Elektrobit

EB GUIDE Studio

ユーザーガイド

バージョン6.9.0.200120181101

Elektrobit Automotive GmbH
Am Wolfsmantel 46
D-91058 Erlangen
GERMANY

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

Legal notice

目次

1. 本書について

1.1. ユーザーマニュアルの対象者

この章では、EB GUIDEプロジェクトに関わる対象者と、それらの対象者が担うタスクについて説明します。

ここでは、タスクの分類と、各タスクの担当者が参照すべきドキュメントについて記載します。

以下の役割があります。

- 1.1.1「対象者: モデラー」
- 1.1.2「対象者: カスタマイズ開発者」

1.1.1. 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

1.1.2. 対象者: カスタマイズ開発者

EB GUIDEモデルをモデリングする方法や顧客固有のアプリケーションを追加する方法では、足りない機能を提供できないケースもあります。このような場合は、新しいウィジェットや専用のレンダラーが必要となります。

カスタマイズ開発者は、以下のタスクを担います。

- EB GUIDE開発チームのメンバーと第3章「サポート」を通じて連絡を取り、問題を解決できるソリューションが既にあるかどうかを確認します。
- フレームワークに従って作業し、新しい機能、EB GUIDE Studio拡張機能、あるいはEB GUIDE GTF拡張機能を開発します。
- 以下のアイテムに対して追加モジュールのコードを書きます。
  - 既存のEB GUIDE GTFモジュール(ウィジェットやシェーダーなど)
  - 既存のEB GUIDE Studio拡張機能(追加のツールバーボタンなど)

カスタマイズ開発者には以下に関する深い知識が必要です。

- EB GUIDEインターフェイス
- 中心となるモジュール同士の対話処理
- 製品のデータ構造

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。
- 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記載します。

- 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。

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

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。

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フォントで表記します。

フォント	説明	使用例
斜体	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。
太字	GUI要素およびキーボードのキー	1. [Project]ドロップダウンリストボックスでProject_Aを選択します。 2. [Enter]キーを押します。
等幅フォント (Courier)	ファイル名やディレクトリ名、および章のタイトルを示す	スクリプトの配置先:`function_name\abcdirectory`。
等幅フォント (Courier)	ユーザーの入力、コード、およびファイルのディレクトリ	`CC_FILES_TO_BUILD =(PROJECT_-PATH)\source\network\can_node.-c CC_FILES_TO_BUILD += $(PROJECT_-PATH)\source\network\can_config.c` このモジュールは`BswM_Dcm_-RequestSessionMode()`関数を呼び出します。 プロジェクト名は、`Project_Test`と入力します。
角括弧[ ]	オプションのパラメータがあるコマンド構文でオプションパラメータを示す	`insertBefore [<opt>]`
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>…]`
縦棒 |	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―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 6.9
  ```

- 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 6.9/platform/win64
  ```

- 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 6.9/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。

  例えば、次のようになります。

  ```
  C:/Documents/Projects/My_exported_model
  ```

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	**Windows 7の長いパス名** 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ヒューマンマシン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. EB GUIDEの基本

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向けのツールとプラットフォームを提供しています。EB GUIDE商品ラインは、インフォテイメントヘッドユニット、計器パネル、特定の工業アプリケーション向けにユーザーインターフェイスを開発するプロジェクトで使うことを意図しています。主に、大量生産、受注生産、プロトタイピングで使用されます。

4.1. EB GUIDE product line

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

- EB GUIDE Studio
- EB GUIDE TF
- EB GUIDE arware

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HMI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や各種組み込みハードウェアで使用できます。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。

EB GUIDE arwareは、運転の快適性を向上するためにオーグメンテッドリアリティソリューションを作成できるソフトウェアフレームワークです。

4.2. EB GUIDE Studio

4.2.1. HMI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ースを指定できます。

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. HMI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を定義します。

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

4.2.3. データの処理

HMI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HMIとアプリケーションの両方に送信され、特定の事象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ション

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 GTF(EB GUIDE Graphics Trarget 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 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、次のハードウェア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、異なるハードウェア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。

4.4. EB GUIDE arware

EB GUIDE arwareは、運転の安全性を高め、運転者に車両への信頼を与え、運転の快適性を向上するためにオーグメンテッドリアリティソリューションを作成できるソフトウェアフレームワークです。EB GUIDE arwareでは、車載GPS(全地球測位システム)とセンサーを使用して、車体の周辺にある物体を捕捉し、識別します。EB GUIDE arwareを使用すると、ヘッドアップディスプレイの視野の中で、運転者に対してリアルタイムでこれらの物体への注意を呼びかけることができます。

EB GUIDE arwareは、次の情報を処理できます。

- ADAS ECUからの物体データなど、車両センサーから提供される情報
- ADASISv3データなど、地図および案内情報プロバイダーから提供される情報
- 地図データやセンサーデータの融合などにより、上記の情報源をもとにEB GUIDE arwareで計算される情報
- EB GUIDE Studioインターフェースを通じて車載用ヘッドユニットから提供される情報

EB GUIDE arwareは2つのメインのソフトウェアパーツで構成されます。

- 車両の周辺を表す仮想モデルを作成し、複数のセンサーから得た測定値を相互に関連付け、データを推定して遅延を補正するデータフュージョン。
- 表示するインターフェース要素を決定し、それらの配置を計算し、ホストOSのグラフィックスサブシステムを使ってそれらを描画するビジュアル化サブシステム。

  EB GUIDE StudioとEB GUIDE GTFを使用して、EB GUIDE arwareのビジュアル化サブシステムを拡張し、カスタマイズすることができます。

EB GUIDE arwareは車両固有のアプリケーションと統合する必要があります。このアプリケーションは、受信したデータメッセージをEB GUIDE arwareで処理できる標準形式に変換する役割を担います。

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

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HMI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. EB GUIDEの起動

EB GUIDEの起動

前提条件:

- EB GUIDEのインストールが完了していること。

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が起動します。プロジェクトセンターが表示されます。

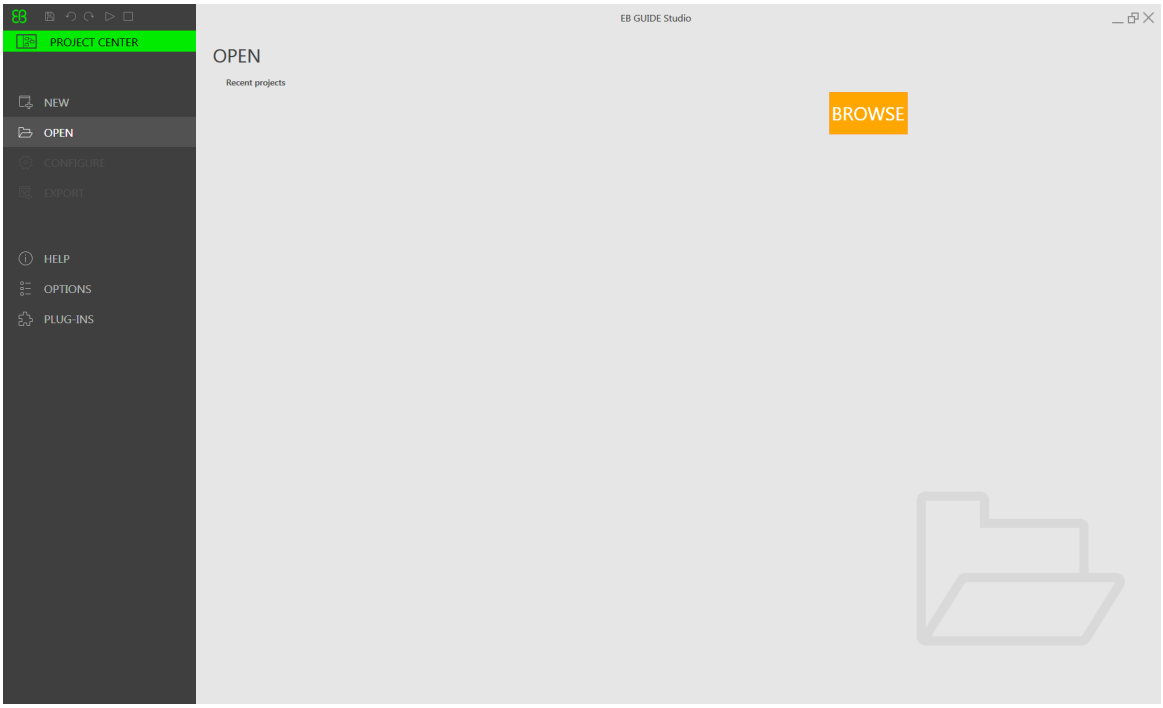

図5.1 プロジェクトセンター

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`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。

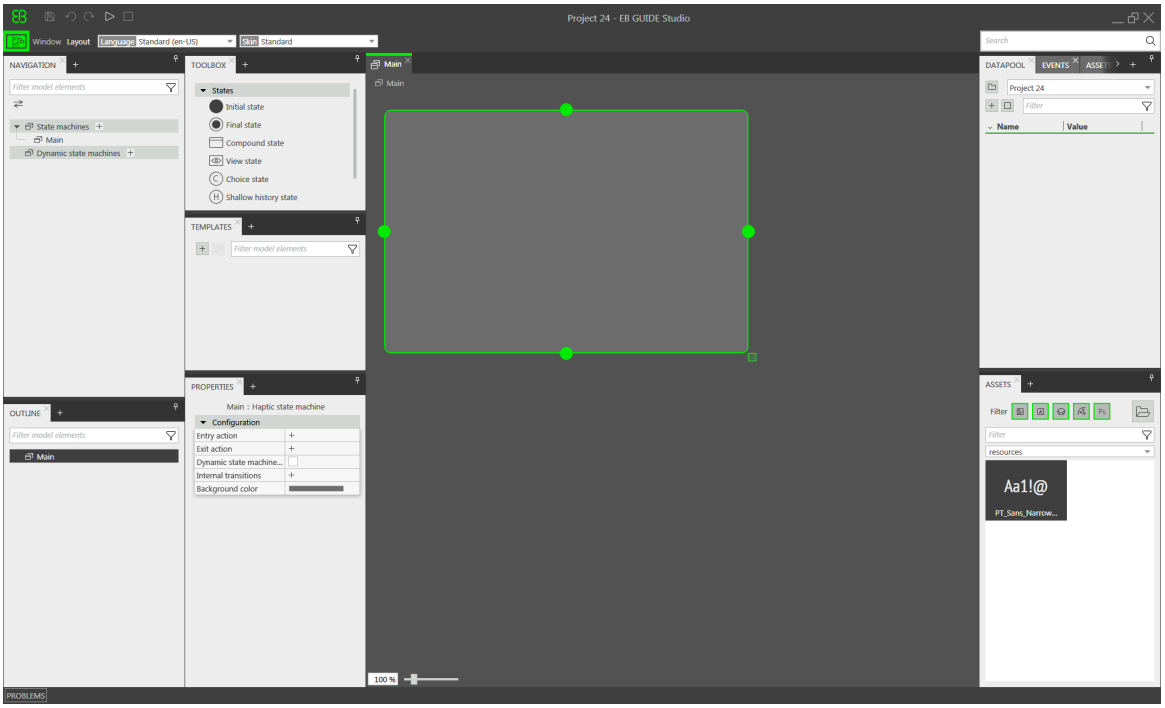

図5.2 [メイン]ステートマシンが表示されたプロジェクトエディター

5.3. HMI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。

`View state 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。

`View state 2` が追加されます。

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。

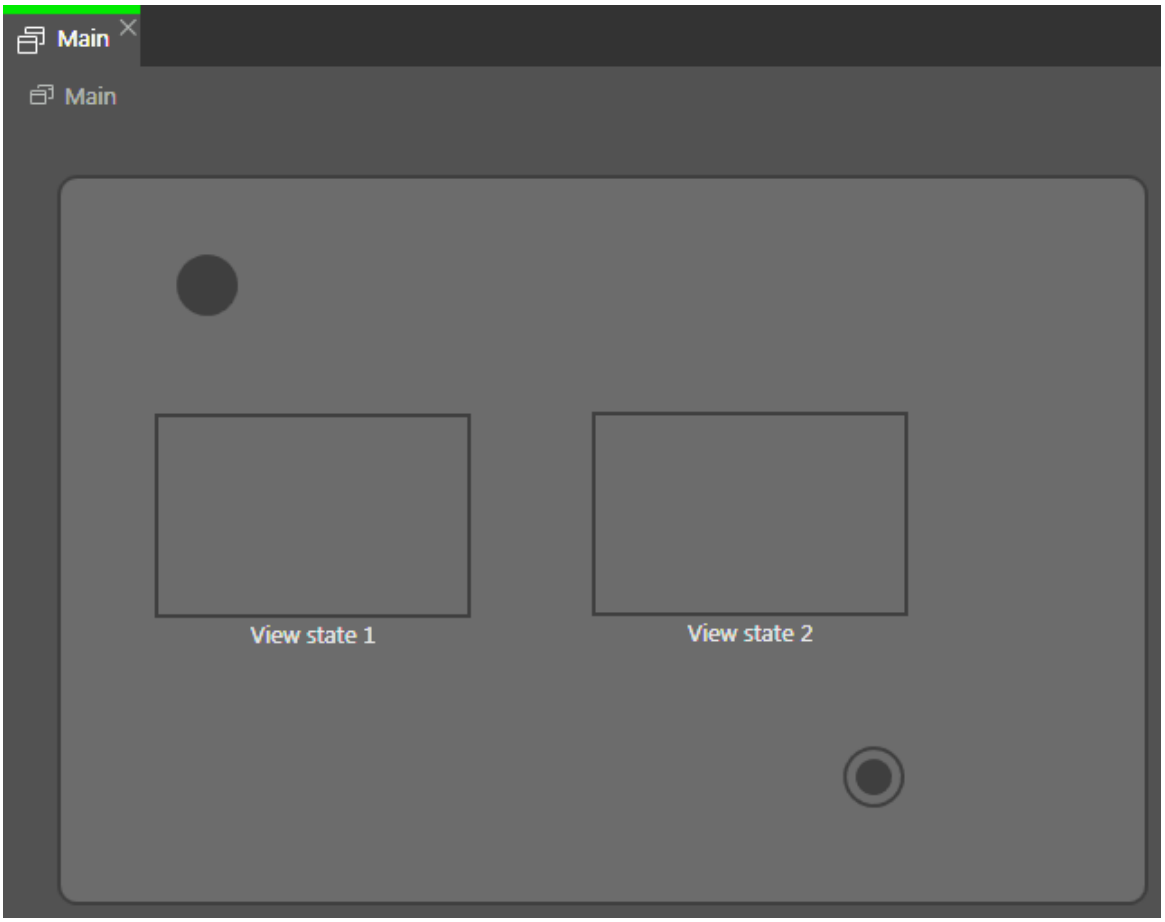

図5.3 ステートが表示されたプロジェクトエディター

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`View state 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
`View state 1`と`View state 2`の間に遷移を追加します。

`View state 1`を選択して、ターゲットステートとして`View state 2`にステップ2～4を繰り返します。

ステップ 6
`View state 1`と`View state 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。

`Event 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。`Event 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
`View state 2`と最終ステートの間に遷移を追加します。

`View state 2`を選択して、ターゲットステートとして最終ステートにステップ2～4を繰り返します。

新規イベント`Event 2`をトリガーとして追加します。

この時点で、ステートマシンは次の図のようになっています。

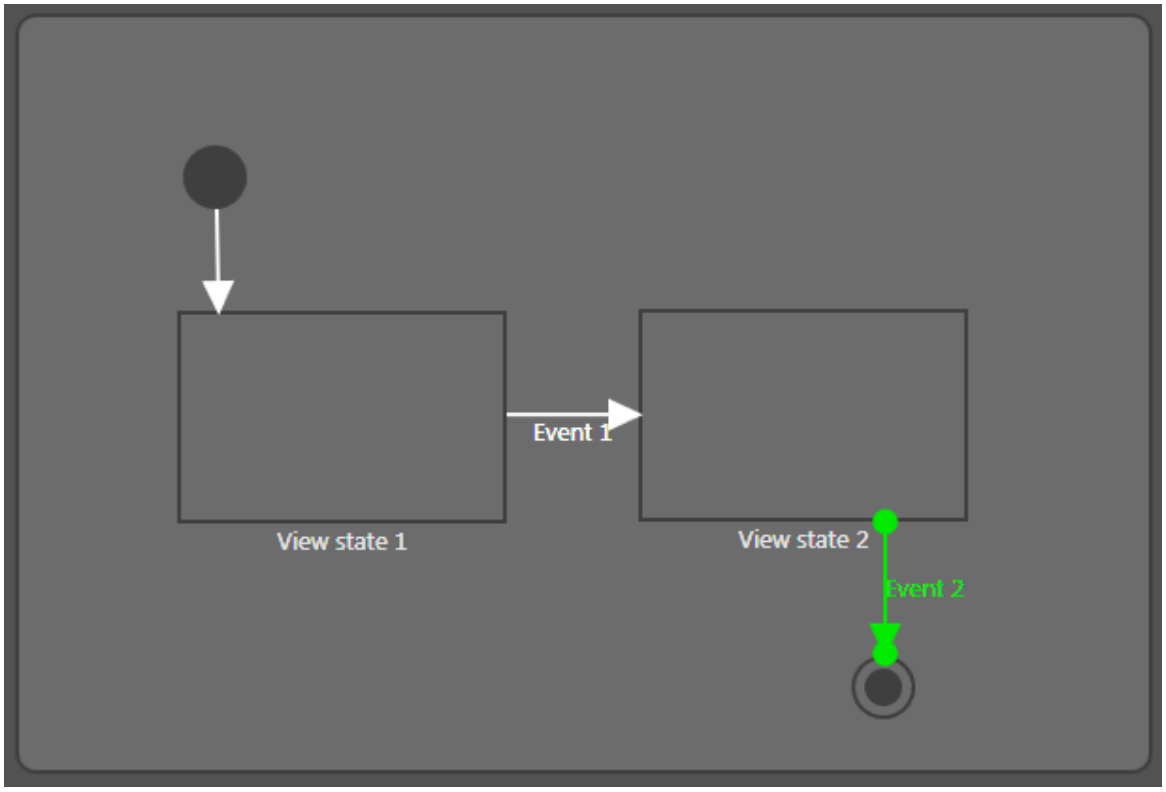

図5.4 イベントを伴う遷移でリンクされたステート

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. HMI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:

- `View state 1` がモデルに追加されていること。

ステップ 1
`View state 1`をダブルクリックします。

コンテンツエリアに`View 1`が表示されていること。

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`View 1`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
[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。

この四角形はモデル実行モードで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
    }
```

モデル実行モードで四角形にタッチすると、500ミリ秒後に`Event 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、View 2をダブルクリックします。

コンテンツエリアにView 2が表示されていること。

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 EB GUIDEスクリプトを持つウィジェットプロパティ

ステップ 12
[承認]をクリックします。

モデル実行モードで四角形にタッチすると、500ミリ秒後にEvent 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、`View 1`が表示されます。赤い四角形をクリックすると、画面が`View 2`に変化します。これは、クリックによって`Event 1`が発行され、`Event 1`によって`View state 1`から`View state 2`への遷移が実行されるためです。

次に`View 2`が表示されます。`View 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`Event 2`が発行され、 `Event 2`によって`View state 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

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 2.0以降のレンダラーに限り、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

ティップ

シーングラフのガンマ補正

`gamma`プロパティを使用すると、モニターまたはディスプレイデバイスの輝度応答に合うようにシーングラフの輝度出力を調整して、視覚に関して最適な結果を得ることができます。値は、0.-0より大きくなければならず、デフォルトの2.2に設定されています。これは、ほとんどのディスプレイに適した値です。

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

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

- 位置、法線、従法線、接線、および1つのテクスチャチャネルを含む3Dオブジェクト

- 指向性光源
- イメージベースドライトソース
- 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。この曲線は、タイプが浮動小数点数、整数、または色であるアニメーション化プロパティに適用されます。リニアキー値補間曲線は、その他のアニメーション曲線と同じように適用することはできません。この曲線は、3Dグラフィックのアニメーションをインポートする場合にのみ使用されます。

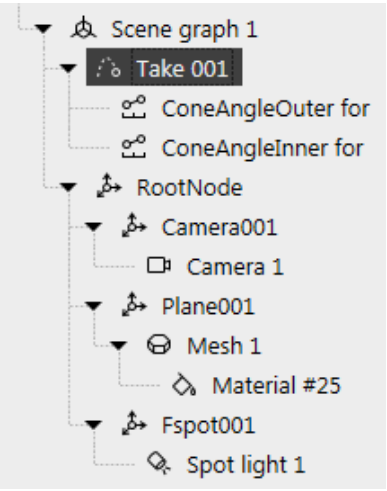

図6.1 [ナビゲーション]コンポーネントに表示されるシーングラフの例

注記

制限

次のことに注意してください。

- 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。
- EB GUIDE Studioでメッシュに複数の材質が追加されている場合は、最上位の材質ウィジェットのみがレンダリングされます。
- Blenderからエクスポートされた.fbxファイルをインポートすると、エミッシブ色が(0, 0, 0)に設定され、このシーンにあるすべてのライトが強度1.0の指向性ライトに変更されます。これが行われるのは、Blenderが必要な材質とライトの情報をエクスポートしないからです。

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、6.24「ウィジェット」、15.9.3「3Dウィジェット」、および15.10.8「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14.7「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。アニメーション機能は、ウィジェットプロパティやデータプールアイテム、あるいは色にさえ適用できます。アニメーションウィジェットは、[ツールボックス]コンポーネントの[基本ウィジェット]カテゴリにあります。

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、ウィジェットプロパティ値を徐々に変化させることを意味します。アニメーションを使うと、ウィジェットをビュー内で移動したり、ウィジェットのサイズを変更したり、ウィジェットの色を徐々に変化させたりすることができます。

アニメーションは曲線で定義されます。曲線には`target`値があります。`target`値として、ウィジェットプロパティまたはデータプールアイテムを使用できます。曲線は、`target`値の時間経過に伴う変化を表します。曲線には複数のタイプがあり、作成するアニメーション効果に適したタイプの曲線を選択できます。例えば、定数曲線、リニア補間曲線、正弦曲線があります。各アニメーションウィジェットには、1つ以上の曲線を関連付けることができます。

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`target`値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します。

アニメーションと曲線のプロパティについては、15.9.2.2「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。

独自のカスタム曲線を作成する場合は、15.9.2.2.6「スクリプト曲線」をご覧ください。詳しいバックグラウンド情報については、6.2.3「スクリプト曲線」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、ビューステート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。

ビューステートおよびビューテンプレートのView Transition Animationを定義します。ビューテンプレートを再利用するたびに、インスタンスはテンプレートのView Transition Animationを継承します。

View Transition Animationにはさまざまなタイプがあります。

表6.1 アニメーションのタイプ

アニメーションのタイプ	説明
開始アニメーション	このアニメーションは、アニメーション付きのビューステートにエントリーすると再生されます。このアニメーションで操作できるのは、追加されたビューのウィジェットプロパティとウィジェット機能プロパティのみです。
終了アニメーション	このアニメーションは、アニメーション付きのビューステートが終了すると再生されます。このアニメーションで操作できるのは、追加されたビューのウィジェットプロパティとウィジェット機能プロパティのみです。
変更アニメーション	このアニメーションはビューステートが変更されると再生されます。このアニメーションでは遷移元ビューと遷移先ビューのプロパティを操作できます。遷移元ビューは、アニメーションが追加されるビューステートです。遷移先ビューは、別のビューステートまたはビューテンプレートにすることができます。
ポップアップオンアニメーション	ビューテンプレートビューと動的ステートマシンビューにのみ使用可能です。このアニメーションは、それぞれの動的ステートマシンが有効化されると(動的ステートマシンがプッシュされると)再生されます。このアニメーションでは、追加されたビューのプロパティとウィジェットプロパティを操作できます。
ポップアップオフアニメーション	ビューテンプレートビューと動的ステートマシンビューにのみ使用可能です。このアニメーションは、それぞれの動的ステートマシンを終了すると(動的ステートマシンがポップされると)再生されます。このアニメーションでは、追加されたビューのプロパティとウィジェットプロパティを操作できます。

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、15.9.1「ビュー」をご覧ください。

手順については、8.7「ビュー遷移のアニメーション化」をご覧ください。

6.2.3. スクリプト曲線

デフォルトのアニメーション曲線には、豊富なカスタマイズオプションがすでに用意されています。また、EB GUIDE Studioではスクリプト曲線機能によって独自のアニメーション曲線を定義することもできます。この機能により、EB GUIDEスクリプトを使用した独自の曲線を定義できます。

スクリプト曲線アニメーションについての理解を深めるには、EB GUIDE Monitorで`v:diff`および`v:t_anim`の出力を取得します。手順については、8.1.2.8「スクリプト曲線によるアニメーションの追加」をご覧ください。チュートリアルについては、14.10「チュートリアル: アニメーションでのスクリプト曲線の使用」をご覧ください。

6.3. アンチエイリアス

EB GUIDE Studioでは、シーン全体またはシーングラフ単位でアンチエイリアスを有効化することができます。したがって、アンチエイリアスをグローバルに有効化または無効化できると同時に、特定のシーングラフについてアンチエイリアスを有効化して設定し、グローバルな設定を上書きすることができます。

アンチエイリアスの設定はハードウェアに依存します。要求した設定がハードウェア側で実現不可能な場合、コンソールログにエラーメッセージと、サポートされない事項に関する情報が表示されます。

アンチエイリアス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、アンチエイリアスを行うと、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。アンチエイリアスなしで開始し、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。アンチエイリアスを高くしても大きな差がない場合は、低い方の設定を使用してください。また、アンチエイリアスによる改善は、解像度の高い小型のディスプレイではわずかな差でしか現れません。

手順については、8.9「アンチエイリアスの有効化」をご覧ください。

6.4. 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ース

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ース(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ースは、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ー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ースの内容

- 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ー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ースを通じてユーザーが設定します。

6.5. 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)が自動的に割り振られます。

手順については、9.9「外部通信の確立」をご覧ください。

6.6. グラフィカルユーザーインターフェイスのコンポーネント

6.6.1. EB GUIDE Studioのグラフィカルユーザーインターフェイス

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

6.6.1.1. 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。

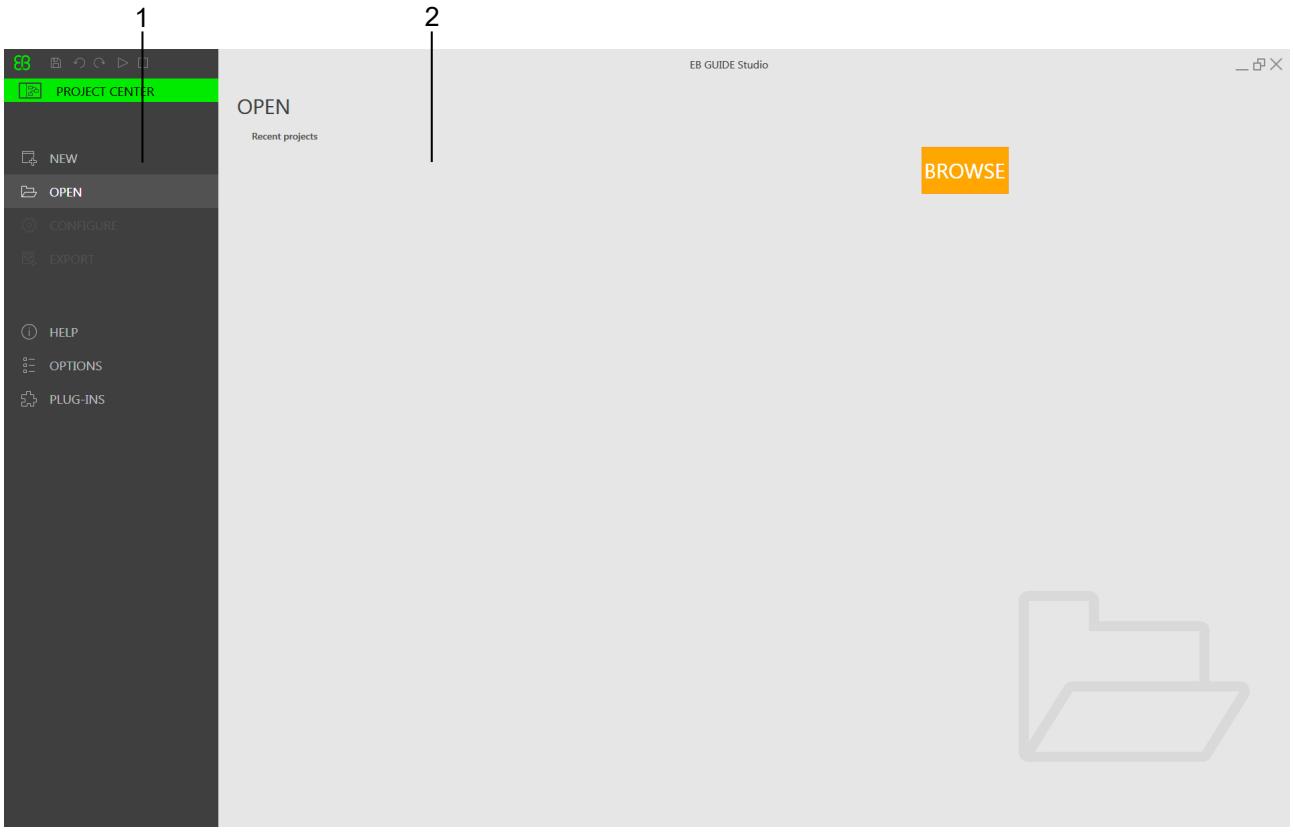

図6.2 ナビゲーションエリア(1)とコンテンツエリア(2)で構成されるプロジェクトセンター

6.6.1.1.1. 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、機能タブで構成されます。ナビゲーションエリア内のタブをクリックすると、対応する機能と設定がコンテンツエリアに表示されます。

タブに次の機能と設定があることを確認してください。

[新規]
　[新規]タブでは、新しいプロジェクトを作成できます。

[開く]
　[開く]タブでは、既存のプロジェクトを開くことができます。

[設定]
　[設定]タブでは、[プロファイル]、[スキン]などの設定を構成できます。

[エクスポート]
　[エクスポート]タブでは、EB GUIDEモデルをエクスポートできます。

[ヘルプ]
　[ヘルプ]タブには、ユーザーマニュアルへのリンクが記載されています。

[オプション]
[オプション]タブでは、EB GUIDE Studioのヒューマンマシンインターフェース言語を切り替えることができます。

[プラグイン]
[プラグイン]タブには、すべての読み込み済みのプラグインが一覧表示されます。

6.6.1.1.2. 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。

6.6.1.2. 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。

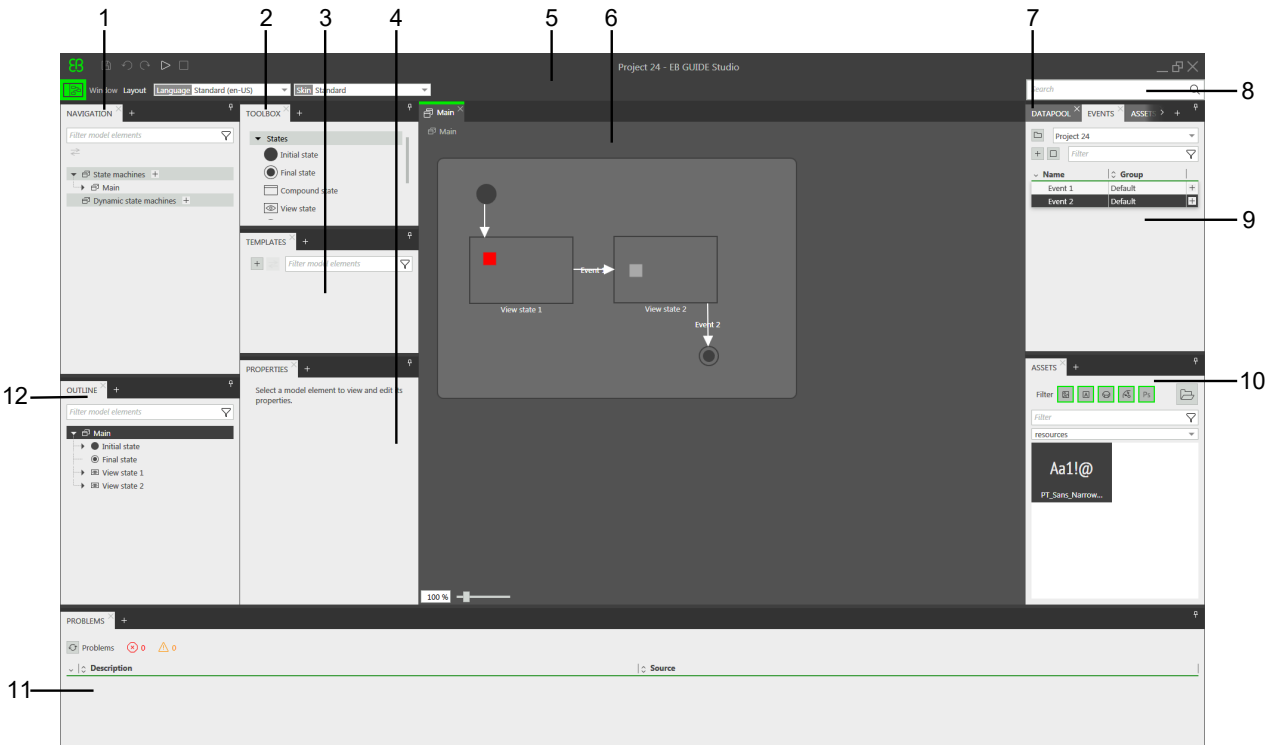

図6.3 プロジェクトエディターとそのエリアおよびコンポーネント

(1) [ナビゲーション]コンポーネント

(2) [ツールボックス]コンポーネント

(3) [テンプレート]コンポーネント

(4) [プロパティ]コンポーネント

(5) コマンドエリア

(6) コンテンツエリア

(7) [データプール]コンポーネント

(8) 検索ボックス

(9) [イベント]コンポーネント

(10) [アセット]コンポーネント

(11) [問題検出]コンポーネント

(12) [概要]コンポーネント

6.6.1.2.1. 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。

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。

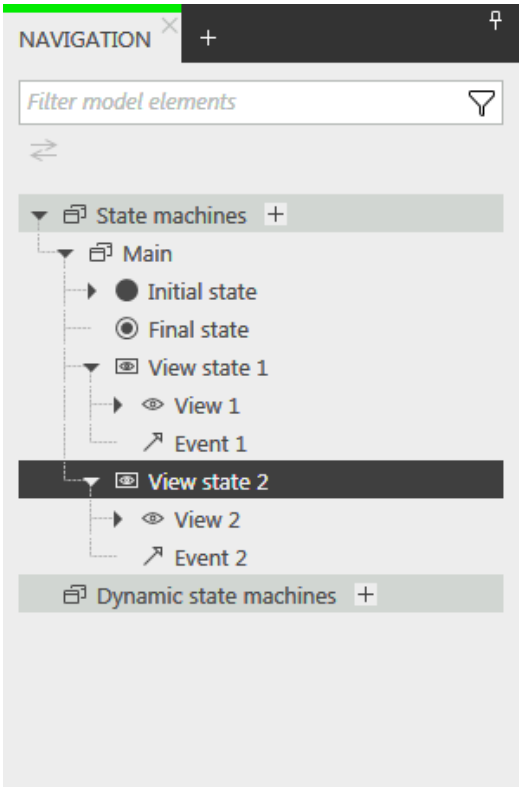

図6.4 プロジェクトエディターの[ナビゲーション]コンポーネント

6.6.1.2.2. [概要]コンポーネント

[概要]コンポーネントは、コンテンツエリアに表示中の構造とモデル要素を一覧表示します。

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。

6.6.1.2.3. 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。

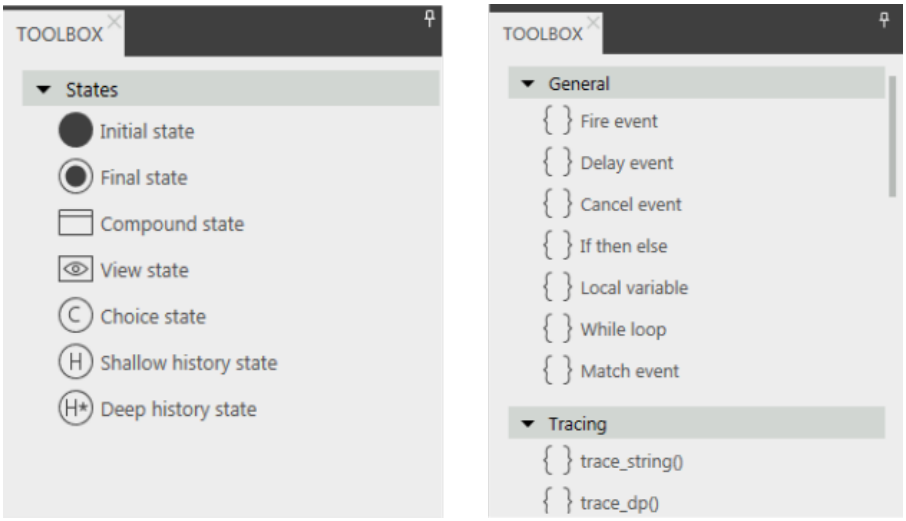

図6.5 プロジェクトエディターのツールボックス

6.6.1.2.4. 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。

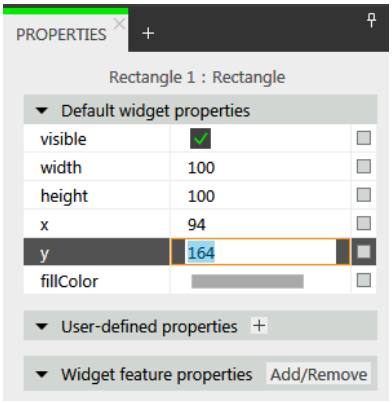

図6.6 ウィジェットのプロパティが表示された[プロパティ]コンポーネント

6.6.1.2.5. 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。

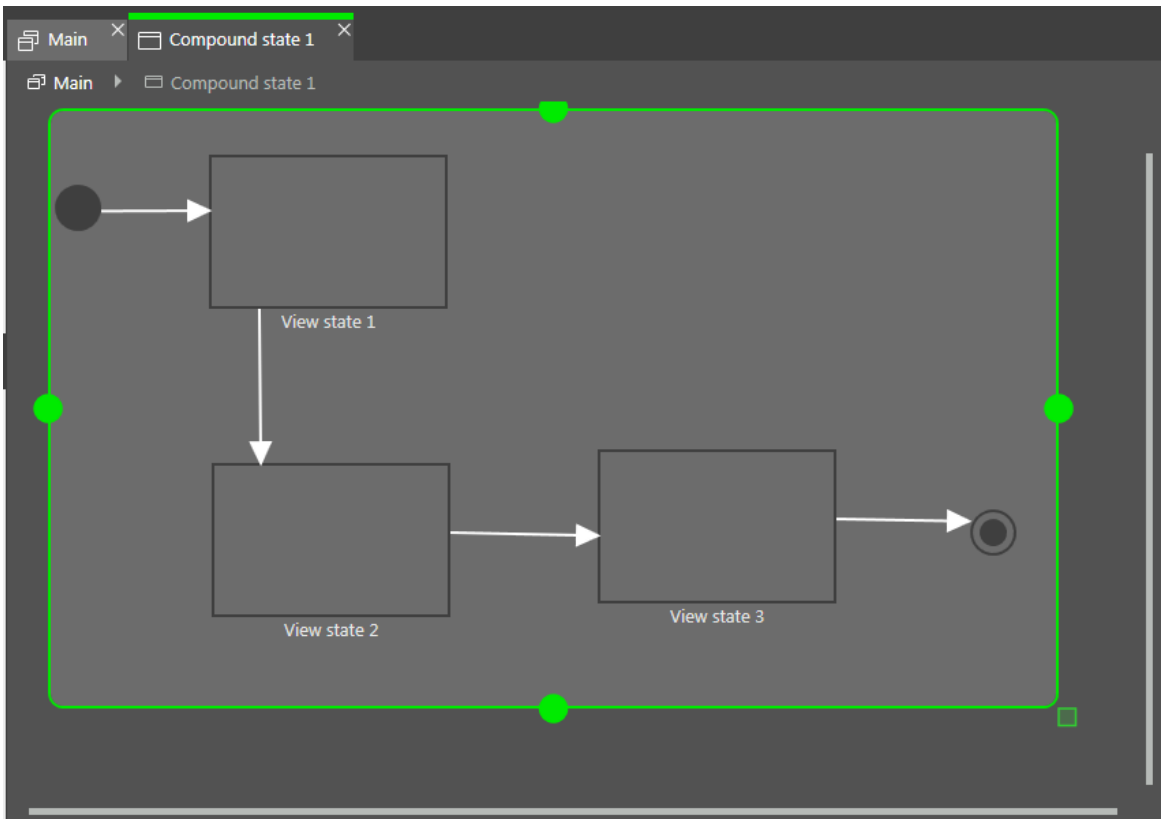

図6.7 プロジェクトエディターのコンテンツエリア

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。

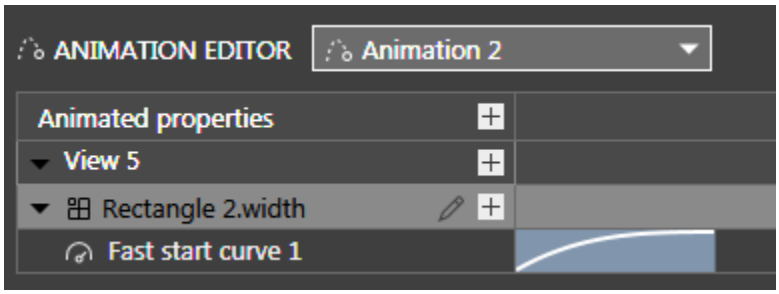

図6.8 アニメーションエディター

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。

6.6.1.2.6. [イベント]コンポーネント

ここでは、選択されている名前空間にイベントを追加したり、プロパティ(イベント表の[名前]、[グループ]、[パラメータ名]、[タイプ]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。

ティップ

モデル要素の複数選択

複数の要素を選択するには、Ctrlキーを押しながら 選択したい要素をクリックします。また、Shiftキーを押しながら、それぞれの要素をクリックしたり、 Up arrowキーまたは Down arrow キーを使用することもできます。

[イベント]コンポーネントでは、以下のビューオプションの切り替えができます。

- 選択されている名前空間のイベントのみを表示する。
- 選択されている名前空間のイベントとそのサブ名前空間を表示する。
- すべての名前空間のイベントを表示する。

複数のモデルインターフェースをEB GUIDEモデルに定義するかインポートする場合は、モデル要素を元のモデルインターフェース別にグループ化したり、グループ化解除 することができます。モデル要素は次の順序でグループ化されます。

1. 1つのモデルインターフェースに属する要素
2. 複数のモデルインターフェースに属する要素
3. インポートされたモデルインターフェースに属する要素
4. どのモデルインターフェースにも属さない要素

6.6.1.2.7. [データプール]コンポーネント

ここでは、選択されている名前空間にデータプールアイテムを追加したり、プロパティ([名前]、[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。

ティップ

モデル要素の複数選択

複数の要素を選択するには、Ctrlキーを押しながら 選択したい要素をクリックします。また、Shiftキーを押しながら、それぞれの要素をクリックしたり、Up arrowキーまたは Down arrow キーを使用することもできます。

[データプール]コンポーネントでは、以下のビューオプションの切り替えができます。

- 選択されている名前空間のデータプールアイテムのみを表示する。
- 選択されている名前空間のデータプールアイテムとそのサブ名前空間を表示する。
- すべての名前空間のデータプールアイテムを表示する。

複数のモデルインターフェースをEB GUIDEモデルに定義するかインポートする場合は、モデル要素を元のモデルインターフェース別にグループ化したり、グループ化解除 することができます。モデル要素は次の順序でグループ化されます。

1. 1つのモデルインターフェースに属する要素
2. 複数のモデルインターフェースに属する要素
3. インポートされたモデルインターフェースに属する要素
4. どのモデルインターフェースにも属さない要素

6.6.1.2.8. [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、.ebiblファイルなどのリソースを追加できます。$GUIDE_PROJECT_PATH/<project name>/resources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。

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。

6.6.1.2.9. [名前空間]コンポーネント

ここでは、名前空間を作成、移動、および削除できます。デフォルトでは、ルート名前空間が追加されています。

注記

デフォルトレイアウト

[名前空間]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開くには、メニューで[レイアウト] > [名前空間]の順に選択します。

6.6.1.2.10. 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。

  または、モデル要素を選択してF3キーを押すこともできます。.検索結果ウィンドウが開き、検索結果が表示されます。
- 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。検索結果をリフレッシュするには、 をクリックします。

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6.2 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親ステートも表示されます。
ビュー	ヒットリストには、見つかったビューの親ステートも表示されます。
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親ステートと親ウィジェットも表示されます。
イベント	プレビューには、イベントのプロパティが表示されます。
データプールアイテム	プレビューには、データプールアイテムのプロパティが表示されます。
スクリプト	プレビューには、テキストを含むスクリプトのコンテンツが表示されます。見つかったテキストは強調表示されます。
プロパティ	プレビューには、プロパティが属するウィジェットが表示されます。
表示遷移アニメーション	プレビューにより、表示遷移アニメーションが属しているビューが表示されます。

6.6.1.2.11. 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。

6.6.1.2.12. VTAコンポーネント

[VTA] (View Transition Animations)コンポーネントでは、ビューステートまたはビューテンプレートのビュー遷移アニメーションを編集できます。さまざまなアニメーションタイプを選択できます。条件がtrueの各アニメーションタイプの最初のタイプなど、関連するすべてのアニメーションタイプが同時に開始されます。

[VTA]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。[VTA]コンポーネントを開くには、[レイアウト]メニューの[VTA (View Transition Animations)]を選択します。

6.6.1.2.13. テンプレートコンポーネント

[テンプレート]コンポーネントでは、ウィジェットテンプレートを作成できます。ウィジェットをEB GUIDEモデルで再利用する場合、テンプレートは便利です。

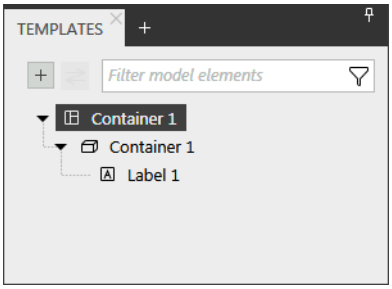

図6.9 プロジェクトエディターの[テンプレート]コンポーネント

6.6.2. EB GUIDE Monitorのグラフィカルユーザーインターフェイス

EB GUIDE Monitorでは、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。

注記

デフォルトのウィンドウレイアウト

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 Monitor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。

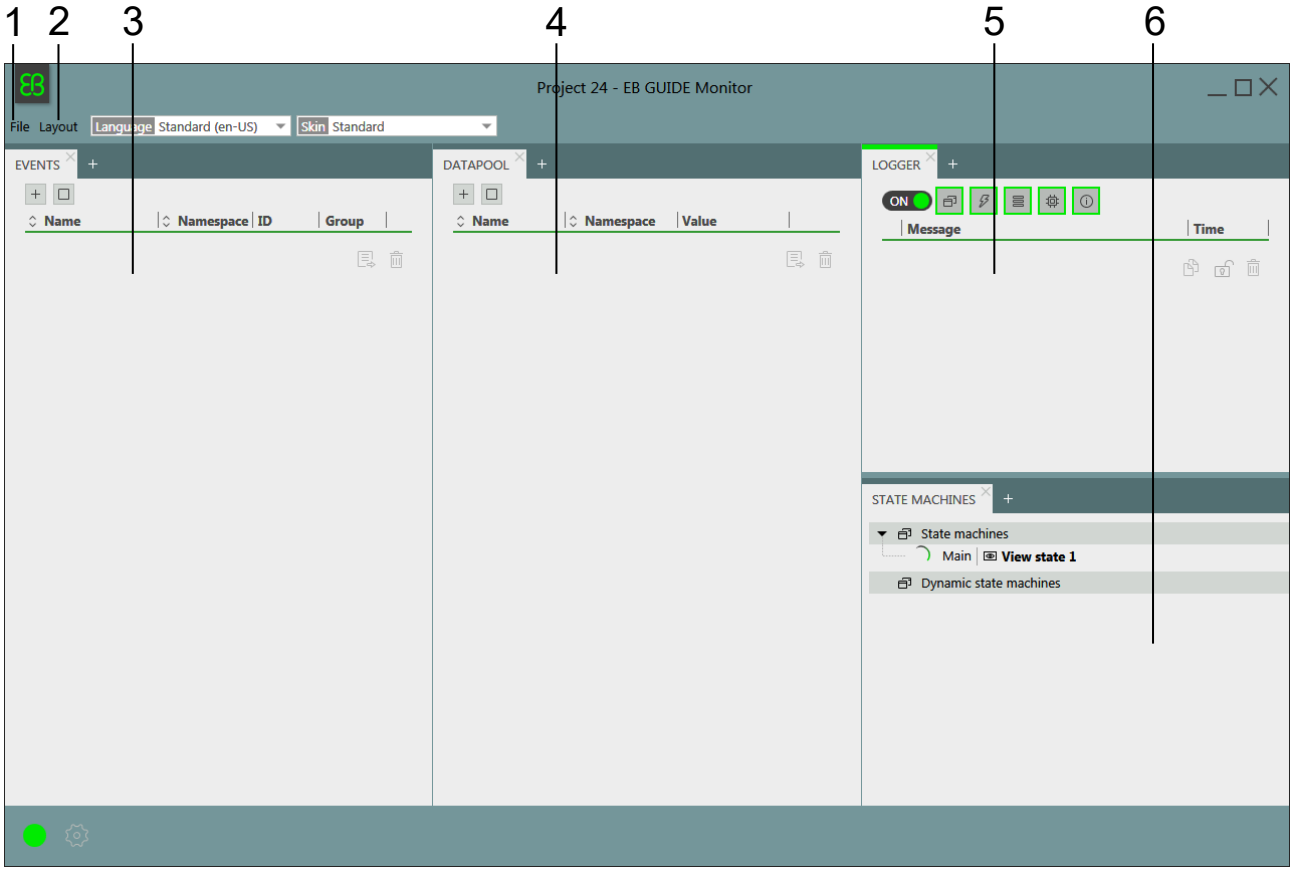

図6.10 EB GUIDE Monitorのデフォルトレイアウト

(1) [ファイル]メニュー

(2) [レイアウト]メニュー

(3) [イベント]コンポーネント

(4) [データプール]コンポーネント

(5) [ロガー]コンポーネント

(6) [ステートマシン]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、イベントを追加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追加し、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タンがあります。自動スクロール機能を変更するには、  または  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。

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。

EB GUIDE Monitorの詳細については、6.9「EB GUIDE Monitor」をご覧ください。

手順については、第11章「EB GUIDE Monitorを操作する」をご覧ください。

6.6.3. ドッキング可能なコンポーネント

EB GUIDE StudioおよびEB GUIDE Monitorのプロジェクトエディターでは、すべてのコンポーネントが、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エディ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。

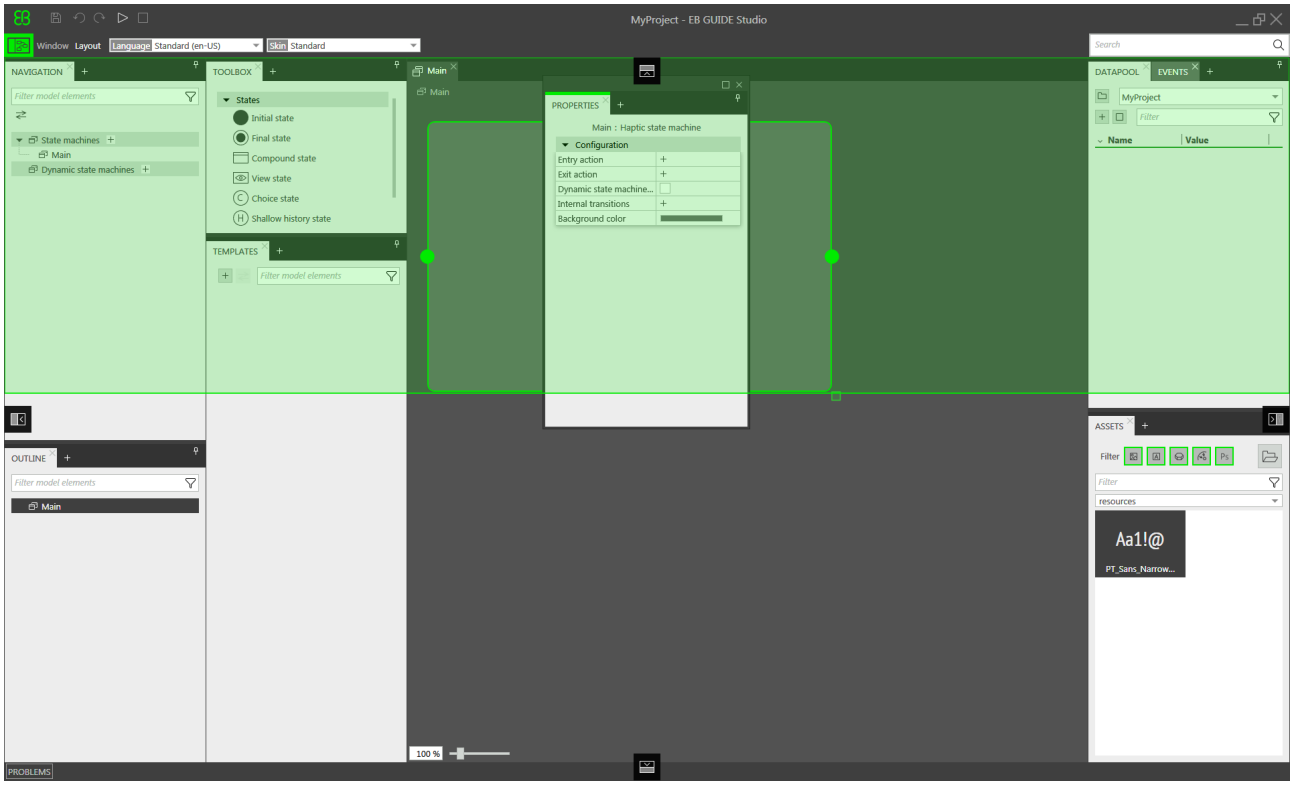

図6.11 ドッキングコントロールとライブプレビュー

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。

6.7. データプール

6.7.1. 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。

6.7.2. 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、9.6「データプールアイテムの追加」をご覧ください。

通信チャネルを開くには、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、9.9「外部通信の確立」をご覧ください。

6.7.3. 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くたびに、ヒューマンマシンインターフェイスは新しい位置をイベントによってメディアアプリケーションに非同期で送信します。メディアアプリケーションは、3つのウィンドウをリストに提供します。3つのウィンドウのそれぞれは、3つのディスプレイの1つに関連付けられています。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。

6.8. 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、EB GUIDEモデルを対象デバイスで実行するために必要な設定で構成されます。エクスポートされたEB GUIDEモデルは、対象デバイスで定義済みのモデル要素を通じて互いに通信することができます。これらのモデル要素はモデルインターフェースで定義できます。

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

6.8.1. ストレージ形式

EB GUIDEプロジェクトは、ファイル形式がEB GUIDE独自のものである複数のファイルに保存されます。このファイル形式は、次の2つのファイル拡張子で表されます。

- `.ebguide`：EB GUIDEプロジェクトファイル用
- `.gdata`：その他すべてのプロジェクトファイル用

EB GUIDE Studioのストレージ形式は、次の疑似EBNF構文で定義されます。

```
INT = [0-9]+ ;

HEXINT = '0' ( 'x' | 'X' ) [a-fA-F0-9]+

FLOAT = <as represented in the C# specification> ;

STRING = " " ; //escape characters are supported as specified in MSDN

SUFFIX = [a-zA-Z_-][a-zA-Z0-9_-]* ;

COLOR = [a-fA-F0-9]{8} ;

IDENTIFIER = ( '_' | [a-zA-Z] ) ( [a-zA-Z] | [0-9] | '_' | '$' | '.' )* ;

file = header object ;

header = 'EBGUIDE' INT '.' INT '.' INT '.' INT SUFFIX ';' ;

object = type '(' objectId ')' '{' propertyList '}' ;

type = identifier [ '<' type { ',' type } '>' ] ;

property = identifier ':' value ;

value = bool
      | int
      | float
      | string
      | color
      | object
      | externalObject
      | nullObject
      | objectReference
      | propertyReference
      | list ;

string : STRING { '\' STRING }  ;

int = [ '+' | '-' ] INT
    | HEXINT ;

color = '#' COLOR ;

float = [ '+' | '-' ] FLOAT ;
```

```
bool = 'true' | 'false' ;

externalObject = '(' objectId ')' ;

nullObject = type '(' 'none' ')' ;

objectReference = '@' objectId '(' type ')' ;

propertyReference = identifier '@' objectId '(' type ')' ;

list = type '[' [ value { ',' value } ] ']' ;

identifier = IDENTIFIER | STRING ;

objectId = GUID ; //encoded as hex digits in
                  //the XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX format
```

EB GUIDEのプロジェクトディレクトリ$GUIDE_PROJECT_PATH/<project name>/には、以下のものが含まれます。

- 以下のものに関連するファイルがあるディレクトリ:
  - 設定
  - モデルインターフェース
  - 名前空間
  - ステートマシン
  - テンプレート
  - View Transition Animation
- プロジェクト固有のリソースがあるresourceディレクトリ。詳しくは、6.18「リソース管理」をご覧ください。
- コンテキスト、データプール、イベントシステム、言語、スキンのデータが含まれる.gdataファイル。
- .xliffファイル、.psdファイル、.fbxファイルの読み込みエラー、移行またはインポートメッセージに関する情報が含まれる.txtファイル。

6.8.2. EB GUIDEプロジェクトの検証基準

EB GUIDE Studioは、EB GUIDEプロジェクトに対して2つのタイプの検証チェックを実行します。

6.8.2.1. EB GUIDEを開くときの検証

EB GUIDEプロジェクトを開くと、EB GUIDE Studioによって構造に関するいくつかの検証が実行されます。例えば、次のようなものです。

- .ebguideプロジェクトファイルが存在しない、または複数の.ebguideファイルが同じフォルダーに置かれている
- オブジェクトIDが重複している
- 子オブジェクトがEB GUIDEプロジェクト内に見つからない
- 重複するプロパティ名がある
- リストアイテムの値に一貫性がない
- .gdataファイルでのEB GUIDE Studioバージョン番号が.ebguideファイルでのEB GUIDE Studioバージョン番号と対応していない
- 不明なタイプが参照されている

これらの基準のいずれかに一致する場合は、EB GUIDEプロジェクトを開くことができず、エラーのタイプとプロジェクトファイル内でのそのエラーの位置を示すログファイル$GUIDE_PROJECT_PATH/<project name>/<project name>_LoadingErrorLog.txtが作成されます。

EB GUIDEプロジェクトを開く方法については、10.2「プロジェクトを開く」をご覧ください。

6.8.2.2. [問題検出]コンポーネントを使用した検証

EB GUIDEプロジェクトがすでに開かれている場合は、[問題検出]コンポーネントでEB GUIDEモデルを検証できます。例えば、次のようなエラーがあります。

- EB GUIDEスクリプトの使用法が無効である
- デフォルト遷移が見つからない
- アイテムのリンク先が見つからない

何らかのエラーが見つかった場合、EB GUIDEモデルのシミュレーションやエクスポートは行えません。

EB GUIDEモデルの検証方法については、10.4「EB GUIDEモデルの検証およびモデル実行」をご覧ください。

6.9. 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。エクスポートされたEB GUIDEモデルを制御したい場合は、EB GUIDE Monitorはスタンドアロンアプリケーションとして使うこともできます。

EB GUIDE Monitor GUIの詳細については、6.6.2「EB GUIDE Monitorのグラフィカルユーザーインターフェイス」をご覧ください。

手順については、第11章「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。

カスタマイズされた拡張機能を作成することにより、EB GUIDE Monitorに機能を追加して強化できます。すぐに利用できる以下のEB GUIDE使用例から、EB GUIDE Monitor拡張機能を作成する方法を習得できます:

- `MonitorRemoteViewPlugin`
- `MonitorUiExtension`
- `MonitorUiExtensionEvents`
- `MonitorUiExtensionDatapool`
- `MonitorUiExtensionTargetViewer`

EB GUIDE拡張子の使用例を以下のEB GUIDEマイクロサイトからダウンロード: https://www.elektrobit.com/ebguide/examples/。手順については、付属の`EB GUIDE Studio Tutorial Using EB GUIDE Studio examples.pdf`ファイルをご覧ください。

6.10. イベント処理

6.10.1. 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、異なるサブスクライバに事前に定義された順序で伝達することはできません。

6.10.2. 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 GTFによって使用されます。

注記

イベントIDの重複

イベントグループ内では、イベントIDが一意でなければなりません。複数のモデルインターフェースを同時にインポートすると、 別のモデルインターフェースに同一のイベントIDが存在する場合に 同じイベントグループ内でIDが重複することになり、検証エラーが発生します。 イベントIDの変更は インポート後にEB GUIDE Studioで行えないため、インポートを元に戻してから、イベントIDの変更を 元のモデルで行い、再びエクスポートとインポートを実行します。あらかじめ、イベントIDの範囲をすべての EB GUIDEモデルについて定義することをお勧めします。

例6.4
イベントの使用

回転ボタンや一連のハードキー(左、右、上、下、Enterなど)があるヒューマンマシンインターフェースでは、ユーザーが次に何を操作すべきか明確ではない場合があります。そのため、通常、このようなシステムでは現在アクティブなディスプレイ要素が強調表示されます。例えば、`YES`ボタンは色付きの枠でアクティブとマークされます。EB GUIDE Studioでは、[フォーカス]ウィジェット機能を使用してこの強調表示機能をモデル化します。現在フォーカスされている要素、つまり`focused`プロパティが`true`に設定されている要素はアクティブな要素でもあります。また、フォーカスパスを形成するこの要素の親もアクティブです。フォーカスされている要素がキー入力や回転入力を処理できない場合、入力は逆方向つまりルート要素に向かってフォーカスパスに沿って処理されます。フォーカスパスのいずれかの要素が入力を処理すると、このフォーカスは処理済みと見なされます。

タッチ入力を使用するヒューマンマシンインターフェースでは、操作は特定の位置で1つの要素を使用して行われます。例えば、タッチスクリーンの`YES`ボタンを押すとき、この入力に不明瞭な点はありません。なぜなら、ディスプレイが押された位置に基づいて、システムは`YES`ボタンで操作が行われたことを認識するからです。

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6.3 許可されるイベントグループとID

イベントグループ	ID	詳細
デフォルト	2	内部グループ、つまりシーンを実行中のコアのみがイベントを受信します。
キー入力イベント	10	次のパラメータを設定できます。 `display` 整数 入力イベントを受信するシーンID `status` 整数 0: キー押下 / 1: キーリリース / 2: キーUnicode
タッチ入力イベント	11	次のパラメータを設定できます。

<table>
<tr><th>イベントグループ</th><th>ID</th><th>詳細</th></tr>
<tr><td></td><td></td><td>
<table>
<tr><th>パラメータ</th><th>タイプ</th><th>詳細</th></tr>
<tr><td>display</td><td>整数</td><td>入力イベントを受信するシーンID</td></tr>
<tr><td>status</td><td>整数</td><td>0: タッチ押下
1: タッチ移動
2: タッチリリース
3: 近接移動
4: 新規タッチ
5: タッチ終了</td></tr>
<tr><td>x</td><td>整数</td><td>タッチイベントのx座標</td></tr>
<tr><td>y</td><td>整数</td><td>タッチイベントのy座標</td></tr>
<tr><td>fingerId</td><td>整数</td><td>マルチタッチサポートの複数の並行タッチ位置をトラッキングする番号</td></tr>
</table>
</td></tr>
<tr><td>回転入力イベント</td><td>12</td><td>次のパラメータを設定できます。
<table>
<tr><th>パラメータ</th><th>タイプ</th><th>詳細</th></tr>
<tr><td>display</td><td>整数</td><td>入力イベントを受信するシーンID</td></tr>
<tr><td>increment</td><td>整数</td><td>増分値</td></tr>
</table>
</td></tr>
<tr><td>システム通知イベント</td><td>13</td><td>Androidなどの画面回転、ライフサイクル管理、シャットダウンといったシステムイベントに使用されます。</td></tr>
</table>

以下の図に、EB GUIDE Studioでタッチ、キー、および回転イベントをモデル化する方法を示します。

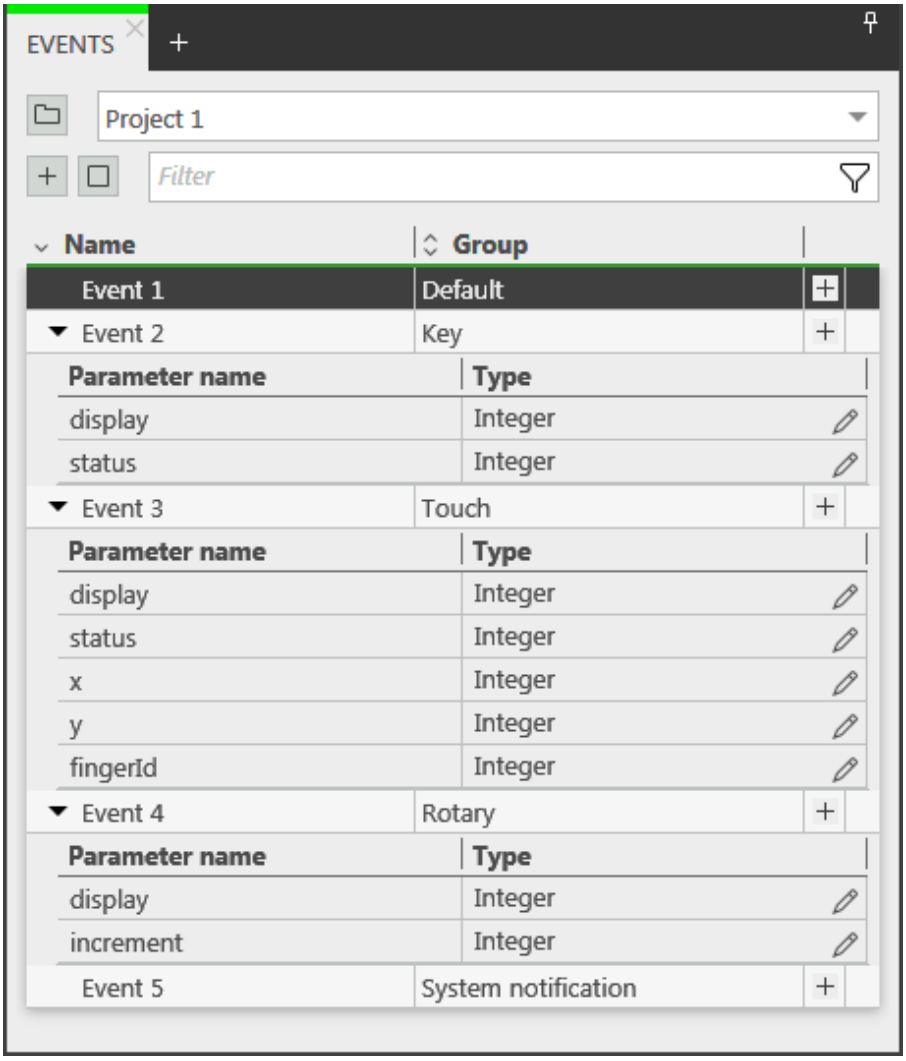

図6.12 イベントグループおよびイベントIDの例

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」

リファレンスについては、15.5「イベント」をご覧ください。

6.11. 拡張機能

6.11.1. 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。一般に、EB GUIDE Studio拡張機能は、カスタムEB GUIDEモデルまたはカスタムUI要素です。手順と詳細な説明については、第12章「EB GUIDE Studioの拡張」をご覧ください。

6.11.2. 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。

- ウィジェットの新機能
- 新しいEB GUIDEスクリプト関数

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

EB GUIDE GTF拡張機能は、サードパーティライブラリを含め、以下の場所に配置してください: `$GUIDE_PROJECT_PATH/<project name>/resources/target`

詳細および手順については、EB GUIDE GTFユーザーガイドをご覧ください。

カスタマイズされたEB GUIDE GTF拡張機能を作成することにより、EB GUIDEモデルのビジュアルと動作をカスタマイズできます。すぐに利用できるEB GUIDEの一連の使用例から、独自のEB GUIDE GTF拡張機能を作成する方法を習得できます。 EB GUIDE拡張子の使用例を以下のEB GUIDEマイクロサイトからダウンロード: https://www.elektrobit.com/ebguide/examples/。手順については、付属の`EB GUIDE Studio Tutorial Using EB GUIDE Studio examples.pdf`ファイルをご覧ください。

クラスとインターフェースの詳細については、EB GUIDE GTFアプリケーションプログラミングインターフェースに関するドキュメントをご覧ください。

6.11.3. EB GUIDE Monitor拡張機能

EB GUIDE Monitor拡張機能はEB GUIDE Monitorに追加の機能を提供します。

以下はEB GUIDE Monitor拡張機能の主な例です。

- EB GUIDE Monitorの追加コンポーネント

- シミュレーション中にスクリーンショットを作成する拡張機能

独自のカスタマイズされた拡張機能を作成できます。すぐに利用できるEB GUIDEの一連の使用例から、独自のEB GUIDE Monitor拡張機能を作成する方法を習得できます。EB GUIDE拡張子の使用例を以下のEB GUIDEマイクロサイトからダウンロード: https://www.elektrobit.com/ebguide/examples/。手順については、付属の`EB GUIDE Studio Tutorial Using EB GUIDE Studio examples.pdf`ファイルをご覧ください。

クラスとインターフェースの詳細については、EB GUIDE Monitorアプリケーションプログラミングインターフェースに関するドキュメントをご覧ください。

6.12. ガンマ補正レンダリング

6.12.1. 概念

ガンマ補正は、レンダリングパイプラインで重要な役割を果たします。ガンマ補正は、画面およびイメージカラーストレージでの色の再現に影響します。ガンマは画面の色値と知覚明度の関係を表します。ガンマの例については、図6.13「ガンマの例」をご覧ください。

人間の視覚システム(HVS)は同じ動作を示します。明るいイメージ領域の輝度差よりも暗いイメージ領域の輝度差により敏感になります。一般的な8ビットのイメージ形式(JPEG、PNG)ではこの事実を利用して非線形伝達関数を使用するsRGB色空間に色を格納し、暗いイメージ領域の精度を向上させています。これは、線形のテクスチャカラー入力に依存している3Dライティング計算とテクスチャアルファブレンディングに影響します。そのため、EB GUIDEはガンマ補正レンダリングを使用して図6.14「sRGBテクスチャの例」に示すようにこれらの影響を相殺します。

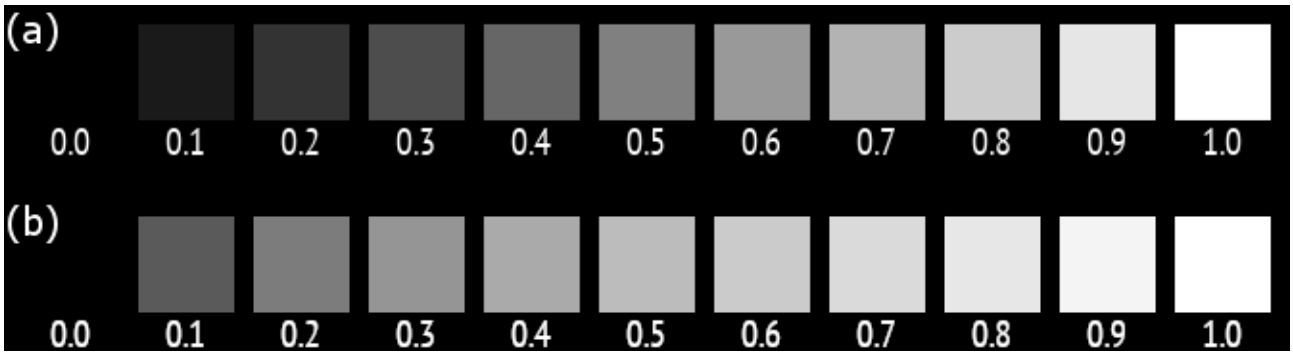

図6.13 ガンマの例

(a)色が付いた四角形の下の値はグレーレベルを示します。色値と表示輝度は非線形関係にあるため、50%の明るさは正しく較正されたモニタでおよそ0.7グレーレベルになります。画面の非線形輝度応答にかかわらず、相対的な差は知覚的に均一です。

(b)表示の前にガンマエンコーディングされる色値です。エンコーディングガンマがディスプレイのガンマを相殺するため、50%の明るさは0.5グレーレベルになります。

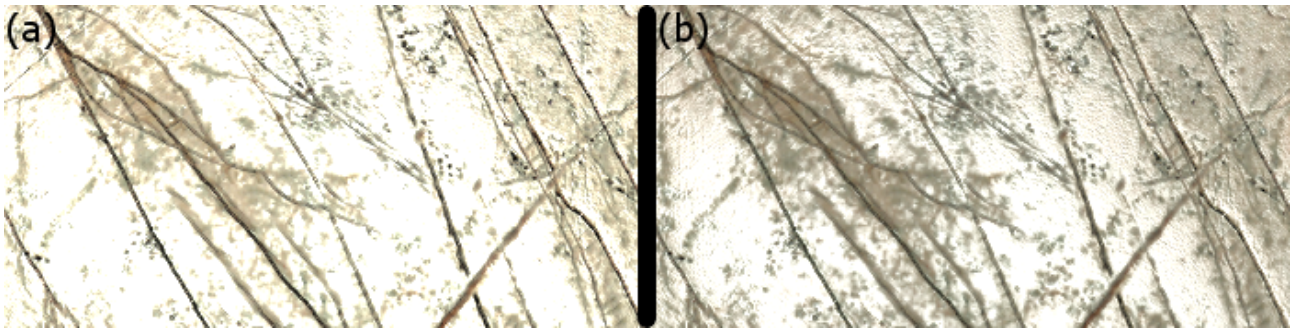

図6.14 sRGBテクスチャの例

(a) sRGBテクスチャはライティング計算で修正されず、出力はガンマ補正されません。ライティングは過飽和状態で、詳細は不鮮明になります。

(b) sRGBテクスチャはライティング前にリニアライズされ、結果がガンマ補正されます。詳細と表面構造が可視化されます。

6.12.2. EB GUIDE Studioでのガンマ補正

EB GUIDE Studioでは、ガンマ補正レンダリングで次のようにレンダリングパイプラインを設定する必要があります。

- 出力設定がディスプレイ自体のガンマエンコーディングを制御します。
- sRGBイメージとして扱うイメージおよびテクスチャリソースを入力設定でEB GUIDE Studioに示し、レンダリングパイプラインがレンダリング操作のためにそれらを正しくリニアライズするようにします。

入力エンコーディングを設定するには、使用される各イメージまたはテクスチャ向けに設定する必要があります。イメージ形式自体はsRGBエンコーディングに関する情報を提供しないことに注意してください。あらかじめこの情報を取得しておく必要があります。手順については、14.8「チュートリアル: ガンマの正しいレンダリング」をご覧ください。

6.13. イメージベースドライティング

イメージベースドライティング(IBL)は、イメージを3Dオブジェクトのライトとして使用できるようにする技法です。EB GUIDE Studioでは、イメージベースドライトウィジェットによってIBLが適用されます。このウィジェットはシーングラフノードに適用できます。1つのシーングラフに複数のイメージベースドライトを持たせることはできません。複数を追加しても、階層で最初のイメージベースドライトのみがシーンで使用されます。

図6.15 イメージベースドライティングの例左: 3点ライトで照らされたセラミックPBR GGX材質のティーポット。中央: イメージベースドライティング(IBL)を使用すると、ティーポットが仮想環境によって照らされ、セラミックPBR GGX材質が写実的になります。右: さらに、材質パラメータを空間的に変動させるためにテクスチャを使用。

IBLデータタイプの詳細については、15.3.7「IBL」をご覧ください。イメージベースドライトウィジェットの詳細については、15.9.3.4「イメージベースドライト」をご覧ください。

6.13.1. IBLGenerator、ファイル形式とインポート

ライティング情報を格納するには、ハイダイナミックレンジイメージデータをサポートするイメージ形式が必要です。EB GUIDE Studioでは、次の2つのIBL形式をサポートしています。

- Portable Float Map(.pfm)
- RGBE(.hdr)

  RGBE形式の場合、EB GUIDEではXYZ色空間をサポートしていません。-Y +X方向のみがサポートされます。

EB GUIDE Studioでこれらの IBLファイルを使用するには、`.ebibl`形式に変換する必要があります。この変換にはIBLGeneratorを使用します。IBLGeneratorは、コマンドラインによって制御され、`$GUIDE_INSTALL_PATH\tools`にあるインストール環境に含まれています。手順については、8.1.6「IBLファイルのインポート」をご覧ください。

IBLファイルには、立方体、球体、または経緯度パラメタリゼーションのいずれかでイメージを含めることができます。IBLGeneratorでは、パラメタリゼーションタイプを選択できます。IBLGeneratorに用意されているすべてのオプションを確認するには、コマンドラインに次のように入力します。`IBLGenerator.exe -h`

6.13.2. OpenGLレンダラーによるIBLの制限

OpenGL 3レンダラーは常にIBLをサポートしています。ただし、OpenGLレンダラーを使用する場合は、OpenGL ES 2.0ドライバが以下のOpenGL拡張機能をサポートしている必要があります。これらの拡張機能が1つでもサポートされていない場合、イメージベースドライトウィジェットは無視されます。

- `GL_EXT_shader_texture_lod`

- `GL_EXT_texture_rg`
- `GL_OES_texture_float`
- `GL_OES_texture_half_float`

6.14. 言語

6.14.1. EB GUIDE Studioの表示言語

EB GUIDE Studioにはグラフィカルユーザーインターフェ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。

6.14.2. 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 Studioによって提供されます。

プロジェクト設定にEB GUIDEモデルの言語を追加します。その後、テキストをエクスポートしてローカライズ担当のサービスプロバイダーに送信し、翻訳済みテキストをEB GUIDEモデルにインポートします。

言語サポートをすべてのデータプールアイテムタイプに追加することが可能です。そうすれば、EB GUIDEモデルによってテキストを異なる言語で表示できます。データプールアイテムは、各言語の値を定義します。エクスポートされるEB GUIDEモデルの言語は、ランタイムに変更できます。

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。

詳細については、8.5.1「EB GUIDEモデルへの言語の追加」および6.14.3「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。

例6.5
他言語ヒューマンマシンインターフェースの言語依存テキスト

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。これで、ヒューマンマシンインターフェースの変化に伴って変化するラベルをモデル化できます。そのためには、ラベルの`text`プロパティを、英語の値`Welcome`、ドイツ語の`Willkommen`とフランス語の`Bienvenue`という値を持つデータプールアイテムにリンクします。

手順については、14.6「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。

6.14.3. 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、翻訳、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とインポート機能を使用します。テキストは.xliffファイルにエクスポートされます。 .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。.xliffは、ローカライズ担当のサービスプロバイダーに送信でき、任意の翻訳ツールで解読できます。

変換後、翻訳済み.xliffファイルをEB GUIDEモデル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。

.xliffファイルの構成は、次のようになっています。

- ヘッダーには、ソースおよびターゲット言語に関するメタデータが含まれています。
  - source-languageおよびtarget-languageタグは、言語名の表現に関するISO 639標準規格と国コードの表現に関するISO 3166-1標準規格に従っています。
  - プロジェクトと言語のすべてペアについて、英数字からなる一意のsourcelanguageidおよびtargetlanguageidが作成されます。これらのIDは、別のプロジェクトまたはターゲット言語からの.xliffファイルの意図しないインポートを防止します。
- trans-unit要素には、ローカライズ可能なデータが含まれます。trans-unit要素は、ソーステキストを格納するsource要素と、翻訳されたテキストを格納するtarget要素を保持します。EB GUIDEモデルに新しい言語が追加されると、target要素にソース言語が設定されます。そのため、.xliffファイルのエクスポート時には、まだ翻訳されていなかったすべてのtarget要素にソース言語が表示されます。

6.15. 名前空間

EB GUIDE Studioでは、名前空間を使用して、データプールアイテムやイベントのようなモデル要素のグループを作成します。こうしたグループには通常、定義された機能があります。それぞれの名前空間では、異なる名前空間のモデル要素に同じ名前を持たせることができるように、モデル要素のネーミングスコープを作成しています。

各モデル要素は、厳密に1つの名前空間のみに属しています。

ルート名前空間は、デフォルトの名前空間であり、削除することも名前を変更することもできません。ルート名前空間はEB GUIDEプロジェクトと同じ名前を持ちます。その他すべての名前空間は、この名前空間から派生したものです。以下の場合は、モデル要素が必ずデフォルトの名前空間に追加されます。

- コンテキストメニューの[データプールアイテムへのリンクの追加]を選択し、新しいデータプールアイテムを作成する場合

- 遷移または内部遷移の[トリガー]コンボボックスでイベントを作成する場合

別の名前空間にモデル要素を移動できます。

注記

モデル要素の移動

ある名前空間からモデル要素を別の名前空間に移動し、移動先の名前空間にすでに同じ名前の要素が含まれていた場合、この移動の操作は失敗し、エラーメッセージが表示されます。

例**6.6**
名前空間ツリー

図6.16「名前空間ツリーの例」に名前異空間の例を示します。`myProject`名前空間は、デフォルトの名前空間であり、EB GUIDEプロジェクトの名前でもあります。一部の名前空間は、他の名前空間内にネストされています。

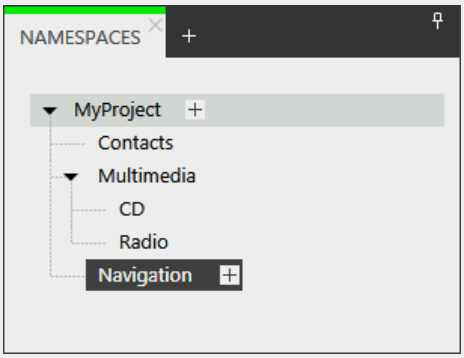

図6.16 名前空間ツリーの例

詳しくは、6.6「グラフィカルユーザーインターフェイスのコンポーネント」をご覧ください。

手順については、9.13「名前空間の操作」をご覧ください。

6.16. モデルインターフェース

EB GUIDEでは、複数のEB GUIDEモデルからHMIを構成することができます。これらのモデルは、個別に開発、テスト、保守、および実行することができます。これを可能にするために、EB GUIDEモデルの1つ以上のインターフェースをエクスポートできます。これらのインターフェースは、他のEB GUIDEモデルにインポートすることができます。複数のモデルインターフェースを別々のEB GUIDEモデルからインポートできます。

基本的に、こうしたインターフェースは、イベントとデータプールアイテムから構成されます。イベントとデータプールアイテムは、モデル間での通信を可能にする機能です。インターフェースを構成するイベントとデータプールアイテムは、ユーザーが定義できます。

モデルには、何も含まれていないデフォルトのモデルインターフェースがありますが、モデルインターフェースを独自に作成し、定義することができます。エクスポートとインポートは、.jsonファイルを使用して行います。インターフェースをインポートしたEB GUIDEモデルでは、そのインターフェースを変更できないことに留意してください。

インターフェースを作成し、イベントとデータプールをモデルインターフェースに追加し、モデルインターフェースをインポートまたはエクスポートする手順については、10.9「モデルインターフェースの操作」をご覧ください。

6.16.1. データプールアイテムのインポート

EB GUIDEモデルは、インポートされたモデルインターフェースのデータプールアイテムの値を読み書きできますが、データプールアイテムを削除したり名前を変更したりすることはできません。インポートされたデータプールアイテムには、▢ボタンがありません。エクスポート時に、すべてのリンク、言語、またはスキンのサポートがデータプールアイテムから削除されます。データプールアイテムにスクリプト値がある場合、エクスポート時にプレーン値に変換されます。データプールアイテムがソースモデルの名前空間に属さない場合は、インポート時に、対応するモデルインターフェースに基づいて命名された名前空間が作成され、この名前空間にデータプールアイテムが追加されます。

6.16.2. イベントのインポート

EB GUIDEモデルは、インポートされたモデルインターフェースのイベントをトリガーできますが、それらのイベントを変更、名前変更、または削除することはできません。インポートされたイベントには、＋ボタンがありません。

注記

イベントIDの重複

イベントグループ内では、イベントIDが一意でなければなりません。複数のモデルインターフェースを同時にインポートすると、 別のモデルインターフェースに同一のイベントIDが存在する場合に 同じイベントグループ内でIDが重複することになり、検証エラーが発生します。 イベントIDの変更は インポート後にEB GUIDE Studioで行えないため、インポートを元に戻してから、イベントIDの変更を 元のモデルで行い、再びエクスポートとインポートを実行します。あらかじめ、イベントIDの範囲をすべての EB GUIDEモデルについて定義することをお勧めします。

イベントがソースモデルの名前空間に属さない場合は、インポート時に、対応するモデルインターフェースに基づいて命名された名前空間が作成され、この名前空間にイベントが追加されます。

6.16.3. イベントグループのインポート

イベントグループがあるモデルインターフェースをインポートすると、イベントグループの所有権は、インターフェースがインポートされたモデルと共有されます。したがって、イベントグループは、モデルインターフェースと関連して特別な扱いが必要です。

- インポートされたモデルインターフェースの一部であるイベントに対応するイベントグループを変更することはできません。

- インポートされたモデルインターフェースの一部である1つ以上のイベントに対応するイベントグループを削除することはできません。
- モデルインターフェースを削除した場合、そのインターフェースに含まれてインポートされたイベントグループは削除されません。
- 名前が変更されたイベントグループが含まれるモデルインターフェースを更新し、再インポートした場合、インポート先のモデルにあるイベントグループの名前は変更されません。
- 既存のイベントグループと同じIDを持つイベントグループをインポートすると、2つのイベントグループは結合されます。

  例えば、次のようになります。EB GUIDEモデルにはイベントグループ"A"があり、IDは65536です。同じ65536をIDとするイベントグループ"B"が含まれるモデルインターフェースをインポートすると、このインポートされたイベントグループにあるすべてのイベントが、ID 65536のイベントグループ"A"に追加されます。

6.16.4. 名前空間のインポート

インポートされたモデルインターフェースに、特定の名前空間に属するイベントまたはデータプールアイテムがある場合、その名前空間もインポートされます。この名前空間は読み取り専用です。つまり、次のような制約が適用されます。

- インポートされた名前空間の名前と内容(データプールアイテムまたはイベント)は変更できません。
- インポートされた名前空間は削除できません。
- インポートされた名前空間にサブ名前空間を追加することはできません。
- インポートされた名前空間にデータプールアイテムまたはイベントを移動することはできません。

6.17. Photoshopファイル形式のサポート

EB GUIDE Studioは、すべての共通の.psdファイル形式をサポートしています。サポートされる色空間は、8ビット、16ビット、32ビットのRGB、それにCMYKです。.psdファイルを直接インポートしたり、.psdファイルからイメージを抽出したりできます。.psb ファイルはサポートしていません。

インポート

.psdファイルにある要素は直接モデル内に配置され、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、.psdファイルのレイヤーから作成されたコンテナー、イメージ、ラベルで構成されます。手順については、8.1.4「ビューに.psdファイルをインポートする」をご覧ください。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、対応するコンテナーまたはイメージのvisibleプロパティの横にあるチェックボックスは選択解除されています。
- .psdファイルのテキストレイヤーは、ラベルとしてインポートされます。対応するラベルは、インポート後にウィジェットツリーで確認できます。

- .psdファイルのイメージレイヤーは、イメージとしてインポートされます。
- .psdファイルのグループレイヤーは、コンテナーとしてインポートされます。コンテナーの名前は対応するグループレイヤーと同じものになります。コンテナーには、イメージ、ラベル、またはその他のコンテナーを含めることができます。

抽出

.psd ファイルにあるイメージが含まれているサブディレクトリが作成されます。ただし、作業しているEB GUIDEモデルは変更されません。手順については、8.1.5「.psdファイルからイメージを抽出する」をご覧ください。

制限事項

EB GUIDE Studioでは、Photoshopファイル形式が持つ次の機能をサポートしていません。

- レイヤーエフェクト、フィルタ、テクスチャ
- RGBおよびCMYK以外のカラーモデル
- マスク
- 1つのレイヤーに適用される複数のマスク(レイヤーマスクおよびベクトルマスク)
- テキストスタイリングおよびフォント
- 色チャネルのみの使用

6.18. 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. .ebibl 3Dグラフィック用ファイル形式
3. イメージ
4. 3Dグラフィック用メッシュ
5. .psd ファイル形式

リソースをプロジェクトで使用するには、リソースファイルを$GUIDE_PROJECT_PATH/<project name>/resourcesに追加します。

6.18.1. フォント

フォントをプロジェクトで使用するには、フォントを$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、8.4「フォント設定の変更」をご覧ください。

6.18.1.1. 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。

6.18.1.2. マルチフォントサポート

EB GUIDE Studioでは、マルチフォントサポートを使用してフォントの独自の組み合わせを作成できます。この機能は、例えば、選択したフォントに必要なすべての文字が用意されていない場合などに役立ちます。そうした場合は、欠落している文字を別のフォントの文字で置き換えることができます。

マルチフォントサポートは、次のモデル要素に追加できます。

- タイプフォントのプロパティとフォントリストのエントリー
- タイプフォントのユーザー定義プロパティとフォントリストのエントリー
- タイプフォントのデータプールアイテムまたはフォントリストのエントリー

指定のフォントで使用するUnicode文字範囲を次のように定義できます。

- 単一のUnicode文字を使用: `0000`など。
- カンマで区切った複数のUnicode文字を使用: `0000, 0001`など。
- Unicode文字の範囲を使用: `0000-FFFF`など。
- カンマで区切った複数の範囲を使用: `0000-0022, 0045-0055`など。

これらの文字は16進数形式を使用して指定します。

フォントは`$GUIDE_PROJECT_PATH/resources`に用意されています。

マルチフォントサポートを追加すると、デフォルトのマルチフォント値が自動的に追加されます。デフォルトのマルチフォント値は削除できません。その優先度や範囲を編集することもできません。ただし、デフォルト値のサイズおよびフォントは編集できます。

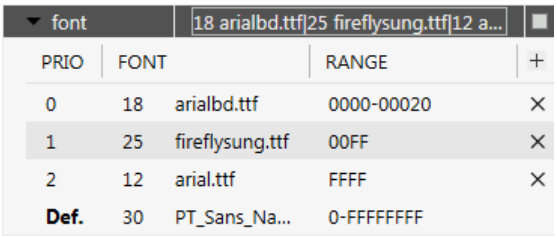

図6.17 マルチフォントサポートを追加したタイプフォントのプロパティの例

マルチフォントサポートの使用方法については、8.4.3「マルチフォントサポートの管理」をご覧ください。

6.18.2. 3Dグラフィックのイメージベースドライティング

EB GUIDE Studioでは、イメージベースドライティングを使用できます。外部コマンドラインツールのIBLGeneratorは、`.pfm`または`.hdr`ファイルを入力データとして受け取り、IBLリソースを表す`.ebibl`ファイルを作成します。IBLリソースは、イメージベースドライトウィジェットの`ibl`プロパティによって使用されます。

`.ebibl`ファイルの取得方法については、8.1.6「IBLファイルのインポート」をご覧ください。

バックグラウンド情報については、6.13「イメージベースドライティング」をご覧ください。

6.18.3. イメージ

イメージをプロジェクトで使用するには、イメージを$GUIDE_PROJECT_PATH/<project name>/resources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。

6.18.3.1. 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、拡大縮小の対象ではありません。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。

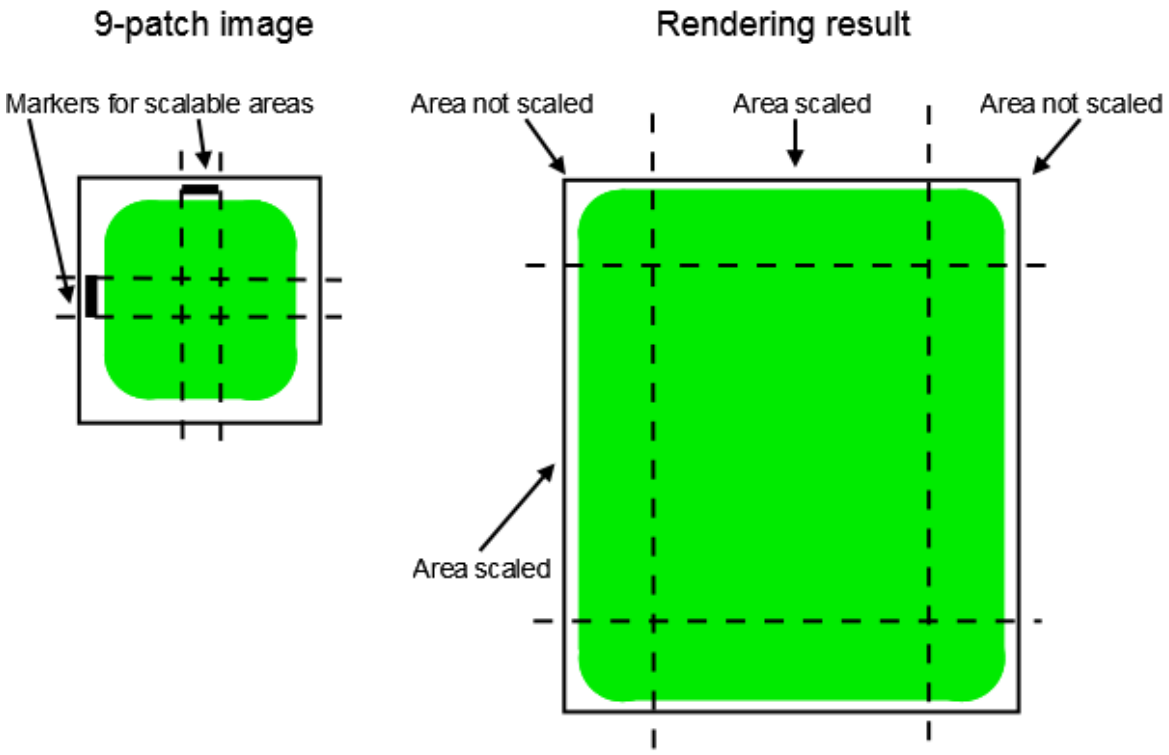

図6.18 9-patchの例

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処理はOpenGL ES 2.0以降のレンダラーでのみ機能します。
- 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。

6.18.4. 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。

6.19. 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。

6.19.1. アプリケーションの機能とエリア

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

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

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

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

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

6.19.2. プレフィックスと識別子

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。これを可能にするには、EB GUIDEスクリプト内のすべての識別子(オブジェクトの名前)がプレフィックスを持つ必要があります。プレフィックスは、オブジェクトのタイプを定義するもので、その後にコロンが続きます。

EB GUIDEスクリプトで使われる一連のプレフィックスは固定されており、新しいプレフィックスを導入することはできません。使用可能なプレフィックス:

- 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。

プレフィックスがない識別子は文字列定数です。

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6.7
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!";
```

6.19.3. 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6.8

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。

6.19.4. 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。

6.19.5. 式

EB　GUIDEスクリプトは、式を基盤として機能します。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6.9
整数値の評価

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

6.19.6. 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通りありますが、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6.10
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
```

6.19.7. 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6.11
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
```

6.19.8. L値とR値

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

- L値はデータプール参照、ローカルウィジェットプロパティ、またはローカル変数です。

- R値はイベントパラメータまたは定数式(文字列定数や整数定数など)です。

6.19.9. 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。変数のプレフィックスはv:です。let式の構文には、次の特徴があります。

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

let式のデータ型と値は、in式の使用時と同じです。

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

例6.12
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;
    }
}
```

6.19.10. 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`はブール型で、実行式は`void`型でなければなりません。`while`式は`void`型であり、代入演算の左辺または右辺に記述することはできません。

例**6.13**
`while`ループの使用方法

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

6.19.11. 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。

注記

if 条件式

`if`式が`&&`または`||`で連結された複数のサブ条件で構成される場合、EB GUIDEスクリプトは、他の一部のプログラミング言語と異なり、すべてのサブ条件を評価します。つまり、あるサブ条件がfalseであり、したがって条件全体がfalseである場合も、残りのすべてのサブ条件が評価されます。

`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**6.14**
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 {}
```

6.19.12. 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6.15
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);
```

6.19.13. 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。プレフィックスdp:を持つ識別子は、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。識別子にはデータプールアイテムの名前のみを含めることができます。あるいは、そのデータプールアイテムがデフォルトの名前空間にない場合、名前空間の名前の後にデータプールアイテムの名前が続きます。

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**6.16**
データプールアイテム値の代入

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

// Assign the value of the datapool item intB to intA.
// Both datapool items must have the same type.
dp:intA = dp:intB;
// Multiply the value of intB by two and assign it to intA.
dp:intA = 2 * dp:intB;
// Use the value of a datapool item in an if-clause.
if( dp:speed > 100 )
{
    // ...
}
```

以下の演算子がデータプールアイテムに使用できます。

- 参照演算子(&)はデータプールアイテムに使用できます。この演算子を使うと、データプールアイテムの値ではなくアドレスを参照できます。参照演算子は、dp_id型のパラメータを渡すために外部関数で使用します。
- リダイレクトリンク演算子(=>)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。

6.19.14. 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6.17
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6.18
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;
}
```

6.19.15. 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6.19
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。

6.19.16. 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6.20
  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6.21
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6.22
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 {}
```

識別子にはイベントの名前のみを含めることができます。あるいは、そのイベントがデフォルトの名前空間にない場合、名前空間の名前の後にイベントの名前が続きます。

EB GUIDE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た場合は、match_event式のin式でそれら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6.23
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;
}
```

6.19.17. 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6.24
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);
```

6.19.18. 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、15.4.3「EB GUIDEスクリプト標準ライブラリ」をご覧ください。

6.20. 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、9.8「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EB GUIDE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。

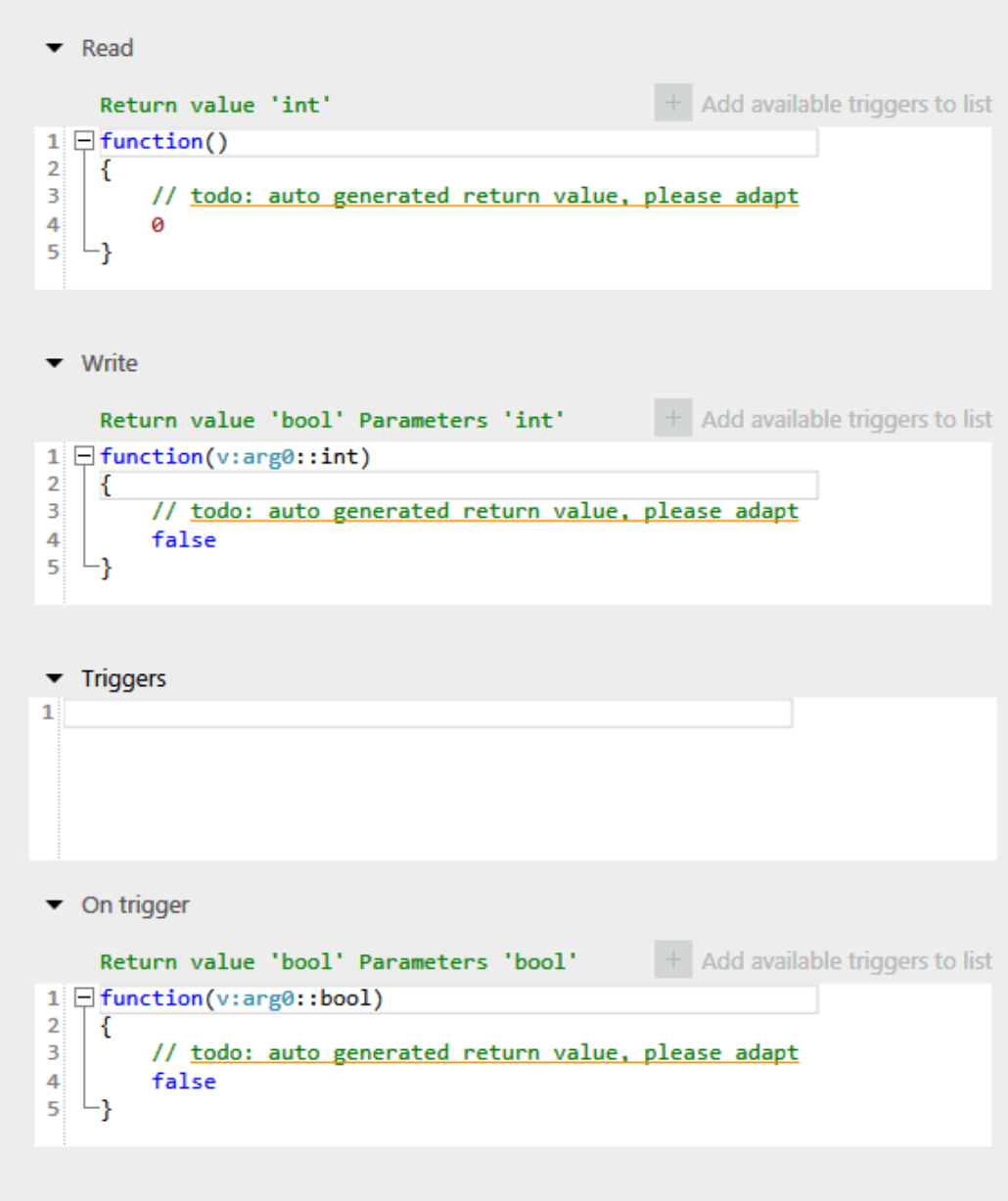

図6.19 EB GUIDE StudioのEB GUIDEスクリプト

- [読み取り]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。

  [読み取り]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。
- [書き込み]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[書き込み]　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。

  [書き込み]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
- [トリガー]スクリプトには、[トリガー時]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。

  [使用可能なトリガーをリストに追加]をクリックすると、対応するスクリプトで強調表示されているすべてのトリガーがトリガースクリプトに追加されます。
- [トリガー時]スクリプトはイベントをトリガーしたのち、またはプロパティを更新したのち、初期化時に呼び出されます。

  [トリガー時]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[トリガー]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。

  [トリガー時]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
- [レングス]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。

  [レングス]スクリプトの戻り値は、リストの現在の長さを表します。

6.21. 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、8.6「スキンのサポートの操作」をご覧ください。

6.22. ステートマシンとステート

注記

ステートマシンとステートの背景色の変更

EB GUIDE Studioでは、次の背景色を変更できます。

- ステートマシン
- ビューステート
- 混合ステート

背景色を変更するには、[プロパティ]コンポーネントの`Background color`ドロップダウンリストボックスから色を選択します。

6.22.1. 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、次のタイプのステートマシンを作成できます。

6.22.1.1. 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。

6.22.1.2. ロジックステートマシン

ロジックステートマシンを使うと、GUIなしでロジックの仕様を提供できます。

6.22.1.3. 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、14.1「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。

6.22.2. 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

6.22.2.1. 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。

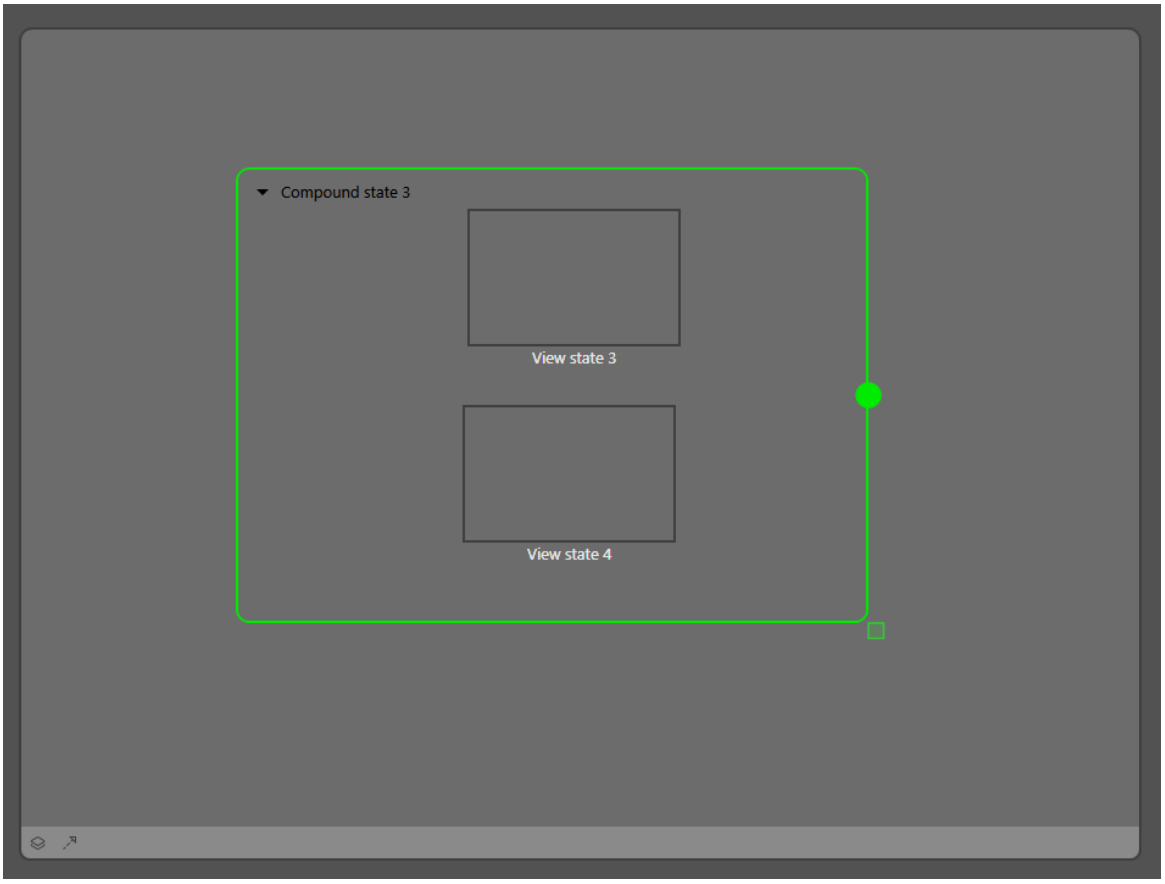

図6.20 混合ステート

[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。

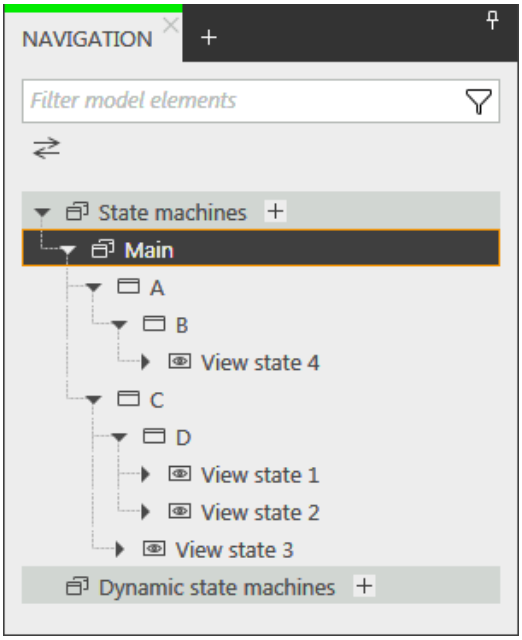

図6.21 ツリー構造で表示されたステート階層

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。

6.22.2.2. 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。

6.22.2.3. 初期ステート

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

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

- 混合ステートへの遷移を使う(初期ステート必須)
- 混合ステートの子ステートへの遷移を使う

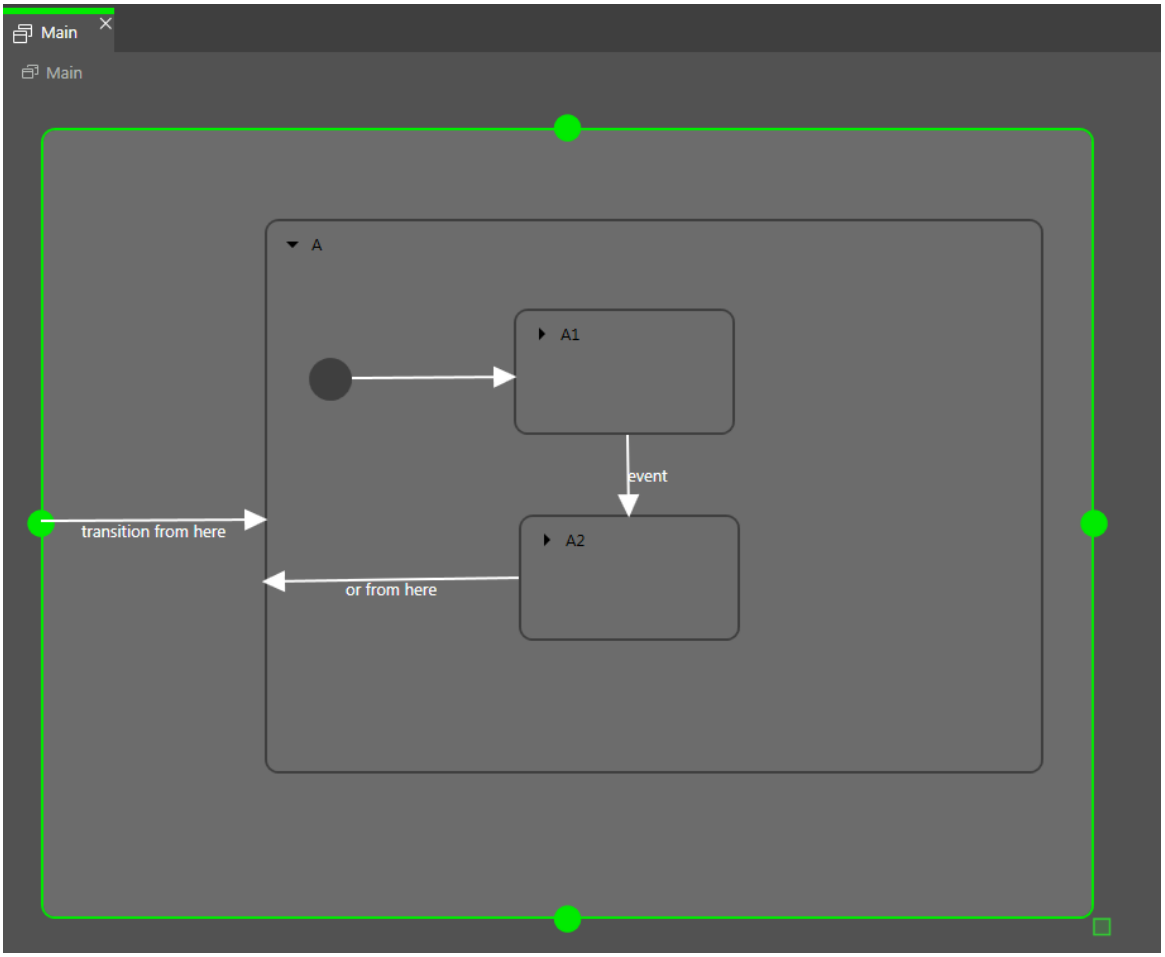

図6.22 初期ステートの例

混合ステートには初期ステートを1つだけ作成できます。

6.22.2.4. 最終ステート

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

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

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

最終ステートを含む混合ステートには、出力遷移が必要です。

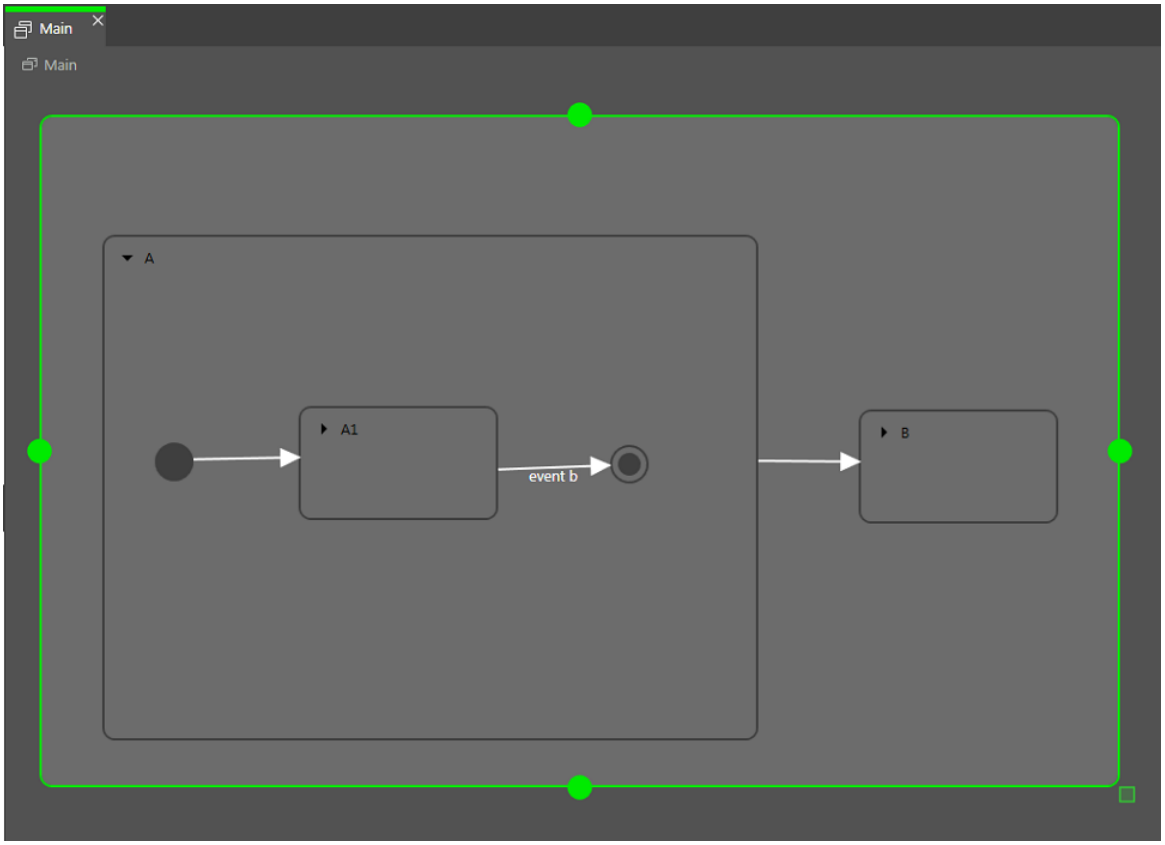

図6.23 混合ステート内での最終ステートの使用方法

混合ステートには最終ステートを1つだけ作成できます。

6.22.2.5. 選択ステート

選択ステートは、動的な条件分岐を実現します。条件に基づいてイベントを発行する場合に使用します。選択ステートは、遷移元のステートと遷移先のステートを接続するものです。1つの選択ステートに複数の入出力遷移を設定できます。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。

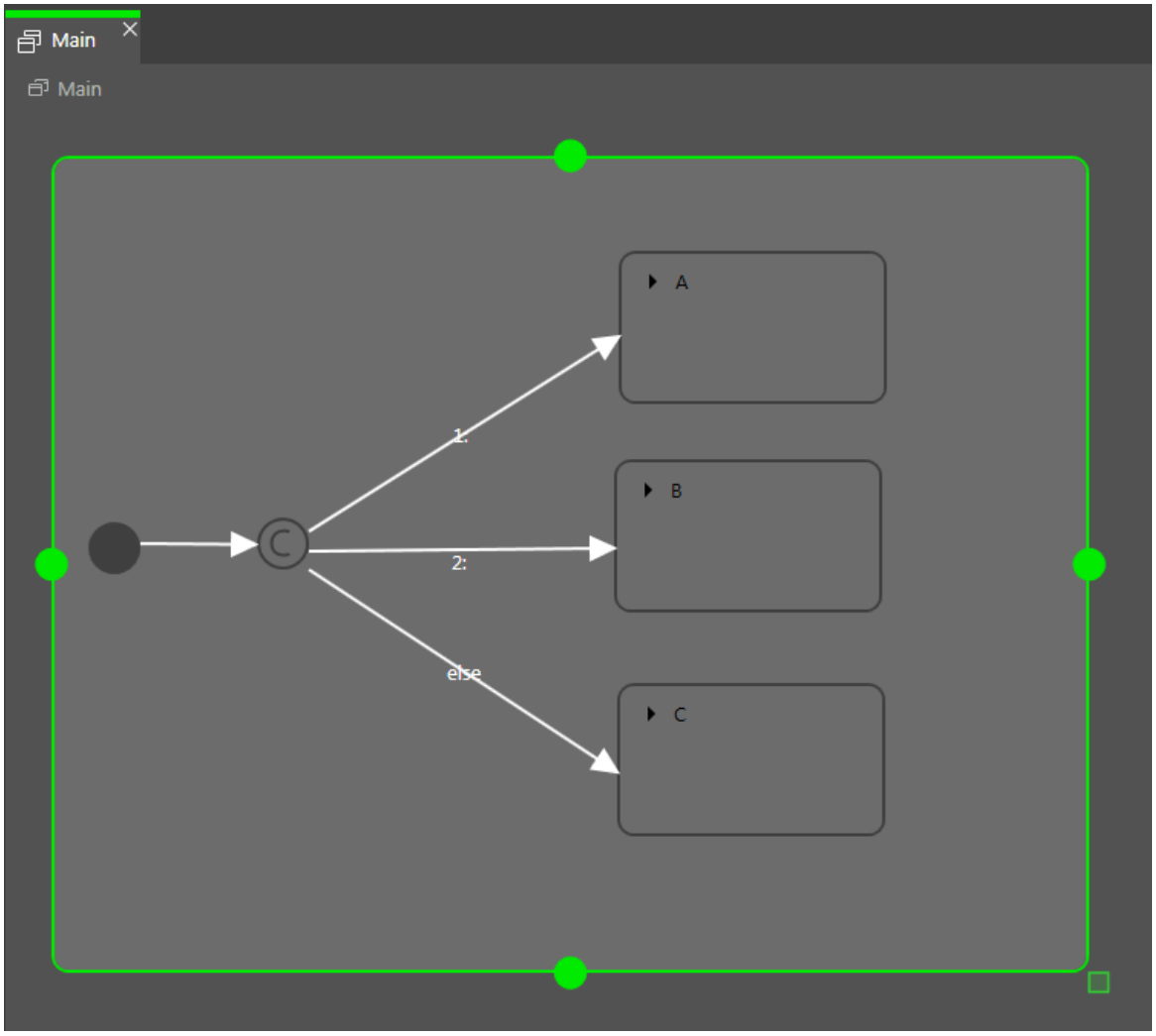

図6.24 入力遷移と出力遷移を持つ選択ステート

6.22.2.6. 履歴ステート

EB GUIDEには、2種類の履歴ステートがサポートされています。

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

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

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

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

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

例6.25
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。

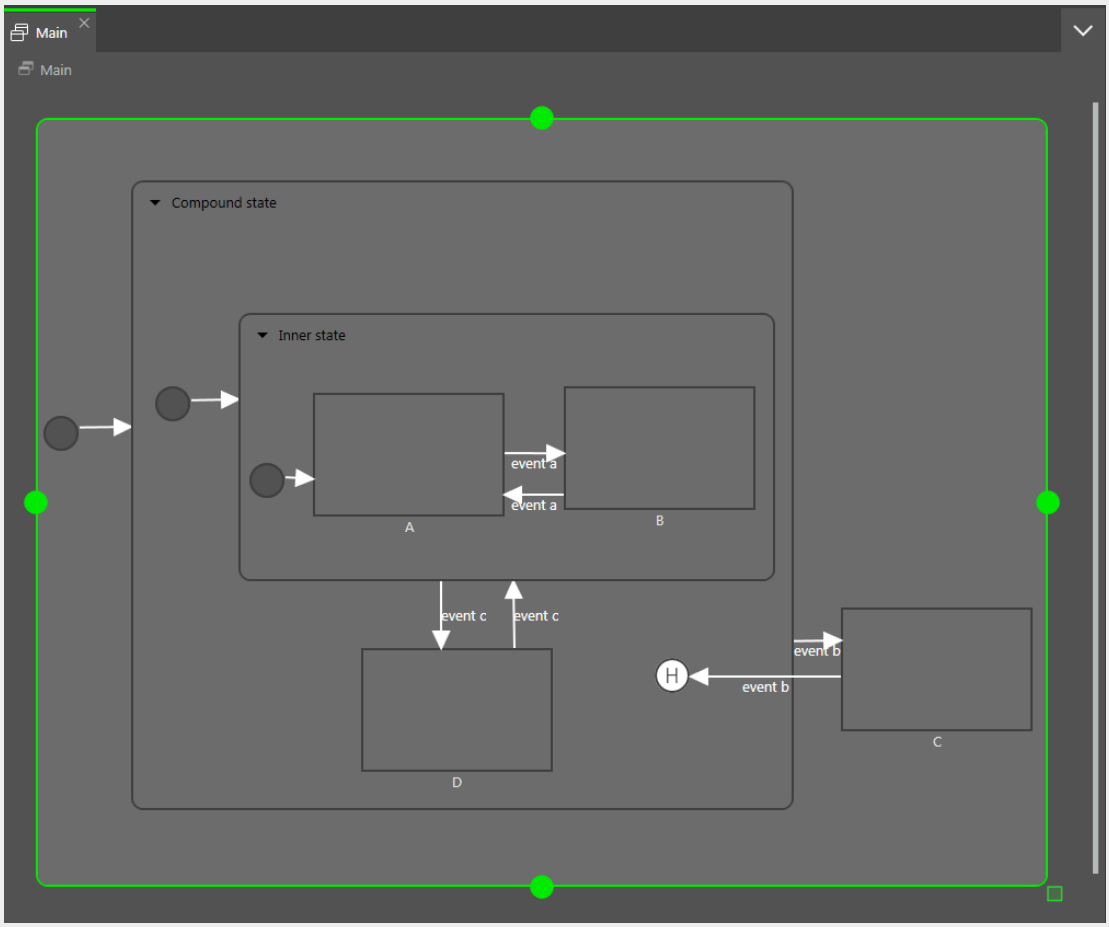

図6.25 浅い履歴ステート

- 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。

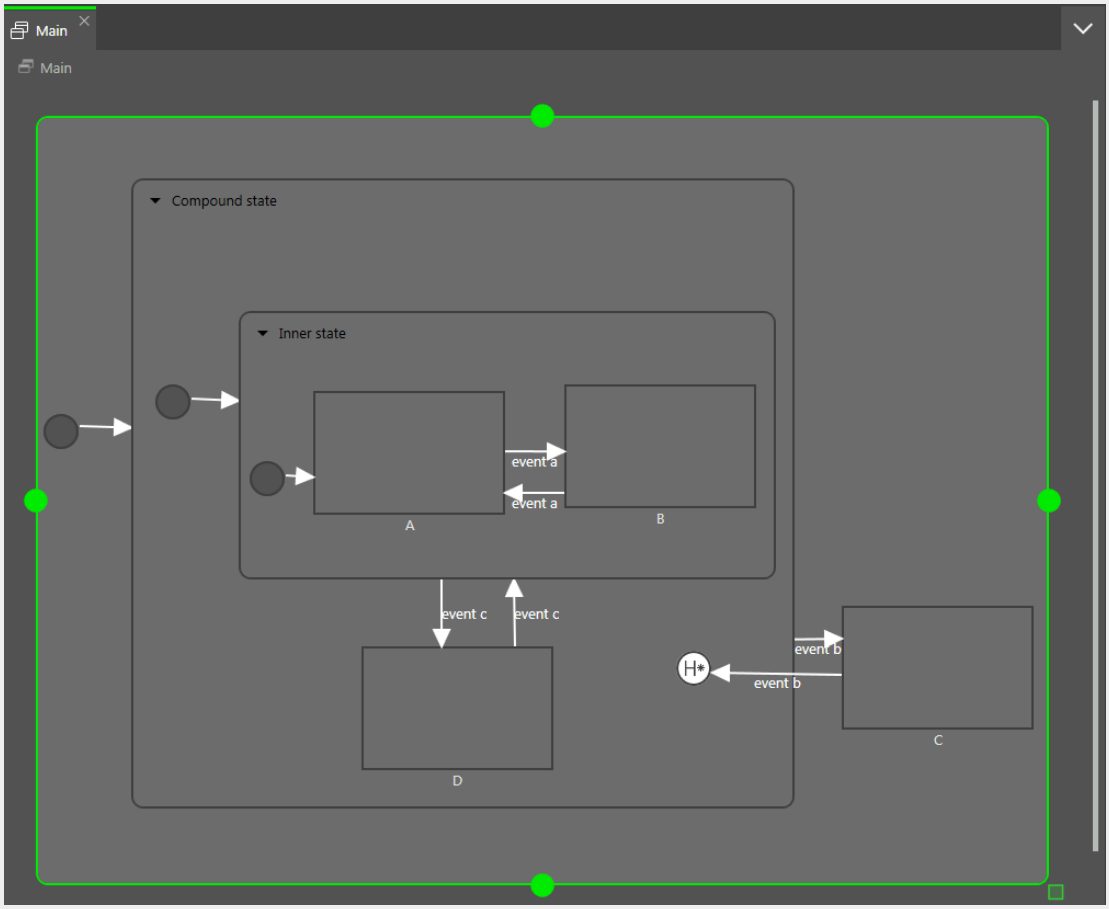

図6.26 深い履歴ステート

- 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。

6.22.3. 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

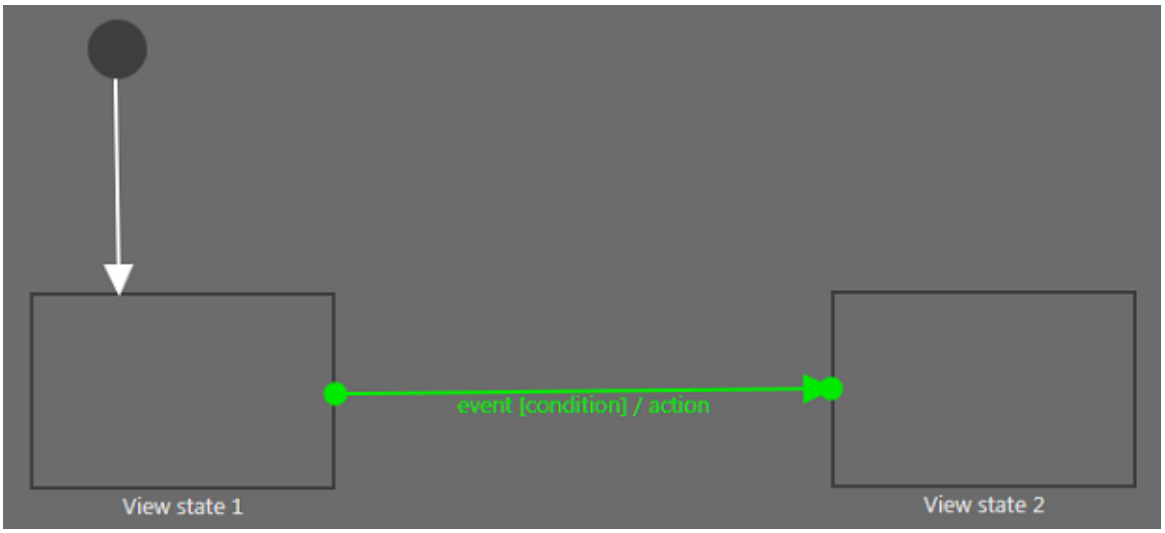

図6.27 遷移

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

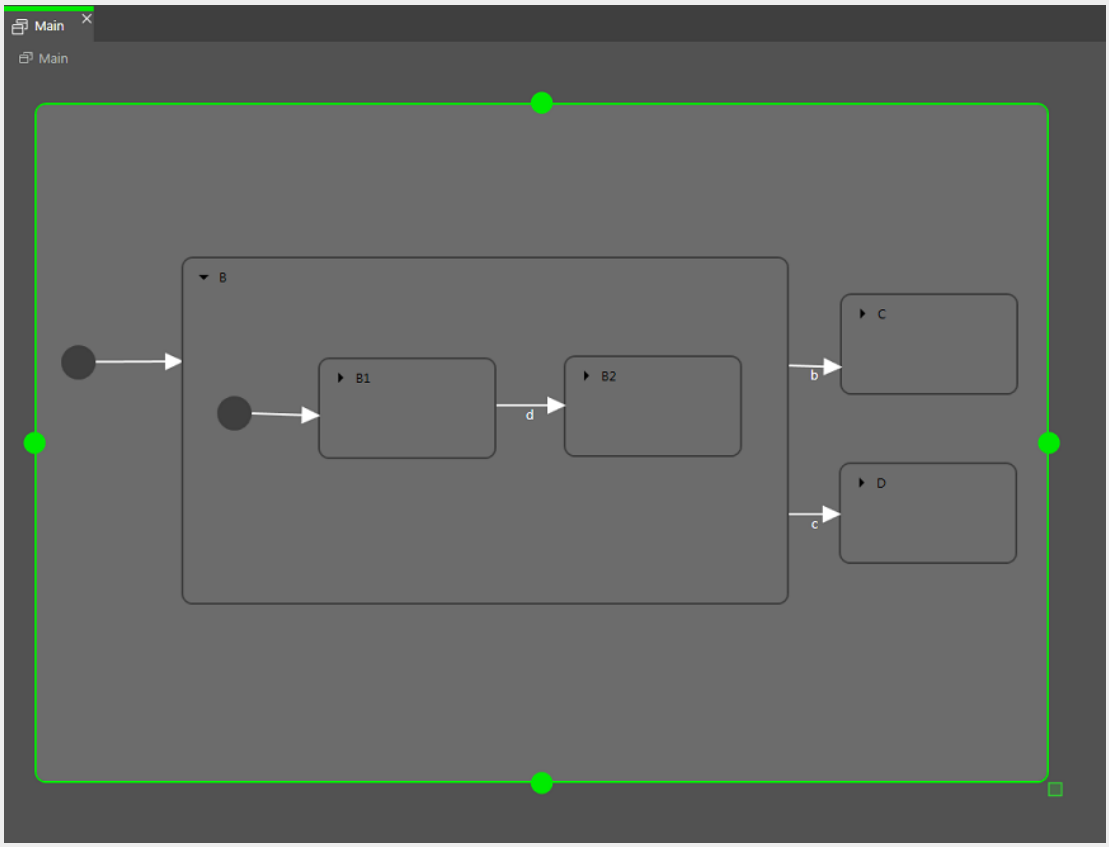

図6.28 遷移の継承

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

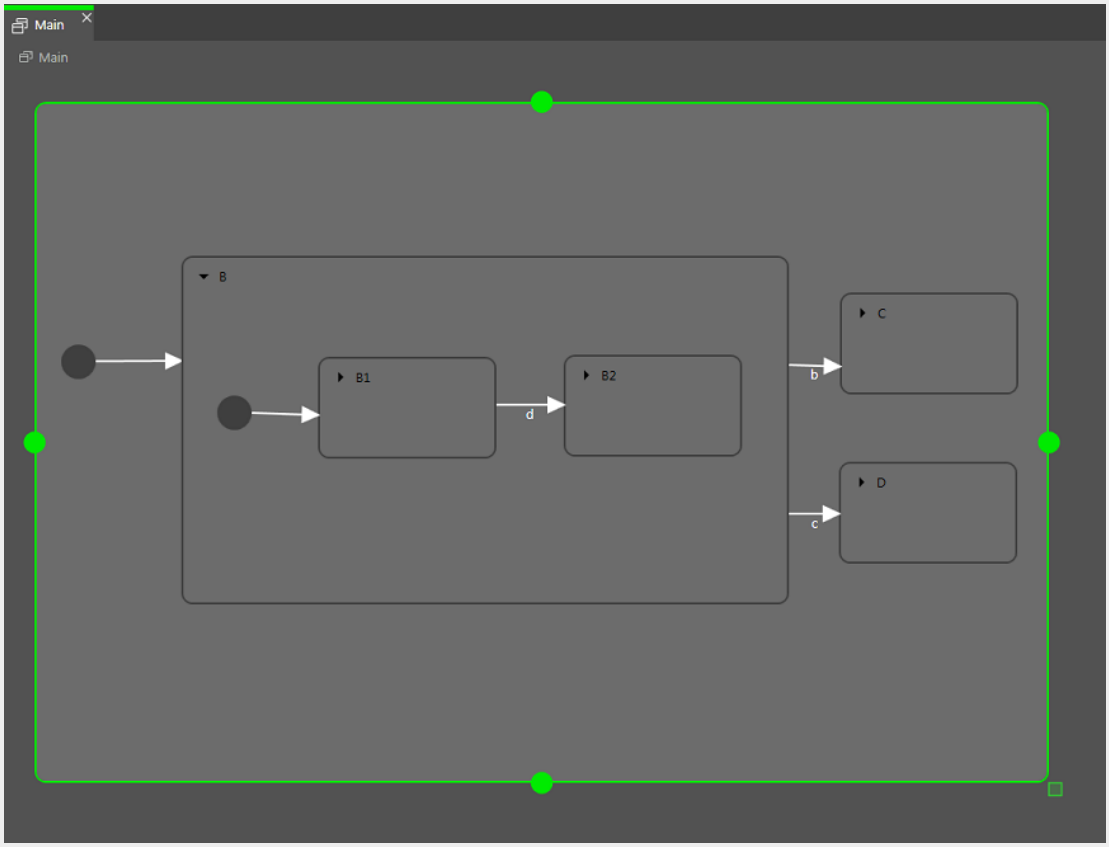

図6.29 遷移のオーバーライド

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

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

注記

実行の階層

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

遷移には次のタイプがあります。

- デフォルト遷移

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

- 選択遷移

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

- else遷移

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

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

- 自己遷移

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

6.22.4. ステートマシンの実行

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

ステートマシンの動作サイクルは、次のようなものです。

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

iii. 遷移が見つからなければ、最上位のステートに達するまで前のステップを繰り返します。

b. 内部遷移が処理されます。

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

c. 遷移が見つかります。

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

ii. 遷移が見つからなければ、親ステートに移動し、遷移を探します。

iii. 条件を満たす遷移が見つかるまで、前のステップを繰り返します。

d. 遷移が処理されます。

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

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

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

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

3. 最終ステートに達した時点で、ステートマシンは停止します。

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

例6.29

遷移の実行

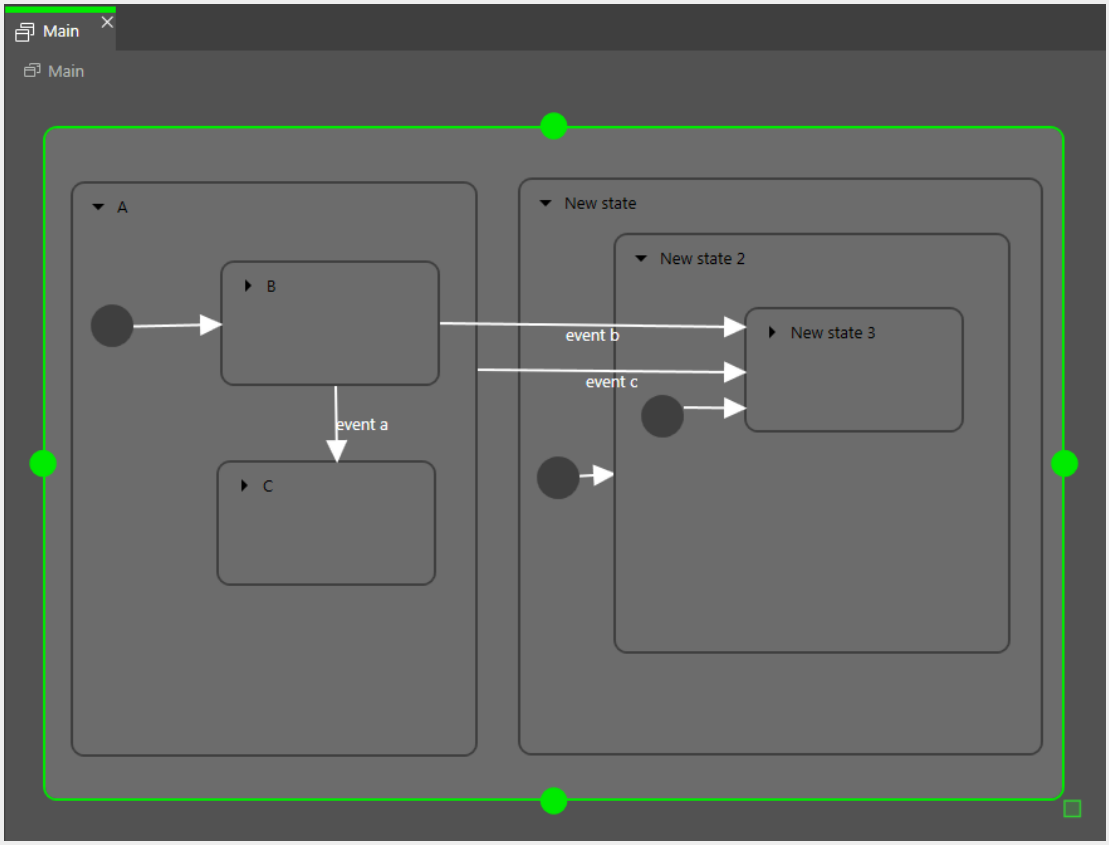

図6.30 遷移の実行

`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

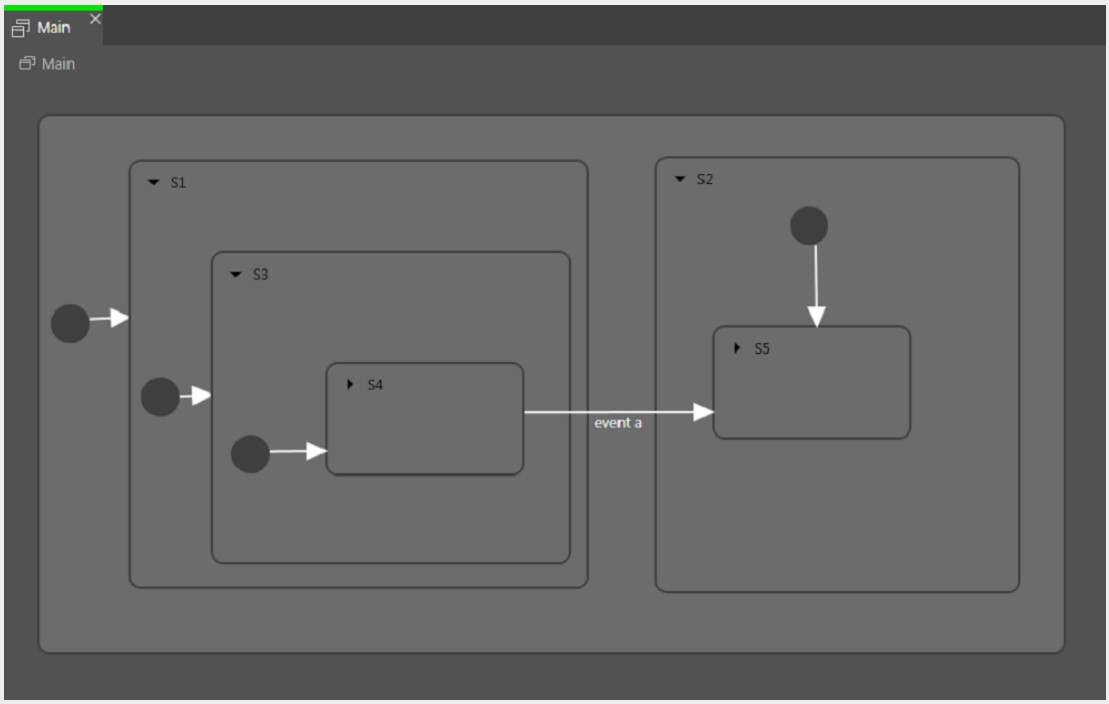

図6.31 遷移の実行

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

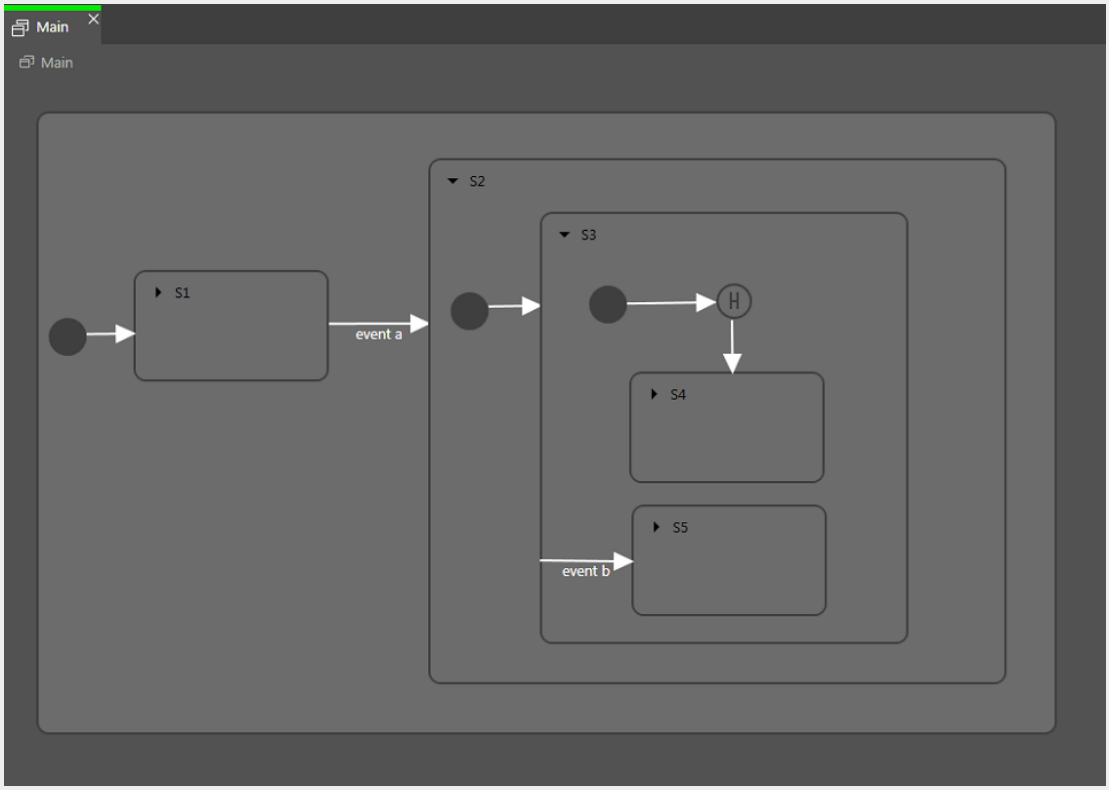

図6.32 遷移の実行

`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。

6.22.5. 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。

6.22.5.1. サポートされている要素

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

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

[a]EB GUIDEでは、外部遷移とローカル遷移を区別しません。

6.22.5.2. 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

6.22.5.3. 偏差

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

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

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

6.23. 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

6.23.1. 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。

6.23.2. 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、14.3「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。

6.23.3. 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。

6.23.4. マルチタッチ入力

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

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

- マルチタッチイベントの処理

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

- Finger ID

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

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

- エンドユーザーは、リストをスクロールしながらボタンを押すなど、インターフェー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。

6.24. 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。

ウィジェットを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。次の要素は、タッチ時または移動時のプロパティの例です。

- サイズ
- 色
- レイアウト
- 動作

ウィジェットを組み合わせることができます。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、次の要素のボタンを作成できます。

- 楕円
- イメージ
- ラベル
- 四角形

ウィジェットを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。

6.24.1. 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、次の要素がビューに配置されます。

- 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

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

ビューのサイズ変更

EB GUIDE Studioでは、一部の領域を詳細に把握したりより広い領域を確認するためにビューのサイズを拡大または縮小することができます。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。

注記

マスターイメージ上の要素の整列

EB GUIDE Studioでは、基本ウィジェットや3Dウィジェットといった要素を均等に整列させるためにビューにマスターイメージを追加できます。ビューにマスターイメージを追加するには、ビューの下部にある をクリックします。マスターイメージを非表示にするには、チェックボックスをオンまたはオフにします。ビューを閉じる場合は、マスターイメージを再度追加する必要があります。

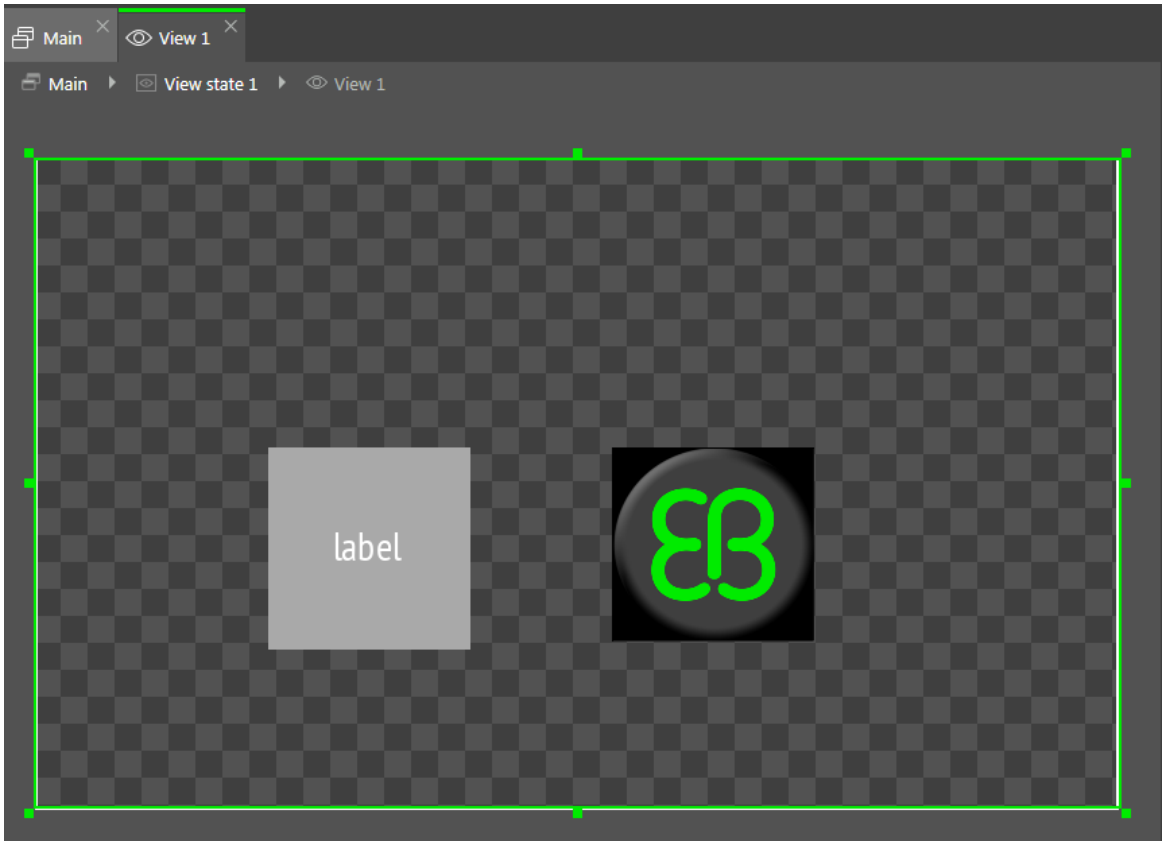

図6.33 四角形、ラベル、イメージを含むビュー

6.24.2. 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

  次の要素は基本ウィジェットです。

  - 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
  - インスタンシエータ
  - ラベル
  - 四角形
- 3Dウィジェット

  3Dグラフィックを表示するには、[3Dウィジェット]カテゴリに含まれるウィジェットを使用します。[3Dウィジェット]は次の要素です。

  - 環境光
  - カメラ
  - 指向性ライト
  - イメージベースドライト
  - 材質
  - メッシュ
  - PBR GGX材質
  - PBR Phong材質
  - 点ライト
  - シーングラフ
  - シーングラフノード
  - スポットライト

  注記

  サポートされているレンダラー

  3Dグラフィックを表示するには、OpenGL ES 2.0以降が必要です。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。
- テンプレート

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

- 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/examples/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。

6.24.3. 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。

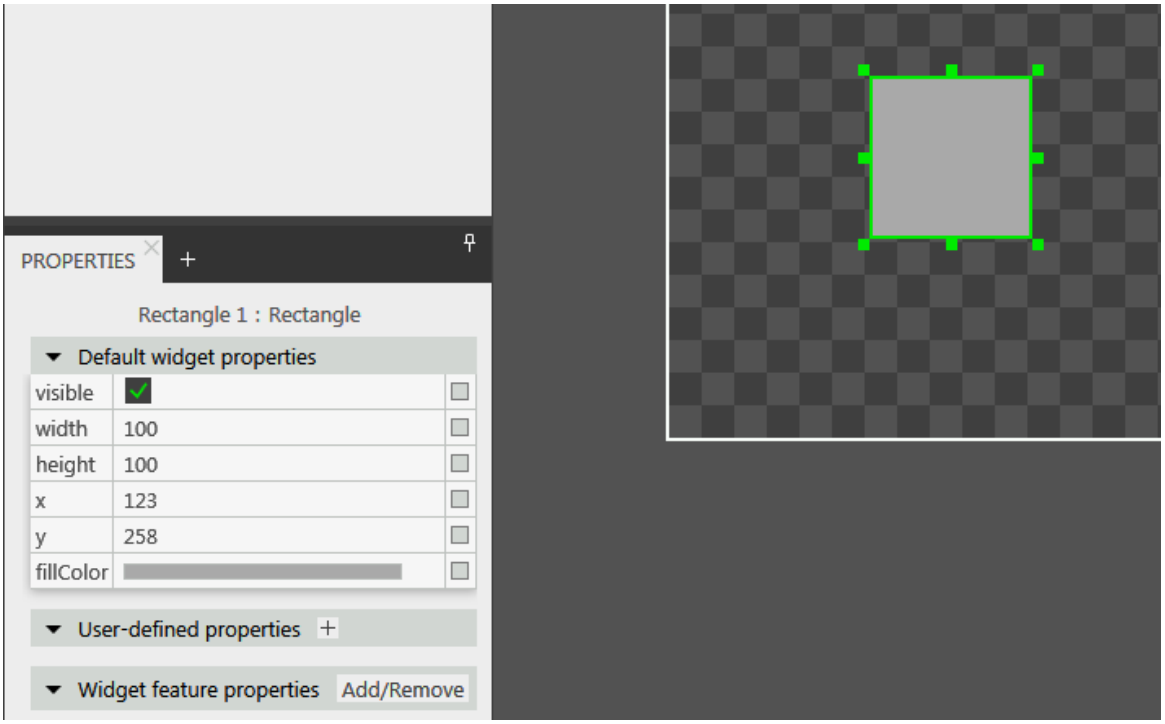

図6.34 四角形とそのプロパティ

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、15.9「ウィジェット」をご覧ください。
- ユーザー定義ウィジェットプロパティは、デフォルトプロパティに加えて、モデラーが作成するプロパティです。

- ウィジェット機能プロパティは、モデラーがウィジェット機能をウィジェットに追加する際に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。`touchPolicy`プロパティは、タッチの処理方法を定義し、`touchBehavior`プロパティはタッチエリアを決定します。

6.24.4. ウィジェットテンプレート

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

ウィジェットテンプレートは、テンプレートインターフェ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。

注記 テンプレートの親ウィジェット
アニメーションウィジェットをテンプレートの親ウィジェットとして使うことはできません。

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ー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、8.8「ウィジェットの再利用」をご覧ください。

6.24.5. ウィジェット機能

ウィジェット機能を使用してウィジェットおよびウィジェットテンプレート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。

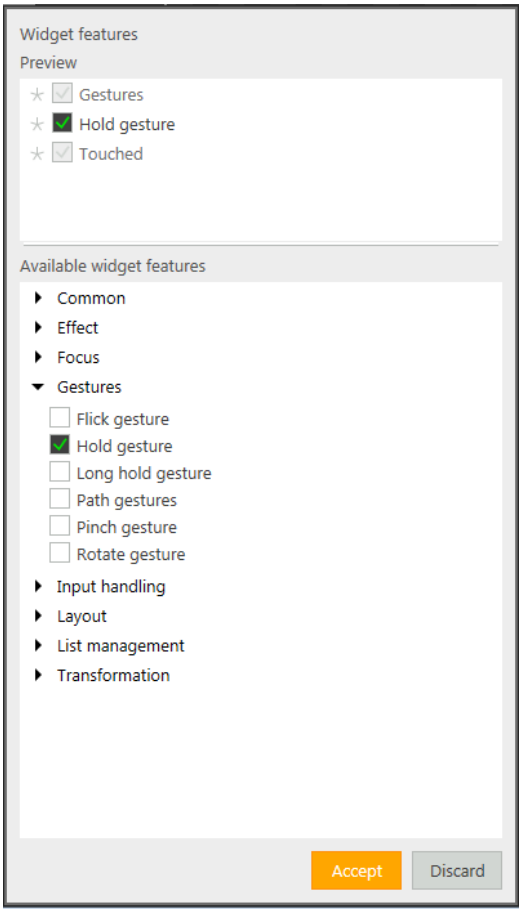

図6.35 ウィジェット機能

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。
- ウィジェット機能を特定タイプのウィジェットに制限できます。
- ランタイム中にウィジェット機能を有効または無効にすることはできません。

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、15.10「ウィジェット機能」をご覧ください。

6.24.5.1. 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を処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。

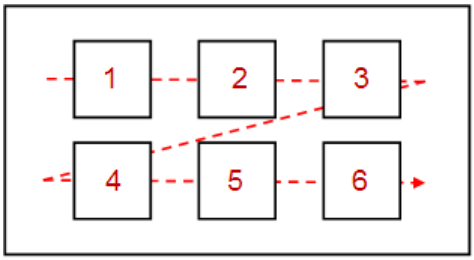

図6.36 [自動フォーカス]ウィジェット機能のポリシー

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と役に立ちます。

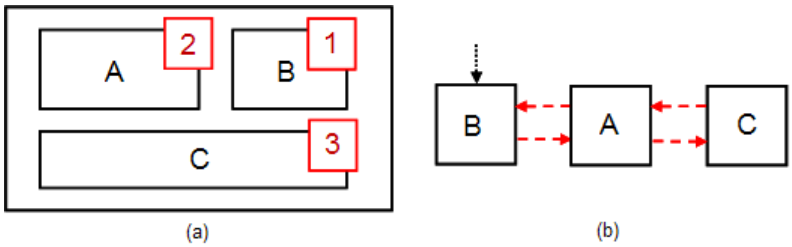

図6.37 [ユーザー定義フォーカス]ウィジェット機能のポリシー

図6.37「[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。フォーカス階層を入れ子にすることができます。

6.24.5.2. 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ン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、インスタン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期されます。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ンスがそのエントリーがどのインスタン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。インスタンシエータがラインテンプレートのインスタン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ンスをゼロとして開始されます。インスタンシエータに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、14.4「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ン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. HMI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
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、14.1「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、＋ 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 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、6.19「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、[終了アクション]プロパティに移動し、＋ 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 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、6.19「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つ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。

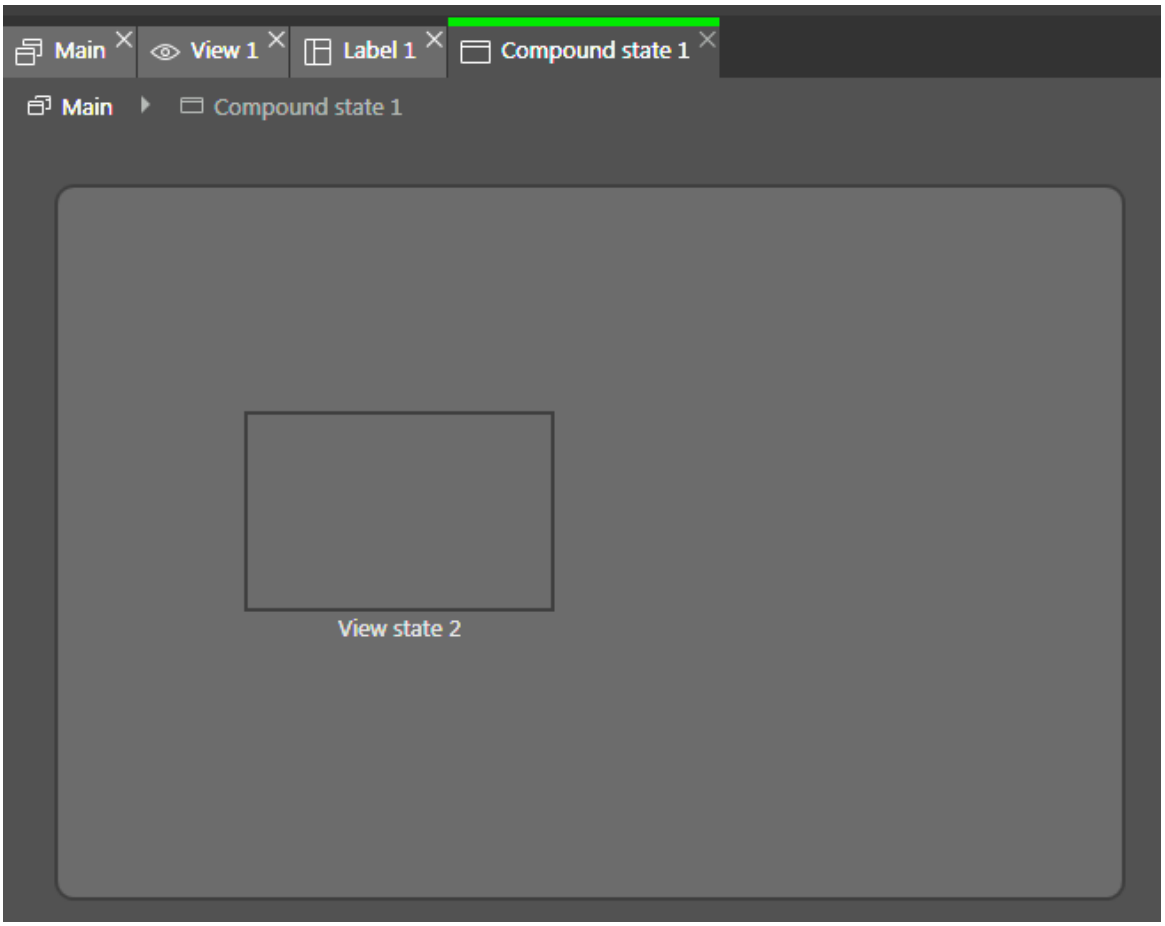

図7.1 入れ子のビューステートを持つ混合ステート

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
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 3
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 4
選択ステートからの出力遷移を追加します。

ステップ 5
[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。

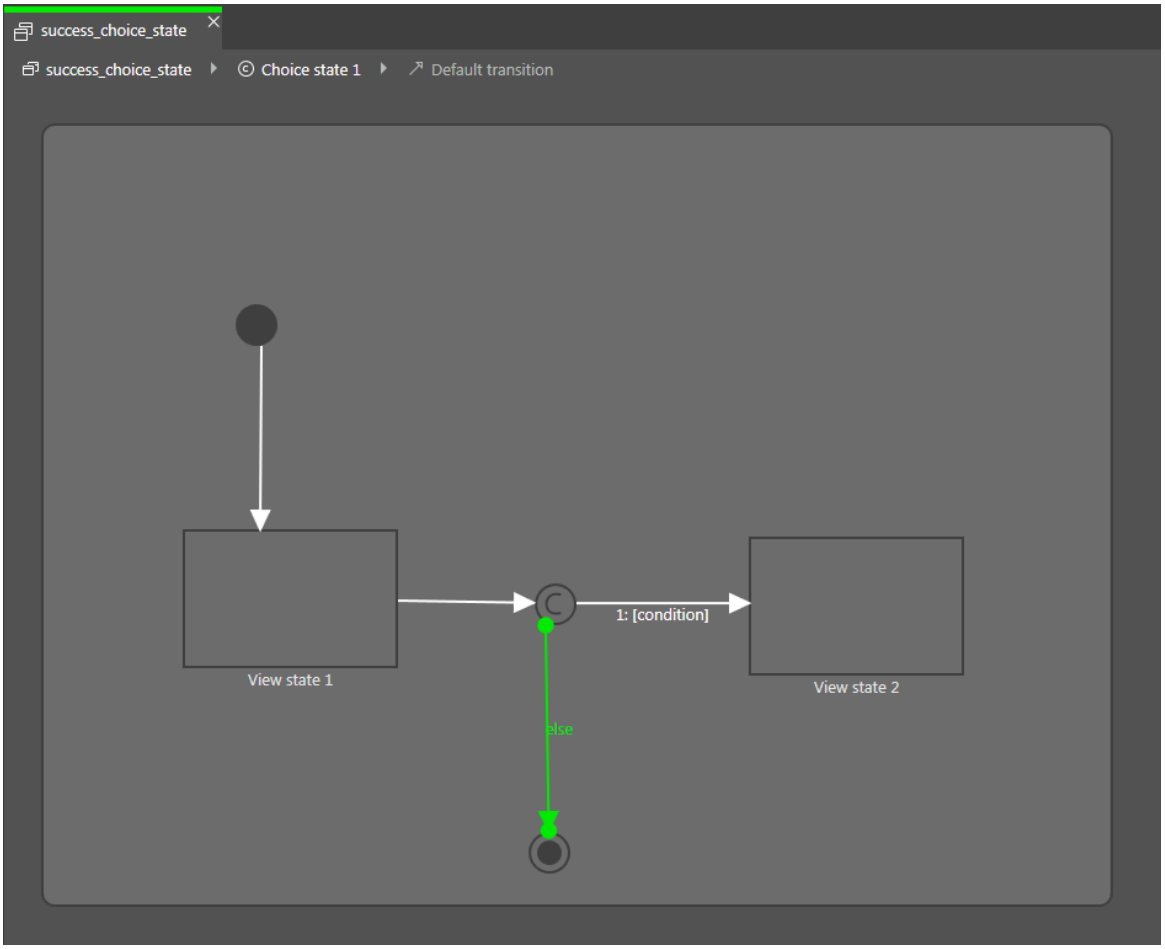

図7.2 選択遷移を持つ選択ステート

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、+ 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 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、6.19「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、+ 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 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、6.19「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。

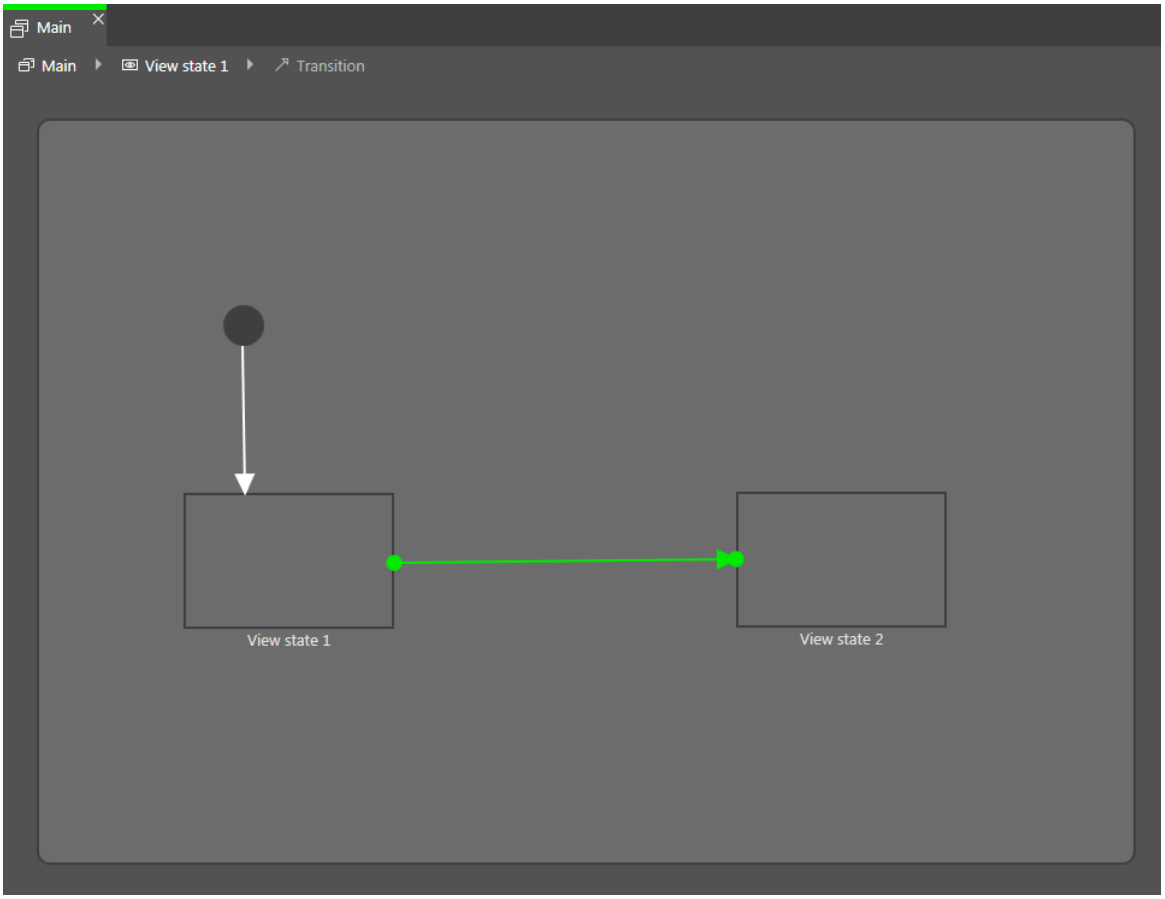

図7.3 遷移

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。

ティップ

スプラインとしての遷移
遷移はスプラインのように形成できます。遷移の形を変更するには、遷移の線をクリックし、マウスボタンを押したままドラッグします。

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]の横にあるボックスをクリックします。

ステップ 3
イベントをクリックします。

新しいイベントを作成する場合は、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。

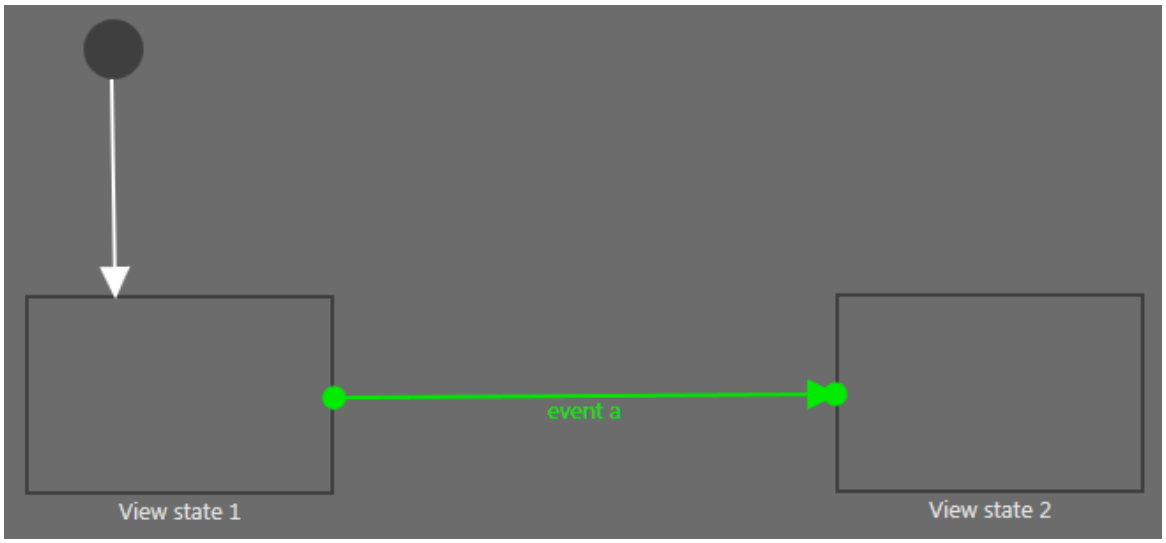

図7.4 トリガー付きの遷移

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+ 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 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、6.19「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。

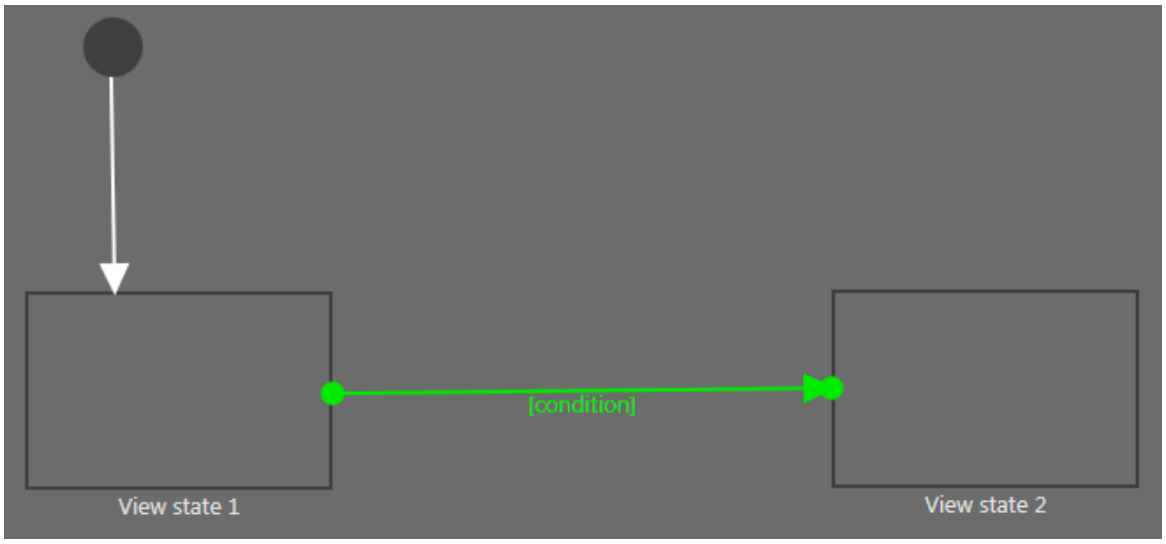

図7.5 条件付きの遷移

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+ 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 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、6.19「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。

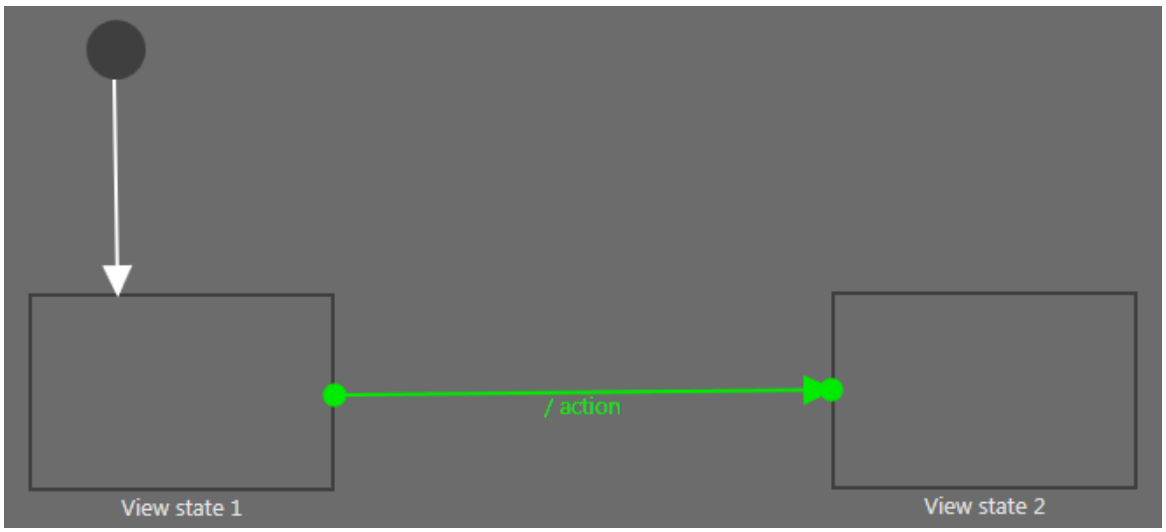

図7.6 アクション付きの遷移

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、[内部遷移]に移動し、＋ 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
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EB GUIDE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、15.9.2「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
幅(0～50)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`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、6.18.3「イメージ」をご覧ください。
- コンテンツエリアにビューが表示されていること。

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

ステップ 2
[プロパティ]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、6.18.3「イメージ」をご覧ください。
- コンテンツエリアにビューが表示されていること。

ステップ 1
[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、6.18.3.1「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

注記

文字置換
ラベルのtextプロパティにテキストを入力すると、以下のような文字の置き換えが行われます。

- シーケンス\\\\は \\に置き換えられます。
- シーケンス\\nは \nに置き換えられます。
- テキストが1行に表示される場合、\nは 1文字の空白に置き換えられます。

[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。

詳しくは、8.4「フォント設定の変更」をご覧ください。

[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、6.18.1「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。

詳しくは、8.4「フォント設定の変更」をご覧ください。

8.1.2.5. コンテナーを追加する

コンテナーを追加する

コンテナーを使用するとウィジェットをグループ化できます。

前提条件:

- コンテンツエリアにビューが表示されていること。

ステップ 1
[ツールボックス]からコンテナーをドラッグし、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、コンテナーの四隅のいずれかをドラッグしてコンテナーを拡大します。

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

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

8.1.2.6. インスタンシエータの追加

インスタンシエータの追加

前提条件:

- コンテンツエリアにビューが表示されていること。

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

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

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

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

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

詳細については、8.3.1「ウィジェット機能の追加」をご覧ください。

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シエータに指定されたレイアウトで、ビューに表示されます。

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シ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 5.1
`lineMapping`を選択し、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、インスタン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。

インスタンシエータの使い方を示す詳細なサンプルについては、14.4「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。
- インスタン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ン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、15.9.2.2「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、[アニメーション化プロパティ]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。

ステップ 5
[承認]をクリックします。

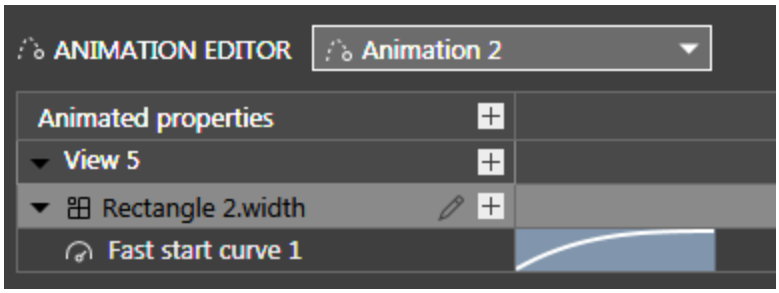

図8.1 サンプル曲線が表示された[アニメーション]エディター

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

ステップ 7
条件スクリプトの横にある[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 8
[トリガー時]セクションに次の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のデフォルト名は[Animation 1]です。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、[トリガー時]スクリプトで名前を書き換えてください。

ステップ 9
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。

ステップ 10
曲線の動作を変更するには、[プロパティ]コンポーネントでアニメーションまたは曲線のプロパティを編集します。

[アニメーション]エディターに曲線の動作のプレビューが表示されます。

アニメーションの具体的なサンプルについては、14.5「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. スクリプト曲線によるアニメーションの追加

スクリプト曲線の出力の取得

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
ビューの名前を`FirstView`に変更し、このビューを開きます。

ステップ 2
[ツールボックス]からアニメーションを`FirstView`内にドラッグします。

ステップ 3
[アニメーションエディター]で、[アニメーション化プロパティ]の横にある+をクリックし、次に`FirstView`をクリックします。

メニューが展開されます。

ステップ 4
`FirstView`の下で`y`プロパティ、[スクリプト曲線]の順に選択します。

ステップ 5
[承認]をクリックします。

アニメーションが[アニメーションエディター]に追加されます。

ステップ 6
新しいアニメーションの名前を`ScriptCurveMonitoring`に変更します。

ステップ 7

[プロパティ]コンポーネントで、[デフォルトウィジェットプロパティ]に移動し、`curve`プロパティの横にある{ } をクリックします。EB GUIDEスクリプトエディターが表示されます。

次のEB GUIDEスクリプトを入力します。

```
function(v:diff::int, v:t_anim::int)
{
f:trace_string("Diff : "+ f:int2string(v:diff) + " t_anim: " + f:int2string(v:t_anim))
       0::int
}
```

[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[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 9
メニューで、`Conditional script`を選択します。

`Conditional script`タイプのユーザー定義プロパティがアニメーションに追加されます。名前を`PlayAnimation`に変更します。

ステップ 10
`PlayAnimation`プロパティの横にある{ } をクリックします。

EB GUIDEスクリプトエディターが表示されます。

ステップ 11
[トリガー時]セクションに次のEB GUIDEスクリプトを入力します。

```
function(v:arg0::bool)
{
f:animation_play(v:this)
false
}
```

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。

EB GUIDE Monitorで、[ロガー]コンポーネントを参照します。`v:diff`は、アニメーションが16ミリ秒ごとに実行されることを示しています。`v:t_anim`は、開始してからこの時点までにアニメーションが実行された時間を示しています。以下の図をご覧ください。

traceString 'Diff : 0 t_anim: 0'	12:21:07.779
traceString 'Diff : 16 t_anim: 16'	12:21:07.779
traceString 'Diff : 16 t_anim: 32'	12:21:07.779
traceString 'Diff : 17 t_anim: 49'	12:21:07.779
traceString 'Diff : 19 t_anim: 68'	12:21:07.779
traceString 'Diff : 16 t_anim: 84'	12:21:07.779
traceString 'Diff : 16 t_anim: 100'	12:21:07.779

図8.2 EB GUIDE Monitorメッセージ

8.1.2.9. 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、15.9.2.1「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。レンダラー(OpenGL ESバージョン2.0以降向け)では、`.png`ファイルおよび `.jpg`ファイルRGBイメージは、グレイ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。グレイ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。

アルファマスク機能は9-patchイメージには適用されません。9-patchイメージは PNGおよびJPEGファイル形式と同じ方法で処理されます。

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ファイルをインポートする

ビューに`.psd`ファイルを追加する

バックグラウンド情報については、6.17「Photoshopファイル形式のサポート」をご覧ください。

前提条件:

- `.psd`ファイルが`$GUIDE_PROJECT_PATH/<project name>/resources`にあること。
- 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
[アセット]コンポーネントで、対応する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。

インポートステータスを示す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。

インポートに成功した場合、[ナビゲーション]コンポーネントに新しいウィジェットツリーが表示されます。最上位要素は、`.psd`ファイルの名前が反映されたコンテナーです。

`$GUIDE_PROJECT_PATH/<project name>/resources`フォルダー内にサブフォルダーが作成されます。ここには、抽出されたイメージのすべてが含まれます。

注記

複数行

テキストレイヤーに1行を超えるテキストが含まれている場合は、[複数行] ウィジェット機能を追加してプロパティを適宜編集する必要があります。詳細については、8.3.1「ウィジェット機能の追加」および15.10.1.5「複数行」をご覧ください。

8.1.5. .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

イメージをインポートせずに.psdファイルから抽出すると、ウィジェットツリーは作成されません。バックグラウンド情報については、6.17「Photoshopファイル形式のサポート」をご覧ください。

前提条件:

- .psdファイルが$GUIDE_PROJECT_PATH/<project name>/resourcesまたはサブフォルダー内にあること。

ステップ 1
[アセット]コンポーネントで、対応するフォルダーを選択します。

ステップ 2
.psdファイルを右クリックし、[.psdファイルからイメージを抽出]を選択します。

インポートステータスを示す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。

$GUIDE_PROJECT_PATH/<project name>/resourcesフォルダーに、抽出されたすべてのイメージが含まれるサブフォルダーが作成されます。このサブフォルダーの名前は、.psdファイルの名前を反映したものになっています。

8.1.6. IBLファイルのインポート

バックグラウンド情報については、6.13「イメージベースドライティング」をご覧ください。

IBLファイルのインポート

IBLファイルをインポートするには、まずIBLファイルを.ebibl形式に変換する必要があります。この処理はIBLGeneratorを使用して行います。

前提条件:

- EB GUIDEプロジェクトが作成されます。
- IBLファイルが.pfmまたは.hdr形式であること。
- 管理者権限があること。
- シーングラフウィジェットがEB GUIDEモデルに追加されていること。
- 3Dファイルがインポートされていること。

ステップ 1
管理者としてコマンドラインプロンプトを開きます。

ステップ 2
IBLGeneratorのインストールパスに移動します。これはtoolsディレクトリ内のEB GUIDE Studioインストールディレクトリ($GUIDE_INSTALL_PATH\tools\IBLGenerator)にあります。

ステップ 3
ファイルを.ebibl形式に変換するコマンドを入力します。そのコマンドは次のようになります。

```
IBLGenerator.exe -i yourfile.hdr -o yourfile.ebibl -p latlong -q 1
```

- -i: 入力ファイル名
- -o: 出力ファイル名
- -p: パラメタリゼーションタイプ。その他のタイプには立方体および球体があります。
- -q: 品質レベル。品質レベルは、1が最低で10が最高の品質となります。品質レベルが高くなると、かなりの処理時間が必要になります。

.ebiblファイルは、指定したディレクトリに配置されます。

ティップ

IBLGeneratorヘルプ
IBLGeneratorのオプションの一覧を確認するには、-hパラメータを指定して実行します。

```
IBLGenerator.exe -h
```

ステップ 4
.ebiblファイルをEB GUIDEモデルのリソースディレクトリにコピーします。これで、.ebiblファイルをシーングラフノードで使用できるようになります。

ステップ 5
EB GUIDE Studioの[ツールボックス]コンポーネントで、[3Dウィジェット]からイメージベースドライトをシーングラフノード内にドラッグします。

ステップ 6
[プロパティ]コンポーネントで、iblプロパティの横にある.ebiblファイルを選択します。

IBLファイルのインポートが終了します。

ティップ

IBLのための最良の結果

最良の結果を得られるよう、イメージベースドライトのプロパティを適合させ、PBR GGX材質またはPBR Phong材質を使用します。

イメージベースドライティングのみでシーンを照らすには、その他すべての光源を無効にします。

8.1.7. 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。

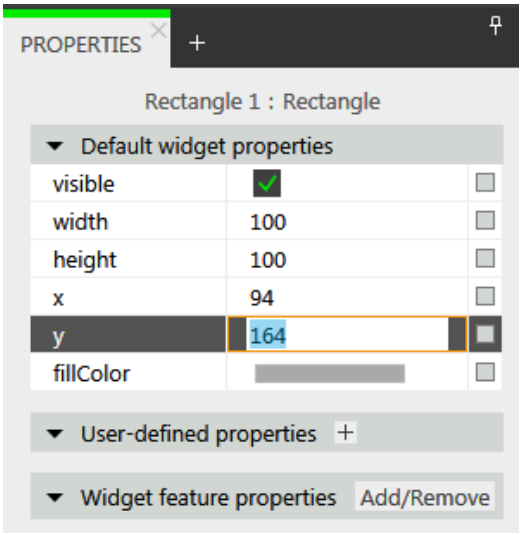

図8.3 四角形のプロパティ

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

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

ステップ 4
テキストボックスの外側をクリックします。

コンテンツエリアで、入力したサイズでウィジェットが表示されます。

注記

負の値

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

ティップ

別の方法

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

8.2.3. ウィジェットプロパティ間のリンク設定

ウィジェットプロパティ間のリンク設定

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

同一ビュー内にあるウィジェットのプロパティのみリンクできます。

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

前提条件:

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

ステップ 1
四角形をクリックします。

[プロパティ]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。

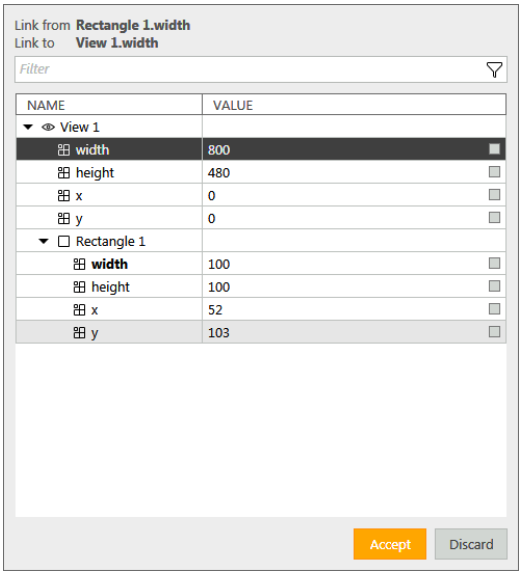

図8.4 ウィジェットプロパティ間のリンク設定

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。

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
データプールアイテムの横にある[値]列を選択し、{} をクリックします。

EB GUIDE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、プロパティ名を右クリックします。

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。

8.2.7. タイプリストのプロパティの編集

タイプリストのプロパティの編集

タイプリストのプロパティの詳細については、15.3.12「リスト」をご覧ください。

リソース管理の詳細については、6.18「リソース管理」および6.6「グラフィカルユーザーインターフェイスのコンポーネント」をご覧ください。

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。
- ビューステートにウィジェットが含まれていること。
- ウィジェットにタイプリストのプロパティがあること。

ステップ 1
[プロパティ]コンポーネントで、タイプリストのプロパティを選択し、その横にあるボタンをクリックします。

[編集]ダイアログが開きます。

ステップ 2
リストに新しいエントリーを追加するには、[追加]ボタンをクリックします。

新しい行が追加されます。

ステップ 3
値を編集するには、[値]列にある対応する行をクリックします。

注記

[アセット]コンポーネント
プロパティがタイプフォントリスト、IBLリスト、イメージリスト、またはメッシュリストのものである場合、[編集]ダイアログには追加の[アセット]コンポーネントが表示されます。[アセット]コンポーネントからアセットを[値]列の対応する行にドラッグアンドドロップできます。

ステップ 4
リストの編集が終了したら、[承認]ボタンをクリックします。

ダイアログが閉じられます。

8.2.8. ウィジェットの順序および可視性の管理

EB GUIDE Studioでは、ウィジェットの順序、レイヤー、または可視性を定義するなど、いくつかの操作が可能です。以下の操作が可能であり、それらを適用できるかどうかはユースケースによって異なります。

ビューにおいて他のウィジェットよりも相対的に上位にあるウィジェットを変更する場合は、[ナビゲーション]ツリーを使用します。[ナビゲーション]ツリーでは、ウィジェットの順序により、どのウィジェットが最上位に表示されるかが決まります。同じ分岐上にあるウィジェットどうしでは、ツリーのより高い位置にあるウィジェットが下に表示されます。異なる分岐上にあるウィジェットどうしでは、より高いレベルのウィジェットが上に表示されます。

ポップアップを他のどの要素よりも上に表示する場合は、動的ステートマシンを使用します。手順については、14.1「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。

ユーザーの選択に応じて表示が変わる複雑なメニューがある場合は、[子の可視性の選択]ウィジェット機能を使用します。このウィジェット機能により、ウィジェットの子ウィジェットの可視性を制御できます。この機能は、影響を受けるウィジェットの`Visibility`プロパティを上書きします。この機能の使用法としては、次の2つが想定されます。

- 1つの子ウィジェットを可視化します。このユースケースでは、ウィジェットツリーにウィジェットのインデックスのみが必要です。手順については、「1つの子ウィジェットの可視化」をご覧ください。
- 複数の子ウィジェットを可視化します。このユースケースでは、子ウィジェットのグループを定義し、`Value`を識別する1つのグループを指定する必要があります。手順については、「複数の子ウィジェットの可視化」をご覧ください。

1つの子ウィジェットの可視化

ウィジェット機能の詳細については、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。

前提条件:

- EB GUIDEプロジェクトがEB GUIDE Studioで開かれていること。
- 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
[子の可視性選択]ウィジェット機能を親ウィジェットに追加します。

ステップ 2
[ナビゲーション]コンポーネントで、子ウィジェットがどの位置にあるか確認します。0 が最初の位置になります。

ステップ 3
[プロパティ]コンポーネントの`containerIndex`テキストボックスに、子ウィジェットの位置を入力します。

これで、この子ウィジェットは表示される唯一の子ウィジェットになります。

複数の子ウィジェットの可視化

ウィジェット機能の詳細については、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。

前提条件:

- EB GUIDEプロジェクトがEB GUIDE Studioで開かれていること。
- EB GUIDEモデルに3つより多いウィジェットが含まれていること。

ステップ 1
[子の可視性選択]ウィジェット機能を親ウィジェットに追加します。

ステップ 2
[プロパティ]コンポーネントで、`containerMapping`を選択してから をクリックします。

エディターが開きます。

ステップ 3
ウィジェットのグループと子ウィジェットとの間のマッピングを定義します。

[インデックス]列には、子ウィジェットのインデックスが含まれます。[値]列に、子ウィジェットのマッピング先にするグループを入力します。

ステップ 4
[承認]をクリックします。

ステップ 5
containerIndexテキストボックスに、可視化する必要があるグループのValueを入力します。

これで、このグループのウィジェットが表示されます。このグループにマッピングされていないウィジェットは表示されません。

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。

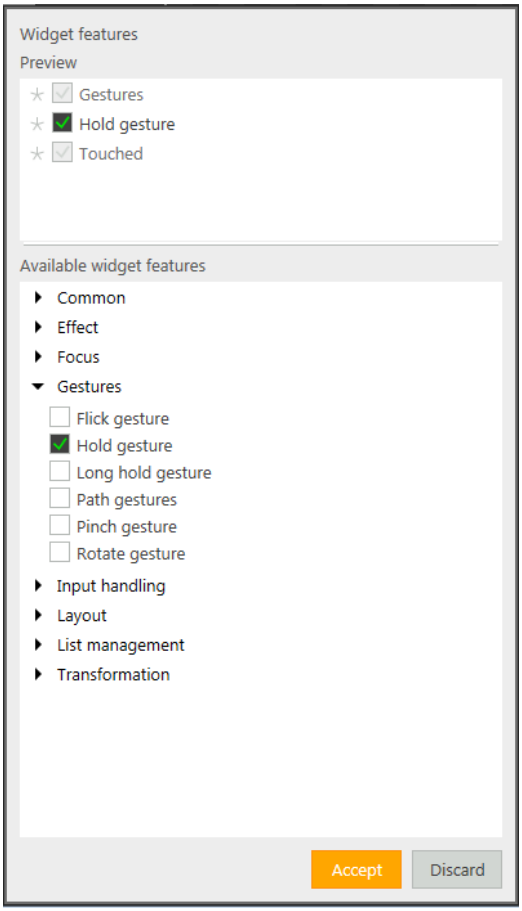

図8.5 ウィジェット機能ダイアログ

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、15.10「ウィジェット機能」をご覧ください。

チュートリアルについては、以下をご覧ください。

- 14.3「チュートリアル: パスジェスチャーをモデル化する」
- 14.4「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」
- 14.2「チュートリアル: EB GUIDEスクリプトを使用したボタン動作のモデル化」

8.3.2. ウィジェット機能の削除

ウィジェット機能の削除

前提条件:

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

ステップ 1
[ナビゲーション]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。

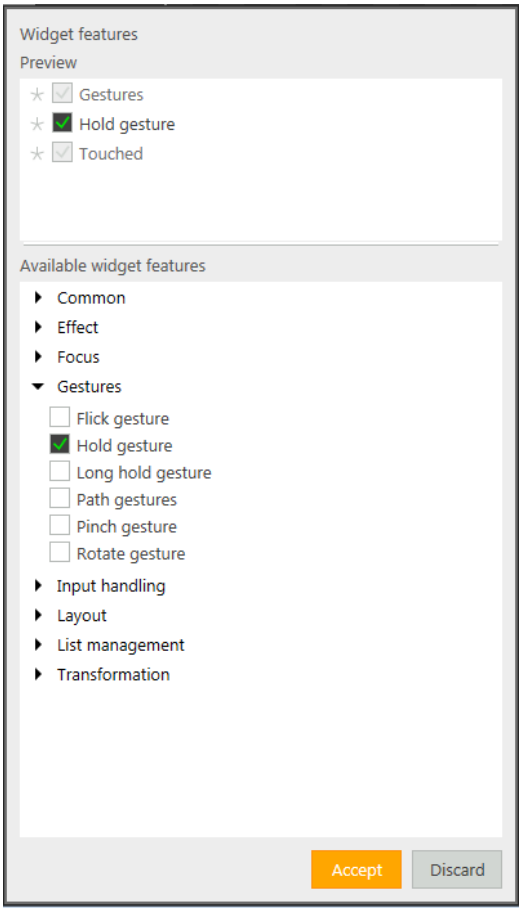

図8.6 ウィジェット機能ダイアログ

ステップ 3
[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. フォント設定の変更

8.4.1. ラベルのフォントの変更

注記

テキストの高さ、行の高さ、および行間の計算

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。フォント設定を変更する際にはこれを参考にしてください。詳しくは、8.4.2「行間の変更」をご覧ください。

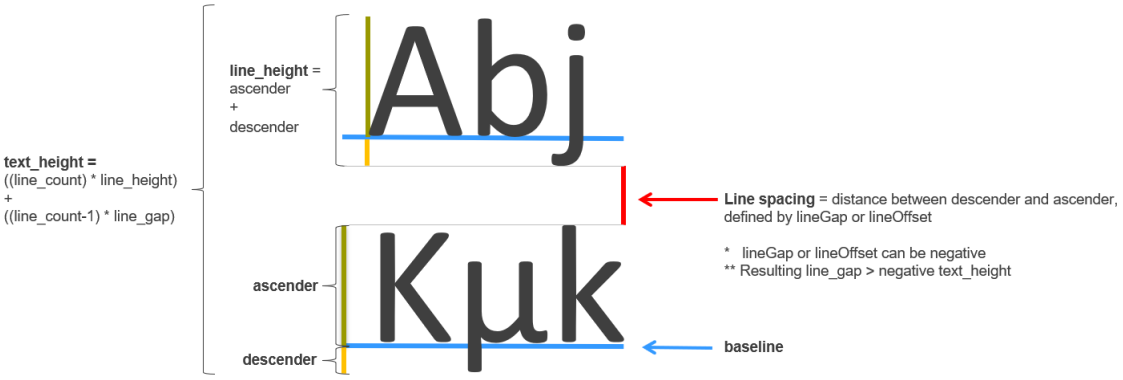

図8.7 テキストの高さ、行の高さ、および行間の計算

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`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、6.18.1「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。

マルチフォントサポートの使用方法については、8.4.3「マルチフォントサポートの管理」をご覧ください。

8.4.2. 行間の変更

フォントには、デフォルトで特定の行間が定義されています。[フォントメトリック]ウィジェット機能の`lineGap`プロパティを使用すると、この行間をラベル単位で変更できます。[複数行]ウィジェット機能もこのラベルに追加する場合は、`lineOffset`プロパティを使って行間を追加で定義することができます。同じラベルに両方のプロパティを設定することができます。その場合、プロパティの設定は相殺されることに留意してください。

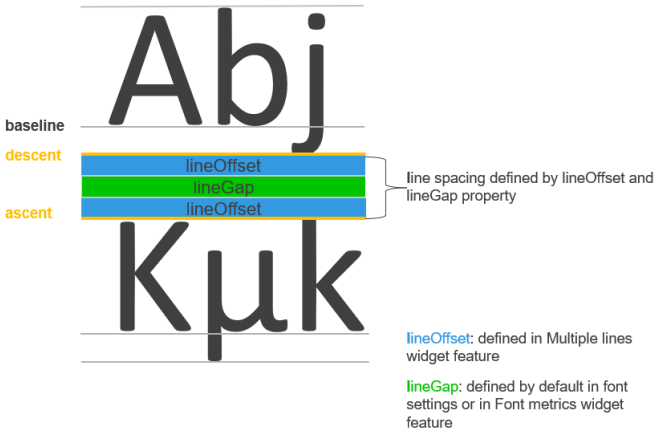

図8.8 行間の定義:`lineOffset`および `lineGap`

8.4.2.1. デフォルトの行間の変更

フォントには、デフォルトで特定の行間が定義されています。[フォントメトリック]ウィジェット機能の`lineGap`プロパティを使用すると、行間をラベル単位で変更できます。

注記

`lineOffset`プロパティの対話処理

[複数行]ウィジェット機能もこのラベルに追加する場合、`lineOffset`プロパティを使って行間を変更することができ、その際は2つのプロパティの設定が相殺されることに留意してください。

[フォントメトリック]ウィジェット機能の`lineGap`プロパティの変更

フォントのデフォルトの行間を変更するには、以下の操作を行います。

前提条件:

- ビューにラベルが含まれていること。
- タイプフォントのデータプールアイテムのプロパティが存在します。

ステップ 1
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[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]で、[共通]カテゴリを展開して[フォントメトリックス]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フォントに追加され、[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、`lineGap`プロパティを選択し、値を変更します。

8.4.2.2. 複数行の行間の変更

[複数行]ウィジェット機能の`lineOffset`プロパティの変更

[複数行]ウィジェット機能を使用すると、ラベル内の改行が可能になります。複数行の行間を変更するには、以下の操作を行います。

注記

`lineGap`プロパティの対話処理
`lineOffset`プロパティを変更すると、[フォントメトリック]ウィジェット機能の`lineGap`プロパティで定義された行間に影響があり、2つのプロパティの効果が相殺される場合があることに留意してください。

前提条件:

- ビューにラベルが含まれ、そのラベルに[複数行]ウィジェット機能が追加されていること。
- タイプフォントのデータプールアイテムのプロパティが存在します。

ステップ 1
ラベルを選択します。

ステップ 2
[プロパティ]コンポーネントでは、[複数行]ウィジェット機能に移動します。

ステップ 3
`lineOffset`プロパティの値を変更します。

8.4.3. マルチフォントサポートの管理

詳しくは、6.18.1.2「マルチフォントサポート」をご覧ください。

タイプフォント向けマルチフォントサポートの追加

前提条件:

- `$GUIDE_PROJECT_PATH/resources`には複数のフォントが用意されています。
- タイプフォントのデータプールアイテムのプロパティが存在します。このプロパティはスクリプト値ではありません。

ステップ 1
マルチフォントサポートを追加するには、次の操作を行います。

- マルチフォントサポートをウィジェットプロパティに追加する場合は、[プロパティ]コンポーネントに移動します。
- マルチフォントサポートをデータプールアイテムに追加する場合は、[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[マルチフォントサポートの追加]をクリックします。

プロパティまたはデータプールアイテムの下に表が表示されます。この表には、デフォルトのマルチフォント値が1つあります。

ステップ 4
新しいマルチフォント値を追加するには、+ボタンをクリックします。

新しい行が追加され、デフォルトフォントに基づいた値が設定されます。

ステップ 5
この行では、以下の編集および定義ができます。

- [優先度]列で、フォントエントリーの優先度を定義します。
- [フォント]列で、サイズとフォントを定義します。
- [範囲]列で、選択されているフォントの影響を受けるUnicode文字を定義します。

ステップ 6
必要なすべてのフォント値が追加されるまで、手順2～5を繰り返します。

ティップ

マルチフォントサポートの削除

マルチフォントサポートを削除するには、ボタンをもう一度クリックします。開いたメニューで、[マルチフォントサポートの削除]をクリックします。

タイプフォントリスト向けマルチフォントサポートの追加

タイプリストのプロパティの操作方法については、8.2.7「タイプリストのプロパティの編集」をご覧ください。

前提条件:

- `$GUIDE_PROJECT_PATH/resources`には複数のフォントが用意されています。
- タイプフォントリストのプロパティまたはデータプールアイテムが存在し、そこに少なくとも1つのフォントリストエントリーがあります。

ステップ 1
マルチフォントサポートを追加するには、次の操作を行います。

- マルチフォントサポートをウィジェットプロパティに追加する場合は、[プロパティ]コンポーネントに移動します。
- マルチフォントサポートをデータプールアイテムに追加する場合は、[データプール]コンポーネントに移動します。

ステップ 2
プロパティまたはデータプールアイテムを選択し、その横にあるボタンをクリックします。

[編集]ダイアログが開きます。

ステップ 3
タイプフォントのエントリー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 4
メニューの[マルチフォントサポートの追加]をクリックします。

プロパティまたはデータプールアイテムの下に表が追加されます。この表には、デフォルトのマルチフォント値が1つあります。

ステップ 5
新しいマルチフォント値を追加するには、ボタンをクリックします。

新しい行が追加され、デフォルトフォントに基づいた値が設定されます。

ステップ 6
この行では、以下の編集および定義ができます。

- [優先度]列で、フォントエントリーの優先度を定義します。
- [フォント]列で、サイズとフォントを定義します。
- [範囲]列で、選択されているフォントの影響を受けるUnicode文字を定義します。

ステップ 7
必要なすべてのフォント値が追加されるまで、手順3～6を繰り返します。

ティップ

マルチフォントサポートの削除

マルチフォントサポートを削除するには、ボタンをもう一度クリックします。開いたメニューで、[マルチフォントサポートの削除]をクリックします。

ステップ 8
リスト内のエントリーの編集が終了したら、[承認]ボタンをクリックします。

ダイアログが閉じられます。

8.5. 言語サポートとの連携

ランタイム中にEB GUIDEモデルの言語を変更するには、言語サポートと言語依存テキストを追加します。

EB GUIDEモデルの言語については、6.14.2「EB GUIDEモデルの言語」をご覧ください。

8.5.1. EB GUIDEモデルへの言語の追加

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

次の手順は、EB GUIDEモデルに言語を追加する方法を示しています。

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで、[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。リスト内の最初の言語は、常にデフォルト言語となります。

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。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 4
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EB GUIDE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にある[言語]ドロップダウンリストボックスで、新しい言語を選択できます。

ランタイム中に言語を切り替えると、異なるデータプール値の効果を確認できます。詳しくは、14.6「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。

8.5.2. データプールアイテムに言語サポートを追加する

データプールアイテムに言語サポートを追加する

次の手順は、EB GUIDEモデルのデータプールアイテムに言語サポートを追加する方法を示しています。

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。
- 少なくとも2つの言語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 language 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、言語サポートがこのデータプールアイテムに追加され、各言語に異なる値を定義できるようになったことを示します。

データプールアイテムに言語サポートが追加されました。これで、このデータプールアイテムで言語依存の値を定義できます。詳しくは、14.6「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。

8.5.3. 言語の削除

注記 デフォルト言語

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。

8.6. 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。

スキンの詳細については、6.21「スキン」をご覧ください。

8.6.1. 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

スキンの名前を入力します。

新しいスキンがEB GUIDE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にある[スキン]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。

8.6.2. 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、各スキン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
表内の各スキンに値を定義します。

8.6.3. 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。

8.6.4. 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。

8.7. ビュー遷移のアニメーション化

8.7.1. 開始アニメーションの追加

開始アニメーションの追加

このセクションでは、開始アニメーションをビューステートに追加する手順を説明します。この手順は、終了アニメーション、ポップアップオンアニメーション、およびポップアップオフアニメーションにも適用されます。詳細については、6.2.2「ビュー遷移のアニメーション」および15.9.1「ビュー」をご覧ください。

前提条件:

- ビューステートとビューがEB GUIDEモデルに追加されます。
- [VTA]コンポーネントが開いていること。

ステップ 1
[VTA]コンポーネントで、＋をクリックします。

ステップ 2
コンテキストメニューで、[開始アニメーション]をクリックします。

[アニメーション]エディターが開きます。

[プロパティ]コンポーネントに、追加した開始アニメーションのプロパティが表示されます。

ステップ 3
依存ビューで利用可能なすべてのウィジェットプロパティをアニメーション化します。

[アニメーション]エディターで開始アニメーションを定義します。

8.7.2. 変更アニメーションの追加

変更アニメーションの追加

このセクションでは、変更アニメーションをビューステートまたはビューテンプレートに追加する手順を説明します。

前提条件:

- [VTA]コンポーネントが開いていること。
- [メイン]ステートマシンに2つのビューステートが含まれていること。

ステップ 1
[メイン]ステートマシンで`View state 1`を選択します。

ステップ 2
[VTA]コンポーネントで、+をクリックします。

ステップ 3
コンテキストメニューで、[変更アニメーション]を選択します。

ダイアログが開きます。

ステップ 4
`View 2`を選択します。

ステップ 5
[承認]をクリックします。

出力先ビューの名前が変更アニメーション名の横に表示されます。

[アニメーション]エディターが開きます。

[プロパティ]コンポーネントに、追加した変更アニメーションのプロパティが表示されます。

出力元ビューがプロジェクトエディターに表示されます。

ステップ 6
出力先ビューを編集するには、[VTA]コンポーネントで  をクリックします。

ステップ 7
アニメーションプロパティを追加するには、[アニメーション]エディターの＋をクリックし、それぞれのカテゴリに追加します。

依存ビューで利用可能なすべてのウィジェットプロパティをアニメーション化します。

8.7.3. アニメーションの再配置

アニメーションの再配置

このセクションでは、ビューステートまたはビューテンプレートのアニメーションを再配置する手順を説明します。この手順は、終了アニメーション、変更アニメーション、ポップアップオンアニメーション、およびポップアップオフアニメーションにも適用されます。

前提条件:

- [VTA]コンポーネントが開いていること。
- 複数の開始アニメーションが追加されていること。

ステップ 1
[VTA]コンポーネントで、最初に開始する開始アニメーションの横にある優先順位テキストボックスを選択します。

ステップ 2
開始アニメーションの値を0に変更します。

開始アニメーションの順序が変更されます。編集した開始アニメーションが最初に開始され、次に続くすべてのエントリーは1つずつ値が増えます。

値0は、そのアニメーションが最初に再生されるように評価されることを意味します。その後の値は、後に続く順序でアニメーションが再生されるように評価されることを意味します。優先順位が最も高く、なおかつ条件を満たす開始アニメーションだけが再生されます。

8.8. ウィジェットの再利用

テンプレートの詳細については、6.24.4「ウィジェットテンプレート」をご覧ください。

8.8.1. テンプレートの追加

テンプレートの追加

ステップ 1
[テンプレート]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。このため、EB GUIDEではテンプレートからテンプレートを作成できます。

ティップ

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

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

8.8.2. 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

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

8.8.3. 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。

8.8.4. テンプレートの削除

テンプレートの削除

ステップ 1
[テンプレート]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。

8.9. アンチエイリアスの有効化

バックグラウンド情報については、6.3「アンチエイリアス」をご覧ください。

8.9.1. アンチエイリアスのグローバルな有効化

アンチエイリアスのグローバルな有効化

前提条件:

- EB GUIDEモデルがあること。

ステップ 1
[プロジェクトセンター]で、[設定] > [プロファイル]の順に選択します。

[プロファイル]メニューが開きます。

ステップ 2
[シーン]タブで、`antiAliasing`ドロップダウンリストボックスから、そのシーンに設定するアンチエイリアスモードを選択します。

このアンチエイリアスモードが、EB GUIDEモデル全体に設定されます。

8.9.2. シーングラフへのアンチエイリアスの有効化

アンチエイリアスのグローバルな有効化

前提条件:

- EB GUIDEモデルにシーングラフが含まれていること。

ステップ 1
シーングラフを選択します。

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
[3D]カテゴリを展開し、[アンチエイリアスモード]ウィジェット機能を選択し、[承認]をクリックします。

[プロパティ]コンポーネントに、[アンチエイリアスモード]ウィジェット機能が表示されます。

ステップ 4
`antiAliasing`ドロップダウンリストボックスで、このシーングラフに指定するモードを選択します。

このシーングラフのアンチエイリアスモードが設定されます。

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

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]コンポーネントの名前空間ドロップダウンリストボックスで、イベントの追加先となる名前空間を選択します。

ステップ 2
+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 3
イベントの名前を変更します。

ステップ 4
イベントIDを変更するには、[プロパティ]コンポーネントに移動し、`Event ID`テキストボックスにIDを入力します。

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キーとの 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+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。

EB GUIDE TFイベントへの対応

前提条件:

- イベントグループが追加されています。
- 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. イベントへのキーのマッピング

イベントを発行するには、キーボードのキーを押すか、または対象デバイスのキーを押すか回転ボタンを使用します。

キーボードのキーを押すと反応するようにするには、モデルでこれらのキーイベントのマッピングを定義する必要があります。

各キーに対して、EB GUIDE GTFはC++ヘッダーファイルで数値コードを定義します。コード番号については、`$INSTALL_PATH$/platform/win64/include/gtf/displayfactory/inputmapper/KeyConstants.h`をご覧ください。

イベントへのキーのマッピング

イベント処理の詳細については、6.10「イベント処理」および15.5「イベント」をご覧ください。

前提条件:

- イベントグループ[キー](ID 10)が[プロジェクトセンター] > [設定] > [イベントグループ]に追加されていること。
- イベントが追加されています。

ステップ 1
`KeyConstant.h`ファイルにマッピングするキーの16進コード番号を検索します。

ステップ 2
10進コード番号を計算します。

ステップ 3
[イベント]コンポーネントの[グループ]列で、[キー](ID 10)を選択します。

ステップ 4
[プロパティ]コンポーネントのEvent IDテキストボックスに、計算された10進コード番号を入力します。

これで、選択したキーがイベントにマップされます。

例**9.1**
イベントへの**F1**キーのマッピング

F1の内部16進コード番号は12です。

12の10進コード番号は18です。

[プロパティ]コンポーネントに移動し、Event IDテキストボックスに18と入力します。

これで、F1キーがイベントにマップされます。

注記

EB GUIDE Monitorで表示されないイベント

イベント自体はEB GUIDE Monitorに表示されませんが、イベントによってトリガーされるEB GUIDEスクリプトは反応します。

10進コード番号の詳細については、15.5.1「キーイベントの10進コード」をご覧ください。

9.5. 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、対応する名前空間を選択します。

ステップ 2
イベントを右クリックし、コンテキストメニューの[削除]を選択します。

イベントが削除されます。

9.6. 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]コンポーネントの名前空間ドロップダウンリストボックスで、データプールアイテムの追加先となる名前空間を選択します。

ステップ 2
＋をクリックします。

メニューが展開されます。

ステップ 3
メニューでデータプールアイテムのタイプをクリックします。

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

ステップ 4
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。

9.7. 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列を選択し、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。

9.8. 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スキン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。
- データプールアイテムはマルチフォントをサポートしません。

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
データプールアイテムの横にある[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

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。

9.9. 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。

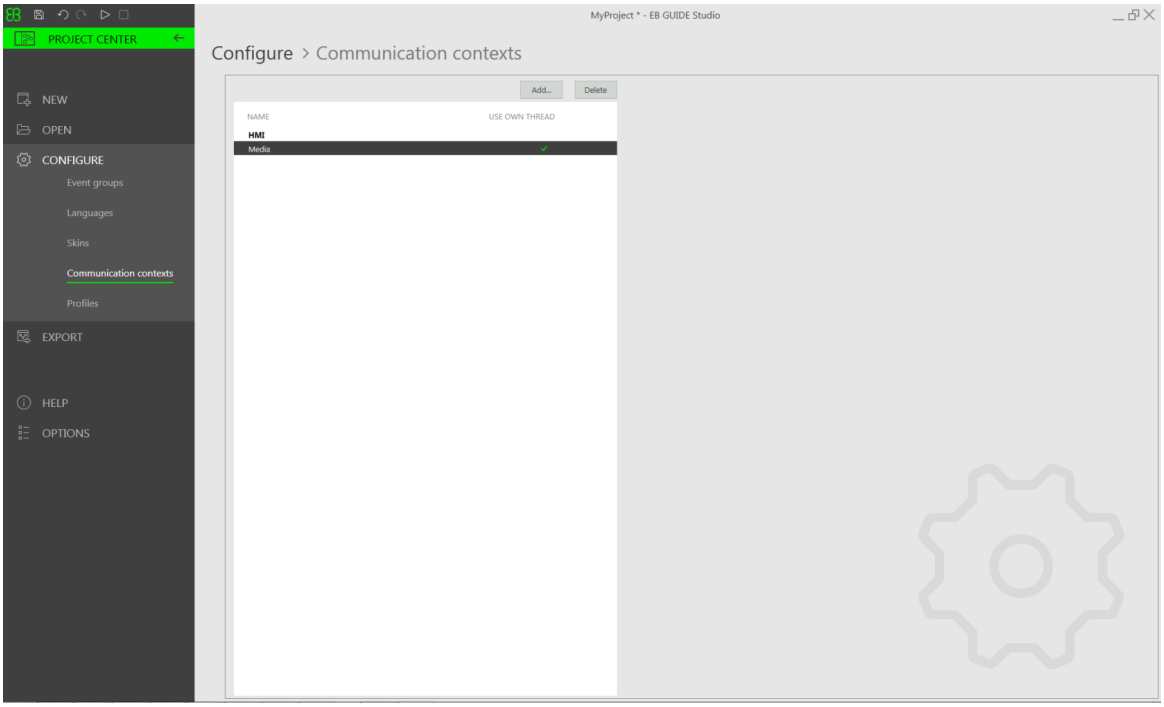

図9.1 通信コンテキストMedia。

9.10. 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キン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。

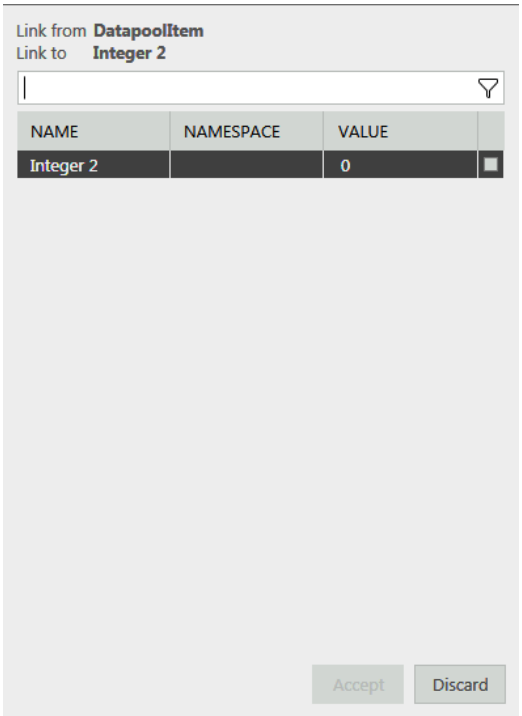

図9.2 データプールアイテム間のリンク設定

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。

9.11. 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、対応する名前空間を選択します。

ステップ 2
データプールアイテムを右クリックし、コンテキストメニューの[削除]を選択します。

データプールアイテムが削除されます。

9.12. モデルインターフェースへのモデル要素の追加

モデルインターフェースの詳細については、6.16「モデルインターフェース」をご覧ください。

モデルインターフェースの操作方法については、10.9「モデルインターフェースの操作」をご覧ください。

9.12.1. モデルインターフェースへのイベントの追加

モデルインターフェースへのイベントの追加

前提条件:

- EB GUIDEモデルにイベントが含まれていること。

ステップ 1
[イベント]コンポーネントに移動します。

ステップ 2
1つのデフォルトモデルインターフェースしかない場合は、イベントを右クリックし、[モデルインターフェースへの追加]を選択します。

複数のモデルインターフェースがある場合は、イベントを右クリックし、[モデルインターフェース]を選択し、このイベントを含めるモデルインターフェースを選択します。

これでイベントがモデルインターフェースに追加されます。イベントの左側にあるカラーバーは、モデルインターフェースに追加済みであることを示します。

9.12.2. モデルインターフェースへのデータプールアイテムの追加

注記

条件スクリプト
条件スクリプト型のデータプールアイテムは、モデルインターフェースに追加できません。

モデルインターフェースへのデータプールアイテムの追加

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。

ステップ 1
[データプール]コンポーネントに移動します。

ステップ 2
1つのデフォルトモデルインターフェースしかない場合は、データプールアイテムを右クリックし、[モデルインターフェースへの追加]を選択します。

複数のモデルインターフェースがある場合は、アイテムを右クリックし、[モデルインターフェース]を選択し、このアイテムを含めるモデルインターフェースを選択します。

これでデータプールアイテムがモデルインターフェースに追加されます。データプールアイテムの左側にあるカラーバーは、モデルインターフェースに追加済みであることを示します。

9.13. 名前空間の操作

名前空間の詳細については、6.6「グラフィカルユーザーインターフェイスのコンポーネント」および6.15「名前空間」をご覧ください。

9.13.1. 名前空間の追加

名前空間の追加

新しい名前空間を子として、ルート名前空間または任意の既存名前空間に追加できます。

ステップ 1
[名前空間]コンポーネントは、EB GUIDEのデフォルトのウィンドウレイアウトでは使用できないので、次の操作を行います。

コマンドエリアで[レイアウト]をクリックし、[名前空間]を選択します。

[名前空間]コンポーネントが表示されます。

ステップ 2
[名前空間]コンポーネントで+をクリックします。

名前空間がツリーに追加されます。

ステップ 3
名前空間の名前を変更します

ティップ

名前空間の移動
名前空間を移動するには、ルート名前空間または別の名前空間にドラッグします。名前空間のネーミングルールに従うことで名前の衝突を回避します。

9.13.2. 名前空間へのモデル要素の追加

名前空間へのイベントの追加方法については、9.1「イベントの追加」をご覧ください。

名前空間へのデータプールアイテムの追加方法については、9.6「データプールアイテムの追加」をご覧ください。

9.13.3. 名前空間どうしでのモデル要素の移動

名前空間どうしでのモデル要素の移動

前提条件:

- モデル要素、イベント、またはデータプールアイテムが名前空間に追加されていること。
- 少なくとも2つの名前空間が存在すること。

ステップ 1
イベントを移動するには、[イベント]コンポーネントに移動します。

データプールアイテムを移動するには、[データプール]コンポーネントに移動します。

ステップ 2
対応する名前空間を選択し、モデル要素を右クリックします。

ステップ 3
コンテキストメニューで、[名前空間への移動...]を選択します。

ダイアログが開きます。

ステップ 4
移動先の名前空間を選択し、[承認]をクリックします。

モデル要素が移動先の名前空間に移動します。

注記

モデル要素の移動
また、モデル要素を別の名前空間にドラッグすることもできます。

9.13.4. 名前空間の削除

警告

名前空間の削除
名前空間を削除すると、この名前空間に含まれているすべてのモデル要素も削除されます。

注記

ルート名前空間
ルート名前空間は削除できません。

名前空間の削除

前提条件:

- 名前空間が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、名前空間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。

名前空間が削除されます。

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 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

注記

無効なプロジェクト

EB GUIDEプロジェクトが有効でない場合、EB GUIDE Studioではそのプロジェクトを開くことができません。エラーメッセージが表示され、エラーを説明するログファイルが`$GUIDE_PROJECT_PATH/<project name>/<project name>_LoadingErrorLog.txt`に作成されます。

詳しくは、6.8.2「EB GUIDEプロジェクトの検証基準」をご覧ください。

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プロジェクトファイルを選択します。EB GUIDE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プロジェクトファイルをダブルクリックします。

プロジェクトがEB GUIDE Studioに開かれます。

10.2.2. プロジェクトをEB GUIDE Studioに開く

プロジェクトをEB GUIDE Studioに開く

前提条件:

- EB GUIDE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プロジェクトファイルを選択します。EB GUIDE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. EB GUIDE StudioでのEB GUIDEモデルの検証

EB GUIDE StudioでのEB GUIDEモデルの検証

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

詳しくは、6.8.2「EB GUIDEプロジェクトの検証基準」をご覧ください。

ステップ 1
[問題検出]コンポーネントを展開するには、[問題]をクリックします。

ステップ 2
[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。

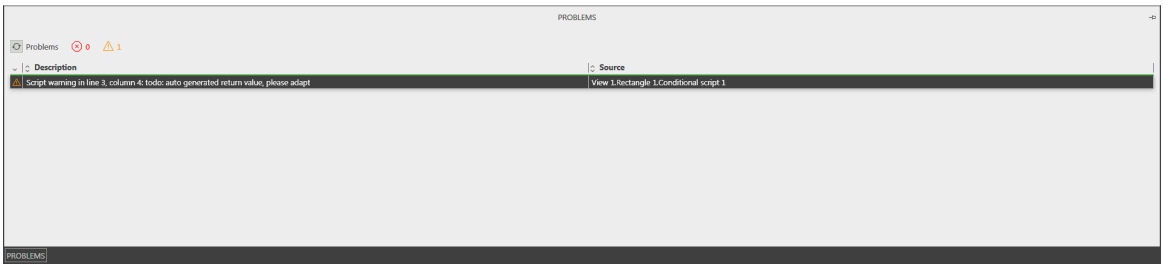

図10.1 問題検出コンポーネント

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。

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. EB GUIDE Studio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

EB GUIDE Studioの表示言語の変更

ステップ 1
 をクリックします。

プロジェクトセンターが表示されます。

ステップ 2
ナビゲーションエリアで[オプション]タブをクリックします。

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

ステップ 4
EB GUIDE Studioを再起動します。

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ます。

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

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

ステップ 3
[複製]をクリックします。

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

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

ステップ 5
[シミュレーションに使用]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ます。
- ナビゲーションエリアで、[設定] > [プロファイル]タブが選択されていること。
- `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>/bin`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。

使用例については、EB GUIDE GTFユーザーガイドの参照セクションをご覧ください。

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ます。
- ナビゲーションエリアで、[設定] > [プロファイル]タブが選択されていること。
- 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。

使用例については、EB GUIDE GTFユーザーガイドの参照セクションをご覧ください。

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ます。
- ナビゲーションエリアで、[設定] > [プロファイル]タブが選択されていること。

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、15.7「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

EB GUIDEでは、ユーザーの選択した言語でテキストを表示できます。そのためには、EB GUIDEモデルに言語サポートを追加します。次に、言語依存テキストを.xliffファイルにエクスポートし、それらのテキストを翻訳したうえで再びモデルにインポートします。

注記

プロジェクトおよび言語固有のID

プロジェクトと言語のすべてペアについて、英数字からなる一意のsourcelanguageidおよびtargetlanguageidが作成されます。これらのIDは、別のプロジェクトまたはターゲット言語からの.xliffファイルの意図しないインポートを防止します。また、それぞれのデータプールアイテムは各言語の一意の英数字IDを受け取ります。

テキストが翻訳のためにエクスポートされている間にEB GUIDE Studioモデルで言語またはデータプールアイテムが変更された場合でも、翻訳されたテキストは、固有のIDによって適切なデータプールアイテムおよび言語に割り当てることができます。すべてのデータプールアイテムが正しく割り当てられているかどうかを確認するには、インポートのログファイルをご覧ください。

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。

詳しくは、10.4.1.1「EB GUIDE StudioでのEB GUIDEモデルの検証」をご覧ください。

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、そのテキストをローカライズ担当のサービスプロバイダーに渡します。

前提条件:

- 翻訳対象言語がEB GUIDEモデルに追加されていること。詳しくは、8.5.1「EB GUIDEモデルへの言語の追加」をご覧ください。
- `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。詳しくは、8.5.2「データプールアイテムに言語サポートを追加する」をご覧ください。
- 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、翻訳対象とな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

エクスポートされたファイルの構造と内容

- プロジェクトセンターで選択する言語ごとに異なる`.xliff`ファイルがエクスポートされます。
- ソース言語はデフォルトの言語です。そのため、テキストがまだ翻訳されていない場合、`target-language`要素には常にソーステキストが含まれます。

EB GUIDEモデルおよび`.xliff`ファイルにおける言語依存テキストの詳細については、6.14「言語」をご覧ください。

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

EB GUIDE Studioを使用した言語依存テキストのインポート

前提条件:

- 選択されているEB GUIDEモデルの翻訳済み.xliffファイルが少なくとも1つは使用できること。
- 翻訳対象のデータプールアイテムとターゲット言語が依然として存在していること。
- 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
[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。

これで、言語サポートがあるすべてのデータプールアイテムに、対応する言語依存テキストが表示されます。インポートの詳細については、ログファイルをご覧ください。

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:

- 選択されているEB GUIDEモデルの翻訳済み`.xliff`ファイルが少なくとも1つは使用できること。
- 翻訳用に送信されたデータプールアイテムが依然として存在していること。
- EB GUIDEモデルにエラーおよび警告がないこと。

ステップ 1
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モデルは変更されます。インポートが失敗した場合、EB GUIDEモデルは変更されません。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. モデルインターフェースの操作

次のセクションでは、モデルインターフェースへのデータプールアイテムとイベントを追加し、このインターフェースをエクスポートしたりインポートしたりする手順について説明します。バックグラウンド情報については、6.16「モデルインターフェース」をご覧ください。

10.9.1. モデルインターフェースの作成

モデルインターフェースの作成

前提条件:

- EB GUIDE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、[設定] > [モデルインターフェース]の順にクリックします。

現在のプロジェクトのすべてのモデルインターフェースが一覧表示されます。

ステップ 3
＋をクリックします。

ユーザー定義のモデルインターフェースが、デフォルトの名前で作成されます。

ステップ 4
モデルインターフェースの名前を変更します。

ステップ 5
モデルインターフェースの横にカラーバーが表示されます。このバーは、イベントやデータプールアイテムの横にも表示されます。このバーの色で、それらが属するモデルインターフェースがわかります。バーの色を変更するには、インターフェースを右クリックし、[色の選択]を選択します。

10.9.2. モデルインターフェースのエクスポート

モデルインターフェースのエクスポート

前提条件:

- EB GUIDEプロジェクトが作成されます。
- モデルインターフェースがEB GUIDEプロジェクトに追加されていること。
- イベントまたはデータプールアイテムがモデルインターフェースに追加されていること。モデルインターフェースへのモデル要素の追加方法については、9.12「モデルインターフェースへのモデル要素の追加」をご覧ください。

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] > [モデルインターフェース]の順にクリックします。

現在のプロジェクトのすべてのモデルインターフェースが一覧表示されます。

ステップ 3
エクスポートするモデルインターフェースを選択します。

ステップ 4
マウスで

をクリックしてエクスプローラウィンドウを開きます。

ステップ 5
保存先を任意に選んで、インターフェース情報が含まれる.jsonファイルを保存します。

これでモデルインターフェースがエクスポートされました。この.jsonファイルは、モデルインターフェースを別のEB GUIDEモデルにインポートするために使用できます。

10.9.3. モデルインターフェースのインポート

モデルインターフェースのインポート

前提条件:

- 2つのEB GUIDEプロジェクトが作成されていること。
- モデルインターフェースが最初のプロジェクトからエクスポートされ、インターフェース情報が含まれる.jsonファイルが作成されていること。
- 2番目のプロジェクトがEB GUIDE Studioに開かれていること。

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] > [モデルインターフェース]の順にクリックします。

[モデルインターフェース]メニューが開きます。

ステップ 3
モデルインターフェースをインポートするには、  をクリックします。

エクスプローラウィンドウが開きます。

ステップ 4
.jsonファイルがある場所に移動し、このファイルを選択し、[開く]をクリックします。

インターフェースがプロジェクトにインポートされます。[モデルインターフェース]メニューにインターフェースが表示され、各インターフェースからいくつのデータプールアイテムやイベントが提供されるのかがわかります。

注記

イベントIDの重複

イベントグループ内では、イベントIDが一意でなければなりません。複数のモデルインターフェースを同時にインポートすると、 別のモデルインターフェースに同一のイベントIDが存在する場合に 同じイベントグループ内でIDが重複することになり、検証エラーが発生します。 イベントIDの変更は インポート後にEB GUIDE Studioで行えないため、インポートを元に戻してから、イベントIDの変更を 元のモデルで行い、再びエクスポートとインポートを実行します。あらかじめ、イベントIDの範囲をすべての EB GUIDEモデルについて定義することをお勧めします。

10.9.4. インポートされたモデルインターフェースの更新

インポートされたモデルインターフェースを更新するには、ソースのEB GUIDEモデルに変更を加えた後で、更新したモデルインターフェースをエクスポートし、それを再インポートします。

モデルインターフェースのエクスポートとインポートの詳細については、10.9.2「モデルインターフェースのエクスポート」と10.9.3「モデルインターフェースのインポート」をご覧ください。

10.9.5. モデルインターフェースの削除

注記

デフォルトモデルインターフェース

デフォルトモデルインターフェースを削除することはできません。

モデルインターフェースの削除

前提条件:

- EB GUIDEモデルに、ユーザー定義のモデルインターフェースまたはインポートされたモデルインターフェースがあること。

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] > [モデルインターフェース]の順にクリックします。

現在のプロジェクトのすべてのモデルインターフェースが一覧表示されます。

ステップ 3
削除するモデルインターフェースを右クリックし、コンテキストメニューの[削除]をクリックします。

このモデルインターフェースがインポートされたものである場合、含まれるすべてのモデル要素を含めてモデルインターフェースがEB GUIDEモデルから削除されます。

このモデルインターフェースが作成されたものである場合、モデルインターフェースだけがEB GUIDEモデルから削除されます。このモデルインターフェースに追加されたすべてのモデル要素は、その後も存在します。

11. EB GUIDE Monitorを操作する

EB GUIDE Monitorの詳細については、6.9「EB GUIDE Monitor」および6.6.2「EB GUIDE Monitorのグラフィカルユーザーインターフェイス」をご覧ください。

注記

デフォルトのウィンドウレイアウト

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 Monitor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. 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。ただし、EB GUIDE Monitorを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。

EB GUIDE Monitorの起動

前提条件:

- EB GUIDEのインストールが完了していること。
- 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
次のファイルをダブルクリックします: `Monitor.exe`

EB GUIDE Monitorが起動します。

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のインストールが完了していること。
- 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`

EB GUIDE Monitorが起動します。

11.2. EB GUIDE Monitorの設定

EB GUIDE Monitorの接続

前提条件:

- EB GUIDE Monitorが起動されていること。
- EB GUIDEモデルが実行中であること。

ステップ 1
EB GUIDE GTFへの接続の設定を変更するには、  をクリックします。

ステップ 2
[ホスト]にホスト名を入力し、[ポート]にポートアドレスを入力します。

ステップ 3
 をクリックします。

EB GUIDE Monitorが接続され、ステータスボタンが緑色になります:  。

ティップ

EB GUIDE Monitorを切断するには、  をクリックします。

EB GUIDE Monitorが切断され、ステータスボタンが赤色になります:  。

EB GUIDE Monitorの表示言語の変更

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]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。

注記

EB GUIDE MonitorによるEB GUIDE Studioからの言語の継承
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。EB GUIDE Monitorの表示言語はEB GUIDE Studioと同じものになります。

EB GUIDE Monitorウィンドウのサイズのリセット

EB GUIDE Monitorウィンドウのサイズと画面上での位置は、`C:\<user>\AppData\Local\Temp\eb_guide_simulation_export\<project>`に個々のEB GUIDEプロジェクトごとに保存されています。

前提条件:

- EB GUIDE Monitorが起動された後で、初期の位置やウィンドウのサイズが変更されることがあります。

ステップ 1
デフォルト値のサイズと位置をリセットするには、`C:\<user>\AppData\Local\Temp\eb_guide_simulation_export\<project>`にある`monitor_layout.xml`および`monitor_settings.xml`を削除します。

ステップ 2
シミュレーションを再起動するか、EB GUIDE Monitorをスタンドアロンアプリケーションとして起動した場合は、EB GUIDE Monitorを再起動します。

新しい`monitor_layout.xml`および`monitor_settings.xml`ファイルが、デフォルトのサイズと位置の値を使用して作成されます。

ログメッセージの数の編集

`$INSTALL_PATH\tools\monitor\Monitor.exe.config`設定ファイルでは、ロガーがメモリ上限に達した場合のEB GUIDE Monitorの動作を定義できます。

ステップ 1
`$INSTALL_PATH\tools\monitor\Monitor.exe.config`設定ファイルを開きます。

ステップ 2
[ロガー]コンポーネントに表示されるエントリーの数を定義するには、`limit`値を変更します。

ステップ 3
上限に達した場合に削除されるエントリーの数を定義するには、`removeCount`値を変更します。

ステップ 4
EB GUIDE Monitorを起動します。

EB GUIDE Monitorは、変更した設定ファイルから新しい設定を使用します。

11.3. EB GUIDE Monitorへの設定の読み込み

EB GUIDE Monitorへの設定ファイルの読み込み

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。
- EB GUIDEモデルが`$EXPORT_PATH`にエクスポートされていること。
- `$EXPORT_PATH`では、`monitor.cfg`設定ファイルが作成されます。

ステップ 1
[ファイル] > [読み込み設定]を選択します。

ダイアログが開きます。

ステップ 2
`$EXPORT_PATH`に移動して、`monitor.cfg`設定ファイルを選択します。

ステップ 3
[開く]をクリックします。

プロジェクトの設定がEB GUIDE Monitorに読み込まれます。

EB GUIDE Monitorへの最近使った設定ファイルの読み込み

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。
- 1つ以上の設定ファイルが最近使用されていること。

ステップ 1
[ファイル] > [最近使った設定ファイル]を選択します。

ダイアログが開きます。

ステップ 2
それぞれの場所に移動して、設定ファイルを選択します。

設定ファイルがEB GUIDE Monitorに読み込まれます。

注記

EB GUIDE MonitorのEB GUIDE GTFからの切断

新しい設定ファイルを読み込む前に、EB GUIDE Monitorは現在のEB GUIDE GTFから自動的に切断されます。

EB GUIDE Monitorは再接続して、新しい設定ファイルを読み込みます。

11.4. EB GUIDE Monitorでのイベント発行

EB GUIDE Monitorでのイベント発行

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、＋ をクリックします。

ダイアログが開きます。

ステップ 2
発行するイベントを選択し、[承認]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。[ロガー]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
イベントにパラメータがある場合は、次の操作を行います。

ステップ 4.1
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。[ロガー]コンポーネントにログメッセージが表示されます。

11.5. EB GUIDE Monitorによるデータプールアイテムの値の変更

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、+ をクリックします。

ダイアログが開きます。

ステップ 2
データプールアイテムを選択し、[承認]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。

11.6. EB GUIDE Monitorでのスクリプトの使用

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

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11.1
EB GUIDE Monitorのサンプルスクリプトファイル

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 occured: " + 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) //❹
        {
            await monitor.WriteDatapool("Boolean 1", true);
        }

        public async Task ReadDatapoolValue(IMonitorContext monitor) //❺
        {
            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"); //❻
        await monitor.Write(string.Format("State {0} left",
                    leftState.Name));

        var enteredState = await monitor.WaitForStateEnter
                    ("Main", "State 2"); //❼
        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"); //❽
        await monitor.Write(string.Format("State Machine {0} started",
                   startedStateMachine.Name));

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

public class Advanced
{
    public async Task CaptureScreenshot(IMonitorContext monitor) //❿
    {
        // 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) //11
        {
            // 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 occured");
        }
    }

    namespace MonitorScripting.EventScripts //12
    {
        using Elektrobit.Guide.Monitor.Scripting.MonitorContext;
        using System.Threading;
        using System.Threading.Tasks;
        using Elektrobit.Guide.Monitor.Model.Event;
        using Elektrobit.Guide.Monitor.Model.Value;

        public class MonitorScripts
        {
            public async Task FireEventInNamespace(IMonitorContext monitor)
            {
                string[] namespacePath = { "Foo_namespace", "go_to_view2" };
                var identifier = new QualifiedIdentifier(namespacePath);

                await monitor.FireEvent(identifier);
            }

            public async Task FireEventInNestedNamespace(IMonitorContext monitor)
            {
                string[] namespacePath =
                    { "Foo_namespace", "sub_namespace_under_foo", "go_to_view4" };
                var identifier = new QualifiedIdentifier(namespacePath);
```

```
                await monitor.FireEvent(identifier);
            }

            public async Task FireEventInRootNamespace(IMonitorContext monitor)
            {
                await monitor.FireEvent("go_to_view3");
            }
        }
    }
}
```

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
⓫ イベントがタイムアウトするまで待機するメソッド
⓬ 名前空間の操作方法の例

11.6.2. EB GUIDE Monitorでのスクリプトの開始

EB GUIDE Monitorでのスクリプトの開始

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、11.6.1「EB GUIDE Monitorでのスクリプトファイルの記述」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。

11.7. ウォッチリストのエクスポートおよびインポート

プロジェクトに使用するイベントおよびデータプールアイテムは、ウォッチリストに格納されます。他のプロジェクトにアイテムを使用するには、ウォッチリストを`.xml`ファイルとしてエクスポートし、後でそれらを新しいプロジェクトにインポートします。

すべてのウォッチリストのエクスポート

前提条件:

- EB GUIDE Monitorが起動されていること。
- EB GUIDEモデルが[データプール]タブまたは[イベント]タブに格納されたアイテムですでにセットアップされていること。

ステップ 1
すべてのウォッチリストをエクスポートするには、[ファイル] > [すべてのウォッチリストをエクスポート]を選択します。

ダイアログが開きます。

ステップ 2
出力先ディレクトリを選択して、ファイル名を入力します。

すべてのデータプールアイテムとイベントがエクスポートされます。

単一のウォッチリストのエクスポート

前提条件:

- EB GUIDE Monitorが起動されていること。

- EB GUIDEモデルが[データプール]コンポーネントまたは[イベント]コンポーネントに格納されたアイテムですでにセットアップされていること。

ステップ 1
エクスポートするアイテムが含まれている[データプール]コンポーネントまたは[イベント]コンポーネントに移動します。

ステップ 2
このタブのアイテムリストを保存するには、[エクスポート]をクリックします。

ダイアログが開きます。

出力先ディレクトリを選択して、ファイル名を入力します。

コンポーネントのデータプールアイテムまたはイベントがエクスポートされます。

ウォッチリストのインポート

前提条件:

- EB GUIDE Monitorが起動されていること。
- エクスポートされたウォッチリストがすでに使用可能であること。

ステップ 1
ウォッチリストをインポートするには、[ファイル] > [ウォッチリストをインポート]を選択します。

ダイアログが開きます。

ステップ 2
インポートするウォッチリストファイルを選択します。

データプールアイテムまたはイベントが新しい[データプール]コンポーネントまたは[イベント]コンポーネントで開きます。

注記

レイアウトはインポートされない
インポートされるのはデータプールアイテムとイベントだけです。レイアウトはインポートされません。

デフォルトレイアウトは、新しく開く[データプール]コンポーネントと[イベント]コンポーネントに使用されます。

12. EB GUIDE Studioの拡張

この章では、EB GUIDE Studioの拡張機能を作成するために理解する必要がある概念を紹介し、作成の手順と作成例を示します。拡張機能の実装時に問題が起きた場合は、サポートまでお問合せください。第3章「サポート」をご覧ください。

12.1. 概念

12.1.1. 依存性の注入

EB GUIDE Studioは依存性の注入を念頭に置いて構築されています。EB GUIDE Studioでは、依存性を管理するために、.NET Frameworkの一部であるManaged Extensibility Framework(MEF)を使用します。

依存性は、属性付きプログラミングモデルに基づいて登録、注入されます。拡張機能は、インターフェースの実装をエクスポートして新しい機能を提供し、インターフェースをインポートして既存のEB GUIDE Studio機能を使用することができます。

依存性をクラスにインポートするには、`ImportingConstructor`属性をコンストラクタに追加し、必要な依存性をコンストラクタの引数として追加します。クラスのインスタンスが必要になると、MEFはすべての依存性を満足させ、マークされたコンストラクタを呼び出そうとします。

MEFは、エクスポートされたクラスの依存性のみを満足させることに注意してください。クラスをエクスポートするには、`Export`属性をクラスに追加します。

例12.1
`Export`属性の例

次の例は、`ImportingConstructor`属性と`Export`属性の一般的な使い方を示しています。`IFooService`インターフェースの実装がエクスポートされ、次にこれが`IBarService`インターフェースの依存性をインポートします。

```
[Export(typeof(IFooService))]
internal class MyFooService : IFooService
{
 [ImportingConstructor]
 public MyFooService(IBarService barService) {}
}
```

Managed Extensibility Frameworkの詳しい概要については、https://docs.microsoft.com/en-us/dotnet/framework/mef/をご覧ください。属性付きプログラミングモデルの詳細については、https://docs.microsoft.com/en-us/dotnet/framework/mef/attributed-programming-model-overview-mefをご覧ください。

12.1.2. EB GUIDEモデルの拡張機能

モデルの一貫性を保つため、すべての変更は一度に1つずつ実行する必要があります。これを確実に行うには、すべての変更をタスクスケジューラで設定し、操作を1つずつ順番に実行します。また、モデルへのすべての変更はセッションで実行する必要もあります。セッションには2つの目的があります。

- 変更を1つの変更セットにグループ化する。こうすると、1セッションで実行されるすべてのものを1ステップで処理できます。
- 基盤となるストレージに、どの要素が変更されたのかを通知する。つまり、セッションで実行された変更だけが、実際にファイルシステムに保存されます。

警告

データの喪失

EB GUIDEモデルの変更にセッションを使わないと、モデルに損傷を与え、データの喪失を招く恐れがあります。

EB GUIDE Studioは、タスクスケジューリングとセッションを簡単に扱えるようにAPIを提供しています。

- `ITaskSchedulerProvider` を通じて、すべてのモデルへの変更を実行するタスクスケジューラにアクセスできます。
- `IEventService`インターフェースは、モデル内でイベントを作成したり、変更したりするメソッドを提供します。
- `ExecuteModelAction` は、`ITaskSchedulerProvider`インターフェースの拡張メソッドです。これは、変更を適切なタスクスケジューラに設定し、セッションを作成します。この拡張メソッドの2番目の引数は、モデルへの変更を実際に実行するデリゲートです。セッションはデリゲートが呼び出される前に作成され、デリゲートが実行された後で自動的にコミットされます。つまり、`ExecuteModelAction`を呼び出してモデルをどのように変更するかを指定すれば、そこから先のタスクスケジューリングとセッションの処理は自動で行われます。

例12.2
拡張機能をEB GUIDEモデルに適用

次の例は、前の例を変更して拡張機能をEB GUIDEモデルに適用する方法を示しています。前の例は、`ITaskSchedulerProvider`と`IEventService`の依存性をインポートするように変更しました。

```
[Export(typeof(IFooService))]
internal class MyFooService : IFooService
{
 private readonly ITaskSchedulerProvider _schedulerProvider;
```

```
  private readonly IEventService _eventService

  [ImportingConstructor]
  public MyFooService(
   ITaskSchedulerProvider schedulerProvider,
   IEventService eventService)
  {
   _schedulerProvider = schedulerProvider;
   _eventService = eventService;
  }

  public async Task ModifyModel(IProjectContext projectContext)
  {
   await _schedulerProvider.ExecuteModelAction(
    projectContext,
    session => _eventService.CreateEvent(
        session,
        projectContext,
        projectContext.Project.RootNamespace,
        "My Event"));
  }
 }
```

すでに説明したとおり、この例では、IEventServiceインターフェースも依存性としてインポートします。モデル要素を直接変更するのではなく、既存のサービスインターフェースを利用してモデルを変更することを強く推奨します。

実行する操作に適した既存のサービスがない場合は、モデルの一貫性を保つため、一定のルールに従う必要があります。セッションはIWriteSessionインターフェースによって表されます。変更をモデルに保存するために、セッションを正しく呼び出す必要があります。

- 可能であれば、モデル要素ツリーを事前に作成し、それをモデルに追加します。こうすると、不要なモデル更新通知が回避され、パフォーマンスが向上します。
- 新しく作成した要素は保存する必要があります。要素ツリー全体を作成する場合は、SaveHierarchyメソッドを使います。すべての子要素が自動的に保存されます。基本的に、新しく作成した要素にはSaveHierarchyを常に呼び出します。
- 変更した要素はSaveメソッドで保存します。既存の要素にSaveHierarchyメソッドを使うことは避けてください。不要なエントリーが変更セット内に大量に作成され、パフォーマンスを低下させます。
- 除外した要素はDeleteメソッドで削除します。このメソッドを直接呼び出すのではなく、IModelElementServiceインターフェースをインポートし、そこにあるDeleteElementsメソッドを使用します。このメソッドは、要素とそのすべての子を再帰的に削除します。
- Commitメソッドは明示的に呼び出さないでください。Commitの呼び出しは、上記のExecuteModelAction拡張メソッドから処理されます。

12.1.3. EB GUIDE Studio UI拡張機能

EB GUIDE Studioは、UIフレームワークとしてWindows Presentation Foundation (WPF)を使用します。WPFの詳細については、https://docs.microsoft.com/en-us/dotnet/framework/wpf/をご覧ください。また、EB GUIDE StudioのUIレイヤーは、Model-View-ViewModel (MVVM)パターンを意識して作成されています。これは、ほとんどの場合に、EB GUIDE Studio UIを拡張するためにビューモデルとビュー実装を提供する必要があることを意味します。メニューのような汎用のUI要素には既存のビューがありますが、カスタムのUI要素についてはビュー実装が必要です。

カスタムビュー実装は、MEFを使ってエクスポートすることにより提供します。WPFで使う予定のビューモデルは、正しく機能するために一定の規則を満たす必要があります。ビューモデルは、`INotifyPropertyChanged`インターフェースを実装する必要があります。このインターフェースは、ビューモデルへの変更をビューに反映するために、WPFのデータバインディングエンジンで使われます。このインターフェースをカスタムビューモデルに実装しない場合、バインディングからは初期値だけがビューレイヤーに転送され、変更は伝達されません。カスタムビューモデルの作成を簡単にするため、EB GUIDE Studioは`INotifyPropertyChanged`インターフェースを実装する`ViewModel`という基本クラスを提供します。

例12.3
カスタムビューモデルの実装

次の例は、`Text`プロパティへの変更をビューに伝達するビューモデルを示しています。また、ビューにバインドできるコマンドプロパティもあります。バインドされたボタンがクリックされるなどの操作によりこのコマンドが実行されるたびに、`Text`プロパティが変更され、ビューが更新されます。

```
internal class MyViewModel : ViewModel
{
 private string _text;

 public string Text
 {
  get => _text;
  set => SetProperty(ref _text, value);
 }

 public ICommand DoSomethingCommand { get; }

 public MyViewModel()
 {
  Text = "Initial text";
  DoSomethingCommand = new DelegateCommand(DoSomething);
 }

 private void DoSomething()
 {
  Text = "Did something";
```

```
  }
}
```

ビューモデルにWPFのDataTemplateを作成すると、ビューが提供されます。DataTemplate は、リソースディクショナリ内にXAMLで定義されます。IResourceProviderインターフェースの実装をエクスポートすることにより、カスタムのリソースディクショナリ.を提供できます。

例12.4
DataTemplate によるカスタムビューモデル

次のコードスニペットは、カスタムビューモデルにDataTemplateを提供する方法を示しています。リソースディクショナリは、アセンブリMyAssembly内にResources.xamlというファイルで定義します。リソースプロバイダーの実装が、URIをXAMLファイルに返します。

```
[Export(typeof(IResourceProvider))]
internal class MyResourceProvider : IResourceProvider
{
 public IEnumerable<Uri> GetResourceUris()
 {
  var uri = new Uri(
   @"MyAssembly;Component/Resources.xaml",
   UriKind.Relative);

  return new[]{ uri };
 }
}
```

例12.5
ResourceDictionary

次のコードスニペットは、XAMLでDataTemplateを使ってResourceDictionaryを作成する方法を示しています。

```
<ResourceDictionary
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:mynamespace="clr-namespace:MyNamespace">
    <DataTemplate
        DataType="{x:Type mynamespace:MyViewModel}">
        <mynamespace:MyView />
    </DataTemplate>
</ResourceDictionary>
```

`IResourceProvider`から返されるURIは、WPFが検出して読み込めるパックURI構文で記述されている必要があります。パックURI構文の詳細については、https://docs.microsoft.com/en-us/dotnet/framework/wpf/app-development/pack-uris-in-wpfをご覧ください。

`IResourceProvider`実装によって提供されるリソースディクショナリ は、アプリケーションレベルで読み込まれます。つまり、ディクショナリに含まれるすべてのリソースは、グローバルに利用可能です。明示的なリソースキーを持つ既存のリソースと名前が衝突するのを回避するため、リソースキーに拡張機能の名前をプレフィックスとして含めます。例えば、特別なボタンにカスタムのボタンスタイルを与える場合、`MyPlugin.MySpecialButton`などと命名します。

EB GUIDE拡張子の使用例を以下のEB GUIDEマイクロサイトからダウンロード: https://www.elektrobit.com/ebguide/examples/。手順については、付属の`EB GUIDE Studio Tutorial Using EB GUIDE Studio examples.pdf`ファイルをご覧ください。

12.2. 新しい拡張プロジェクトを作成

Visual Studioプロジェクトの作成

前提条件:

- EB GUIDE Studioインストールディレクトリが書き込み可能であること。`C:\Program Files\Elektrobit`の下にあるデフォルトのインストールディレクトリは、Windowsによってプロテクトされています。拡張機能を開発するには、書き込みがプロテクトされていないインストールディレクトリを使います。
- Visual Studio 2017以降がインストールされていること。
- .NET Framework 4.7 SDKがインストールされていること。

ステップ 1
Visual Studioを開きます。

ステップ 2
新しいプロジェクトを作成します。

ステップ 3
[新しいプロジェクト]ダイアログで、次のプロジェクトテンプレートを選択します。

コア拡張機能の場合は、`Class Library (.NET Framework)`を選択します。

UI拡張機能の場合は、`WPF User Control Library (.NET Framework)`を選択します。

ステップ 4
拡張プロジェクトの名前を入力し、[OK]をクリックします。

プロジェクトが作成されます。

ステップ 5
[ソリューションエクスプローラ]でプロジェクトを右クリックし、[NuGetパッケージの管理...]を選択します。

ステップ 6
[参照]をクリックし、「Elektrobit.Guide.Studio.」を検索します。

ステップ 7
コア拡張機能の場合、パッケージElektrobit.Guide.Studio.Coreをインストールします。

UI拡張機能の場合、パッケージElektrobit.Guide.Studio.Uiをインストールします。

プロジェクトの設定はこれで完了し、コーディングを始めることができます。

12.3. アセンブリのコピーを無効化

アセンブリのコピーを無効化

プロジェクトを作成すると、デフォルトで、参照先のNuGetパッケージからすべてのアセンブリが出力ディレクトリにコピーされます。EB GUIDE NuGetパッケージから提供されるアセンブリは、拡張機能の実行に使うEB GUIDEインストールにすでに含まれているため、コピーする必要はありません。次の手順でNuGetパッケージの参照を変更し、アセンブリのコピーを無効化することができます。

前提条件:

- プラグインプロジェクトがVisual Studioで作成されたものであること。

ステップ 1
拡張プロジェクトのプロジェクトファイル(.csproj)をテキストエディターに開きます。

ステップ 2
EB GUIDE NuGetパッケージのPackageReferenceエントリーを探します。

ステップ 3
IncludeAssetsプロパティを追加し、値をcompileに設定します。

ステップ 4
プロジェクトファイルを保存し、Visual Studioにプロジェクトを再読み込みします。

これでPackageReferenceエントリーの内容は、次のサンプルスニペットのようになります。

```
<PackageReference Include="Elektrobit.Guide.Studio.Ui">
    <Version>6.9.0</Version>
    <IncludeAssets>compile</IncludeAssets>
</PackageReference>
```

12.4. 拡張機能を実行

拡張機能を実行

前提条件:

- EB GUIDE Studioインストールディレクトリが書き込み可能であること。`C:\Program Files\Elektrobit`の下にあるデフォルトのインストールディレクトリは、Windowsによってプロテクトされています。拡張機能を開発するには、書き込みがプロテクトされていないインストールディレクトリを使います。
- 拡張プロジェクトは、ソリューションの起動プロジェクトとして設定されます。

ステップ 1
拡張プロジェクトのプロジェクト設定を開き、[デバッグ]タブに移動します。

ステップ 2
`Start external program`を選択し、EB GUIDE Studioインストールディレクトリにある `Studio.exe`を選択します。

ステップ 3
[ビルドイベント]タブに移動し、次のポストビルドスクリプトを入力します。

```
copy /Y $(TargetPath) <extension dir>
copy /Y $(TargetDir)$(TargetName).pdb <extension dir>
```

このスクリプトは、拡張機能をビルドしてから、それをEB GUIDE Studioプラグインディレクトリにコピーします。

ステップ 4
`<extension dir>`は、実際のEB GUIDE Studioインストールのパスで置き換えてください。

コア拡張機能の場合は、`$GUIDE_INSTALL_PATH\studio\lib\core`を使用します。

UI拡張機能の場合は、`$GUIDE_INSTALL_PATH\studio\lib\ui`を使用します。

これで、拡張機能はVisual Studioから実行できるようになりました。

13. ベストプラクティス

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 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。

13.1. ベストプラクティス: スクリプト値の処理

データプールアイテムおよびウィジェットのプロパティがスクリプト値に変換されている場合、そのプロパティの読み取りのたびにEB GUIDEスクリプトが実行されます。一部のユースケースでは、EB GUIDEスクリプトの実行回数を最小限に抑えてパフォーマンスを向上するために、次の操作を行います。

1. スクリプト値がある倍は、そのプロパティタイプをリセットします。スクリプト値に変換されておらず、プレーン値を持つプロパティを使用します。詳細については、9.8「プロパティのスクリプト値への変換」および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。
2. 現在値を計算および設定するには、`Conditional script`タイプのユーザー定義プロパティを追加します。このアクションは、例えば、初期化の際や入力プロパティの変更時など、必要な場合にのみ実行する必要があります。

14. 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
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。

14.1. 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
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]に移動して、＋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
初期ステートから`Volume`ビューステートへの遷移を追加します。

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ーをモデリングする手順を説明します。スライダーインジケーターはランタイム中に音量を表示します。

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

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
[ナビゲーション]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。

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。

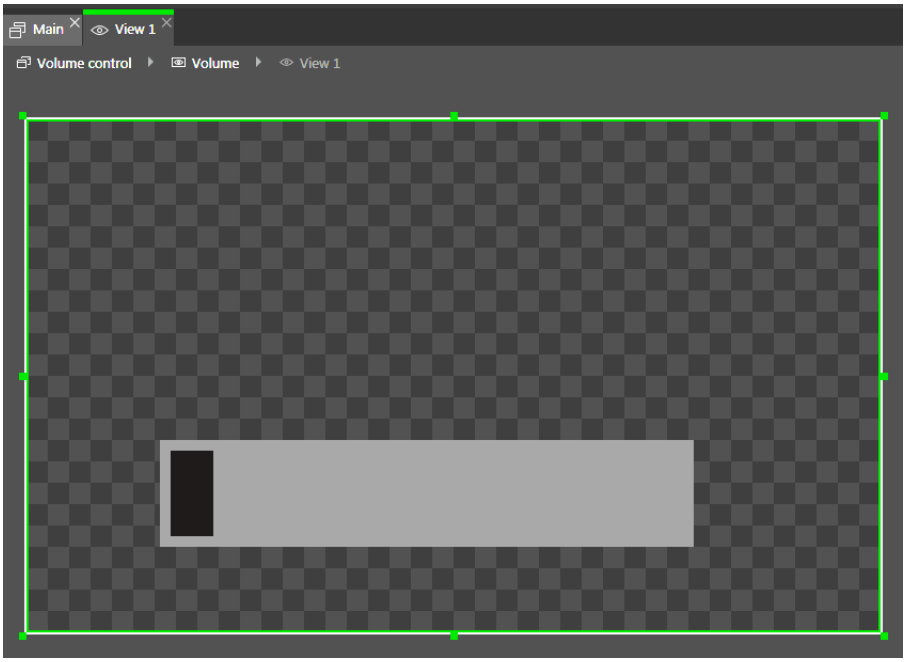

図14.1 2つの四角形を持つView 1の外観

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、+ 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イベントを検索してダブルクリックします。

ステップ 4.3
[アクション]プロパティの横にある+ 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`イベントを検索してダブルクリックします。

ステップ 5.3
[アクション]プロパティの横にある+ 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`イベントを検索してダブルクリックします。

ステップ 6.3
[アクション]プロパティの横にある+ 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。

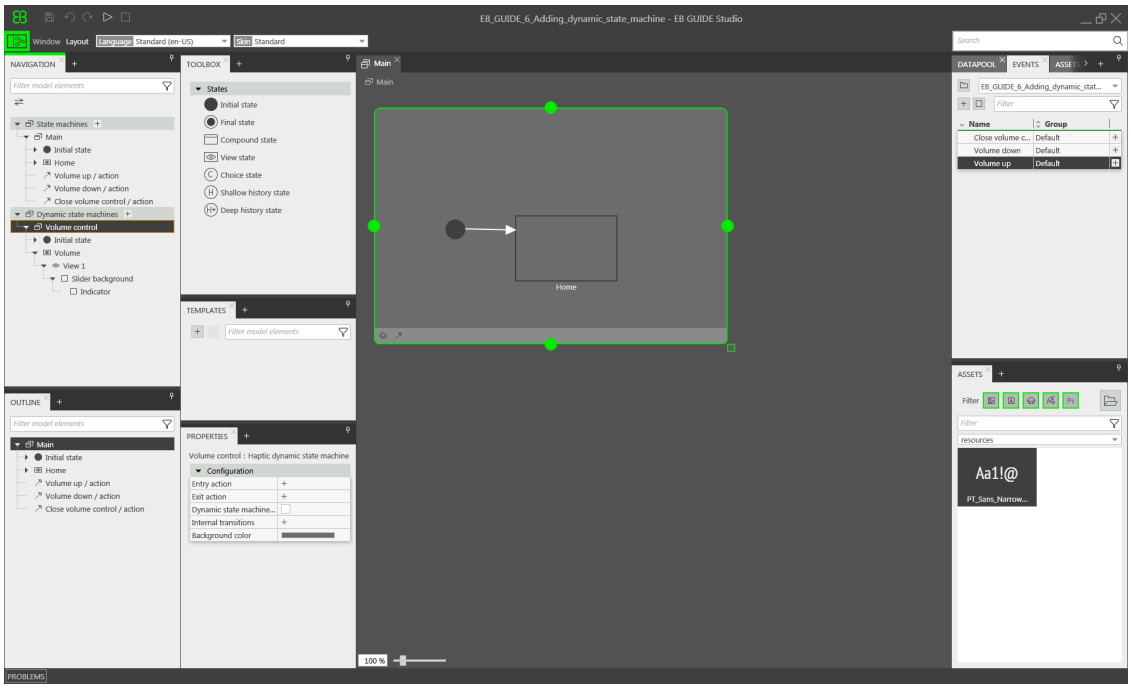

図14.2 すべてのモデル要素が揃ったEB GUIDEモデル

EB GUIDEモデルのシミュレーションとテストを開始する

前提条件:

- 前のセクションの手順を完了していること。

シミュレーションを開始するには、コマンドエリアで▷をクリックします。

シミュレーションとEB GUIDE Monitorが開始されます。EB GUIDEモデルに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、＋ をクリックし、Volume upイベントを選択して、このイベントを発行するための をクリックします。

動的ステートマシンが起動し、スライダーインジケーターが表示されます。動的ステートマシンは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イベントを発行すると、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。

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

注記

デフォルトのウィンドウレイアウト

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。

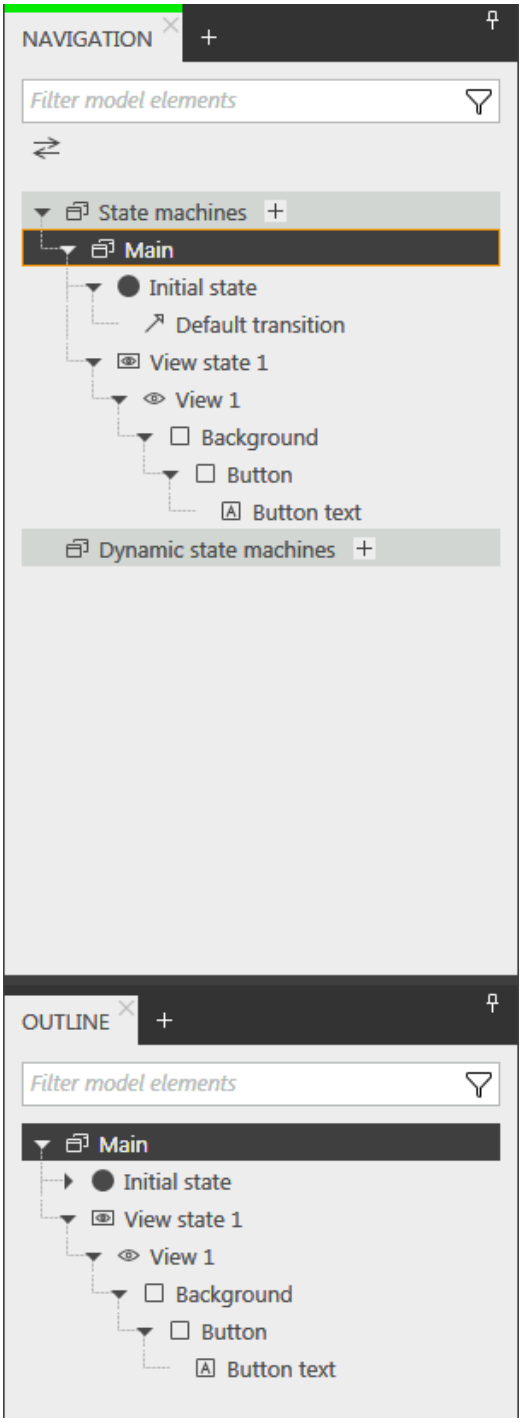

図14.3 ウィジェットの階層

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[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

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center(1)を選択します。

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。

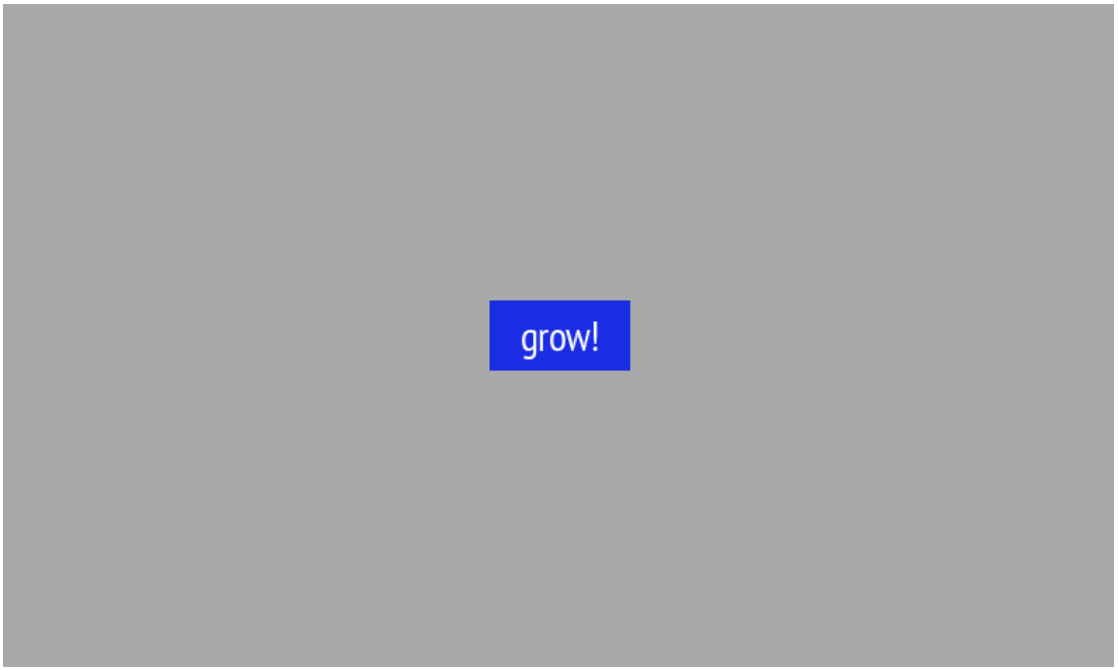

図14.4 結果

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。

14.3. 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 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[値]列を選択し、{} をクリックします。

EB GUIDEスクリプトエディターが開きます。

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[値]列を選択し、{} をクリックします。

EB GUIDEスクリプトエディターが開きます。

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[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

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。

14.4. 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

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。

インスタンシエータを使うと、ランタイム中にリストを動的に作成することができます。リストタイプのデータプールアイテムに基づき、インスタンシエータが事前定義済みのレイアウトでリストのすべての要素を表示します。データプールアイテムのコンテンツが変更されると、インスタンシエータの表示も変更されます。

このセクションでは、動的コンテンツを使用してリストを作成する手順について説明します。リストの各要素は、ラベルの付いた四角形です。

所要時間: 15分

データプールアイテムの追加

このセクションでは、String listタイプのデータプールアイテムを追加する手順について説明します。データプールアイテムは、インスタンシエータのすべてのリスト要素に値を提供します。データプールアイテムのコンテンツが変更されると、インスタンシエータの表示も変更されます。

前提条件:

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

ステップ 1
リストのコンテンツを表示するために、String listタイプのデータプールアイテムを追加します。

[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
[値]列を選択し、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]からインスタンシエータをビューにドラッグしてドロップします。インスタンシエータの名前を`MyInstantiator`に変更します。

ステップ 4
[ツールボックス]から四角形をドラッグし、インスタンシエータにドロップします。四角形の名前を`MyRectangle`に変更します。

ステップ 5
[ツールボックス]からラベルをドラッグし、四角形にドロップします。ラベルの名前を`MyLabel`に変更します。

ウィジェットの階層が、次のように変わります。

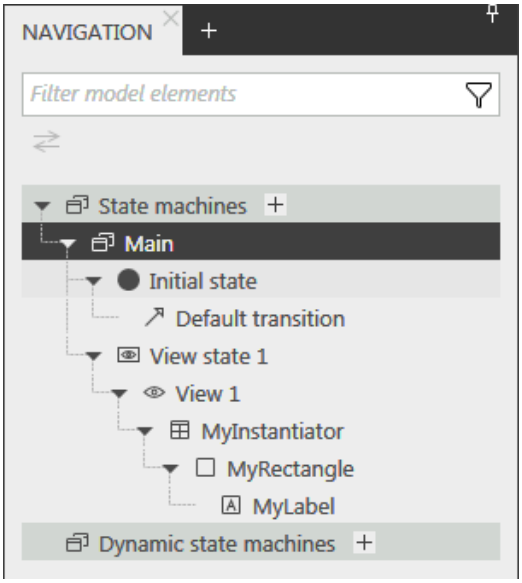

図14.5 インスタンシエータを含むウィジェット階層

インスタンシエータの設定

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
MyInstantiatorのプロパティを変更するには、インスタンシエータを選択し、[プロパティ]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
プロパティの横にある[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 5.4
[トリガー]の下に、dp:MyStringListと入力します。

ステップ 5.5
[トリガー時]スクリプトに次のように入力します。

```
function(v:arg0::bool)
{
 v:this.numItems = length dp:MyStringList;
 false
}
```

MyStringListのコンテンツに応じてリストエントリー数を自動的に変更するスクリプトが追加されました。

ステップ 6
インスタンシエータ内のラベルをすべて整列させるために、レイアウトを追加します。

[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 6.1
[使用可能なウィジェット機能]の下から[レイアウト]カテゴリを展開し、[ボックスレイアウト]ウィジェット機能を選択してラベルを横に整列させます。

ステップ 6.2
[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インスタンシエータに追加され、[プロパティ]コンポーネントに表示されます。

ステップ 6.3
`gap`テキストボックスに`5`と入力して、各リスト要素の間隔を5ピクセルに設定します。

ステップ 6.4
`layoutDirection`ドロップダウンリストボックスから`vertical (1)`を選択して、ラベルを整列させます。

リストの外観を定義し、リストアイテム数を動的に適用するインスタンシエータが設定されました。

リスト要素テキストの設定

前提条件:

- 前のセクションの手順を完了していること。

ステップ 1
ラベルの外観を変更するには、`MyLabel`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
`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。

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center(1)を選択します。

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シエータのwidthプロパティへのリンクを追加します。

ステップ 3
heightテキストボックスに50と入力します。

ステップ 4
リストの各ラインに対して一意の位置を定義するために、[ラインインデックス]ウィジェット機能を追加します。

ステップ 4.1
[ウィジェット機能プロパティ]カテゴリで[追加/削除]をクリックします。

[ウィジェット機能]ダイアログが表示されます。

ステップ 4.2
[使用可能なウィジェット機能]から[リスト管理]カテゴリを展開し、[ラインインデックス]ウィジェット機能を選択します。

lineIndexプロパティが四角形のプロパティに追加されます。

ステップ 5
リストのラベルにMyStringListのコンテンツを入れるため、条件スクリプトを追加します。

ステップ 5.1
[ユーザー定義プロパティ]カテゴリの横にある+をクリックします。

メニューが展開されます。

ステップ 5.2
メニューから[条件スクリプト]をクリックします。

ステップ 5.3
プロパティの名前をsetTextに変更します。

ステップ 5.4
setTextプロパティの横にある[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 5.5
[トリガー]の下に、v:this.lineIndexおよびdp:MyStringListと入力します。

ステップ 5.6
[トリガー時]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。

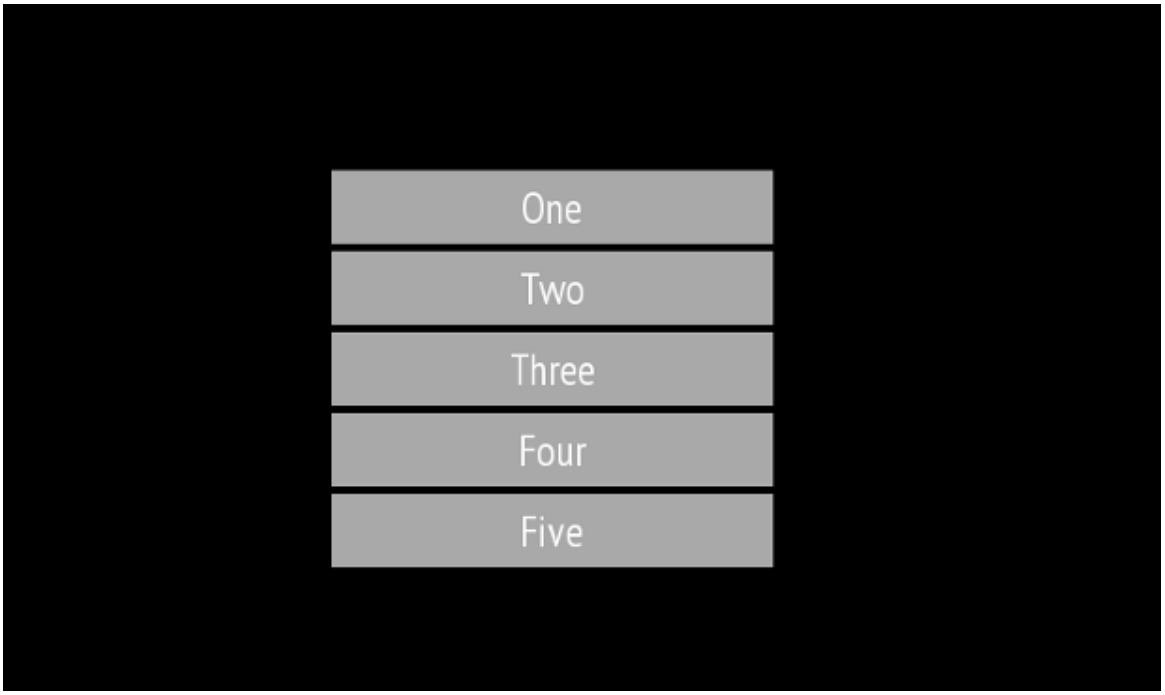

図14.6 インスタンシエータで作成されたリスト

14.5. 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

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。

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[値]列を選択し、{} をクリックします。

EB GUIDEスクリプトエディターが開きます。

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。
- コンテンツエリアにView 1ビューが表示されます。

ステップ 1
[アニメーションエディター]に移動します。[アニメーション化プロパティ]の横にある+をクリックして、View 1を選択します。

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。

14.6. 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

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。
- 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。
- 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[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

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:setLanguage(l:German, true)
     dp:SelectedLanguage = 1
    }
    else if (dp:SelectedLanguage == 1)  // German selected
    {
     f:setLanguage(l:French, true)
     dp:SelectedLanguage = 2
    }
    else if (dp:SelectedLanguage == 2)  // French selected
    {
     f:setLanguage(l:Standard, true)
     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。

14.7. 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 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 2.0以降のレンダラーに限り、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。

14.8. チュートリアル: ガンマの正しいレンダリング

注記

デフォルトのウィンドウレイアウト

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、以下に対するガンマ補正を実行できます。

- ディスプレイ
- イメージ
- テクスチャ

このセクションでは、ガンマ補正の設定方法について説明します。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。

所要時間: 15分

ディスプレイのガンマエンコーディングの設定

このセクションでは、ガンマエンコーディングされた値をディスプレイに出力するようEB GUIDE Studioを設定する手順を説明します。

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。

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] > [プロファイル]の順にクリックします。

ステップ 3
sRGBを認識できる各プロファイルについて、以下のように設定します。

ステップ 3.1
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 3.2
`colorMode`ドロップダウンリストボックスから、以下を選択します。

- レンダリングパイプラインがグラフィックス処理ユニットのハードウェアsRGBサポートを使用する場合は、次のものを選択します。`32-bit sRGB (4)`
- レンダリングハードウェアがsRGBをサポートしていない場合は、`32-bit sRGB (Emulated) (5)`を選択します。

編集モードで変更を適用するには、EB GUIDE Studioを再起動します。

注記

ハードウェアのレンダリング

OpenGL 3レンダラーには必ずハードウェアsRGBサポートがあります。ハードウェアが適切なOpenGL ES拡張機能によってハードウェアsRGBをサポートしている場合、OpenGL ES 2.0 APIを使用するOpenGLレンダラーはハードウェアsRGBサポートのみを使用します。これは自動的に検出されます。OpenGL ES 2.0ハードウェアがsRGBをサポートしていない場合、レンダラーは変換にフラグメントシェーダーを使用する`32-bit sRGB (Emulated)(5)`に自動的に戻ります。システムがOpenGL ES 3.0をサポートしていても、ハードウェアsRGBモードはすべてのシステムで機能するわけではないことに注意してください。その場合は`32-bit sRGB (Emulated)(5)`に切り替えます。

イメージのガンマエンコーディングの設定

前提条件:

- 前のセクションの手順を完了していること。
- イメージファイルはリソースディレクトリで利用可能です。

ステップ 1
プロジェクトエディターに移動して、ビューをダブルクリックします。[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`ドロップダウンリストボックスからイメージファイルを選択します。

たいていの場合、イメージは非常に明るく表示されます。

これはシーンプロパティで設定されたsRGBカラーモードが原因です。すでにガンマ補正されているイメージにガンマ補正が適用されています。

ステップ 3
ガンマエンコーディングされるようにイメージを設定するには、[プロパティ]コンポーネントで`sRGB`を選択します。

これで、イメージがブレンド操作で正しく表示および処理されます。

テクスチャのガンマエンコーディングの設定

前提条件:

- 前のセクションの手順を完了していること。
- 少なくとも1つのテクスチャ3Dオブジェクトを含む3Dファイル(`.fbx`ファイルなど)がディフューズテクスチャで使用できること。

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。

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、3Dファイルを選択します。

ステップ 3
[ナビゲーション]コンポーネントに、インポートされたシーングラフを配置します。シーングラフ構造内で、ディフューズテクスチャを使用する材質ウィジェットを選択します。

ステップ 4
[プロパティ]コンポーネントの[ウィジェット機能プロパティ]で、[追加/削除]をクリックします。

ステップ 5
[3D]カテゴリから、[ディフューズテクスチャ]を選択します。

ステップ 6
`diffuseSRGB`プロパティを選択します。

このテクスチャはガンマエンコーディングされたイメージとして扱われ、ライトの計算で使用される前にリニアライズされます。

14.9. チュートリアル: 表示遷移アニメーションの使用

注記	デフォルトのウィンドウレイアウト 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。

表示遷移アニメーション(VTA)は、あるビューから別のビューへの移動時に実行されるアニメーションです。以下では、こうしたアニメーションを作成する手順を説明します。ビューの変更時に再生されるビューやアニメーションによってモデルを作成することになります。以下の要素を作成します。

- 2つのビューステート
- ナビゲーション要素(ボタン、ラベルなど)
- ビューステートの変化をトリガーするイベント
- 別のステートへの遷移時に再生されるアニメーション

所要時間: 30分

最初のビューステートの作成

最初のビューとボタンを作成します。

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
[ナビゲーション]コンポーネントで、ビューステートの名前をFirstStateに、ビューの名前をFirstViewにそれぞれ変更します。

ステップ 2
FirstViewを開きます。

ステップ 3
[ツールボックス]コンポーネントから、四角形をFirstView内にドラッグし、その名前をRectNextViewに変更します。

この四角形は、遷移をトリガーするボタンのためのものです。

ステップ 4
[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 5
[使用可能なウィジェット機能]の下で、[エフェクト]カテゴリを展開し、[枠]を選択します。

ステップ 6
[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[プロパティ]コンポーネントに表示されます。

ステップ 7
[プロパティ]コンポーネントで、次のように入力します。

- widthテキストボックスに、220と入力します。
- heightテキストボックスに、70と入力します。
- xテキストボックスに、290と入力します。
- yテキストボックスに、150と入力します。
- fillColorを黒に設定します。
- borderThicknessテキストボックスに、2と入力します。
- borderColorを白に設定します。

ステップ 8
[ツールボックス]コンポーネントから、ラベルを[ナビゲーション]コンポーネント内にドラッグし、FirstViewの子ウィジェットとして追加します。

ステップ 9
ラベルの名前をLabelNextViewに変更します。

ステップ 10
[プロパティ]コンポーネントで、次のように入力します。

- textテキストボックスに、Go to the next viewと入力します。
- fontテキストボックスに、25と入力します。
- horizontalAlignを中央に設定します。

ステップ 11
ラベルの寸法を四角形の寸法にリンクします。次のプロパティをリンクします。

- LabelNextViewのwidthプロパティをRectNextViewのwidthにリンクします。
- LabelNextViewのheightプロパティをRectNextViewのheightにリンクします。
- LabelNextViewのxプロパティをRectNextViewのxにリンクします。
- LabelNextViewのyプロパティをRectNextViewのyにリンクします。

図14.7 ボタン付きのFirstView

2番目のビューステートの作成

2番目のビューにはボタンが含まれています。このビューを作成するには、すでに作成した要素をコピーして名前を変更します。

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
[メイン]タブを選択します。[]

ステップ 2
FirstStateステートをコピーして貼り付けます。

ステップ 3
[ナビゲーション]コンポーネントで、作成した新しいステートを探し、次のウィジェットの名前を変更します。

- ステートの名前をSecondStateに変更します。
- ビューの名前をSecondViewに変更します。
- RectNextViewという名前をRectGoBackに変更します。
- LabelNextViewという名前をLabelGoBackに変更します。

ステップ 4
LabelGoBackをダブルクリックし、textテキストボックスにGo backと入力します。

図14.8 ボタン付きのSecondView

遷移とイベントの作成

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
[メイン]ステートマシンをダブルクリックします。

ステップ 2
[メイン]ステートマシンの端から双方のビューステートへの遷移を作成します。

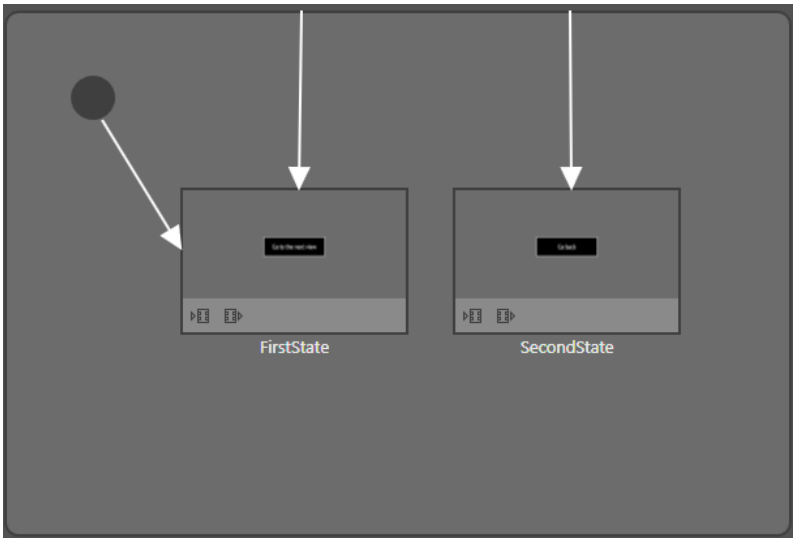

図14.9 遷移付きのMainステートマシン

ステップ 3
[メイン]ステートマシンからFirstStateへの遷移を選択します。

ステップ 4
[プロパティ]コンポーネントで、[トリガー]コンボボックスにgoToFirstStateと入力し、[イベントの追加]をクリックします。

新しいイベントが作成されます。

ステップ 5
SecondStateへの遷移を選択します。

ステップ 6
[プロパティ]コンポーネントで、[トリガー]コンボボックスにgoToSecondStateと入力し、[イベントの追加]をクリックします。

新しいイベントが作成されます。

ボタンとイベントの接続

ここで、ボタンのクリック時に別のステートへの遷移がトリガーされるという動作を定義します。そのためには、EB GUIDEスクリプトを使用します。

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、LabelNextViewをダブルクリックします。

ステップ 1.1
[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 1.2
[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。

ステップ 1.3
[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 1.4
`touchShortReleased`プロパティの横にある[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 1.5
ボタンのクリック時に`goToSecondState`イベントを発行する次の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 ev:goToSecondState()
    false
}
```

ステップ 1.6
[承認]をクリックします。

ステップ 2
[ナビゲーション]コンポーネントで、`LabelGoBack`をダブルクリックします。

ステップ 2.1
[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 2.2
[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。

ステップ 2.3
[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 2.4
[touchShortReleased]プロパティの横にある[値]列を選択し、{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 2.5
ボタンのクリック時に`goToFirstState`イベントを発行する次の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 ev:goToFirstState()
    false
}
```

ステップ 2.6
[承認]をクリックします。

FirstView開始アニメーションの作成

[VTA]コンポーネントを有効にし、ボタンが右側から入ってくるアニメーションを作成します。アニメーションを作成するには、アニメーション化されるプロパティ、アニメーションの継続時間、その開始と終了の場所を定義する必要があります。このチュートリアルでは、xプロパティのみを使用します。

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
[VTA]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。有効にする必要があります。

コマンドエリアで[レイアウト]>[VTA]の順にクリックします。

[VTA]コンポーネントが表示されます。

ステップ 2
[ナビゲーション]コンポーネントで、FirstStateをダブルクリックします。

ステップ 3
[VTA]コンポーネントで、+をクリックし、[開始アニメーション]を選択します。

[開始アニメーション]の表が表示されます。

[アニメーションエディター]がコンテンツエリアの下に表示されます。

ステップ 4
+をクリックし、[終了アニメーション]を選択します。

[終了アニメーション]の表が表示されます。

ステップ 5
[アニメーションエディター]のドロップダウンリストで、Entry animation 1を選択します。

ステップ 6
+をクリックし、[遷移先:]FirstViewを選択します。[アニメーションプロパティ]ダイアログが表示されます。

ステップ 7
RectNextView、x、[高速開始曲線]、および[承認]の順にクリックします。

新しいアニメーションが[アニメーション化プロパティ]リストに追加されます。

ステップ 8
[プロパティ]コンポーネントで、次のように入力します。

- startテキストボックスに、900と入力します。
- endテキストボックスに、290と入力します。

シミュレーションを開始すると、ボタンが移動してくることを確認できます。

FirstView終了アニメーションの作成

ボタンが右側に出て行くアニメーションを作成します。

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]のドロップダウンリストで、Exit animation 1を選択します。

ステップ 2
+をクリックし、[遷移元:]FirstViewを選択します。[アニメーションプロパティ]ダイアログが表示されます。

ステップ 3
RectNextView、x、[高速開始曲線]、および[承認]の順にクリックします。

新しいアニメーションが[アニメーション化プロパティ]リストに追加されます。

ステップ 4
[プロパティ]コンポーネントで、次のように入力します。

- durationテキストボックスに、500と入力します。
- startテキストボックスに、290と入力します。
- endテキストボックスに、800と入力します。

SecondView開始アニメーションの作成

ボタンが右側から入ってくるアニメーションを作成します。

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、SecondViewをダブルクリックします。

SecondView がコンテンツエリアに表示されます。

ステップ 2
[VTA]タブで、以下をクリックします。＋

ステップ 3
[開始アニメーション]と[終了アニメーション]を追加します。

[アニメーションエディター]がコンテンツエリアの下に表示されます。

ステップ 4
[アニメーションエディター]のドロップダウンリストで、Entry animation 2を選択します。

ステップ 5
＋をクリックし、[遷移先:]SecondViewを選択します。[アニメーションプロパティ]ダイアログが表示されます。

ステップ 6
RectGoBack、x、[高速開始曲線]、および[承認]の順にクリックします。

新しいアニメーションが[アニメーション化プロパティ]リストに追加されます。

ステップ 7
[プロパティ]コンポーネントで、次のように入力します。

- startテキストボックスに、900と入力します。
- endテキストボックスに、290と入力します。

SecondView終了アニメーションの作成

ボタンが右側に出て行くアニメーションを作成します。

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]のドロップダウンリストで、Exit animation 2を選択します。

ステップ 2
＋をクリックし、[遷移元:]SecondViewを選択します。[アニメーションプロパティ]ダイアログが表示されます。

ステップ 3
RectGoBack、x、[高速開始曲線]、および[承認]の順にクリックします。

新しいアニメーションが[アニメーション化プロパティ]リストに追加されます。

ステップ 4
[プロパティ]コンポーネントで、次のように入力します。

- durationテキストボックスに、500と入力します。
- startテキストボックスに、290と入力します。

- endテキストボックスに、800と入力します。

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。

Go to the next viewをクリックします。ビューが変化し、アニメーションが再生されます。

Go backをクリックします。ビューが最初のビューに再び変化し、アニメーションが再生されます。

14.10. チュートリアル: アニメーションでのスクリプト曲線の使用

注記

デフォルトのウィンドウレイアウト
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。

アニメーションで独自の曲線を定義する場合は、スクリプト曲線を使用します。その他のアニメーション曲線では適さない場合や、カスタム曲線を定義したい場合には、独自の曲線の定義が必要になることがあります。このチュートリアルでは、2つのアニメーション用に2つのスクリプト曲線を持つ簡単なモデルを作成します。以下の要素を作成します。

- ビューステート
- 2つの四角形ウィジェット
- スクリプト曲線によって四角形ウィジェットの位置をアニメーション化する2つのアニメーションウィジェット

結果として、2つの四角形を持つモデルが生成されます。一方の四角形は下に移動します。他方は横に移動します。

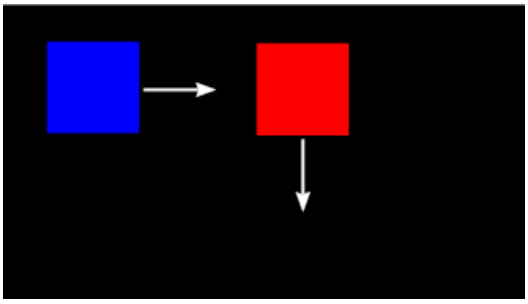

図14.10 移動方向が指定された四角形

所要時間: 15分

最初のスクリプト曲線の作成

前提条件:

- [メイン]ステートマシンに、初期ステート、`FirstState`というビューステート、および`FirstView`というビューが含まれていること。
- 初期ステートに`FirstState`への遷移があること。
- コンテンツエリアに`FirstView`ビューが表示されます。

ステップ 1
[ツールボックス]コンポーネントからこのビューに四角形をドラッグし、その名前を次のように変更します。
`BlueRectangle`

ステップ 2
[プロパティ]コンポーネントで、`fillColor`を青に設定します。

ステップ 3
[ツールボックス]コンポーネントからこのビューにアニメーションをドラッグし、その名前を`MoveAnimation`に変更します。

ステップ 4
[データプール]コンポーネントで、`Float`型のデータプールアイテムを追加し、その名前を`xFloat`に変更します。

ステップ 5
[ナビゲーション]コンポーネントで`BlueRectangle`を選択します。

ステップ 6
[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。

メニューが開きます。

ステップ 7
メニューから[条件スクリプト]を選択します。

`Conditional script 1` が[ユーザー定義プロパティ]に追加されます。

ステップ 8
Conditional script 1という名前をStartBlueAnimationに変更します。

ステップ 9
StartBlueAnimationの横にある{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 10
次のスクリプトを入力します。

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

ステップ 11
[ナビゲーション]コンポーネントでFirstViewを選択します。

ステップ 12
[アニメーションエディター]で、[アニメーション化プロパティ]の横にある+をクリックし、FirstViewを選択します。

[アニメーションプロパティ]ダイアログが開きます。

ステップ 13
BlueRectangleの下で、xプロパティを選択し、さらに次のものを選択します。Script curve

ステップ 14
[承認]をクリックします。

Script curve 1 が[アニメーションエディター]に追加されます。

ステップ 15
Script curve 1という名前をBlueCurveに変更します。

ステップ 16
[プロパティ]コンポーネントで、curveプロパティの横にある{ } をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 17
次のスクリプトを入力します。

```
function(v:diff::int, v:t_anim::int)
{
 dp:xFloat+=0.2
 f:floor(dp:xFloat*dp:xFloat)
}
```

2番目のスクリプト曲線の作成

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
[ツールボックス]コンポーネントから[FirstView]内に四角形をドラッグし、その名前を`RedRectangle`に変更します。

ステップ 2
[プロパティ]コンポーネントで、`fillColor`プロパティを赤に設定します。

ステップ 3
[データプール]コンポーネントで、`Integer`型のデータプールアイテムを追加し、その名前を`1_diff`に変更します。

ステップ 4
`Integer`型のデータプールアイテムをもう1つ追加し、その名前を`2t_anim`に変更します。

ステップ 5
`RedRectangle`を選択します。

ステップ 6
[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、条件スクリプトタイプのプロパティを追加します。

`Conditional script 2` が追加されます。

ステップ 7
`Conditional script 2`という名前を`StartRedAnimation`に変更します。

ステップ 8
`StartRedAnimation`の横にある{} をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 9
次のスクリプトを入力します。

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

ステップ 10
[ナビゲーション]コンポーネントで`FirstView`を選択します。

ステップ 11
[アニメーションエディター]で、[アニメーション化プロパティ]の横にある+をクリックし、`FirstView`を選択します。

[アニメーションプロパティ]ダイアログが開きます。

ステップ 12
`RedRectangle`の下で、yプロパティ、`Script curve`の順に選択します。

ステップ 13
[承認]をクリックします。

`Script curve 2` が[アニメーションエディター]に追加されます。

ステップ 14
`Script curve 2`という名前を`RedCurve`に変更します。

ステップ 15

[プロパティ]コンポーネントで、`curve`プロパティの横にある{ }をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 16
次のスクリプトを入力します。

```
function(v:diff::int, v:t_anim::int)
{
 dp:"1_diff"=v:diff
 dp:"2t_anim"=v:t_anim
    v:t_anim/2::int
}
```

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。

シミュレーションの開始時にアニメーションが再生されます。

14.11. チュートリアル: 水平プログレスバーの作成

注記

デフォルトのウィンドウレイアウト
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、次のようなプログレスバーをモデリングする手順を説明します。

図14.11 プログレスバー

ウィジェットテンプレートライブラリにあるプログレスバーテンプレートも参考になります。https://www.elektrobit.com/ebguide/examples/をご覧ください。

所要時間: 10分

ウィジェットの追加

ここからは、プログレスバーにウィジェットを追加する手順について説明します。

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。
- 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
[テンプレート]コンポーネントで、＋ をクリックしてからContainer。

コンテナーが含まれるテンプレートが作成されます。

ステップ 2
テンプレートの名前をT_ProgressBarに変更します。

ステップ 3
コンテナーの名前をProgressBar_Containerに変更します。

ステップ 4
四角形をコンテナーの中にドラッグしてドロップし、その名前をBackground_Rectangleに変更します。

ステップ 5
別の四角形をコンテナーの中にドラッグしてドロップし、その名前をProgress_Rectangleに変更します。

この四角形が、操作の進捗をビジュアル化します。

ステップ 6
ラベルをコンテナーの中にドラッグしてドロップし、その名前をPercentage_Textに変更します。

プログレスバーのプロパティの入力

ここからは、プロパティを設定し、ウィジェットにスクリプト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
[テンプレート]コンポーネントでProgressBar_Container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width, height, x, yのプロパティを追加します。

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
[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、+をクリックしてIntegerを選択します。

Integer型のユーザー定義プロパティがコンテナーに追加されます。

ステップ 4
プロパティの名前をprogressに変更します。

ステップ 5
progressをテンプレートインターフェースに追加します。

ステップ 6
[テンプレート]コンポーネントでBackground_Rectangleを選択します。

ステップ 7
widthをProgressBar_Containerのwidthプロパティにリンクします。

プロパティを別のプロパティにリンクするには、[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[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 8
ダイアログで、ProgressBar_Containerのwidthプロパティを選択し、[承認]をクリックします。

ステップ 9
Background_Rectangleのheightプロパティを、ProgressBar_Containerのheightプロパティにリンクします。

ステップ 10
[テンプレート]コンポーネントでProgress_Rectangleを選択します。

ステップ 11
heightプロパティをProgressBar_Containerのheightプロパティにリンクします。

ステップ 12
fillColorを緑色に設定します。

ステップ 13
widthプロパティの横にあるをクリックしてから、Convert to scriptを選択します。

widthプロパティは、ProgressBar_Containerの幅に対するパーセント値として幅を定義します。

ステップ 14
{ }をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 15
次のEB GUIDEスクリプトスクリプトを[読み取り]セクションに入力します。

```
function()
{
 v:this->^.width * v:this->^.progress / 100
}
```

このスクリプトは、progressプロパティを100で除算します。

ステップ 16
[使用可能なトリガーをリストに追加]をクリックします。

widthとprogressに対応する2つのトリガーが追加されます。

ステップ 17
[テンプレート]コンポーネントでPercentage_Textを選択します。

ステップ 18
widthプロパティとheightプロパティをProgressBar_Containerの幅と高さにリンクします。

ステップ 19
horizontalAlignをcenter (1)に設定します。

ステップ 20
textプロパティをスクリプトに変換します。

このテキストは、コンテナーの幅のパーセント値を表示します。

ステップ 21
{ }をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 22
[読み取り]セクションに、次のスクリプトを入力します。

```
function()
{
 f:int2string(v:this->^.progress) + "%"
}
```

このスクリプトは、パーセント値を文字列に変換し、その数値の後に%記号を追加します。

ステップ 23
[使用可能なトリガーをリストに追加]をクリックします。

progressに対応するトリガーが追加されます。

ステップ 24
テンプレートにあるすべてのウィジェットのxプロパティとyプロパティを0に設定します。

ステップ 25
[ナビゲーション]コンポーネントでビューをダブルクリックします。

ステップ 26
[ツールボックス]コンポーネントで、T_ProgressBarをドラッグして、コンテンツエリアにドロップします。

テンプレートがビューに追加されます。これで、操作の進捗を動的に表示するためにアニメーションをビューに追加する準備が整いました。

プログレスバーのアニメーション化

ここからは、プログレスバーをアニメーション化する手順について説明します。アニメーション化によって、パーセント値が変わったときに起こる表示の変化がわかりやすくなります。

前提条件:

- 前のセクションの手順を完了していること。
- コンテンツエリアにビューが表示されます。

ステップ 1
アニメーションをビューにドラッグしてドロップします。

ステップ 2
アニメーションの名前をLoading_Animationに変更します。

ステップ 3
[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、Conditional scriptを選択します。

ステップ 4
条件スクリプトの名前をanimateProgressに変更します。

ステップ 5
条件スクリプトの横にある{ }をクリックします。

EB GUIDEスクリプトエディターが開きます。

ステップ 6
[トリガー時]セクションに、次のスクリプトを入力します。

```
function(v:arg0::bool)
{
 f:animation_play(v:this)
 false
}
```

ステップ 7
[ナビゲーション]コンポーネントで、Loading_Animationをダブルクリックして[アニメーションエディター]を開きます。

ステップ 8
[アニメーションエディター]で、[アニメーション化プロパティ]の横にある+をクリックし、View 1を選択します。

[アニメーションプロパティ]ダイアログが開きます。

ステップ 9
T_ProgressBar 1の下で、progressプロパティ、Linear interpolation curveの順に選択します。[承認]をクリックします。

ステップ 10
[プロパティ]コンポーネントで、endプロパティを100に設定します。

プログレスバーが100%に達すると、進捗のアニメーションは停止します。

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。

15. 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。

EB GUIDE GTF固有のパラメータ、プロパティ、識別子については、EB GUIDE GTFユーザーガイドを、ご覧ください。

15.1. コマンドラインオプション

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

次の表に、EB GUIDE StudioのStudio.Console.exeで使用できるコマンドラインオプションとその意味を示します。未定義のコマンドラインオプションは、エラーメッセージをプロンプトしません。

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15.1 のコマンドラインオプション Studio.Console.exe

オプション	説明
-c <logfile dir>	EB GUIDEモデルを検証し、ログファイルを次のオプションで指定されたディレクトリに書き込む: logfile dir
-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>	エクスポート時に、profile で指定されたプロファイルを使用

例15.1
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」

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

次の表に、EB GUIDE Monitorの`Monitor.Console.exe`で使用できるコマンドラインオプションとその意味を示します。未定義のコマンドラインオプションは、エラーメッセージをプロンプトしません。

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

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`(簡体字中国語)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15.2
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、第11章「EB GUIDE Monitorを操作する」をご覧ください。

15.2. データプールアイテム

表15.3 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

15.3. 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。

15.3.1. 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、15.3.12「リスト」をご覧ください。

15.3.2. 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、15.3.12「リスト」をご覧ください。

15.3.3. 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]セクションでは、[トリガー時]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、またはウィジェットプロパティを選択できます。
- [トリガー時]セクションには、初期化時、イベントトリガー時、またはデータプールアイテムやウィジェットプロパティの値を更新した後に呼び出されるEB GUIDEスクリプトを追加できます。

  [トリガー時] EB GUIDEスクリプトのパラメータは、スクリプトを実行する要因を表します。

  `arg0`は、EB GUIDEスクリプトが初期化時やトリガーによって実行されるかどうかを参照します。注意事項を以下に示します。

  - 初期化時にEB GUIDEスクリプトが実行される場合、`arg0`はtrueになります。
  - トリガーによってEB GUIDEスクリプトが実行される場合、`arg0`はfalseになります。

  [トリガー時] EB GUIDE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。

  [トリガー時] EB GUIDEスクリプトの戻り値によって、EB GUIDEスクリプトで通知を生成する必要があるかどうかを制御します。注意事項を以下に示します。

  - 戻り値がtrueの場合、通知が生成されます。
  - 戻り値がfalseの場合、通知は生成されません。

[トリガー時]スクリプトを実行可能にするには、以下の場合に条件を満たす必要があります。

- 初期化時。例えば、データプールアイテムの場合はEB GUIDEモデル起動時、ウィジェットプロパティの場合はビュー作成時。
- トリガースクリプトからイベントを処理するとき。イベントが一致するたびにEB GUIDEスクリプトが実行されます。
- トリガースクリプトから1つ以上のアイテムのデータプール通知を処理するとき。複数の通知を一度に処理できます。
- トリガースクリプトから1つ以上のウィジェットプロパティの通知を処理するとき。複数の通知を一度に処理できます。

15.3.4. 浮動小数点数

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、15.3.12「リスト」をご覧ください。

15.3.5. 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次の場所にコピーします: `$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、15.3.12「リスト」をご覧ください。

15.3.6. `Function () : bool`

`Function () : bool`を使って、独自の関数を作成できます。

このデータタイプに使用できる操作は、すべてのプロパティの読み取り/実行操作です。

15.3.7. IBL

IBLは、IBLGeneratorによって生成されたライティング情報を格納するデータフォーマットです。

EB GUIDEプロジェクトにIBLを追加するには、`.ebibl`ファイルを次の場所にコピーします: `$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) (=)

IBL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、15.3.12「リスト」をご覧ください。

15.3.8. 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次の場所にコピーします: `$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、15.3.12「リスト」をご覧ください。

15.3.9. 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、15.3.12「リスト」をご覧ください。

15.3.10. 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。

EB GUIDEプロジェクトにメッシュを追加するには、.ebmeshファイルを次の場所にコピーします: `$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) (=)

メッシュ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、15.3.12「リスト」をご覧ください。

15.3.11. 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、15.3.12「リスト」をご覧ください。

15.3.12. 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。

- 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
- IBLリスト
- イメージリスト
- 整数リスト
- メッシュリスト
- 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: ([])

15.4. EB GUIDEスクリプト

15.4.1. EB GUIDEスクリプトのキーワード

以下はEB GUIDEスクリプトの予約されているキーワードの一覧です。スクリプト内でこれらの単語を識別子として使用する場合は、単語を引用符で囲む必要があります。

キーワード	説明
`cancel_fire`	`fire_delayed`によって発行されるイベントをキャンセルします。
`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。`f:setLanguage(l:English,true)`で使用します。
`else`	`if`条件部が完了しています。代わりに後続のブロックが実行されます。
`ev:`	イベントが後に続きます。
`f:`	ユーザー定義関数が後に続きます。

キーワード	説明
`false`	ブールリテラル値
`fire`	イベントを発行します
`fire_delayed`	指定の時間後にイベントが発行されます。時間はミリ秒単位で指定します。
`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
`s:`	スキンが後に続きます。`f:setSkin(mySkin, true)`で使用します。
`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

15.4.2. EB GUIDEスクリプトの演算子の優先順位

以下は、EB GUIDEスクリプトの演算子とそれらの優先順位および結合性の一覧です。演算子は上から下へ優先順位が高いものから低いものの順に示されています。

表15.4 EB GUIDEスクリプトの演算子の優先順位

演算子	結合性
(())、({})	なし
([])	なし
(->)	左
(.)	なし
(::)	左

演算子	結合性
長さ	なし
(&)	右
(!)、(-) 単項マイナス	右
(*)、(/)、(%)	左
(+)、(-)	左
(<)、(>)、(<=)、(>=)	左
(!=)、(==)、(=Aa=)	左
(&&)	左
(||)	左
(=)、(+=)、(-=)、(=>)	右
(,)	右
(;)	左

15.4.3. 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。

15.4.3.1. EB GUIDEスクリプトの関数A～B

15.4.3.1.1. abs

この関数は整数xの絶対値を返します。

表15.5 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

15.4.3.1.2. absf

この関数は浮動小数点数xの絶対値を返します。

表15.6 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

15.4.3.1.3. acosf

この関数はxの逆余弦の主値を返します。

表15.7 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

15.4.3.1.4. 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15.8 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。

15.4.3.1.5. 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15.9 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
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。

15.4.3.1.6. 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15.10 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。

15.4.3.1.7. 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15.11 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。

15.4.3.1.8. 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15.12 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。

15.4.3.1.9. animation_pause

この関数はアニメーションを一時停止します。

表15.13 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。

15.4.3.1.10. 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15.14 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。

15.4.3.1.11. 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15.15 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。

15.4.3.1.12. 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15.16 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。

15.4.3.1.13. animation_set_time

この関数はアニメーションの現在の時間を設定します。アニメーションをスキップまたは再生するために使用できます。

表15.17 のパラメータ animation_set_time

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

パラメータ	タイプ	説明
<return>	ブール値	trueである場合、関数が成功しました。

15.4.3.1.14. asinf

この関数はxの逆正弦の主値を計算します。

表15.18 のパラメータ asinf

パラメータ	タイプ	説明
x	浮動小数点数	逆正弦を返す数
<return>	浮動小数点数	戻り値

15.4.3.1.15. 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、y/xの逆正接の主値を計算します。

表15.19 のパラメータ atan2f

パラメータ	タイプ	説明
y	浮動小数点数	引数 y
x	浮動小数点数	引数 x
<return>	浮動小数点数	戻り値

15.4.3.1.16. 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、y/xの逆正接の主値を計算します。

表15.20 のパラメータ atan2i

パラメータ	タイプ	説明
y	整数	引数 y
x	整数	引数 x
<return>	浮動小数点数	戻り値

15.4.3.1.17. atanf

この関数はxの逆正接の主値を計算します。

表15.21 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

15.4.3.1.18. bool2string

この関数は、ブール変数を文字列trueまたはfalseのいずれかに変換します。

表15.22 のパラメータ bool2string

パラメータ	タイプ	説明
x	ブール値	文字列に変換する値
<return>	文字列	true xがtrueの場合、その他はfalse

15.4.3.2. EB GUIDEスクリプトの関数C～H

15.4.3.2.1. ceil

この関数は引数以上の最小整数値を返します。

表15.23 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

15.4.3.2.2. 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15.24 のパラメータ changeDynamicStateMachinePriority

パラメータ	タイプ	説明
stack	ポップアップスタックID	動的ステートマシンのリスト
sm	ステートマシンID	動的ステートマシン

パラメータ	タイプ	説明
priority	整数	リスト内の動的ステートマシンの優先順位。数値が大きいほど優先順位が高いことに注意してください。

15.4.3.2.3. 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15.25 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
<return>	整数	Unicode値としての文字 エラーの場合は0

15.4.3.2.4. clampf

この関数は、浮動小数点値を定義済みの範囲[xmin, xmax]にクランプします。つまり、この関数はmax (xmin, min (xmax, x))を計算します。

表15.26 のパラメータ clampf

パラメータ	タイプ	説明
x	浮動小数点数	クランプする値
xmin	浮動小数点数	最小範囲
xmax	浮動小数点数	最大範囲
<return>	浮動小数点数	[xmin, xmax]の範囲にクランプされた値x

15.4.3.2.5. clampi

この関数は、整数値を定義済みの範囲[xmin, xmax]にクランプします。つまり、この関数はmax (xmin, min (xmax, x))を計算します。

表15.27 のパラメータ clampi

パラメータ	タイプ	説明
x	整数	クランプする値
xmin	整数	最小範囲

パラメータ	タイプ	説明
xmax	整数	最大範囲
<return>	整数	[xmin, xmax]の範囲にクランプされた値x

15.4.3.2.6. 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15.28 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

15.4.3.2.7. color2string

この関数は色を8桁の16進数値に変換します。

表15.29 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。

15.4.3.2.8. 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15.30 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

パラメータ	タイプ	説明
<return>	浮動小数点数	戻り値

15.4.3.2.9. deg2rad

この関数は角度を度数からラジアンに変換します。

表15.31 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

15.4.3.2.10. expf

この関数はe(自然対数の底)のx乗の値を返します。

表15.32 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

15.4.3.2.11. 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15.33 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

15.4.3.2.12. 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15.34 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

パラメータ	タイプ	説明
<return>	整数	丸めた値

15.4.3.2.13. fmod

この関数は、浮動小数点除算x/yの剰余を計算します。

表15.35 のパラメータ fmod

パラメータ	タイプ	説明
x	浮動小数点数	浮動小数点分子
y	浮動小数点数	浮動小数点分母
<return>	浮動小数点数	除算の剰余 x/y

15.4.3.2.14. focusMoveTo

この関数は、フォーカスマネージャーのフォーカスを専用のフォーカス可能な要素へ強制的に進めます。

表15.36 のパラメータ focusMoveTo

パラメータ	タイプ	説明
widget	ウィジェット	フォーカスを移動するウィジェット
<return>	void	

15.4.3.2.15. 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15.37 のパラメータ focusNext

パラメータ	タイプ	説明
<return>	void	

15.4.3.2.16. 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15.38 のパラメータ focusPrevious

パラメータ	タイプ	説明
<return>	void	

15.4.3.2.17. format_float

この関数は浮動小数点値をフォーマットします。

表15.39 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。
useDotAsDelimiter	ブール値	区切り記号を定義します。 使用可能な値: ▶ true: ドットを区切り記号として使用します。 ▶ false: カンマを区切り記号として使用します。
value	浮動小数点数	フォーマットする数

警告

C++のprintf仕様の順守

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。

15.4.3.2.18. format_int

この関数は整数値をフォーマットします。

表15.40 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:

パラメータ	タイプ	説明
		%[フラグ] [幅] [.精度]タイプ ▶ フラグ: 出力配置、および符号、空白、先行ゼロ、小数点、8進数と16進数のプレフィックスの出力を制御するオプションの1文字または複数文字。 ▶ 幅: 出力される最小文字数を指定するオプションの10進数。 ▶ 精度: 出力される最小桁数を指定するオプションの10進数。 ▶ タイプ: 関連引数が文字、文字列、整数、または浮動小数点数として解釈されるかどうかを判別する必須の変換指定子文字。
value	整数	フォーマットする数

警告

C++のprintf仕様の順守

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。

15.4.3.2.19. frac

この関数は、浮動小数点値の端数部分を計算します。戻り値は、[0, 1]の範囲になります。例えば、この関数はパラメータ値がx=1.5またはx=-1.5の場合、0.5を返します。

表15.41 のパラメータ frac

パラメータ	タイプ	説明
x	浮動小数点数	浮動小数点値
<return>	浮動小数点数	浮動小数点値の端数部分

15.4.3.2.20. getAllLanguages

この関数は、コアスコープまたはモデルスコープから取得した言語UIDのリストをデータプールアイテムに設定します。

表15.42 のパラメータ getAllLanguages

パラメータ	タイプ	説明
itemId	dp_id	言語UIDを保存するデータプールアイテムのID。データプールアイテムの型は、文字列リストである必要があります。

パラメータ	タイプ	説明
isCoreScope	ブール値	スコープを指定します。 使用可能な値: ▶ true: コアスコープ ▶ false: モデルスコープ
<return>	void	

15.4.3.2.21. getAllSkins

この関数は、コアスコープまたはモデルスコープから取得したスキンUIDのリストをデータプールアイテムに設定します。

表15.43 のパラメータ getAllSkins

パラメータ	タイプ	説明
itemId	dp_id	スキンUIDを保存するデータプールアイテムのID。データプールアイテムの型は、文字列リストである必要があります。
isCoreScope	ブール値	スコープを指定します。 ▶ true: コアスコープ ▶ false: モデルスコープ
<return>	void	

15.4.3.2.22. getConfigItem

この関数は、設定アイテム値でデータプールアイテムを埋めます。

表15.44 のパラメータ getConfigItem

パラメータ	タイプ	説明
itemId	dp_id	設定アイテムが格納されているデータプールID
name	文字列	設定アイテム名
<return>	ブール値	データプールアイテムが設定アイテム値で正常に埋められている場合はtrueになります。

15.4.3.2.23. getFontAscender

この関数はパラメータとして渡されたフォントのアセンダーを返します。

表15.45 のパラメータ getFontAscender

パラメータ	タイプ	説明
x	フォント	評価されるフォント マルチフォントサポートを追加している場合、デフォルトフォントのみが評価されることに注意してください。
<return>	整数	フォントのアセンダー

15.4.3.2.24. getFontDescender

この関数はパラメータとして渡されたフォントのディセンダーを返します。

表15.46 のパラメータ getFontDescender

パラメータ	タイプ	説明
x	フォント	評価されるフォント マルチフォントサポートを追加している場合、デフォルトフォントのみが評価されることに注意してください。
<return>	整数	フォントのディセンダー

15.4.3.2.25. getFontLineGap

この関数はパラメータとして渡されたフォントの行間を返します。

表15.47 のパラメータ getFontLineGap

パラメータ	タイプ	説明
x	フォント	評価されるフォント マルチフォントサポートを追加している場合、デフォルトフォントのみが評価されることに注意してください。
<return>	整数	フォントの行間

15.4.3.2.26. getImageHeight

この関数は、パラメータとして渡されたイメージの高さをピクセル単位で返します。

表15.48 のパラメータ getImageHeight

パラメータ	タイプ	説明
x	イメージウィジェット	評価するウィジェット
<return>	整数	ピクセル単位のイメージの高さ

15.4.3.2.27. getImageWidth

この関数は、パラメータとして渡されたイメージの幅をピクセル単位で返します。

表15.49 のパラメータ getImageWidth

パラメータ	タイプ	説明
x	イメージウィジェット	評価されるウィジェット
<return>	整数	ピクセル単位のイメージの幅

15.4.3.2.28. getLabelTextHeight

この関数はラベルテキストの全高をピクセル単位で返します。全高は次の式で計算されます。

```
total_height = line_height * line_count + line_spacing * (line_count - 1)
```

line_spacingは、フォントのlineGapプロパティと[複数行]ウィジェット機能のlineOffsetプロパティの合計として計算されます。フォントのlineGapとlineOffsetプロパティの両方とも負にすることができます。

表15.50 のパラメータ getLabelTextHeight

パラメータ	タイプ	説明
widget	ラベルウィジェット	評価されるウィジェット
<return>	整数	ピクセル単位のテキストの高さ

15.4.3.2.29. getLabelTextWidth

この関数はラベルテキストの最長行の幅をピクセル単位で返します。

表15.51 のパラメータ getLabelTextWidth

パラメータ	タイプ	説明
widget	ラベルウィジェット	評価するウィジェット

パラメータ	タイプ	説明
<return>	整数	ピクセル単位のテキストの最長行の幅

15.4.3.2.30. getLanguage

この関数は、コアスコープまたはモデルスコープの現在の言語を返します。

表15.52 のパラメータ getLanguage

パラメータ	タイプ	説明
isCoreScope	ブール値	スコープを指定します。 ► true: コアスコープ ► false: モデルスコープ
<return>	文字列	言語のUID。

15.4.3.2.31. getLanguageName

この関数は、指定された言語UIDの名前を返します。

表15.53 のパラメータ getLanguageName

パラメータ	タイプ	説明
languageUid	文字列	名前を要求する言語。
<return>	文字列	言語の名前。

15.4.3.2.32. getLanguageTag

この関数は、指定された言語UIDのタグを返します。

表15.54 のパラメータ getLanguageTag

パラメータ	タイプ	説明
languageUid	文字列	タグを要求する言語。
<return>	文字列	言語のタグ。

15.4.3.2.33. getLineCount

この関数はラベルテキストの行数を返します。

表15.55 のパラメータ getLineCount

パラメータ	タイプ	説明
widget	ラベルウィジェット	評価されるウィジェット
<return>	整数	テキストの行数

15.4.3.2.34. getLineHeight

この関数は、パラメータとして渡されたフォントで書かれた行の高さを返します。

表15.56 のパラメータ getLineHeight

パラメータ	タイプ	説明
x	フォント	評価されるフォント マルチフォントサポートを追加している場合、デフォルトフォントのみが評価されることに注意してください。
<return>	整数	指定のフォントで書かれた行の高さ

15.4.3.2.35. getProductString

この関数はEB GUIDE GTFの製品名を含む文字列を返します。

表15.57 のパラメータ getProductString

パラメータ	タイプ	説明
<return>	文字列	製品名

15.4.3.2.36. getSkin

この関数は、コアスコープまたはモデルスコープの現在のスキンを返します。

表15.58 のパラメータ getSkin

パラメータ	タイプ	説明
isCoreScope	ブール値	スコープを指定します。 ► true: コアスコープ ► false: モデルスコープ

パラメータ	タイプ	説明
<return>	文字列	スキンのUID。

15.4.3.2.37. getSkinName

この関数は、指定されたスキンUIDの名前を返します。

表15.59 のパラメータ getSkinName

パラメータ	タイプ	説明
skinUid	文字列	名前を要求するスキン。
<return>	文字列	スキンの名前。

15.4.3.2.38. 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。高さはフォントのアセンダーとディセンダーの合計を表します。

表15.60 のパラメータ getTextHeight

パラメータ	タイプ	説明
text	文字列	評価するテキスト
font	フォント	評価するフォント
<return>	整数	テキストの高さ 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。

注記

getTextHeight
この関数は常にテキストを1行と想定して高さの値を計算します。

15.4.3.2.39. getTextLength

この関数はテキスト内の文字数を返します。

表15.61 のパラメータ getTextLength

パラメータ	タイプ	説明
text	文字列	評価するテキスト

パラメータ	タイプ	説明
<return>	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。

15.4.3.2.40. 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15.62 のパラメータ getTextWidth

パラメータ	タイプ	説明
text	文字列	評価するテキスト
font	フォント	評価するフォント
<return>	整数	テキストの幅 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。

注記

この関数は常にテキストを1行と想定して幅の値を計算します。

15.4.3.2.41. getVersionString

この関数はEB GUIDE GTFのバージョン番号を含む文字列を返します。

表15.63 のパラメータ getVersionString

パラメータ	タイプ	説明
<return>	文字列	バージョン文字列

15.4.3.2.42. has_list_window

この関数はタイプリストのデータプールアイテムでインデックスが有効かどうかを確認します。ウィンドウ表示リストでは、インデックスが少なくとも1つのウィンドウ内にあるかどうかも確認します。

表15.64 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。

15.4.3.2.43. hsba2color

この関数はHSB/HSV色をEB GUIDE GTF色に変換します。

表15.65 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EB GUIDE GTF色

15.4.3.3. EB GUIDEスクリプトの関数I～R

15.4.3.3.1. 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15.66 のパラメータ int2float

パラメータ	タイプ	説明
value	整数	浮動小数点数に変換する値
<return>	浮動小数点数	浮動小数点数に変換された整数値

15.4.3.3.2. int2string

この関数は単純整数を文字列に変換します。

表15.67 のパラメータ int2string

パラメータ	タイプ	説明
value	整数	文字列に変換する値
<return>	文字列	文字列に変換された整数値(10進表記)

15.4.3.3.3. isDynamicStateMachineActive

この関数は、ある動的ステートマシンが動的ステートマシンのリストに含まれるかどうかをチェックします。

表15.68 のパラメータ isDynamicStateMachineActive

パラメータ	タイプ	説明
stack	ポップアップスタックID	動的ステートマシンのリスト
sm	ステートマシンID	動的ステートマシン

15.4.3.3.4. isWidgetOnActiveStatemachine

この関数は、ウィジェットがアクティブなステートマシンに属しているかどうかを確認します。

表15.69 のパラメータ isWidgetOnActiveStatemachine

パラメータ	タイプ	説明
widget	ウィジェット	評価されるウィジェット
<return>	ブール値	ウィジェットがアクティブなステートマシンに属している場合はtrueになります。

15.4.3.3.5. language

(旧式): setLanguageを代わりに使用してください。

この関数はコアスコープの言語を切り替えます。この処理は同期して実行されますが、この変更へのモデルの反応は非同期です。

表15.70 のパラメータ language

パラメータ	タイプ	説明
language	文字列	切り替える言語(など) f:setLanguage(l:German,true)

パラメータ	タイプ	説明
<return>	void	

15.4.3.3.6. lerp

この関数は、次の式を使用して2つの値xおよびyのリニア補間を計算します。(1-s) * x + s * y

表15.71 のパラメータ lerp

パラメータ	タイプ	説明
x	浮動小数点数	最初の値
y	浮動小数点数	2番目の値
s	浮動小数点数	値xおよびyの間をリニア補間する値
<return>	浮動小数点数	次のリニア補間を返します。(1-s) * x + s * y

15.4.3.3.7. 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15.72 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

15.4.3.3.8. 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15.73 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

15.4.3.3.9. 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15.74 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

15.4.3.3.10. 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15.75 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

15.4.3.3.11. 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15.76 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

15.4.3.3.12. 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15.77 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

15.4.3.3.13. 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15.78 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

15.4.3.3.14. log10f

この関数はxの10を底とする対数を返します。

表15.79 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

15.4.3.3.15. logf

この関数はxの自然対数を返します。

表15.80 のパラメータ logf

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

15.4.3.3.16. maxf

この関数は2つの浮動小数点値の最大値を計算します。

表15.81 のパラメータ maxf

パラメータ	タイプ	説明
x	浮動小数点数	最初の値
y	浮動小数点数	2番目の値
<return>	浮動小数点数	xおよびyの最大値

15.4.3.3.17. `maxi`

この関数は2つの整数値の最大値を計算します。

表15.82 のパラメータ `maxi`

パラメータ	タイプ	説明
`x`	整数	最初の値
`y`	整数	2番目の値
`<return>`	整数	xおよびyの最大値 `y`

15.4.3.3.18. `minf`

この関数は2つの浮動小数点値の最小値を計算します。

表15.83 のパラメータ `minf`

パラメータ	タイプ	説明
`x`	浮動小数点数	最初の値
`y`	浮動小数点数	2番目の値
`<return>`	浮動小数点数	次の最小値: `x`および `y`

15.4.3.3.19. `mini`

この関数は2つの整数値の最小値を計算します。

表15.84 のパラメータ `mini`

パラメータ	タイプ	説明
`x`	整数	最初の値
`y`	整数	2番目の値
`<return>`	整数	次の最小値: `x`および `y`

15.4.3.3.20. `nearbyint`

この関数は最も近い整数に丸めます。

表15.85 のパラメータ nearbyint

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

15.4.3.3.21. popDynamicStateMachine

この関数は動的ステートマシンを動的ステートマシンリストから削除します。

表15.86 のパラメータ popDynamicStateMachine

パラメータ	タイプ	説明
stack	ポップアップスタックID	動的ステートマシンのリスト
sm	ステートマシンID	動的ステートマシン

15.4.3.3.22. powf

この関数はxのy乗の値を返します。

表15.87 のパラメータ powf

パラメータ	タイプ	説明
x	浮動小数点数	引数 x
y	浮動小数点数	引数 y
<return>	浮動小数点数	戻り値

15.4.3.3.23. pushDynamicStateMachine

この関数は動的ステートマシンを動的ステートマシンリストに挿入します。

表15.88 のパラメータ pushDynamicStateMachine

パラメータ	タイプ	説明
stack	ポップアップスタックID	動的ステートマシンのリスト
sm	ステートマシンID	動的ステートマシン

パラメータ	タイプ	説明
priority	整数	リスト内の動的ステートマシンの優先順位。数値が大きいほど優先順位が高いことに注意してください。

15.4.3.3.24. rad2deg

この関数は角度をラジアンから度数に変換します。

表15.89 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

15.4.3.3.25. rand

この関数は0～2^{31}-1の間の乱数値を取得します。

表15.90 のパラメータ rand

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

15.4.3.3.26. rgba2color

この関数はRGB色空間からEB GUIDE GTF色に変換します。

表15.91 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
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EB GUIDE GTF色に変換され、アルファ値が適用された色

15.4.3.3.27. 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15.92 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

15.4.3.4. EB GUIDEスクリプトの関数S～W

15.4.3.4.1. saturate

この関数は、浮動小数点値を[0, 1]の範囲にクランプします。つまり、この関数はmax (0, min (1, x))を計算し、clampf(0, 1, x)の略式表記としての役割を果たします。clampf(0, 1, x)

表15.93 のパラメータ saturate

パラメータ	タイプ	説明
x	浮動小数点数	クランプする値
<return>	浮動小数点数	[0, 1]の範囲にクランプされた値x

15.4.3.4.2. seed_rand

この関数は乱数生成のシードを設定します。

表15.94 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	

15.4.3.4.3. setLanguage

この関数はモデルスコープの言語を切り替えます。この処理は同期して実行されますが、この変更へのモデルの反応は非同期です。

表15.95 のパラメータ setLanguage

パラメータ	タイプ	説明
languageUid	文字列	切り替える言語(f:language(l:German)など)
isCoreScope	ブール値	スコープを指定します。 ▶ true: コアスコープ ▶ false: モデルスコープ
<return>	void	

15.4.3.4.4. setSkin

この関数はモデルスコープの言語を切り替えます。この処理は同期して実行されますが、この変更へのモデルの反応は非同期です。

表15.96 のパラメータ setSkin

パラメータ	タイプ	説明
skinUid	文字列	切り替えるスキン(f:setSkin(x,y)など)
isCoreScope	ブール値	スコープを指定します。 ▶ true: コアスコープ ▶ false: モデルスコープ
<return>	void	

15.4.3.4.5. 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。

15.4.3.4.6. 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15.97 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

15.4.3.4.7. skin

(旧式): setSkinを代わりに使用してください。

この関数はコアスコープのスキンを切り替えます。この処理は同期して実行されますが、この変更へのモデルの反応は非同期です。

表15.98 のパラメータ skin

パラメータ	タイプ	説明
skin	文字列	切り替えるスキン(など) f:setSkin(x,y)
<return>	void	

15.4.3.4.8. smoothstep

この関数は、値が[xmin, xmax]の範囲内にある場合、滑らかなエルミート補間3z² - 2z³ with z = (x - xmin) / (xmax - xmin)を計算します。この範囲内にない場合は0を返します。この関数は[0,1]の範囲の値を返します。

表15.99 のパラメータ smoothstep

パラメータ	タイプ	説明
xmin	浮動小数点数	xmin値
xmax	浮動小数点数	xmax値
x	浮動小数点数	補間される値
<return>	浮動小数点数	次のエルミート補間を返します。3z² - 2z³ with z = (x-xmin) / (xmax-xmin)

15.4.3.4.9. sqrtf

この関数はxの負ではない平方根を返します。

表15.100 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

パラメータ	タイプ	説明
<return>	浮動小数点数	戻り値

15.4.3.4.10. 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15.101 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
<return>	浮動小数点数	戻り値

15.4.3.4.11. 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15.102 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

15.4.3.4.12. string2string

この関数は、文字列の切り捨てによって指定の文字数のみを残すために使用します。

表15.103 のパラメータ string2string

パラメータ	タイプ	説明
str	文字列	切り捨て前の文字列
len	整数	文字列の最大長
<return>	文字列	切り捨て後の文字列

15.4.3.4.13. 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) abcを返します。
- substring("abc", 0, -2) abを返します。
- substring ("abcd", 1, 3) bcを返します。

表15.104 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
<return>	文字列	言語の文字列

15.4.3.4.14. 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15.105 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

15.4.3.4.15. 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15.106 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

15.4.3.4.16. 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15.107 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

15.4.3.4.17. 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15.108 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	

15.4.3.4.18. 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15.109 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	

15.4.3.4.19. 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15.110 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

15.4.3.4.20. 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15.111 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

15.4.3.4.21. 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15.112 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

15.4.3.4.22. 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15.113 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

15.4.3.4.23. 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15.114 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

15.4.3.4.24. 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15.115 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
<return>	整数	子ウィジェットの数

15.5. イベント

表15.116 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

プロパティ名	説明
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。

15.5.1. キーイベントの10進コード

表15.117 テンキーの10進コード

テンキー	10進コード
0	5
1	6
2	7
3	8
4	9
5	10
6	11
7	12
8	13
9	14

表15.118 ファンクションキーの10進コード

ファンクションキー	10進コード
F1キー	18
F2キー	19
F3キー	20
F4キー	21
F5キー	22
F6キー	23
F7キー	24
F8キー	25
F9キー	26
F10キー	27

ファンクションキー	10進コード
F11キー	28
F12キー	29

表15.119 ASCIIキーの10進コード

ASCIIキー	10進コード
スペース	32
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

15.6. ボタンおよびアイコン

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

表15.120 共通のアイコン

共通のアイコン	説明
	元に戻す
	やり直し
	保存
	プロジェクトを検証。
	シミュレーションを開始。
	シミュレーションを停止。
	プロジェクトセンターを開閉。
	イベント、データプールアイテム、ステートマシンなどの要素を追加。
	イベント、データプールアイテム、ステートマシンなどの要素を追加。

表15.121 プロジェクトセンターのアイコン

プロジェクトセンターのアイコン	説明
	新しいプロジェクトを作成できるタブを示す。
	既存のプロジェクトを開くことができるタブを示す。
	モデルインターフェース、イベントグループ、言語およびスキンなど、複数のプロジェクトオプションを設定できるタブを示す。
	EB GUIDEモデルのエクスポートを作成できるタブを示す。
	ユーザーマニュアルにアクセスできるタブを示す。
	ユーザーインターフェース言語を変更できるタブを示す。
	読み込まれたプラグインを確認できるタブを示す。
	プロジェクトインターフェースをインポート。

プロジェクトセンターのアイコン	説明
	プロジェクトインターフェースをエクスポート。

表15.122 プロジェクトエディターのアイコン

プロジェクトエディターのアイコン	説明
	コンポーネントを非表示。
	非表示のコンポーネントを表示。
	コンテンツエリアと[ナビゲーション]または[テンプレート]のコンポーネントを同期。
	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 ボタンの色の意味は次のとおりです。  プロパティがローカルである  プロパティが別のプロパティにリンクされている  プロパティがデータプールアイテムにリンクされている  プロパティ値テンプレート値と等しい
	EB GUIDEスクリプトエディターを開く。
	名前空間オプションを開くか、選択された名前空間を表示。
	すべての名前空間を表示。
	サブ名前空間を含める。
	設定を開く。
	データプールアイテムを示す。
	遷移を示す。
	動的ステートマシンのリストを示す。
	エントリーアクションを示す。
	終了アクションを示す。
	内部遷移を示す。
	テンプレートを示す。

プロジェクトエディターのアイコン	説明
	このウィジェットのテンプレートにジャンプ。
	検索結果またはリスト要素をフィルタ。
	Photoshopファイルを示す。
	ウィジェット機能プロパティに変更が加えられたことを示す。
	トリガーのリストを示す。
	モデルインターフェース別のグループ化と解除を切り替え。

表15.123 ステートのアイコン

ステートのアイコン	説明
	選択ステートを追加。
	深い履歴ステートを示す。
	最終ステートを示す。
	初期ステートを示す。
	ステートマシンを示す。
	浅い履歴ステートを示す。
	ビューステートを示す。

表15.124 基本ウィジェットのアイコン

基本ウィジェットのアイコン	説明
	アルファマスクを示す。
	コンテナーを示す。
	カスタムウィジェットを示す。
	楕円を示す。

基本ウィジェットのアイコン	説明
	イメージを示す。
	インスタンシエータを示す。
	ラベルを示す。
	四角形を示す。

表15.125 アニメーションのアイコン

アニメーションのアイコン	説明
	アニメーションウィジェットを示す。
	コンスタント曲線によるアニメーションを示す。
	高速開始曲線によるアニメーションを示す。
	リニア曲線によるアニメーションを示す。
	リニア補間曲線によるアニメーションを示す。
	二次曲線によるアニメーションを示す。
	スクリプト曲線によるアニメーションを示す。
	正弦曲線によるアニメーションを示す。
	低速開始曲線によるアニメーションを示す。
	変更アニメーションを示す。
	開始アニメーションを示す。
	終了アニメーションを示す。
	アニメーションエディターを展開。
	アニメーションエディターを最小化。
	ポップアップオンアニメーションを示す。

アニメーションのアイコン	説明
	ポップアップオフアニメーションを示す。

表15.126 3Dウィジェットのアイコン

3Dウィジェットのアイコン	説明
	環境光を示す。
	カメラを示す。
	指向性ライトを示す。
	イメージベースドライトを示す。
	材質を示す。
	メッシュを示す。
	点ライトを示す。
	シーングラフを示す。
	シーングラフノードを示す。
	スポットライトを示す。

表15.127 [問題]検出コンポーネントのアイコン

[問題]検出コンポーネントのアイコン	説明
	モデルを検証。
	既知の問題を示す。
	情報を示す。
	正常実行を示す。
	警告を示す。
	破損したリンクを示す。

表15.128 EB GUIDE Monitorのアイコン

EB GUIDE Monitorのアイコン	説明
	イベントを発行するか、イベントが発行済みであることを示す。
	キーイベントが発行済みであることを示す。
	ホストへの接続が確立されているかどうかを示す。
	接続の設定を開く。
	ログの自動スクロールをオンまたはオフ。
	すべてのログメッセージをコピー。
	ログメッセージを削除。
	ウォッチリストをエクスポート。
	ログメッセージを示す。
	モデルインターフェース別のグループ化と解除を切り替え。

15.7. シーン

表15.129 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)

プロパティ名	説明
	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キー ▶ 16-bit (2): RGB565キー ▶ 24-bit (3): RGB888キー ▶ 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。
antiAliasing	使用可能な値: ▶ Off (0): アンチエイリアスなし ▶ MSAA 2x (1): 2倍のアンチエイリアス ▶ MSAA 4x (2): 4倍のアンチエイリアス 6.3「アンチエイリアス」もご覧ください。
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: ▶ 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

プロパティ名	説明
	▶ `Console & on screen (3)`: 画面およびコンソールにFPSを表示する ▶ `On screen (large text) (4)` ▶ `Console & on screen (large text) (5)`
`Renderer`	シーンのレンダラーを定義します。 使用可能な値: ▶ `OpenGLRenderer` ▶ `OpenGL3Renderer`

注記

シーン設定での`sceneID`の使用

シーン設定で同じ`sceneID`を使用している場合、複数のステートマシンが入力処理に同時に反応します。

これを避け、1つのステートマシンのみが入力処理に反応するようにするには、シーン設定で各ステートマシンに異なる`sceneID`値を割り当てます。

15.8. ショートカット

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

表15.130 ショートカット

ショートカット	説明
Ctrl+Aキー	すべての要素を選択
Ctrl+Cキー	選択内容をコピー
Ctrl+Fキー	検索ボックスにジャンプ
Ctrl+Sキー	保存
Ctrl+Vキー	コピーした選択内容を貼り付け
Ctrl+Yキー	やり直し
Ctrl+Zキー	元に戻す
Enterキー	表の中で、入力値を確認し、次のセルに移動します。
Ctrl+Enterキー	トリガーフィルタボックスに新しいイベントを追加します。 表の中で、入力値を確認し、現在のセルに留まります。
Ctrl+Shift+Insertキー	[名前空間]コンポーネントで、新しい名前空間を子として既存の名前空間に追加します。

ショートカット	説明
Alt+F4キー	アクティブなウィンドウを閉じる
Shift+F1キー	EB GUIDE TFのユーザーマニュアルを開く
F1キー	EB GUIDE Studioのユーザーマニュアルを開く
F2キー	選択された要素の名前変更
Shift+F2キー	選択した要素が使用されているすべての場所(EB GUIDEスクリプトなど)で、それらの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。
F3キー	EB GUIDEモデル内における選択した要素への参照をすべて検索
F4キー	元のテンプレートへジャンプ
F5キー	シミュレーションの開始
F6キー	検証
Deleteキー	選択された要素を削除。
-	ツリーの中で、選択された要素を折りたたむ。
+	ツリーの中で、選択された要素を展開。
*	ツリーの中で、選択された要素とその要素のすべての子を展開。
上/下/左/右方向キー	コンテンツエリアの中で、選択されたステートまたはウィジェットを上、下、左、または右へ1ピクセル移動。 表の中では、要素を通過。
Ctrlキーを押しながら左マウスボタンをクリック Shiftキーを押しながら、左マウスボタンをクリック、あるいは上矢印または下矢印キー	複数の要素を選択。
Ctrlキーを押しながら回転ボタンを回す Ctrl++キー Ctrl+-キー Ctrl+0キー	コンテンツエリアの中で、表示を拡大、縮小、または100%にリセット

15.9. ウィジェット

15.9.1. ビュー

表15.131 ビューのプロパティ

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`	ウィジェットのx座標
`y`	ウィジェットのy座標

ビューステートとビューテンプレートには、View Transition Animation用に追加のプロパティがあります。View Transition Animationは、開始アニメーション、終了アニメーション、変更アニメーション、ポップアップオンアニメーション、およびポップアップオフアニメーションに適用されます。

表15.132 ビュー遷移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`プロパティで定義されます。
`scale`	アニメーション時間を乗じる際の係数。
`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。 ► 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`

プロパティ名	説明
	▶ 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`

15.9.2. 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
- インスタンシエータ
- 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います。

15.9.2.1. 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15.133 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。
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: ► 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。レンダラー(OpenGL ESバージョン2.0以降向け)では、.pngファイルおよび .jpgファイルRGBイメージは、グレイ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。グレイ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。

アルファマスク機能は9-patchイメージには適用されません。9-patchイメージは PNGおよびJPEGファイル形式と同じ方法で処理されます。

15.9.2.2. 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15.134 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。

プロパティ名	説明
	繰り返し回数はrepeatで定義されます。
scale	アニメーション時間を乗じる際の係数
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。 ► 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

15.9.2.2.1. 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15.135 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

15.9.2.2.2. 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15.136 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

15.9.2.2.3. 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15.137 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

15.9.2.2.4. 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15.138 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

15.9.2.2.5. 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15.139 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

15.9.2.2.6. スクリプト曲線

スクリプト曲線は、EB GUIDEスクリプトによって独自に定義できる曲線です。その他の曲線では使用できないアニメーションや独自のカスタムアニメーションを使用したい場合には、スクリプト曲線を使用します。この曲線が特に役立つのは、カスタマイズされた軌跡をウィジェットの移動で使用したい場合です。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15.140 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	EB GUIDEスクリプトで曲線の関数を定義します。次の2つのパラメータを指定します。 ► diff: 前回の実行からのミリ秒単位の経過時間です。アニメーションの開始時点ではdiffが0になっています。 ► t_anim: アニメーション開始以降のミリ秒単位の経過時間です。
target	結果値が適用されるターゲットプロパティ

15.9.2.2.7. 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15.141 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

プロパティ名	説明
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

15.9.2.2.8. 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15.142 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

15.9.2.3. コンテナー

コンテナーは、複数のウィジェットを子ウィジェットとして格納し、それらをグループ化します。

表15.143 コンテナーのプロパティ

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

15.9.2.4. 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15.144 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
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

15.9.2.5. イメージ

イメージは、画像をビューに配置します。

表15.145 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。OpenGL ES 2.0以降のレンダラーは、.pngファイルと.jpgファイルをサポートしています。

15.9.2.6. インスタンシエータ

インスタンシエータは、ランタイムにウィジェットインスタンスを作成します。インスタンシエータを使用して、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。インスタンシエータの子ウィジェットは、ランタイムに作成されるリストまたは表のラインテンプレートとして利用されます。デフォルトでは、インスタンシ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15.146 インスタンシ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。

15.9.2.7. ラベル

ラベルは、テキストをビューに配置します。

注記

文字置換

ラベルのtextプロパティにテキストを入力すると、以下のような文字の置き換えが行われます。

- シーケンス\\\\は \\に置き換えられます。
- シーケンス\\nは \nに置き換えられます。
- テキストが1行に表示される場合、\nは 1文字の空白に置き換えられます。

表15.147 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

プロパティ名	説明
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

15.9.2.8. 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15.148 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

15.9.3. 3Dウィジェット

15.9.3.1. 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、PBR GGX材質、およびPBR Phong材質のambient色プロパティに影響します。

表15.149 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。

プロパティ名	説明
color	光の色です。
intensity	光の強度です。下限値である0.0は、環境光がないことを示します。

15.9.3.2. 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15.150 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。測定単位は、サードパーティの3Dモデリングソフトウェアで3Dモデルを作成する際に定義されます。
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。測定単位は、サードパーティの3Dモデリングソフトウェアで3Dモデルを作成する際に定義されます。
fieldOfView	カメラの垂直方向の視野角度。最大値は180
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。 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。

15.9.3.3. 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15.151 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。下限値である0.0は、指向性ライトがないことを示します

15.9.3.4. イメージベースドライト

イメージベースドライティングは、.pfmまたは.hdrファイルに保存された現実世界のライティング情報に基づいてシーンを照らすライトです。.pfmまたは.hdrファイルは、.ebiblファイルを作成する際にIBLGeneratorの入力データとして利用されます。

表15.152 イメージベースド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。
ibl	手動で作成されたIBLファイル.ebibl
intensity	ライトの強度です。0.0はイメージベースドライトがないことを示します。

15.9.3.5. 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15.153 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。[ディフューズテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
emissive	オブジェクトの自己発光色。[エミッシブテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
shininess	光沢要素 有効なのは0.0と1.0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。 [光沢テクスチャ]ウィジェット機能が使用されている場合、shininessプロパティは無視されます。
specular	表面に光沢があるオブジェクトが反射する色。[スペキュラテクスチャ]ウィジェット機能が追加されている場合、またはshininessプロパティが0.0に設定されている場合、specularプロパティは無効です。
opacity	オパシティ値 有効なのは0.0と1.0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。

15.9.3.6. メッシュ

メッシュは3Dオブジェクトの形状を定義します。

表15.154 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。

15.9.3.7. 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15.155 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[アンビエントテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
`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。[ディフューズテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
`emissive`	オブジェクトの自己発光色。[エミッシブテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
`specular`	表面に光沢があるオブジェクトが反射する色。[スペキュラテクスチャ]ウィジェット機能が追加されている場合、または`shininess`プロパティが`0.0`に設定されている場合、`specular`プロパティは無効です。
`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。

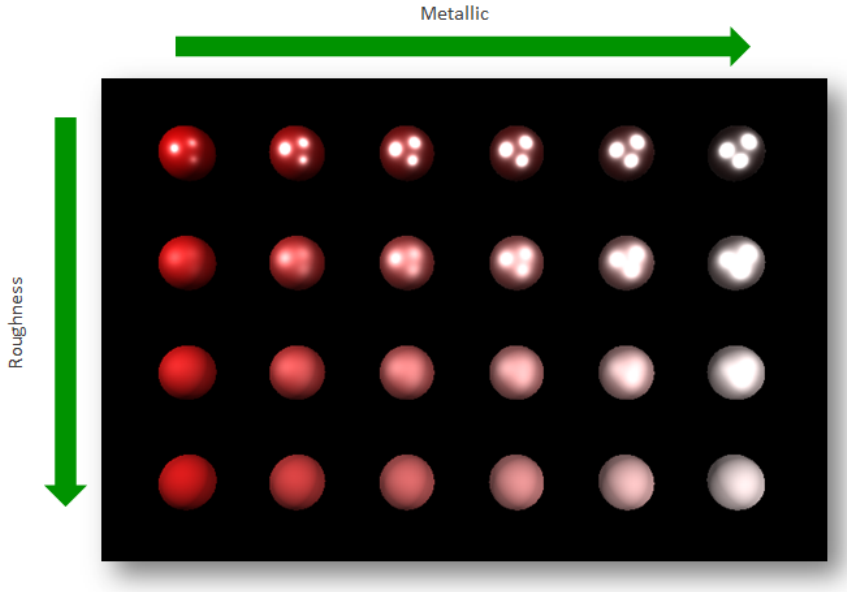

図15.1 物理ベース材質のサンプル

15.9.3.8. 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15.156 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[アンビエントテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
`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。[ディフューズテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
`emissive`	オブジェクトの自己発光色。[エミッシブテクスチャ]ウィジェット機能が追加されている場合、このプロパティは無効です。
`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。[スペキュラテクスチャ]ウィジェット機能が追加されている場合、または`shininess`プロパティが`0.0`に設定されている場合、`specular`プロパティは無効です。
`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。

プロパティ名	説明
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。

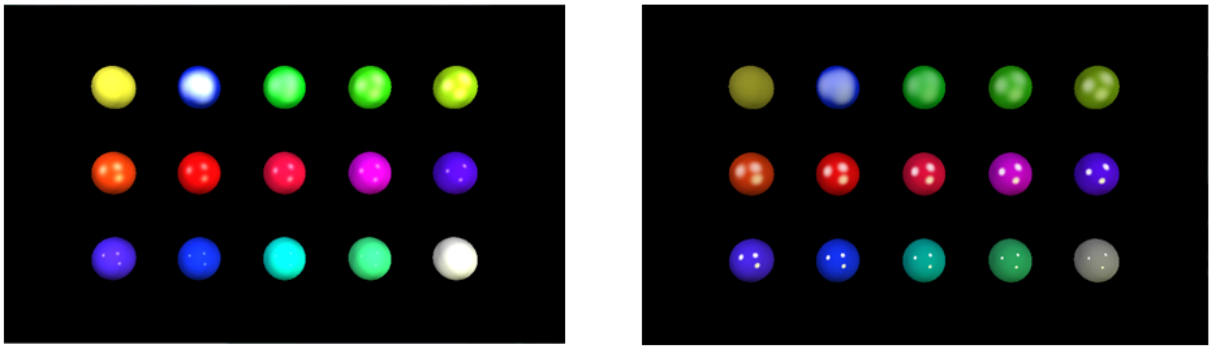

図15.2 正規化されていない材質(左)と正規化された材質(右)のサンプル

15.9.3.9. 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15.157 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。下限値0.0は点ライトがないことを示し、上限値は減衰係数によって異なります。
attenuationConstant	距離が離れるにつれライトの強度を減退させる定数係数。値が0.0の場合、この係数は使用されません。
attenuationLinear	距離が離れるにつれライトの強度を減退させる線形係数。値が0.0の場合、この係数は使用されません。
attenuationQuadratic	距離が離れるにつれライトの強度を減退させる二次係数。値が0.0の場合、この係数は使用されません。

15.9.3.10. 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15.158 シーングラフのプロパティ

プロパティ名	説明
visible	trueである場合、ウィジェットおよび子ウィジェットが表示されます。
width	ピクセル単位のウィジェットの幅
height	ピクセル単位のウィジェットの高さ
x	親ウィジェットを基準にしたウィジェットのx座標
y	親ウィジェットを基準にしたウィジェットのy座標
gamma	シーングラフの輝度出力を補正します。デフォルト値は2.2に設定されています。

15.9.3.11. 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
- イメージベースドライト
- メッシュ
- 点ライト
- スポットライト

表15.159 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

15.9.3.12. 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15.160 スポットライトのプロパティ

プロパティ名	説明
enabled	trueである場合、ウィジェットが有効になります。
color	ライトの色
intensity	ライトの強度です。下限値0.0はスポットライトがないことを示し、上限値は減衰係数によって異なります。
attenuationConstant	距離が離れるにつれライトの強度を減退させる定数係数
attenuationLinear	距離が離れるにつれライトの強度を減退させる線形係数
attenuationQuadratic	距離が離れるにつれライトの強度を減退させる二次係数
coneAngleInner	ライトの円錐の内側の角度(度単位)。最大値は180
coneAngleOuter	ライトの円錐の外側の角度(度単位)。最大値は180

15.10. ウィジェット機能

次のリストには、実装されているすべてのウィジェット機能の説明と、これらをEB GUIDEモデルで使用する方法に関する簡単な説明が含まれます。

15.10.1. 共通

15.10.1.1. 子の可視性の選択

[子の可視性の選択]ウィジェット機能は、子ウィジェットの可視性を処理します。表示する1つのウィジェットを定義したり、同時に表示する子ウィジェットのグループを定義したりできます。グループを定義するには、子ウィジェットのインデックスを同じグループ値にマッピングします。

表15.161 [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	EB GUIDE GTFによって設定
containerIndex	子ウィジェットの可視性を制御します。 containerMappingが設定されていない場合、containerIndexによって1つの子ウィジェットが表示さ	入力

プロパティ名	説明	EB GUIDE GTFによって設定
	れます。表示される子ウィジェットは、ウィジェットツリーにおけるそのウィジェットの順序によって識別されます。最上位の子のcontainerIndexが0、次のcontainerIndexが1というように続きます。 containerMappingが設定されている場合、containerIndexは子ウィジェットのグループを参照します。containerMappingでグループを定義します。	
containerMapping	子ウィジェットのグループを作成するには、このプロパティを使用します。[インデックス]列は子ウィジェットを識別します。[値]列はグループを定義します。 行の数は、子ウィジェットの数と一致している必要があります。そうでない場合、マッピングは使用されません。	入力

15.10.1.2. 有効

[有効]ウィジェット機能は、ウィジェットにenabledプロパティを追加します。

表15.162 [有効]ウィジェット機能のプロパティ

プロパティ名	説明	EB GUIDE GTFによって設定
enabled	trueである場合、ウィジェットはタッチ入力および押下入力に反応します。	入力

15.10.1.3. フォーカス

[フォーカス]ウィジェット機能は、ウィジェットに入力フォーカスを設定できるようにします。

表15.163 [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	EB GUIDE GTFによって設定
focusable	ウィジェットにフォーカスするかどうかを定義します。使用可能な値: ▶ not focusable (0) ▶ only by touch (1) ▶ only by key (2)	入力

プロパティ名	説明	EB GUIDE GTFによって設定
	▶ focusable (3)	
focused	trueである場合、ウィジェットがフォーカスされます。	x

15.10.1.4. フォントメトリック

[フォントメトリック]ウィジェット機能を使うと、ラベルに使われるフォントの設定を変更できます。

lineGapを変更する方法については、8.4.2「行間の変更」をご覧ください。

制限:

- [フォントメトリック]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15.164 [フォントメトリック]ウィジェット機能のプロパティ

プロパティ名	説明	EB GUIDE GTFによって設定
ascender	フォントのベースラインより上に出ている文字の一部。	入力
descender	フォントのベースラインより下に出ている文字の一部。	入力
lineGap	すべてのフォントにデフォルトで含まれる行間。正の値は行間を増やし、負の値は行間を減らします。	入力

15.10.1.5. 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。改行は、ラベルウィジェットに設定されたwidthプロパティに従い、一連の単語または文字の間に設定されます。行の終端にマーク付けするため、強制改行文字\nを使うこともできます。

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15.165 [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	EB GUIDE GTFによって設定
lineOffset	行間の空白のサイズ。正の値は行間を増やし、負の値は行間を減らします。 lineOffset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	入力

プロパティ名	説明	**EB GUIDE GTF**によって設定
	ントスタイルがPT_Sans_Narrow、サイズが30に設定されていて、lineOffsetが-50と定義されている場合に発生します。	
maxLineCount	表示行の最大数。0 = 制限なし	入力

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、15.4.3.2.33「getLineCount」をご覧ください。

注記

文字置換

ラベルのtextプロパティにテキストを入力すると、以下のような文字の置き換えが行われます。

- シーケンス\\\\は \\に置き換えられます。
- シーケンス\\nは \nに置き換えられます。
- テキストが1行に表示される場合、\nは 1文字の空白に置き換えられます。

15.10.1.6. 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15.166 [押下]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
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。	x

[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。

15.10.1.7. 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ヒューマンマシンインターフェース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15.167 [選択]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`selected`	trueである場合、ウィジェットが選択されます。 [選択グループ]ウィジェット機能を追加すると、この機能によってbuttonIDとbuttonValueが同じであるかどうかが評価されます。trueである場合、ボタンが選択されます。	入力 [選択グループ]ウィジェット機能を追加すると、このプロパティはEB GUIDE GTFによって自動的に設定されます。

15.10.1.8. 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15.168 [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	**EB GUIDE GTF**によって設定
`buttonId`	ボタングループ内のボタンを識別するID	入力
`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。	入力

15.10.1.9. 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15.169 [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	EB GUIDE GTFによって設定
currentValue	現在の回転値	x
maxValue	currentValueプロパティの最大値	入力
minValue	currentValueプロパティの最小値	入力
incValueTrigger	trueである場合、currentValueプロパティが1増加します。	入力
incValueReaction	currentValueプロパティの増加に対する反応	入力
decValueTrigger	trueである場合、現在の値が1減少します。	入力
decValueReaction	currentValueプロパティの減少に対する反応	入力
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	入力
valueWrapAround	使用可能な値: ▶ true: minValueまたはmaxValueを超えた場合、currentValueプロパティは逆側に進みます。 ▶ 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。	入力

15.10.1.10. テキストの切り捨て

[テキストの切り捨て]ウィジェット機能は、textプロパティのコンテンツがウィジェットエリアに収まらない場合にそのコンテンツを切り捨てます。ウィジェット機能を使用すると、デフォルト設定のtrailingとは異なる切り捨てが可能になります。

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15.170 [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	EB GUIDE GTFによって設定
truncationPolicy	1行のテキストの場合、truncationPolicyプロパティは切り捨ての位置を定義します。使用可能な値: ▶ leading (0): テキストはテキストの先頭で置き換えられます。 ▶ trailing (1): テキストはテキストの末尾で置き換えられます。	入力

プロパティ名	説明	EB GUIDE GTFによって設定
	複数行のテキストの場合、truncationPolicyプロパティは、テキストを置き換える位置を定義します。使用可能な値: ▶ leading (0): 先頭にある行が置き換えられ、最初に表示される行のテキストがテキストの先頭で切り捨てられます。 ▶ trailing (1) 末尾にある行が置き換えられ、最後に表示される行のテキストがテキストの末尾で切り捨てられます。	
truncationSymbol	置き換えられたテキスト部分の代わりに表示される文字列	入力

15.10.1.11. 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15.171 [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	EB GUIDE GTFによって設定
touchable	trueである場合、ウィジェットはタッチ入力に反応します。	入力
touched	trueである場合、ウィジェットは現在タッチされています。	x
touchPolicy	ウィジェットの境界を超えるタッチおよび動きを処理する方法を定義します。使用可能な値: ▶ Press then react (0): 最初に押下すると、ウィジェットが反応します。移動および解放の通知はウィジェットエリア内でのみアクティブです。 ▶ Press and grab (1): 押すとコンタクトをグラブできます。コンタクトは、ウィジェットエリアの外部に移動しても、グラブされたままです。 ▶ Press then react on contact (2): 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。	入力
touchBehavior	タッチ評価を定義します。使用可能な値: ▶ Whole area (0): タッチされたウィジェットを識別するために、レンダラーはウィジェットのクリッピング四角形を評価します。 ▶ Visible pixels (1): タッチされたウィジェットを識別するために、レンダラーはタッチされたピクセルが属するウィジェットを評価します。	入力

プロパティ名	説明	**EB GUIDE GTF**によって設定
	アルファ透過性を持つ透過ピクセル、またはOまたはAなどの文字内のピクセルにはタッチできません。 `Visible pixels`値はラベルに対して無効であるという点に注意してください。	

[タッチ]ウィジェット機能を[押下]ウィジェット機能に組み合わせると、押ボタンをモデル化できます。

ティップ

パフォーマンスに関する推奨事項
プロジェクトにおいてパフォーマンスが重要な問題である場合、`touchBehavior`プロパティに`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。

15.10.2. 効果

15.10.2.1. 枠

[枠]ウィジェット機能は、設定可能な枠をウィジェットに追加します。枠はウィジェットの境界から始まり、ウィジェット内に配置されます。

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15.172 [枠]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`borderThickness`	ピクセル単位の枠の厚さ	
`borderColor`	枠のレンダリングに使用する色	
`borderStyle`	枠のレンダリングに使用するスタイル	

15.10.2.2. 配色

[配色]ウィジェット機能は、ウィジェットおよびウィジェットサブツリーに色を付けます。また、アルファ値が不透明でない場合は透過性にも影響します。

例15.3
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15.173 [配色]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
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	

15.10.2.3. 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15.174 [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	**EB GUIDE GTF**によって設定
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	

15.10.3. 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。

15.10.3.1. 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15.175 [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	**EB GUIDE GTF**によって設定
focusNext	フォーカスインデックスが増加する条件	
focusPrev	フォーカスインデックスが減少する条件	
focusFlow	フォーカスの動作は階層内で変化します。使用可能な値: - stop at hierarchy (0) - wrap within hierarchy level (1) - step up in hierarchy (2)	
focusedIndex	フォーカス可能なn番目の子ウィジェットとして現在フォーカスされている子ウィジェットのインデックス	x
initFocus	このインデックスは、初期化時にフォーカスされる子ウィジェットを定義します。ウィジェットがフォーカス不可能である場合、次にフォーカス可能な子が使用されます。	

15.10.3.2. 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15.176 [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	**EB GUIDE GTF**によって設定
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	

プロパティ名	説明	**EB GUIDE GTF**によって設定
`focusOrder`	`focusOrder`プロパティを使用すると、フォーカスを割り当てる際に子ウィジェットをスキップできます。子ウィジェットのIDは、サブツリー内の位置に対応しています。フォーカス不可能な子ウィジェットはデフォルトでスキップされます。子ウィジェットがフォーカスされる順序は、次のとおりです。 ▶ 定義済み: ユーザー定義のウィジェット順序を使用します。 ▶ 未定義: デフォルトのウィジェット順序を代わりに使用します。 各子ウィジェットには[フォーカス]ウィジェット機能が必要です。この機能がない場合、ウィジェットはフォーカス管理で無視されます。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。	
`focusPrev`	フォーカスを前の子に割り当てるトリガー	
`focusFlow`	フォーカスの動作は階層内で変化します。使用可能な値: ▶ `stop at hierarchy level (0)` ▶ `wrap within hierarchy level (1)` ▶ `step up in hierarchy (2)`	
`focusedIndex`	インデックスは、`focusOrder`リスト内の子ウィジェットの位置を定義します。ウィジェットがフォーカス不可能である場合、リスト内の次の子が使用されます。	x
`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	

15.10.4. ジェスチャー

15.10.4.1. フリックジェスチャー

表面を素早くなでるコンタクト

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15.177 [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	EB GUIDE GTFによって設定
onGestureFlick	ジェスチャーが認識されたときにトリガーされる反応 反応引数は、次のとおりです。 ▶ speed: フリックジェスチャーの相対速度 ピクセル/ミリ秒の速度をflickMinLength/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	x
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	

15.10.4.2. ホールドジェスチャー

動きのないホールドジェスチャー

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15.178 [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	EB GUIDE GTFによって設定
onGestureHold	ジェスチャーが認識されたときにトリガーされる反応反応はコンタクトごとに1回のみトリガーされます。つまり、holdDurationが期限切れになったときに、コンタクトがまだ最初のタッチ位置の周囲の小さな境界ボックス内にある場合です。 反応引数は、次のとおりです。 ▶ x: コンタクト位置のx座標 ▶ y: コンタクト位置のy座標	x

プロパティ名	説明	**EB GUIDE GTF**によって設定
`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	

15.10.4.3. ロングホールドジェスチャー

動きのないロングホールドジェスチャー

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15.179 [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	**EB GUIDE GTF**によって設定
`onGestureLongHold`	ジェスチャーが認識されたときにトリガーされる反応反応はコンタクトごとに1回のみトリガーされます。つまり、`longHoldDuration`が期限切れになったときに、コンタクトがまだ最初のタッチ位置の周囲の小さな境界ボックス内にある場合です。 反応引数は、次のとおりです。 - `x`: コンタクト位置のx座標 - `y`: コンタクト位置のy座標	x
`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	

15.10.4.4. パスジェスチャー

1つのコンタクトが描画した形状を、既知の形状集合とマッチングします。

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15.180 [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	**EB GUIDE GTF**によって設定
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。 ► gestureId: マッチングされたパスのID	x
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。	x
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。	x
pathMinXBox	パスジェスチャー認識機能が入力を始めるためにコンタクトが移動する必要があるピクセル単位の最小距離のx座標	
pathMinYBox	パスジェスチャー認識機能が入力を始めるためにコンタクトが移動する必要があるピクセル単位の最小距離のy座標	

15.10.4.4.1. ジェスチャーID

ジェスチャー識別子は、パスジェスチャー認識機能の設定によって決まります。次の表は、EB GUIDEに含まれるサンプル設定を示しています。

表15.181 EB GUIDEに含まれるパスジェスチャの設定の例

ID	形状	説明
0		左から右への屋根形状
1		右から左への屋根形状
2		左から右への水平線

ID	形状	説明
3		右から左への水平線
4		チェックマーク
5		左から右への波形状
6		右から左への波形状

15.10.4.5. ピンチジェスチャー

2つのコンタクトが近づくまたは離れる動き

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15.182 [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	EB GUIDE GTFによって設定
`onGesturePinchStart`	ジェスチャーの開始が認識されたときにトリガーされる反応反応引数は、次のとおりです。 - `ratio`: 最初のコンタクトの距離に対する現在のコンタクトの距離の比率 - `centerX`: 2つのコンタクト間の現在の中心点のX座標 - `centerY`: 2つのコンタクト間の現在の中心点のY座標	x

プロパティ名	説明	**EB GUIDE GTF**によって設定
`onGesturePinchUpdate`	ピンチ率または中心点が変更されたときにトリガーされる反応。反応引数は、次のとおりです。 ▶ `ratio`: 最初のコンタクトの距離に対する現在のコンタクトの距離の比率 ▶ `centerX`: 2つのコンタクト間の現在の中心点のX座標 ▶ `centerY`: 2つのコンタクト間の現在の中心点のY座標	x
`onGesturePinchEnd`	ジェスチャーが終了したときにトリガーされる反応。反応引数は、次のとおりです。 ▶ `ratio`: 最初のコンタクトの距離に対する現在のコンタクトの距離の比率 ▶ `centerX`: 2つのコンタクト間の現在の中心点のX座標 ▶ `centerY`: 2つのコンタクト間の現在の中心点のY座標	x
`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	

15.10.4.6. 回転ジェスチャー

2つのコンタクトの円に沿った動き

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15.183 [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	**EB GUIDE GTF**によって設定
`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	x
`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	x
`onGestureRotateEnd`	ジェスチャーが終了したときにトリガーされる反応	x
`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

15.10.5. 入力処理

15.10.5.1. 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。

15.10.5.2. 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15.184 [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	**EB GUIDE GTF**によって設定
`keyPressed`	キー押下に対するウィジェットの反応 反応引数は、次のとおりです。 - `keyId`: 処理されるキーのID	x

15.10.5.3. 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15.185 [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	**EB GUIDE GTF**によって設定
`keyShortReleased`	キーリリースに対するウィジェットの反応 反応引数は、次のとおりです。 - `keyId`: 処理されるキーのID	x

15.10.5.4. 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15.186 [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	**EB GUIDE GTF**によって設定
`keyStatusChanged`	キー押下またはキーリリースに対するウィジェットの反応 反応引数は、次のとおりです。 - `keyId`: 処理されるキーのID - `status`: ステータス変更の数値ID	x

15.10.5.5. 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15.187 [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	**EB GUIDE GTF**によって設定
`keyUnicode`	Unicodeキー入力に対するウィジェットの反応 反応引数は、次のとおりです。 ► `keyId`: 処理されるキーのID	x

15.10.5.6. 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15.188 [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	**EB GUIDE GTF**によって設定
`moveIn`	境界内への動きに対するウィジェットの反応 反応引数は、次のとおりです。 ► `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID ► `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	x

15.10.5.7. 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15.189 [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	**EB GUIDE GTF**によって設定
`moveOut`	境界外への動きに対するウィジェットの反応	x

プロパティ名	説明	**EB GUIDE GTF**によって設定
	反応引数は、次のとおりです。 ▶ `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID ▶ `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	

15.10.5.8. 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15.190 [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	**EB GUIDE GTF**によって設定
`moveOver`	境界内の動きに対するウィジェットの反応 反応引数は、次のとおりです。 ▶ `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID ▶ `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	x

15.10.5.9. 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15.191 [可動]ウィジェット機能のプロパティ

プロパティ名	説明	EB GUIDE GTFによって設定
moveDirection	ウィジェットが移動する方向使用可能な値: ▶ horizontal (0) ▶ vertical (1) ▶ free (2)	

15.10.5.10. 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15.192 [回転]ウィジェット機能のプロパティ

プロパティ名	説明	EB GUIDE GTFによって設定
rotaryReaction	回転に対するウィジェットの反応を定義します。trueである場合、ウィジェットは入力された回転イベントに反応します。 反応引数は、次のとおりです。 ▶ rotaryId: 整数ID ▶ increment: 入力されたイベントが送信されたときに回転入力が移動する単位数	x

15.10.5.11. 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。

▶ [タッチの喪失]を追加する場合、[タッチ]ウィジェット機能のtouchPolicyドロップダウンリストボックスでPress and grabを選択します。

[タッチの喪失]は、他のタッチポリシーでは機能しません。

表15.193 [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	**EB GUIDE GTF**によって設定
`onTouchGrabLost`	タッチコンタクトの喪失に対するウィジェットの反応 反応引数は、次のとおりです。 ▶ `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID ▶ `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	x

15.10.5.12. 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15.194 [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	**EB GUIDE GTF**によって設定
`touchMoved`	タッチでの移動に対するウィジェットの反応 反応引数は、次のとおりです。 ▶ `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID ▶ `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	x

15.10.5.13. 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15.195 [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	**EB GUIDE GTF**によって設定
`touchPressed`	押下に対するウィジェットの反応 反応引数は、次のとおりです。 - `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID - `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	x

15.10.5.14. 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15.196 [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	**EB GUIDE GTF**によって設定
`touchShortReleased`	リリースに対するウィジェットの反応 反応引数は、次のとおりです。 - `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID - `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	x

15.10.5.15. 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15.197 [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	**EB GUIDE GTF**によって設定

`touchStatusChanged`	タッチのステータス変更に対するウィジェットの反応
	
	反応引数は、次のとおりです。
	
	- `touchId`: ユーザーがクリックまたはリリースしたタッチスクリーンのID
	- `x`: x座標
	- `y`: y座標
	- `touchStatus`: タッチのタイプのID
	
	使用可能な値:
	
	  - `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
	- `fingerId`: ウィジェット内を移動するコンタクトのID
		x

15.10.6. レイアウト

15.10.6.1. 絶対レイアウト

親ウィジェットの[絶対レイアウト]ウィジェット機能は、子ウィジェットの位置およびサイズを定義します。非表示になっている子ウィジェットは無視されます。追加されたウィジェット機能プロパティは整数リストで構成されます。各リスト要素は1つの子ウィジェットにマップされます。

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15.198 [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	**EB GUIDE GTF**によって設定
`itemLeftOffset`	子ウィジェットの左枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。	
`itemTopOffset`	子ウィジェットの上枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。	
`itemRightOffset`	子ウィジェットの右枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。	
`itemBottomOffset`	子ウィジェットの下枠からのオフセットを格納する整数リスト。各リスト要素は1つの子ウィジェットにマップされます。	

15.10.6.2. ボックスレイアウト

[ボックスレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよびサイズプロパティは、親ウィジェットによって設定されます。非表示になっている子ウィジェットは計算では無視されます。

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15.199 [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	**EB GUIDE GTF**によって設定
`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:	

プロパティ名	説明	EB GUIDE GTFによって設定
	▶ horizontal (0) ▶ vertical (1)	

15.10.6.3. フローレイアウト

[フローレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよびサイズプロパティは、親ウィジェットによって設定されます。非表示になっている子ウィジェットは計算では無視されます。

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15.200 [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	EB GUIDE GTFによって設定
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: ▶ 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: ▶ leading (0): 子ウィジェットは左側に配置されます。 ▶ center (1): 子ウィジェットは中央に配置されます。 ▶ trailing (2): 子ウィジェットは右側に配置されます。	
verticalChildAlign	子ウィジェットの垂直方向の位置揃え使用可能な値: ▶ 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。	

15.10.6.4. グリッドレイアウト

[グリッドレイアウト]ウィジェット機能は、各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよびサイズプロパティは、親ウィジェットによって設定されます。非表示になっている子ウィジェットは計算では無視されます。

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15.201 [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	**EB GUIDE GTF**によって設定
`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。	

15.10.6.5. レイアウト余白

[レイアウト余白]ウィジェット機能は、[フローレイアウト]、[絶対レイアウト]、[ボックスレイアウト]、または[グリッドレイアウト]ウィジェット機能を使用するウィジェットに設定可能な余白を追加します。

表15.202 [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	**EB GUIDE GTF**によって設定
`leftMargin`	左枠の余白	
`topMargin`	上枠の余白	
`rightMargin`	右枠の余白	
`bottomMargin`	下枠の余白	

15.10.6.6. リストレイアウト

[リストレイアウト]ウィジェット機能は、ピクセル単位で各子ウィジェットの位置およびサイズを定義します。

子ウィジェットの位置プロパティおよび[リストインデックス]ウィジェット機能のlistIndexプロパティは、親ウィジェットによって設定されます。

子ウィジェットを作成するインスタンシエータとともに使用するのが最も適しています。

[リストインデックス]ウィジェット機能の詳細については、15.10.7.2「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、インスタン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15.203 [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	**EB GUIDE GTF**によって設定
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: - horizontal (0) - vertical (1)	
scrollOffset	リストをスクロールするピクセルの数	x
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	x
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。	x
scrollValue	現在のスクロール値(ピクセル単位)	x
scrollValueMax	リストの終了位置にマップされる最大スクロール値(ピクセル単位)	
scrollValueMin	リストの開始位置にマップされる最小スクロール値(ピクセル単位)	

プロパティ名	説明	**EB GUIDE GTF**によって設定
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。	x
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切り捨てられます。	
segments	水平レイアウト方向の場合: 行の数 垂直レイアウト方向の場合: 列の数	
listLength	リスト要素の数	
wrapAround	使用可能な値: ▶ true: scrollValueMinまたはscrollValueMaxを超えた場合、scrollValueプロパティは逆側に進みます。 ▶ false: scrollValueMinまたはscrollValueMaxを超えた場合、scrollValueプロパティは増加/減少しません。	

15.10.6.7. 拡大縮小モード

[拡大縮小モード]ウィジェット機能は、イメージのサイズがウィジェットのサイズと異なる場合にイメージを表示する方法を定義します。

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15.204 [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	**EB GUIDE GTF**によって設定
scaleMode	イメージの拡大縮小モード。使用可能な値: ▶ 0 = original size ▶ 1 = fit to size ▶ 2 = keep aspect ratio	

15.10.7. リスト管理

15.10.7.1. ラインインデックス

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

表15.205 [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	**EB GUIDE GTF**によって設定
`lineIndex`	表内の現在のラインのインデックス	x

15.10.7.2. リストインデックス

[リストインデックス]ウィジェット機能は、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15.206 [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	**EB GUIDE GTF**によって設定
`listIndex`	リスト内の現在のウィジェットのインデックス	x

15.10.7.3. テンプレートインデックス

[テンプレートインデックス]ウィジェット機能は、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15.207 [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	**EB GUIDE GTF**によって設定
`lineTemplateIndex`	使用されているラインテンプレートのインデックス	x

15.10.7.4. 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、コンテナーまたはリストとともに使用するよう設計されています。
- [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15.208 [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	**EB GUIDE GTF**によって設定
`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	

15.10.8. 3D

[3D]カテゴリのウィジェット機能は、3Dウィジェットに対してのみ使用することができます。

15.10.8.1. アンチエイリアスモード

表15.209 [アンチエイリアスモード]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`antiAliasing`	シーングラフのアンチエイリアスモードを定義します。この定義はシーンの設定に優先します。シーンで設定された値を使用する場合は、`Global (5)`を使用します。 - `Off (0)` - `MSAA 2x (1)` - `MSAA 4x (2)` - `MSAA 8x (3)`	

プロパティ名	説明	**EB GUIDE GTF**によって設定
	▶ FXAA (4) ▶ Global (5)	

15.10.8.2. 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15.210 [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	**EB GUIDE GTF**によって設定
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)	

15.10.8.3. クリアコート

[クリアコート]ウィジェット機能は、反射レイヤーを追加して複数レイヤー面をシミュレートします。

制限:

▶ [クリアコート]ウィジェット機能は、PBR GGX材質およびPBR Phong材質で使用できます。

表15.211 [クリアコート]ウィジェット機能のプロパティ

プロパティ名	説明
clearCoatStrength	クリアコートレイヤーの強度。使用可能な値は0.0～1.0の範囲。
clearCoatRoughness	クリアコートレイヤーの知覚される粗度。使用可能な値は0.0～1.0の範囲。

15.10.8.4. 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。
- [アンビエントテクスチャ]が追加されている場合、ambientプロパティは無視されます。

表15.212 [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	**EB GUIDE GTF** によって設定
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: - 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: - 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。	
ambientFilterMode	テクスチャのフィルタリングモード。使用可能な値: - 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。 - 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。	

プロパティ名	説明	**EB GUIDE GTF** によって設定
	▶ `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。	
`diffuseSRGB`	このプロパティが有効になっている場合、`ambientTexture`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。 sRGB機能を使用するには、プロジェクトセンターを開き、[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。	

15.10.8.5. 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。
- [ディフューズテクスチャ]が追加されている場合、`diffuse`プロパティは無視されます。

表15.213 [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	**EB GUIDE GTF** によって設定
`diffuseTexture`	テクスチャのファイル名	
`diffuseTextureAddressModeU`	u方向のテクスチャのアドレスモード。使用可能な値: ▶ `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。	
`diffuseTextureAddressModeV`	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	**EB GUIDE GTF** によって設定
	▶ `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。	
`diffuseFilterMode`	テクスチャのフィルタリングモード。使用可能な値: ▶ `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。 ▶ `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。	
`diffuseSRGB`	このプロパティが有効になっている場合、`diffuseTexture`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。 sRGB機能を使用するには、プロジェクトセンターを開き、[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。	

15.10.8.6. 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。

- [エミッシブテクスチャ]が追加されている場合、emissiveプロパティは無視されます。

表15.214 [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	**EB GUIDE GTF** によって設定
emissiveTexture	テクスチャのファイル名	
emissiveTextureAddressModeU	u方向のテクスチャのアドレスモード。使用可能な値: - 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。	
emissiveTextureAddressModeV	v方向のテクスチャのアドレスモード。使用可能な値: - 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。	
emissiveFilterMode	テクスチャのフィルタリングモード。使用可能な値: - 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。 - 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。	

プロパティ名	説明	**EB GUIDE GTF** によって設定
diffuseSRGB	このプロパティが有効になっている場合、emissiveTexture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。 sRGB機能を使用するには、プロジェクトセンターを開き、[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。	

15.10.8.7. 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15.215 [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	**EB GUIDE GTF** によって設定
lightMapTexture	テクスチャのファイル名	
lightMapTextureAddressModeU	u方向のテクスチャのアドレスモード。使用可能な値: - 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。	
lightMapTextureAddressModeV	v方向のテクスチャのアドレスモード。使用可能な値: - 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	

プロパティ名	説明	**EB GUIDE GTF** によって設定
	▶ `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。	
`lightMapFilterMode`	テクスチャのフィルタリングモード。使用可能な値: ▶ `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。 ▶ `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。	

15.10.8.8. 金属テクスチャ

[金属]ウィジェット機能は、材質に拡張設定値を追加します。このテクスチャは、PBR GGX材質とPBR Phong材質ウィジェットの金属パラメータを制御します。

制限:

- [金属テクスチャ]ウィジェット機能は、PBR GGX材質およびPBR Phong材質で使用できます。
- [金属テクスチャ]はグレースケールイメージです。RGBカラーイメージでは、赤のチャネルのみが使用されます。
- [金属テクスチャ]が追加されている場合、`metallic`プロパティは無視されます。

表15.216 [金属テクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF** によって設定
`metallicTexture`	テクスチャのファイル名	
`metallicMinFactor`	テクスチャ値を補間する浮動小数点数として最小の金属パラメータ	
`metallicMaxFactor`	テクスチャ値を補間する浮動小数点数として最大の金属パラメータ	

プロパティ名	説明	**EB GUIDE GTF** によって設定
metallicTextureAddressModeU	u方向のテクスチャのアドレスモード。使用可能な値: ▶ 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。	
metallicTextureAddressModeV	v方向のテクスチャのアドレスモード。使用可能な値: ▶ 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。	
metallicFilterMode	テクスチャのフィルタリングモード。使用可能な値: ▶ 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。 ▶ 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。	

15.10.8.9. 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15.217 [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	**EB GUIDE GTF** によって設定
`normalMapTexture`	テクスチャのファイル名	
`normalMapTextureAddressModeU`	u方向のテクスチャのアドレスモード。使用可能な値: - `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。	
`normalMapTextureAddressModeV`	v方向のテクスチャのアドレスモード。使用可能な値: - `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。	
`normalMapFilterMode`	テクスチャのフィルタリングモード。使用可能な値: - `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。 - `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。	

プロパティ名	説明	**EB GUIDE GTF** によって設定
	▶ `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。	

15.10.8.10. 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15.218 [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	**EB GUIDE GTF** によって設定
`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: ▶ `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: ▶ `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:	

プロパティ名	説明	**EB GUIDE GTF** によって設定
	▶ `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。 ▶ `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。	

15.10.8.11. 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15.219 [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	**EB GUIDE GTF** によって設定
`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: ▶ `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。 ▶ `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。	

プロパティ名	説明	**EB GUIDE GTF** によって設定
	▶ `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。	

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`

イメージファイルは、同一サイズで二次形状のものでなければなりません。

15.10.8.12. ラフネス(粗さ)テクスチャ

[ラフネス(粗さ)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。このテクスチャは、PBR GGX材質ウィジェットのラフネス(粗さ)パラメータを制御します。

制限:

- [ラフネス(粗さ)テクスチャ]ウィジェット機能は、PBR GGX材質に対して使用できます。
- [ラフネス(粗さ)テクスチャ]はグレースケールイメージです。RGBカラーイメージでは、赤のチャネルのみが使用されます。
- [ラフネス(粗さ)テクスチャ]がアクティブの場合、`roughness`プロパティは無視されます。

表15.220 [ラフネス(粗さ)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	**EB GUIDE GTF** によって設定
`roughnessTexture`	テクスチャのファイル名	
`roughnessMinFactor`	テクスチャ値を補間する浮動小数点数として最小のラフネス(粗さ)パラメータ	
`roughnessMaxFactor`	テクスチャ値を補間する浮動小数点数として最大のラフネス(粗さ)パラメータ	

プロパティ名	説明	**EB GUIDE GTF** によって設定
roughnessTextureAddressModeU	u方向のテクスチャのアドレスモード。使用可能な値: ▶ 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。	
roughnessTextureAddressModeV	v方向のテクスチャのアドレスモード。使用可能な値: ▶ 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。	
roughnessFilterMode	テクスチャのフィルタリングモード。使用可能な値: ▶ 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。 ▶ 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。	

15.10.8.13. 光沢テクスチャ

[光沢テクスチャ]ウィジェット機能は、材質に拡張設定値を追加します。このテクスチャは、スカラーshininessプロパティを持つテクスチャ値を増加させることで光沢強度を調整します。

制限:

- [光沢テクスチャ]ウィジェット機能は、材質およびPBR Phong材質で使用できます。
- [光沢テクスチャ]はグレースケールイメージです。RGBカラーイメージでは、赤のチャネルのみが使用されます。
- [光沢テクスチャ]ウィジェット機能が使用されている場合、shininessプロパティは無視されます。

表15.221 [光沢テクスチャ]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF** によって設定
shininessTexture	テクスチャのファイル名	
shininessMinFactor	テクスチャ値を補間する浮動小数点数として最小の光沢パラメータ	
shininessMaxFactor	テクスチャ値を補間する浮動小数点数として最大の光沢パラメータ	
shininessTextureAddressModeU	u方向のテクスチャのアドレスモード。使用可能な値: - 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。	
shininessTextureAddressModeV	v方向のテクスチャのアドレスモード。使用可能な値: - 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。	
shininessFilterMode	テクスチャのフィルタリングモード。使用可能な値:	

プロパティ名	説明	**EB GUIDE GTF** によって設定
	▶ `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。 ▶ `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。	

15.10.8.14. 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。
- [スペキュラテクスチャ]が追加されている場合、`specular`プロパティは無視されます。

表15.222 [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	**EB GUIDE GTF** によって設定
`specularTexture`	テクスチャのファイル名	
`specularTextureAddressModeU`	u方向のテクスチャのアドレスモード。使用可能な値: ▶ `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。	
`specularTextureAddressModeV`	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	**EB GUIDE GTF** によって設定
	▶ `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。	
`specularFilterMode`	テクスチャのフィルタリングモード。使用可能な値: ▶ `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。 ▶ `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。	
`diffuseSRGB`	このプロパティが有効になっている場合、`specularTexture`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。 sRGB機能を使用するには、プロジェクトセンターを開き、[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。	

15.10.8.15. Texture coordinate transformation

[Texture coordinate transformation]ウィジェット機能を使用すると、材質テクスチャの座標を変更できます。この機能は、材質、PBR Phong材質、およびPBR GGX材質で使用できます。

表15.223 [Texture coordinate transformation]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`uOffset`	テクスチャ座標のu方向のオフセットを定義します。	
`vOffset`	テクスチャ座標のv方向のオフセットを定義します。	
`uScale`	テクスチャ座標のu方向の拡大縮小を定義します。	
`vScale`	テクスチャ座標のv方向の拡大縮小を定義します。	

例15.4
[Texture coordinate transformation]の例

uOffset: 1.0 vOffset: 1.0 uScale: 1.0 vScale: 1.0	0.5 1.0 1.0 1.0	1.0 0.5 1.0 1.0	1.0 1.0 0.5 0.5	1.0 1.0 3.0 3.0
				

15.10.8.16. 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15.224 [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	**EB GUIDE GTF**によって設定
`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。	入力

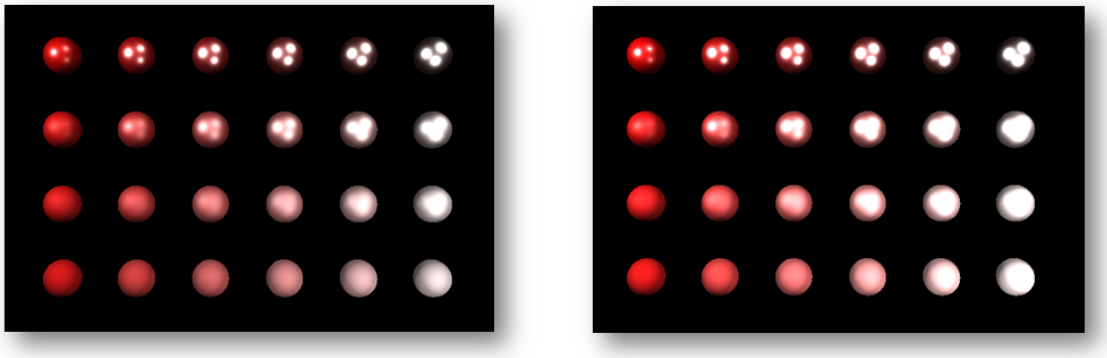

図15.3 トーンマッピングなしイメージ(左)とトーンマッピングありイメージ(右)のサンプル

15.10.8.17. カメラブルーム

[カメラブルーム]ウィジェット機能は、後処理効果です。この機能は、イメージの明るい部分の縁から光の筋を外側に向かって伸ばし、シーンをとらえたカメラや目を圧倒する強烈な光の錯覚を生成します。

制限:

- [カメラブルーム]ウィジェット機能は、カメラにのみ使用できます。

表15.225 [カメラブルーム]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`enabled`	trueの場合にブルーム効果がカメラに適用されます。	
`threshold`	効果が適用されるエリアの境目を定義する強度(輝度)値。使用可能な値は0.0～1.0の範囲。 [トーンマッピング]ウィジェット機能が親シーングラフに追加された場合、1.0より大きな値を`threshold`に使用できます。	入力
`strength`	ブルーム効果の強度。	入力
`radius`	光彩の半径。	入力

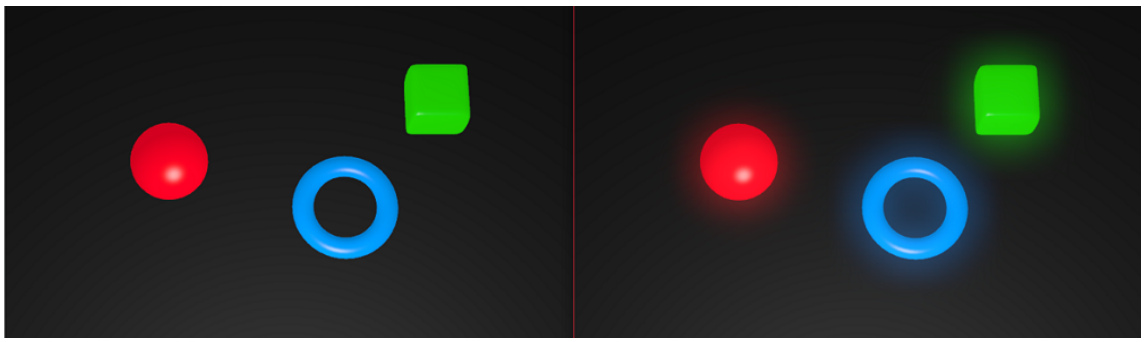

図15.4 ブルーム効果を適用した物体(左)と、適用しない物体(右)の例。

15.10.8.18. 被写界深度

[被写界深度]ウィジェット機能は、後処理効果です。この機能は、特定の距離にある1つの物体にのみ明確にフォーカスを合わせ、カメラレンズのフォーカス特性をシミュレートします。カメラに近い、または遠い物体は、ぼやけて見えます。

制限:

- [被写界深度]ウィジェット機能は、カメラにのみ使用できます。

表15.226 [被写界深度]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`enabled`	trueの場合に被写界深度効果がカメラに適用されます。	
`focusDistance`	ワールド空間におけるカメラ位置からの距離。この値は、カメラの`nearPlane`プロパティから`farPlane`プロパティまでの範囲に収まる必要があります。	入力
`fstop`	フォーカス領域のサイズを定義します。	入力

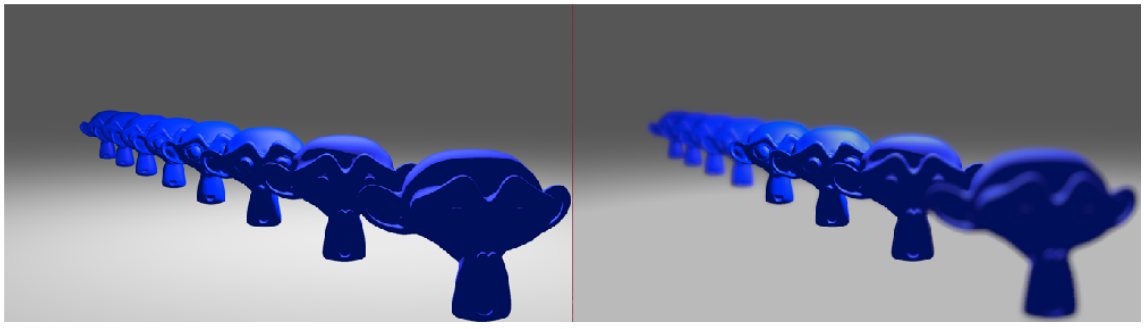

図15.5 被写界深度効果を適用した物体(左)と、適用しない物体(右)の例。

15.10.8.19. スクリーンスペースアンビエントオクルージョン

[スクリーンスペースアンビエントオクルージョン]ウィジェット機能は、後処理効果です。シーンへの環境光の作用を計算し、間接光の近似値を提供します。

制限:

- [スクリーンスペースアンビエントオクルージョン]ウィジェット機能 は、シーングラフにのみ使用できます。

表15.227 [スクリーンスペースアンビエントオクルージョン]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
`radius`	サンプル球体の半径を定義します。ここで`samplesCount`が、POIの周囲に使用されます。この値は0.0～1.0の範囲の正数です。	
`fallOff`	最小距離を定義する内側の球体。オクルージョンを開始する点を表します。この値は0.0～1.0の範囲の正数です。この値は`radius`値より小さい必要があります。	
`samplesCount`	サンプル球体に含まれるサンプルの数。この数が多いほど、オクルージョンの品質が高くなります。8～64個のサンプルを指定できます。	

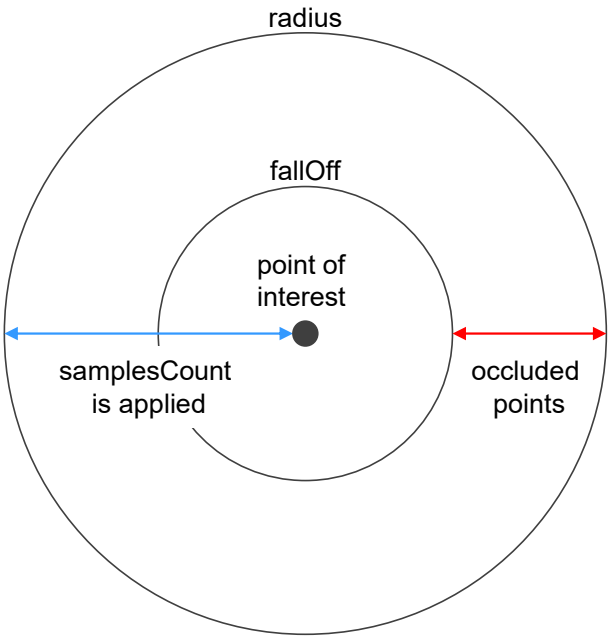

図15.6 サンプル球体と[スクリーンスペースアンビエントオクルージョン]ウィジェット機能のプロパティ

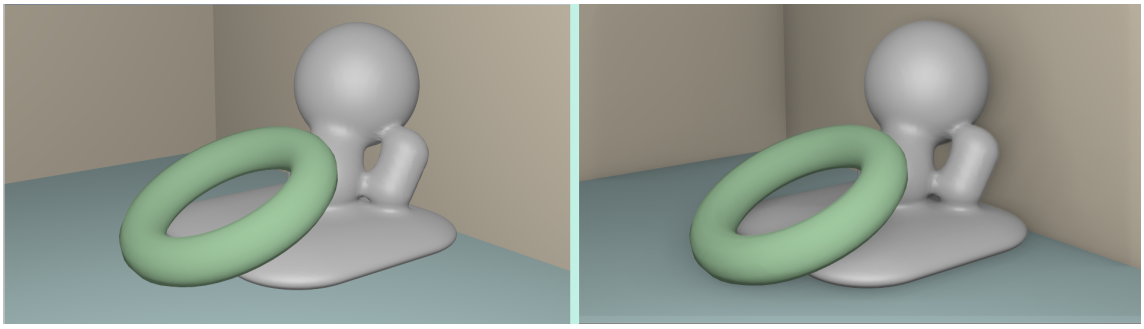

図15.7 [スクリーンスペースアンビエントオクルージョン]効果を適用した物体(左)と、適用しない物体(右)の例

15.10.9. 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

15.10.9.1. ピボット

[ピボット]ウィジェット機能は、ウィジェットに適用される変形のピボット点を定義します。ピボット点が設定されていない場合、デフォルトのピボット点は(0.0, 0.0, 0.0)になります。

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15.228 [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	**EB GUIDE GTF**によって設定
`pivotX`	親ウィジェットを基準にしたx軸のピボット点	

プロパティ名	説明	**EB GUIDE GTF**によって設定
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	

15.10.9.2. 回転

[回転]ウィジェット機能は、ウィジェットおよびサブツリーを回転させるために使用します。

表15.229 [回転]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
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。	

15.10.9.3. 拡大縮小

[拡大縮小]ウィジェット機能は、ウィジェットおよびサブツリーを拡大縮小するために使用します。

表15.230 [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	**EB GUIDE GTF**によって設定
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	

15.10.9.4. せん断

[せん断]ウィジェット機能は、ウィジェットとそのサブツリーを変形するために使用します。

表15.231 [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	**EB GUIDE GTF**によって設定
shearingEnabled	せん断を使用するかどうかを定義します。	
shearingXbyY	y軸によるx軸のせん断	
shearingXbyZ	ウィジェットがシーングラフである場合、z軸によるx軸のせん断	
shearingYbyX	x軸によるy軸のせん断	
shearingYbyZ	ウィジェットがシーングラフである場合、z軸によるy軸のせん断	
shearingZbyX	ウィジェットがシーングラフである場合、x軸によるz軸のせん断	
shearingZbyY	ウィジェットがシーングラフである場合、y軸によるz軸のせん断	

15.10.9.5. 変換

[変換]ウィジェット機能は、ウィジェットおよびサブツリーを変換するために使用します。これにより、ウィジェットはx、y、およびz方向に移動します。

表15.232 [変換]ウィジェット機能のプロパティ

プロパティ名	説明	**EB GUIDE GTF**によって設定
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	

16. EB GUIDE Studioのインストール

16.1. バックグラウンド情報

16.1.1. 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

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。

16.1.2. システム要件

以下の構成を考慮してください。

表16.1 EB GUIDE Studioの推奨構成

ハードウェア	クアッドコアCPU(最低2 GHz)および8 GB RAMを搭載するPC
オペレーティングシステム	Windows 7(64ビット)、Windows 10(64ビット)
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

表16.2 EB GUIDE SDKの推奨構成

ソフトウェア開発キット	Microsoft Visual Studio 2013以降
ファイル統合	CMake

16.2. EB GUIDEのダウンロード

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。

16.3. EB GUIDEのインストール

EB GUIDEのインストール

前提条件:

- セットアップファイル`studio_setup.exe`のダウンロードが完了しています。
- オペレーティングシステムの管理者権限を持っていること。

ステップ 1
セットアップファイル`studio_setup.exe`をダブルクリックします。

ダイアログが開きます。

ステップ 2
[Yes]をクリックします。

[Setup - EB GUIDE Studio]ダイアログが開きます。

ステップ 3
ライセンス使用許諾契約書に同意し、[Next]をクリックします。

ステップ 4
インストール先のフォルダを選択します。

デフォルトのインストールディレクトリは、`C:/Program Files/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。

16.4. EB GUIDEのアンインストール

EB GUIDEのアンインストール

注記

EB GUIDEの永久的な削除
以下の手順に従うと、EB GUIDEがPCから永久的に削除されます。

前提条件:

- EB GUIDEのインストールが完了していること。
- オペレーティングシステムの管理者権限を持っていること。

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

ステップ 2
[Elektrobit]メニューで、アンインストールするバージョンをクリックします。

ステップ 3
サブメニューで、[Uninstall]をクリックします。

用語集

#

3Dグラフィック

3Dグラフィックは、3Dシーンの仮想画像です。3Dシーンは、3Dモデル(メッシュや形状)、材質、光源、カメラをまとめたものです。材質は色やテクスチャ、仮想ライト効果下での動作などで3Dモデルの外観を定義します。カメラは、3Dシーンの仮想画像を撮影する視点となります。

A

ADAS ECU

先進運転支援システム電子制御ユニット

ISO 26262のハードウェア設計とソフトウェアアーキテクチャを基盤とする、オープンな拡張可能プラットフォームです。Autoliv、顧客企業、またはサードパーティのアルゴリズムをホストします。

ADASIS

先進運転支援システムインターフェース仕様

アプリケーションプログラミングインターフェース

アプリケーションプログラミングインターフェース

アプリケーション

EB GUIDEのコンテキストでのアプリケーションとは、例えば、イベントシステムやデータプールなどを介して1つ以上のEB GUIDEモデルとのやり取りをEB GUIDE GTFのランタイムに行うコンピューターソフトウェアです。例えば、メディアプレーヤーのようなエンターテイメントソフトウェアや、電話のような通信ソフトウェアなどは、アプリケーションです。
アプリケーションプログラミングインターフェース参照

外見

EB GUIDEでの外見とは、EB GUIDE GTFのランタイムに適用される、EB GUIDEモデルの外観に関する変更です。EB GUIDEモデルに対してさまざまな外観を定義できるスキンと、言語という2つのタイプの外見が存在します。

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 arware	EB GUIDE arwareは、運転の快適性を向上するためにオーグメンテッドリアリティソリューションを作成できるソフトウェアフレームワークです。
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SDK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モデルとは、ヒューマンマシンインターフェースの外観と動作を定義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 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イスに変換したりするために必要なソフトウェアライブラリおよびツールの集まりです。
EB GUIDEプロジェクト	EB GUIDEプロジェクトは、EB GUIDEモデルと、EB GUIDEモデルを対象デバイスで実行するために必要な設定で構成されます。
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のランタイム環境であり、EB GUIDE GTF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。
拡張機能	EB GUIDEでの拡張機能とは、任意のEB GUIDE製品に対する追加要素です。拡張機能は、特定の機能をEB GUIDE Studio、EB GUIDE GTF、またはEB GUIDE Monitorに追加するライブラリ(.dllまたは.soファイル)の形をとるプラグインです。そのような機能としては、例えば、データエクスポーターや追加のウィジェット機能が考えられます。

G

GL	グラフィックスライブラリ
GPS	全地球測位システム
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
HMIモデル	HMIモデルとは、ヒューマンマシンインターフェー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。HMIソフトウェアツールを使用して作成します。

I

IPC	プロセス間通信
IBL	イメージベースドライティング
IBLGenerator	IBLGeneratorは、環境ライティング情報を処理するツールです。

L

ライブラリ	ライブラリとは、EB GUIDEで使用されるプリコンパイルされたソフトウェア部品、サブルーチン、またはプログラムのコレクションです。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。サポートされるファイルタイプとしては.dllおよび.soの2種類があります。

M

MEF	Managed Extensibility Framework。https://docs.microsoft.com/en-us/dotnet/framework/mef/をご覧ください。

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
モデルインターフェース	対象デバイスにエクスポートされたEB GUIDEモデル間での通信に使われる、定義済みの一連のモデル要素です。
マルチフォントサポート	異なる文字範囲に対する複数のフォントの集約であり、1つのフォントとして機能します。
MVC	モデルビューコントローラ

N

名前空間	EB GUIDE Studioでは、名前空間を使用して、データプールアイテムやイベントのようなモデル要素のグループを作成します。こうしたグループには通常、定義された機能があります。それぞれの名前空間では、異なる名前空間のモデル要素に同じ名前を持たせることができるように、モデル要素のネーミングスコープを作成しています。

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。
RomFS	読み取り専用メモリファイルシステム

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。
VTA	ビュー遷移アニメーション

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。
WPF	Windows Presentation Foundation https://docs.microsoft.com/en-us/dotnet/framework/wpf/をご覧ください。

インデックス

シンボル

A

E

F

G

H

I

L

P

U

V