーション, 152
 - リファレンス, 271
- 高速開始曲線
 - リファレンス, 274

E

- EB GUIDE GTF, 329
- EB GUIDE GTF SDK, 329
- EB GUIDE Monitor, 50, 170, 179, 179, 180, 181, 185
 - イベント, 179

- イベントコンポーネント, 179
- コマンドライン, 185
- スクリプトコンポーネント, 181
- スタンドアロン, 185
- タブ, 50
- データプールアイテム, 180
- データプールコンポーネント, 180
- EB GUIDE product line, 329
- EB GUIDE SDK, 329
- EB GUIDE Studio, 329
- EB GUIDE Studio SDK, 329
- EB GUIDE TF, 329
- EB GUIDE プロジェクト, 53
- EB GUIDE モデル, 53, 329
 - モデル要素, 53
- EB GUIDE 拡張機能, 55
- EB GUIDESクリプト, 60, 161, 329
 - if-then-else, 66
 - L値, 64
 - R値, 64
 - Whileループ, 66
 - イベント, 71
 - ウィジェットプロパティ, 69
 - コメント, 61
 - スクリプト値, 73
 - チュートリアル, 195
 - データプールアクセス, 68
 - データ型, 62
 - ネームスペース, 61
 - リスト, 70
 - ローカル変数, 65
 - 外部関数呼び出し, 67
 - 式, 62
 - 文字列の書式設定, 72
 - 標準ライブラリ, 73
 - 識別子, 61

F

- finger ID, 98

G

- GL, 330

GUI, 330

H

HMI, 330

M

model.json, 255

O

OS, 331

P

platform.json, 264

T

todo

EB GUIDEスクリプト, 61

U

UI, 332

V

VTA, 152