Interstage

FUJITSU

Interstage Interaction Manager V9.1.1

Ajaxフレームワーク UI部品リファレンス

Windows/Solaris/Linux

B1FW-5361-07Z0(00)
2009年10月

まえがき

本書は、以下の製品のAjaxフレームワークを対象としています。

- Interstage Interaction Manager V9.1.1

本書の目的

本書は、Ajaxフレームワークが提供するUI部品について説明しています。

本書の読者

本書は、以下の読者を対象としています。

- Ajaxフレームワークを利用してWebアプリケーションを開発する人
- Ajaxフレームワークを利用してWebシステムを構築する人

前提知識

本書を読むにあたっては、以下の知識が必要です。

- HTML/XHTML/JavaScript/Java/CSSに関する基本的な知識
- Webアプリケーションに関する基本的な知識

本書の構成

本書の構成は以下のとおりです。

関連マニュアルの表記

本書に記載されているマニュアルの名称は、以下のように省略して表記します。

略称	正式名称
ユーザーズガイド	Ajaxフレームワーク ユーザーズガイド

略称

本書に記載されている製品の名称は、以下のように省略して表記します。

なお、本書では、システム名または製品名に付記される登録表示((TM)または(R))は、省略しています。

略称	正式名称
Internet Explorer	Microsoft(R) Internet Explorer(R) 6、 Microsoft(R) Windows(R) Internet Explorer(R) 7、および Microsoft(R) Windows(R) Internet Explorer(R) 8
Firefox	Mozilla(R) Firefox(R) 2.0、および Mozilla(R) Firefox(R) 3.0

本書のコメント

本書ではブラウザによる固有情報を区別するために、以下のアイコンを使用しています。

ブラウザ	アイコン
Internet Explorer 6	
Internet Explorer 7	
Internet Explorer 8	
Internet Explorer 8 標準	
Internet Explorer 8 互換	
Firefox 2.0	
Firefox 3.0	

Internet Explorer 8の各モードについては、以下の組み合わせを対象としています。

	ブラウザモード	ドキュメントモード
Internet Explorer 8 標準	Internet Explorer 8	Internet Explorer 8 標準
Internet Explorer 8 互換	Internet Explorer 8 互換表示	Internet Explorer 7 標準

商標

輸出許可

当社ドキュメントには、外国為替および外国貿易管理法に基づく特定技術が含まれていることがあります。特定技術が含まれている場合は、当該ドキュメントを輸出または非居住者に提供するとき、同法に基づく許可が必要となります。

発行年月

2009年 10月

改版履歴

2007年 6月 初版
2007年 8月 第2版
2007年10月 第3版
2008年 6月 第4版
2008年 7月 第5版
2009年 1月 第6版
2009年10月 第7版

お願い

本書を無断で他に転載しないようお願いします。
本書は予告なしに変更されることがあります。

著作権

目　次

第1章 UI部品の概要

本章では、UI部品の概要を説明します。

1.1 UI部品とは

UI部品とは、クライアントで使用するAjaxフレームワークの部品群の総称で、画面に表示される画面部品と、画面上に表示されない機能部品および機能付加部品があります。これらの部品を組み合わせて、多彩な操作性をもつ、Webアプリケーションの画面を容易に作成できます。

画面部品

入力フィールド、チェックボックス、テーブルなど、画面に表示される機能を提供する部品です。

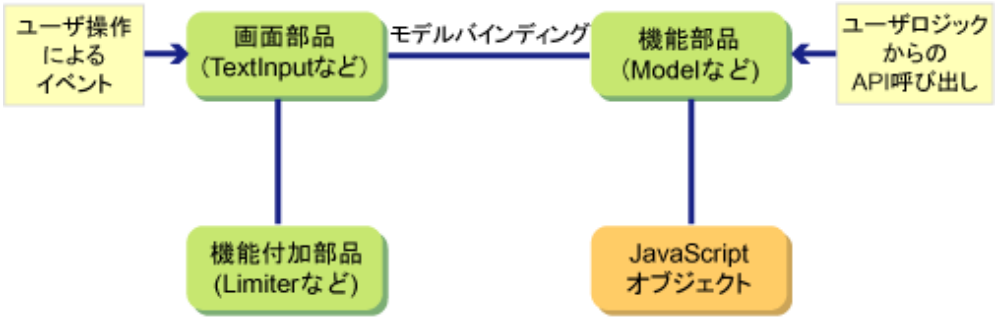
機能部品

データモデルを定義するなど、画面上で利用する機能を提供する部品です。

機能付加部品

入力フィールドへの入力時に文字列を補完するオートコンプリーション機能、フォーカス制御機能など、画面部品に対して機能を付加する部品です。

図1.1 UI部品の相関関係



1.2 UI部品一覧

画面部品

画面部品は、テキスト入力フィールドや、テーブルなど画面に表示される部品群です。HTMLに決められた記述方式で部品を指定すると、Webブラウザでの実行時に動的に指定されたコンポーネントに展開され、その結果が表示されます。
以下に画面部品の一覧を示します。

表1.1 画面部品一覧

名前	概要
Text	テキストを表示する部品です。 HTMLのテキストに相当する部品です。
TextInput	単一行のテキストを入力および編集するための部品です。 HTMLの<input type="text" value="値">および<input type="password" value="値">に相当する部品です。
CheckBox	オンまたはオフの状態を表すチェックボックス部品です。 HTMLの<input type="checkbox" value="値">に相当する部品です。
RadioButton	ラジオボタン部品です。 HTMLの<input type="radio" value="値">に相当する部品です。
Button	ボタン部品です。 HTMLの<button type="タイプ名">に相当する部品です。

名前	概要
	HTMLのbutton要素と同様に、子要素としてHTMLの構造をもつことができ、画像ボタンも使用できます。
TextArea	単一行または複数行のテキストを、入力および編集するための部品です。 HTMLの<textarea>に相当する部品です。
Select	単一選択および複数選択が可能な選択リスト部品です。 HTMLの<select> <option>に相当する部品です。
ComboBox	入力フィールドと選択リストから構成され、項目の選択および選択されている項目の表示を行う部品です。 入力フィールドに対して直接テキストを入力することもできます。
DateInput	TextInputの一種であり、日付および時間データを、入力および編集するための部品です。
NumberInput	TextInputの一種であり、数値を入力および編集するための部品です。
MaskedTextInput 	TextInputの一種であり、フォーマットパターンを設定して、穴埋め方式でテキストを入力および編集するための部品です。 (注) Internet Explorerでだけ利用できます。
MaskedDateInput 	TextInputの一種であり、日時のフォーマットパターンを設定して、入力文字を数字だけに制約し、穴埋め方式でテキストを入力および編集するための部品です。 (注) Internet Explorerでだけ利用できます。
SelectList	単一選択および複数選択が可能な選択リスト部品です。 項目の表示内容をテキスト以外のものにカスタマイズすることができます。
CheckList	チェックボックス付きのリスト部品です。
ViewContainer	汎用コンテナ部品です。 子要素にHTML要素を持つことができ、それらをまとめて部品として扱うことができます。 ViewContainer自体は表示部分を持ちません。
Panel	タイトル部、ボディ部から成り立っているタイトルバー付きコンテナ部品です。 子要素にHTML要素を持つことができ、子要素に定義された内容はボディ部の内容となります。
ViewStack	同じ位置で表示切替えを行う場合に利用する部品です。 複数の画面情報(コンテナ)を持つことができ、同じ位置で切替え表示を行います。
TabPanel	タブにより表示切替えを行う場合に利用する部品です。 複数の画面情報(コンテナ)を持つことができ、タブにより切替え表示を行います。
FragmentContainer	ページ表示後、任意のタイミングで外部から画面情報を読み込み表示するためのコンテナ部品です。
Window	ブラウザ内で移動(ドラッグ)可能な内部ウィンドウ部品です。
TableView	2次元のデータを表形式で表示する部品です。昇順、降順でソートすることができます。
TableEdit	2次元のデータを表形式で表示し、編集することもできる部品です。
DataGrid 	2次元のデータを表形式で表示し、編集することもできる部品です。テーブル内にチェックボックスやツリーなどの表示ができます。 (注) Internet Explorerでだけ利用できます。

名前	概要
ViewColumn	TableViewおよびTableEditのテーブルで列を定義する部品です。
ViewColumnGrid IE 6 IE 7 IE 8	DataGridのテーブルで列を定義する部品です。 (注) Internet Explorerでだけ利用できます。
ViewColumnGroup	ViewColumnおよびViewColumnGridをグループ化する部品です。
ViewColumnCheck IE 6 IE 7 IE 8	DataGridのテーブルで指定した列にチェックボックスを配置する部品です。 (注) Internet Explorerでだけ利用できます。
ViewColumnTree IE 6 IE 7 IE 8	DataGridのテーブルで指定した列にツリー展開用のボタンを配置する部品です。 (注) Internet Explorerでだけ利用できます。
ViewColumnSelect IE 6 IE 7 IE 8	DataGridのテーブルで指定した列にプルダウン(リストボックス)を配置する部品です。 (注) Internet Explorerでだけ利用できます。
ViewColumnImage IE 6 IE 7 IE 8	DataGridのテーブルで指定した列に画像ファイルを配置する部品です。 (注) Internet Explorerでだけ利用できます。
Calendar	カレンダを表示し、日付の選択もできる部品です。
PopupCalendar	ダイアログ表示するカレンダ部品です。
CalendarButton	PopupCalendarを表示するボタン部品です。PopupCalendarと組み合わせて利用します。
TreeView	ツリー形式で項目リストを表示する部品です。
ScrapingView	Webアプリケーションからスクレイピングしたコンテンツを表示する部品です。
ContextMenu	マウスの右クリックによりコンテキストメニューを表示する部品です。

機能部品

機能部品は、画面に表示されない機能を提供し、単体で使用できます。
以下に機能部品の一覧を示します。

表1.2 機能部品一覧

名前	概要
Model	データモデルを定義する部品です。
TreeModel	TreeViewで使用するデータモデルを定義する部品です。
MenuModel	ContextMenuで使用するデータモデルを定義する部品です。

機能付加部品

機能付加部品は、画面部品に機能を付加します。
以下に機能付加部品の一覧を示します。

表1.3 機能付加部品一覧

名前	概要
AutoCompleter	オートコンプリーション機能部品です。 入力フィールドにオートコンプリート機能を付加します。
Limiter IE 6 IE 7 IE 8	入力文字制限機能部品です。 入力フィールドに入力する文字に制限を付加します。 (注) Internet Explorerでだけ利用できます。
NumeralOnlyLimiter IE 6 IE 7 IE 8	Limiterの一種であり、入力文字を数値だけに制限する機能部品です。 (注) Internet Explorerでだけ利用できます。
EnableCharTypeLimiter IE 6 IE 7 IE 8	Limiterの一種であり、指定された有効文字種により検証を行う機能部品です。 (注) Internet Explorerでだけ利用できます。
ValidationHelper	検証を行うトリガとなるイベントを指定し、そのイベント発生時にデータの検証を実行する機能部品です。
FocusManager	フォーカス制御機能部品です。 フォーカス移動の順序、およびフォーカス移動のためのキー操作を定義します。
CheckBoxGroup	チェックボックスグループ定義部品です。 複数のチェックボックスを1つのグループとして登録します。
RadioButtonGroup	ラジオボタングループ定義部品です。 複数のラジオボタンを1つのグループとして登録します。

1.3 UI部品の使い方

UI部品を使用した画面作成は、以下の手順で行います。

1. HTMLファイルへの記述
    1. 基本的な記述方法
       “1.3.1 UI部品の記述方法”を参照してください。
    2. スタイルの設定
       “1.3.2 スタイルの設定”を参照してください。
    3. モデルオブジェクトの定義
       “1.3.3 画面部品とモデルオブジェクトの関係”を参照してください。
    4. JavaScriptから操作
       “1.3.4 JavaScriptからの操作”を参照してください。
2. 入力データの検証
   “ユーザーズガイド”を参照してください。

1.3.1 UI部品の記述方法

Ajaxフレームワークで提供するUI部品は、HTMLの タグおよびタグを使って記述します。

記述形式: タグの場合

```
<div rcf:id="部品ID" rcf:type="部品名" rcf:プロパティ名="値" ... ></div>
```

記述形式: タグの場合

```
<span rcf:id="部品ID" rcf:type="部品名" rcf:プロパティ名="値" ... ></span>
```

属性	説明
rcf:id	部品IDを指定します。(省略可)
rcf:type	部品名を指定します。
rcf:プロパティ名	部品ごとに定義されているプロパティ、スタイルプロパティ、イベントリスナなどを指定します。

例えば、TextInputに、maxLengthプロパティ、colorプロパティ、およびonChangeイベントリスナを指定する場合は、以下のように記述します。

```
<span rcf:id="id1" rcf:type="TextInput" rcf:maxLength="10" rcf:color="blue" rcf:onChange="alert('変更されました')">
</span>
```

画面部品は、 タグおよびタグの両方で記述できる部品と、 タグでだけ記述できる部品があります。 タグ、タグの両方で記述できる部品に関しては、表示上、以下の違いがあります。

- タグ
  部品の前後に改行コードが挿入されます。
- タグ
  部品の前後に改行コードは挿入されません。

各画面部品の詳細は、“第2章 画面部品”を参照してください。

機能部品と機能付加部品は、 タグおよびタグのどちらで記述しても違いはありません。

ポイント

UI部品を記述する際には、以下の事柄に注意してください。

- **終了タグの記述**
  UI部品を記述する場合、子要素を持たない部品を空要素で記述せずに、必ず終了タグを記述してください。
  空要素で記述した場合、それ以降の部品が表示されない場合があります。
  - 正しい例

    ```
    <span rcf:id="id1" rcf:type="TextInput"></span>
    ```

  - 間違った例(空要素記述)

    ```
    <span rcf:id="id2" rcf:type="TextInput"/>
    ```

- **“rcf-”から始まる文字列**
  Ajaxフレームワークでは“rcf-”から始まる文字列を特別な意味で利用しています。そのため、以下の値に“rcf-”から始まる文字列は指定しないでください。
  - HTMLのid属性
  - HTMLのstyle属性
  - rcf:id属性

- **rcf:id属性およびHTMLのid属性**
  rcf:id属性により部品IDを指定する場合、指定する値は以下の条件を満たしている必要があります。

  ー 画面で一意であること
  例えば、画面の遅延読込み(画面の一部分だけを別に用意しておき、ユーザが画面操作をしている間に裏で画面情報を読み込むこと)を行う場合、遅延読込みで読み込んだ画面情報を含めて、部品IDは一意である必要があります。

  ー 画面のHTMLのid属性の値と重複しないこと
  HTML要素にid属性でIDを指定する場合、その画面で指定する部品IDと重複する値は指定できません。重複する値を指定した場合、動作が不定になります。

- **配列要素の記述**
  StringやNumberの配列を値として指定するプロパティの場合、属性値を";"で区切って配列の要素を指定することができます。
  例1: FocusManagerのtargetsプロパティの場合

```
<div rcf:type="FocusManager" rcf:targets="text1; text2; text3; ..."></div>
```

  例2: SelectListのselectedIndexesプロパティの場合

```
<div rcf:type="SelectList" rcf:selectedIndexes="0;2;..." rcf:multiple="true"></div>
```

  配列の要素が空の場合は、属性値に空文字列("")を指定してください。
  そのほかの型(Object型など)の配列を、属性値として直接指定することはできません。配列を持つモデルへのバインディング式を指定するか、部品の初期化後にAPIを使用して値を指定してください。

1.3.2 スタイルの設定

画面部品のスタイルの指定には、以下の3つの方法があります。

- スタイルプロパティでの設定
- CSSでの設定
- JavaScript APIでの設定

スタイルプロパティでの設定

画面部品では、その部品で指定できるスタイルをプロパティ(スタイルプロパティ)として設定できます。
例えば、TextInput部品で背景色、文字色、フォントの大きさを変更したい場合は、以下のように指定します。

```
<span rcf:type="TextInput"
 rcf:backgroundColor="#FFFFFF" rcf:color="#000000" rcf:fontSize="large"></span>
```

その画面部品で指定できないスタイルプロパティが指定された場合は、無視されます。

スタイルプロパティには、部品全体に対するスタイルプロパティと部品の特定のパーツに対するスタイルプロパティがあります。
スタイルプロパティの命名規則については、"2.9 スタイルプロパティ"のポイントを参照してください。
また、各部品のスタイルプロパティについては、各部品の説明を参照してください。

CSSでの設定

各画面部品は、styleClassプロパティによってクラス名が指定できます。クラス名を利用することで、スタイルをCSSによって記述できます。

以下に、SelectList部品での記述例を示します。

```
<style type="text/css">
  .myClass .rcf-SelectList{
    width: 100px;
    height: 200px;
    color: blue;
    border-color: black;
    background-color: lightgrey;
  }
```

```
  .myClass .rcf-SelectList-optionSelected {
    color: red;
  }
</style>

<div rcf:type="SelectList" rcf:styleClass="myClass" rcf:options="foo;bar">
</div>
```

styleClassプロパティによってクラス名を指定すると、その部品のスタイルプロパティで指定できるスタイルをCSSで記述することができます。

CSSで指定する場合のルールは、以下のとおりです。

- スタイルプロパティ名がすべて小文字のもの (例: width、height)
  CSSでは、同じ名前で指定できます。
- スタイルプロパティ名に大文字を含んでいるもの (例: backgroundColor、fontSize)
  CSSでは、大文字の前に“-”を挿入し、大文字を小文字にした名前になります。
  - backgroundColor → background-color
  - fontSize → font-size
- スタイルプロパティでプレフィックス(パーツ名)が前にあるもの (例: titleHeight、bodyPadding)
  特定のクラス名とCSSで指定できます。(.myClassは、styleClassで指定したクラス名)

  Panelの titleHeightの場合

```
.myClass .rcf-Panel-title {
   height: 30px;
}
```

  Panelの bodyBackgroundColorの場合

```
.myClass .rcf-Panel-body {
   background-color: white;
}
```

  特定のクラス名は、部品ごとに異なります。クラス名の命名規則は、以下のようになります。

```
"rcf-部品名-パーツ名"
```

  なお、画面部品のスタイルプロパティで指定できないスタイルをCSSに記述した場合の動作は不定です。

JavaScript APIでの設定

JavaScriptから、rcf:idで指定した値を変数としたオブジェクトとしてUI部品を操作することで、スタイルを設定することができます。
詳細は、“1.3.4 JavaScriptからの操作”を参照してください。

1.3.3 画面部品とモデルオブジェクトの関係

Ajaxフレームワークでは MVCモデルを採用しており、画面部品(V)とモデル(M)を分離しています。モデルを定義する場合、サーバ側に1回の送信で送る必要があるデータを、まとめて1個のモデルとして定義することをお勧めします。

以下に、従来のHTMLで記述した場合と、Ajaxフレームワークで記述した場合の例を示します。

■従来の HTML での記述

```
<FORM>
  名前：<INPUT type="text" name="name1"/><BR/>
  住所：<INPUT type="text" name="address1" /><BR/>
  性別：<INPUT type="radio" name="gender1" value="男性" /> 男性 /
        <INPUT type="radio" name="gender1" value="女性" /> 女性 <BR/>
</FORM>
```

submit

```
name1="xxxx"
address1="xxxxxxxx"
gender1= 男性 or 女性
```

■Ajax フレームワークでの記述

```
名前：<span rcf.type="TextInput" rcf.value="{model1.name1}"></span><br/>
住所：<span rcf.type="TextInput" rcf.value="{model1.address1}"></span><br/>
性別：<span rcf:type="RadioButton" rcf:id="radioMan" rcf:value="男性"
                    rcf:label="男性"></span>  /
      <span rcf:type="RadioButton" rcf:id="radioWoman" rcf:value="女性"
                    rcf:label="女性"></span><br/>

<div rcf:type="Model" rcf:id="model1" rcf:object="object1"></div>
<div rcf:type="RadioButtonGroup" rcf:targets="radioMan; radioWoman;"
                   rcf:selectedValue="{model1.gender1}"></div>
```

データバインディングによる同期

```
var object1={
  name1:"xxxx"
  address1:"xxxxxxxx"
  gender1: 男性 or 女性
}
```

非同期でサーバに送信

従来のHTMLでは、<FORM>タグで囲むことにより、サーバ側に送信するデータの単位を示していました。

Ajaxフレームワークでは、モデルを定義し、各画面部品にバインディング式を記述することによって、JavaScriptのオブジェクトと画面部品の入力データが同期します。このJavaScriptのオブジェクトを、従来のHTMLでの記述と同様にサーバ側に送信するデータ単位にしておくことで、サーバへの送信時におけるデータ取得や加工の手間を削減することができます。

モデルオブジェクトとバインディングされる画面部品のプロパティは、“第2章 画面部品”の各部品のプロパティの説明を参照してください。

1.3.4 JavaScriptからの操作

UI部品は、JavaScriptからrcf:idで指定した値を変数としたオブジェクトとして操作することができます。JavaScriptのオブジェクトとして操作するには、rcf:idで指定した値を用いる方法と、RCF.getComponent関数を用いる方法があります。

rcf:idで指定した値を用いる方法

rcf:idには、以下の文字列を指定してください。

- 最初の文字は半角英文字
- 2文字目以降の文字は、半角英文字、半角数字、アンダーバーのどれか

以下のようにUI部品を記述した場合、JavaScriptからは“textInput1”という変数名でTextInput部品のAPIを実行することができます。

```
<script type="text/javascript">
//<![CDATA[
   function setProperty() {
    // valueプロパティの設定
    textInput1.setProperty("value", "hoge");
    // colorスタイルの設定(スタイルを設定する場合はsetStyleメソッドを使用します。)
    textInput1.setStyle("color", "yellow");
   }
//]]>
</script>

 ...
<span rcf:id="textInput1" rcf:type="TextInput" rcf:maxLength="10"></span>
```

```
<div rcf:type="Button" rcf:onClick="setProperty()">実行</div>
 ...
```

各UI部品で使用できるJavaScript APIについては、各部品の説明を参照してください。

RCF.getComponent関数を利用する方法

以下に、RCF.getComponent関数を利用した例を示します。

```
<script type="text/javascript">
//<![CDATA[
   function func1() {                                    // (1)
      for (var i = 0; i < 10; i++) {
         var textInput = RCF.getComponent("textInput"+i);
         if (textInput.getProperty("value") == '') {
            textInput.setStyle("backgroundColor", "red");
         }
      }
   }
//]]>
</script>
   (省略)
<div rcf:id="textInput0" rcf:type="TextInput"></div>
<div rcf:id="textInput1" rcf:type="TextInput"></div>
   (省略)
<div rcf:id="textInput9" rcf:type="TextInput"></div>

<div rcf:type="Button" rcf:onClick="func1()">実行</div>    // (2)
```

上記の例では、(2)のボタンがクリックされると、(1)のfunc1関数が実行されます。
func1関数では、textInput0～textInput9に対して、入力値が空であった場合、背景色を赤に変更します。

RCF.getComponent関数を利用すると、ID文字列を指定して、部品のオブジェクトを取得することもできます。
RCF.getComponent関数は、部品のID文字列を動的に生成し、そのオブジェクトを取得する場合に利用できます。

1.3.5 UI部品の タグおよびタグで利用できる属性

AjaxフレームワークのUI部品の タグおよびタグには、“rcf:”で始まる属性とstyle属性だけが記述できます。

style属性に記述可能なCSSスタイルプロパティは以下に限定されます。

- position
- top
- left
- z-index

以下の部品では、styleプロパティが利用できません。

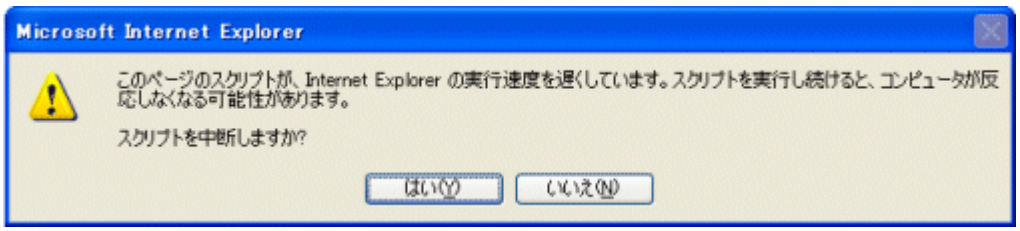
- Window
- PopupCalendar
- ContextMenu

1.3.6 ページあたりの画面部品の個数

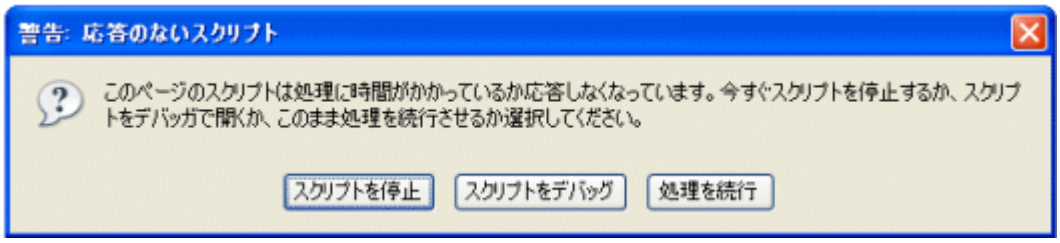
ViewStackやFragmentContainerなどにより最初は表示しない部分も含め、ページあたりの画面部品数は50個以下を推奨します。

部品数が多い場合、初期表示に時間がかかったり、初期表示およびページ切替え時に以下のようなエラーになることがあります。

- IE 6 IE 7 IE 8



- Firefox2 Firefox3



なお、TableView、TableEdit、DataGrid、Calendar、TreeViewなどの複雑な部品を多用した場合や、マシン性能などの要因で、画面部品が50個以下であっても、初期表示に時間がかかったり、エラーになったりする場合があります。

1.3.7 コンテナ部品内のレイアウト

Ajaxフレームワークの動作定義(basePoint_inContainer)により、コンテナ部品内にUI部品およびHTML要素を配置したときの基準位置が変わります。

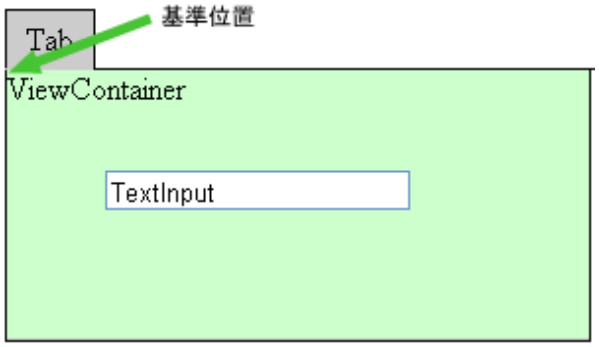
以下に例を示します。

ソース例

```
<div rcf:id="tp1" rcf:type="TabPanel" rcf:width="300px" rcf:height="170px"
     style="left: 10px; top: 10px; position: absolute">
  <div rcf:type="ViewContainer" rcf:label="Tab" rcf:backgroundColor="#DDFFDD" rcf:width="296px" rcf:height="136px">
  ViewContainer
    <div rcf:id="ti1" rcf:type="TextInput" rcf:width="155px" rcf:height="20px"
         style="left: 50px; top: 50px; position: absolute" rcf:value="TextInput"></div>
  </div>
</div>
```

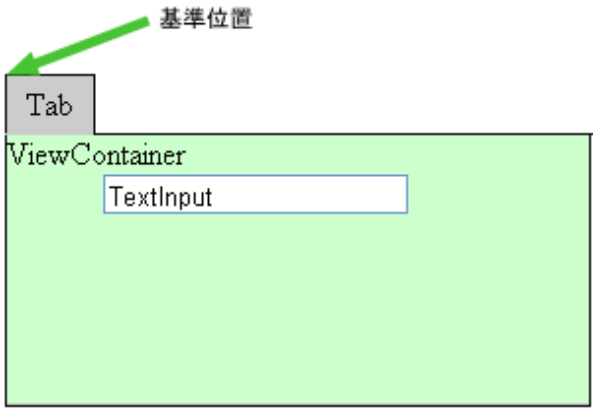
basePoint_inContainerがtrueの場合

基準位置は、以下の図のとおりです。



basePoint_inContainerがfalse(デフォルト)の場合

基準位置は、以下の図のとおりです。



Ajaxフレームワークの動作定義の詳細は、“ユーザーズガイド”を参照してください。

第2章 画面部品

本章では、画面部品について説明します。

2.1 フォーム部品

フォーム部品は、Webアプリケーションを開発する際に主に入力部品として利用するもので、テキストボックスやラジオボタンなどがあります。

ここでは、フォーム部品の設定内容および設定方法について説明します。

2.1.1 Text

Textは、テキストを表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

表示例

テキスト表示

記述形式

```
<div rcf:type="Text" ... ></div>
```

または

```
<span rcf:type="Text" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は、以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味

表の項目の意味は、以下のとおりです。

- 省略:省略できるプロパティかどうかを示します。
- 省略値:省略した場合の値を示します。

- 属性指定:属性値として指定できる内容を示します。
  - ー 値:値を指定可能
  - ー バインド:バインディング式を記述可能
- 更新:APIまたはモデルバインディングにより、外部から更新可能かどうかを示します。
- 部分更新:プロパティ値の一部を更新することが可能かどうかを示します。更新する場合は、データプロバイダを利用します。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
value	String	テキストを指定します。	可	""	値、バインド	可	不可
title	String	ツールチップで表示されるテキストを指定します。	可	""	値	可	不可
labelProvider	getLabel関数を持つオブジェクト	テキストを表示するときに、フォーマットするためのオブジェクトを指定します。詳細は、“フォーマッタの指定方法”を参照してください。 getLabel関数を持たないオブジェクトが指定された場合、エラーになります。 nullを指定した場合は、valueに指定された文字列がそのまま使用されます。	可	null	値	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

フォーマッタの指定方法

labelProviderプロパティでフォーマッタを指定する例を以下に示します。

【処理概要】

1. テキストをフォーマットするオブジェクトに、getLabel関数を用意します。
   getLabel関数は、引数としてテキストを受け取り、戻り値に表示する文字列を返します。
2. labelProviderプロパティに、getLabel関数を持つオブジェクトを指定します。例では“lp1”を指定しています。
3. “テキスト”という文字列がフォーマットされ、画面には“formatted"テキスト"”と表示されます。
   なお、フォーマットされた文字列は、画面に表示する際に特殊文字の変換が行われます。詳細は、“表示するテキストの変換規則”を参照してください。

```
<script type="text/javascript">
//<![CDATA[
  var lp1 = {
    getLabel: function(value) {
      return 'formatted ¥"' + value + '¥"';
    }
 };
//]]>
</script>

  ...
```

```
<div rcf:id="text1" rcf:type="Text" rcf:value="テキスト" rcf:labelProvider="lp1"></div>
```

表示するテキストの変換規則

valueプロパティに指定されたテキストは、labelProviderによってフォーマットされたあとに特殊文字の変換(エスケープ処理)を行い、結果をHTMLとして表示します。変換ルールは以下のとおりです。

元文字列	変換後の文字列
&	&amp;
<	&lt;
>	&gt;
"	&quot;
'	&#39;
改行文字 (¥r¥n および単独の ¥r, ¥n)	
空白(半角空白およびタブ)	

変換例:labelProviderがフォーマットを行った結果、<formatted"テキスト">という文字列を返した場合、内部で以下のように変換されます。

```
[変換前]
      <formatted"テキスト">
       ↑          ↑        ↑ ↑
      (1)        (2)      (2) (3)
```

```
[変換後]
      &lt;formatted&quot;テキスト&quot;&gt;
       ↑              ↑             ↑     ↑
      (1)            (2)           (2)   (3)
```

適用される変換規則は以下のとおりです。

1. < を < に変換
2. " を " に変換
3. > を > に変換

変換後の文字列をHTMLとしてブラウザで表示します。結果、<formatted"テキスト">と正しく表示されます。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-Text	• カラー • フォント • テキスト(textIndent、textAlign、whiteSpaceを除く)(注) • ボーダー • パディング

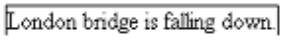
注) Firefox3
Firefox 3.0の場合は、wordSpacingも除きます。

詳細は、“2.9 スタイルプロパティ”を参照してください。

注意

Internet Explorerでボーダーを指定した場合の注意事項 IE 6 IE 7 IE 8 互換

Internet Explorer 6、Internet Explorer 7、またはInternet Explorer 8 互換でTextにボーダーを指定した場合、以下のようにボーダーの一部が欠けて表示される場合があります。



この場合、以下のどちらかの方法で回避できます。

- Text部品の下に空行を追加します。

```
<div rcf:type="Text" rcf:value="文字列"></div>
<br/> 
```

- Text部品を記述した タグまたはタグに、style属性でposition:relativeを指定します。

```
<div rcf:type="Text" style="position:relative" rcf:value="文字列"></div>
```

イベントリスナ

固有のイベントリスナはありません。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.1.2 TextInput

TextInputは、テキスト(単一行)の入力フィールドを表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

ポイント

検証を実行するValidationHelper、入力制限を行うLimiterなどの機能付加部品を利用することができます。

表示例

本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

記述形式

```
<div rcf:type="TextInput" ... ></div>
```

または

```
<span rcf:type="TextInput" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
value	String	テキストを指定します。 “初期化時の処理順序”を参照してください。	可	""	値、バインド	可	不可
title	String	ツールチップで表示されるテキストを指定します。	可	""	値	可	不可
labelProvider	getLabel関数を持つオブジェクト	テキストを表示するときに、フォーマットするためのオブジェクトを指定します。 getLabel関数を持たないオブジェクトが指定された場合、エラーになります。“フォーマッタについて”を参照してください。	可	null	値	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効 ・ false:無効 操作できなくなりイベントも発生しません。	可	true	値、バインド	可	不可
readOnly	Boolean	書込み禁止/許可を指定します。 ・ true:禁止 マウスオーバーなどの各種イベントは発生します。 ・ false:許可	可	false	値、バインド	可	不可
password	Boolean	パスワードを入力する形式にするかどうかを指定します。表示後に変更することはできません。	可	false	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ true:パスワード入力形式 ブラウザにより“*”や“●”で表示されます。 ・ false:パスワード入力形式でない					
maxLength	Number	最大文字数を指定します。2バイト文字も1文字と数えます。本プロパティを指定したあとに、“指定なし”の状態に戻すことはできません。 ・ 1以上の数字:文字数を指定 (0以下は無効) ・ 指定なし:ブラウザに依存 “5.1.2 サロゲートペア”を参照してください。	可	指定なし(-1)	値	可	不可
autoEscape	Boolean	自動脱出機能の有効/無効を指定します。 自動脱出機能とは、maxLengthに指定された文字数が入力された場合に、FocusManagerで設定された次の移動先に、自動的にフォーカスを移動させる機能です。 ・ true:有効 ・ false:無効	可	false	値	可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可
uppercase	Boolean	半角および全角の英文字を大文字にするかどうかを指定します。表示後に変更することはできません。 ・ true:入力文字を大文字に変換する 最初に画面に表示されるとき、入力確定時(フォーカスが外れたとき)、およびsetPropertyなどによりvalueプロパティが更新された場合に変換されます。 ・ false:入力文字を変換しない	可	false	値	不可	不可
imeMode IE 6 IE 7 IE 8	String	IMEのモードを指定します。以下の値以外が指定された場合は、無視されます。 Internet Explorer以外のブラウザでも、本プロパティは無視されます。 ・ auto:自動 ・ active:フォーカス時に日本語入力をONにする	可	auto	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ inactive:フォーカス時に日本語入力をOFFにする ・ disabled:IMEを無効化					

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

ポイント

・ maxLengthプロパティを指定してIMEで入力したときの動作について、以下に示します。

　― IE 6 IE 7 IE 8
　変換前の文字列が存在する場合、maxLengthプロパティで指定した文字数に達した段階で強制的に文字列が確定します。

　― Firefox2 Firefox3
　変換前の文字列が存在する場合、maxLengthプロパティで指定された文字数以上の文字列が入力できます。ただし、変換が確定した時点で、maxLengthプロパティに指定された文字数以上の文字列は無視されます。

・ 自動脱出機能を有効にするには、以下の条件を満たす必要があります。

　― autoEscapeプロパティにtrueが指定されていること

　― maxLengthプロパティに最大文字数を指定してあること

　― 本部品がFocusManagerの対象となっており、FocusManagerに次のフォーカスの移動先が指定されていること

　― imeModeプロパティに“disabled”が指定されていること(Internet Explorerの場合)

初期化時の処理順序

画面の初期化時に、valueプロパティの値が表示されるときの処理順序を以下に示します。

1. 文字列が大文字に変換されます。(uppercaseプロパティがtrueの場合)
2. フォーマッタ処理が実行されます。(labelProviderプロパティが指定されている場合)

フォーマッタについて

本部品へのフォーカスの有無によって、以下のようにフォーマットされます。

― フォーカスがない場合
labelProviderプロパティで指定されたオブジェクトによってフォーマットされたvalueプロパティの値が表示されます。

― フォーカスがある場合
valueプロパティに指定された値が表示されます。

フォーマッタの指定方法は、“フォーマッタの指定方法”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-TextInput	・ サイズ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
			• カラー • フォント(lineHeightを除く) • テキスト(textIndent、whiteSpaceを除く)(注) • ボーダー • パディング

注) Firefox2 Firefox3
Firefoxの場合は、wordSpacingも除きます。

詳細は、"2.9 スタイルプロパティ"を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	
onChange	部品がフォーカスを失い、かつ値が変更されたときに呼ばれます。	
onValueChange	値が入力され、かつ値が変更されたときに呼ばれます。	ValueChangeEvent

部品共通のイベントリスナもあります。部品共通のイベントリスナは、"2.8.2 画面部品共通イベントリスナ"を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、"2.8.3 画面部品共通JavaScript API"を参照してください。

補足事項

TextInputがフォーカスを得たとき、Internet ExplorerとFirefoxでは動作に違いがあります。

IE 6 IE 7 IE 8

- 常に入力テキストが選択状態になります。

Firefox2 Firefox3

- TABキーによるブラウザのデフォルトのフォーカス移動によりフォーカスを得た場合は、入力テキストが選択状態になります。
- マウスクリックによりフォーカスを得た場合は、クリックした位置にカーソルが移動します。
- FocusManagerによるフォーカス移動によりフォーカスを得た場合は、入力テキストは選択状態にはならず、カーソルが表示されます。

2.1.3 CheckBox

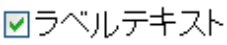
CheckBoxは、チェックボックスを表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

ポイント

CheckBoxGroupと組み合わせて、チェックボックスグループを形成することができます。
グループ内で選択されているチェックボックスの値を一括して取得したり、機能付加部品のFocusManagerに一括して登録したりできます。

表示例



記述形式

```
<div rcf:type="CheckBox" ... ></div>
```

または

```
<span rcf:type="CheckBox" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
value	String	チェックボックスの値を指定します。	可	""	値、バインド	可	不可
checked	Boolean	チェックの有無を指定します。 ・ true:チェックする ・ false:チェックしない	可	false	値、バインド	可	不可
title	String	ツールチップで表示されるテキストを指定します。	可	""	値	可	不可
label	String	チェックボックスの右側に表示されるラベルテキストを指定します。	可	""	値、バインド	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効	可	true	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ false:無効 操作できなくなりイベントも発生しません。					
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに関する注意事項

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

labelプロパティに関する注意事項

ラベルテキストにCheckBoxの幅より長い文字列を指定した場合、デフォルトでは自動改行が行われ、2行目がチェック部の下から始まります。
自動改行の方法は、ブラウザにより異なります。

フォーカスに関する注意事項

部品にフォーカスがある場合、フォーカス枠が表示されますが、Internet ExplorerとFirefoxでは、フォーカス枠が表示される部分が異なります。

- Firefox2 Firefox3 IE 8 標準
  チェック部にフォーカス枠が表示されます。
- IE 6 IE 7 IE 8 互換
  ラベル部にフォーカス枠が表示されます。

なお、Windows Vista上のFirefoxでは、チェックボックスにフォーカス枠が表示されません。
これは、HTMLの<input type="checkbox">でも同様の現象が発生しており、CheckBox部品固有の問題ではありません。

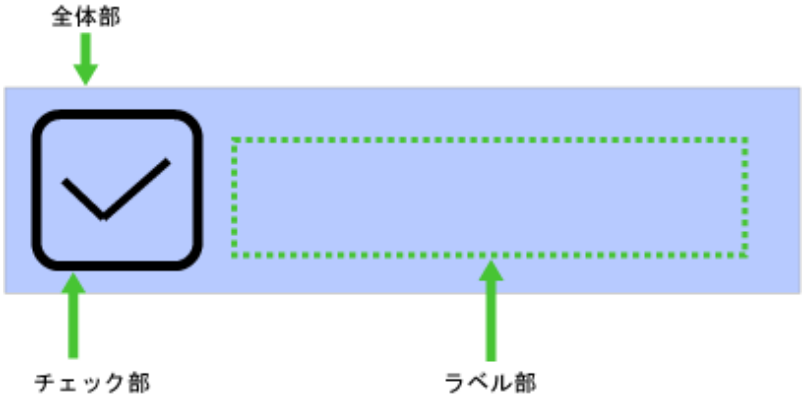
スタイルプロパティ

スタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
全体部	なし	rcf-Checkbox	・ サイズ ・ カラー ・ ボーダー ・ パディング
ラベル部	label	rcf-CheckBox-label	・ カラー ・ フォント

パーツ名	プレフィックス	クラス名	使用可能なスタイル
			・ テキスト(textIndent、textAlign、whiteSpaceを除く)

図2.1 CheckBoxの部品構成



詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	
onCheckStatusChange	チェックの有無が変更されたときに呼ばれます。	CheckStatusChangeEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

ポイント

本部品では、clickイベントを直接定義しても、動作しません。
イベントリスナの定義方法については、“ユーザーズガイド”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.1.4 RadioButton

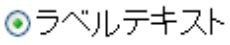
RadioButtonは、ラジオボタンを表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

ポイント

RadioButtonGroupと組み合わせて、相互に排他的なラジオボタングループを形成することができます。
グループ内で選択されているラジオボタンの値を取得したり、機能付加部品のFocusManagerに一括して登録したりできます。

表示例



記述形式

```
<div rcf:type="RadioButton" ... ></div>
```

または

```
<span rcf:type="RadioButton" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
value	String	ラジオボタンの値を指定します。	可	""	値、バインド	可	不可
checked	Boolean	チェックの有無を指定します。 ・ true:チェックする ・ false:チェックしない	可	false	値、バインド	可	不可
title	String	ツールチップで表示されるテキストを指定します。	可	""	値	可	不可
label	String	ラジオボタンの右側に表示されるラベルテキストを指定します。	可	""	値、バインド	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効	可	true	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ false:無効 操作できなくなりイベントも発生しません。					
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに関する注意事項

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

labelプロパティに関する注意事項

ラベルテキストにRadioButtonの幅より長い文字列を指定した場合、デフォルトでは自動改行が行われ、2行目がチェック部の下から始まります。
自動改行の方法は、ブラウザにより異なります。

tabIndexプロパティに関する注意事項 IE 6 IE 7 IE 8

tabIndexにマイナスの数字を指定しても、Tabキーで移動する対象となります。

フォーカスに関する注意事項

部品にフォーカスがある場合、フォーカス枠が表示されますが、Internet ExplorerとFirefoxでは、フォーカス枠が表示される部分が異なります。

- Firefox2 Firefox3 IE 8 標準
  チェック部にフォーカス枠が表示されます。
- IE 6 IE 7 IE 8 互換
  ラベル部にフォーカス枠が表示されます。

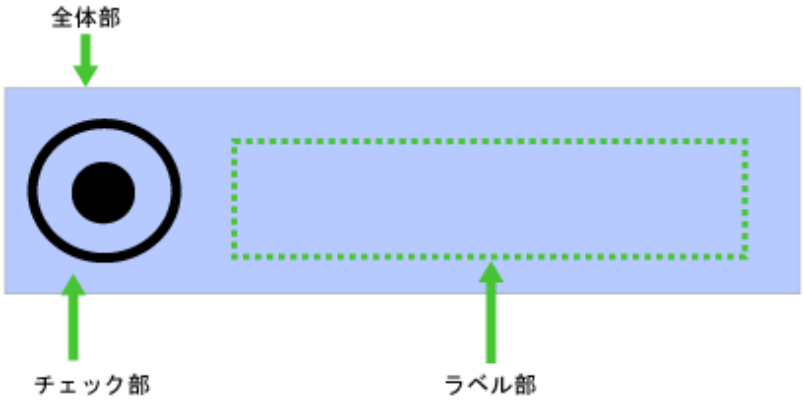
スタイルプロパティ

スタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
全体部	なし	rcf-RadioButton	・ サイズ ・ カラー ・ ボーダー ・ パディング

パーツ名	プレフィックス	クラス名	使用可能なスタイル
ラベル部	label	rcf-RadioButton-label	・ カラー ・ フォント ・ テキスト(textIndent、textAlign、whiteSpaceを除く)

図2.2 RadioButtonの部品構成



詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	
onCheckStatusChange	チェックの有無が変更されたときに呼ばれます。	CheckStatusChangeEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

ポイント

本部品では、clickイベントを直接定義しても、動作しません。
イベントリスナの定義方法については、“ユーザーズガイド”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.1.5 Button

Buttonは、ボタンを表示する部品です。ボタンには、テキストだけでなくイメージを表示することもできます。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

表示例

記述形式

```
<div rcf:type="Button" ... >子要素</div>
```

ポイント

- 本部品は、前後に改行コードは挿入されません。
- 子要素には、ブロック要素またはインライン要素を指定できます。
  ただし、以下のものは子要素に指定できません。
  - <a>タグ
  - <form>タグ
  - <input>タグ
  - <select>タグ
  - <textarea>タグ
  - <label>タグ
  - <button>タグ
  - <isindex>タグ
  - <fieldset>タグ
  - <iframe>タグ
  - UI部品

記述例	表示例
`<div rcf:type="Button"> Click! </div>`	Click!

`<div rcf:type="Button">`
`  <font color="red"> Click! </font>`
`</div>`
	Click!

`<div rcf:type="Button">`
`  <img src="calendar.png" />`
`</div>`
	

`<div rcf:type="Button">` `  <center>` `  <img src="calendar.png" /> ` `  click!` `  </center>` `</div>`	

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
title	String	ツールチップで表示されるテキストを指定します。	可	""	値	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効 ・ false:無効 操作できなくなりイベントも発生しません。	可	true	値、バインド	可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可
label	String	ボタンに表示する文字列を指定します。 このプロパティは、Button部品 タグの子要素として指定されたボタン表示内容を上書きします。setPropertyやバインディングにより、初期値("")以外の文字列に変更されたとき、Button部品 タグの子要素の内容を破棄し、この文字列に差し替えます。	可	""	値、バインド	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

スタイルプロパティ

スタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-Button	・ カラー ・ サイズ ・ テキスト(textIndent、textDecorationを除く) ・ フォント(lineHeightを除く) ・ ボーダー ・ パディング ・ verticalAlign

詳細は、“2.9 スタイルプロパティ”を参照してください。

注意

ボーダーに関する注意事項

ボーダーは、Internet ExplorerとFirefoxで、省略値が異なります。そのため、borderWidth、borderStyle、borderColorの一部だけを指定した場合、ブラウザによって表示が異なる場合があります。

whiteSpaceに関する注意事項 Firefox2 Firefox3

whiteSpaceは、Firefoxでだけ有効です。Internet Explorerでは、指定しても無視されます。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

イベント処理の記述例

ボタンが押されてclickイベントが発生したときの記述例を、以下に示します。

【処理概要】

1. clickイベントに対する処理の実行に、clickButton関数を指定します。
2. clickイベントが発生すると、clickButton関数が実行されます。
3. アラートが表示されます。

```
<script type="text/javascript">
//<![CDATA[
  function clickButton() {
    // ボタンが押されたときの処理
    alert("click button");
  }

  var eventMap = {

    button1: {
      click: clickButton
    }
  };

  RCF.addInitializedListener(function(eventObject) {
    rcf.event.EventRegistrar.registerEvents(eventMap, "eventMap");
  });

//]]>
</script>

  ...

<div rcf:id="button1" rcf:type="Button">
  <span>ボタン1</span>
</div>

  ...
```

注意

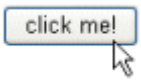
マウスイベントに関する注意事項 IE6 IE7

Internet Explorer 6およびInternet Explorer 7では、ボタンとそのボタンの内容の間に、ボタンをクリックできない部分があります。これは、通常のHTMLの<button>および<input type="button">の場合にもあります。

このクリックできない部分にマウスがある場合、以下の注意事項があります。

- この部分でボタンをクリックしても、click、mouseup、mousedown、dblclickの各イベントは発生しません。
- マウスをゆっくり動かした場合、この部分で、mouseoutおよびmouseoverイベントが発生する場合があります。

Internet Explorer 6の場合の表示例

クリックできる場合	クリックできない部分にマウスがある場合
	

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.1.6 TextArea

TextAreaは、テキスト(単一行または複数行)の入力フィールドを表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

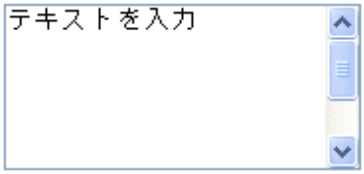
注意

- **文字列の折り返しについて** Firefox2 Firefox3
  Firefoxでは、長い文字列を入力した場合、連続した英数字の途中では折り返されません。表示される文字がフィールドの横幅を超えた場合、横スクロールバーが表示されます。
  Tabキーやマウスクリックによりフォーカスを移動した直後のカーソル位置および選択状態は、ブラウザおよびlabelProviderの指定により異なります。
- **改行コードについて**
  TextAreaに入力した文字列に改行を含む場合、valueプロパティの文字列値では“¥n”(LF、文字コード10)として表現されます。APIやモデルとのバインディングで設定された文字列に“¥r”がある場合、“¥r”は取り除かれます。これにより、改行コードが“¥r¥n”(CR+LF、文字コード13,10)の場合は、“¥n”に変換されます。

ポイント

機能付加部品のValidationHelperを利用して、検証を実行することができます。

表示例



本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

記述形式

```
<div rcf:type="TextArea" ... ></div>
```

または

```
<span rcf:type="TextArea" ... ></span>
```

注意

- 子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- rcf:valueに改行を含む文字列を直接記述した場合、ブラウザによっては先頭と末尾の改行が取り除かれることがあります。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
value	String	テキストを指定します。	可	""	値、バインド	可	不可
title	String	ツールチップで表示されるテキストを指定します。	可	""	値	可	不可
labelProvider	getLabel関数を持つオブジェクト	テキストを表示するときにフォーマットするためのオブジェクトを指定します。詳細は、“フォーマッタについて”を参照してください。 getLabel関数を持たないオブジェクトが指定された場合、エラーになります。nullを指定した場合は、valueに指定された文字列がそのまま使用されます。	可	null	値	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効 ・ false:無効 操作できなくなりイベントも発生しません。	可	true	値、バインド	可	不可
readOnly	Boolean	書込み禁止/許可を指定します。 ・ true:禁止 マウスオーバーなどの各種イベントは発生します。 ・ false:許可	可	false	値、バインド	可	不可
rows	Number	行数を指定します。1以上の値を指定できます。 スタイルプロパティのheightに部品の高さが指定された場合は、heightの値が優先されます。	可	2	値	可	不可
cols	Number	テキストエリアの平均的文字幅による文字数を指定します。1以上の値を指定できます。 スタイルプロパティのwidthに部品の幅が指定された場合は、widthの値が優先されます。	可	20	値	可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可
imeMode  	String	IMEのモードを指定します。以下の値以外が指定された場合は、無視されます。Internet Explorer以外のブラウザでは本プロパティは無効となります。 ・ auto:自動 ・ active:フォーカス時に日本語入力をONにする ・ inactive:フォーカス時に日本語入力をOFFにする ・ disabled:IMEを無効化	可	auto	値、バインド	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

titleプロパティについて

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

colsプロパティに関する注意事項

colsにおける平均的文字幅の値は、ブラウザにより若干異なります。デフォルトでは、FirefoxがInternet Explorerよりわずかに長くなっています。

rowsプロパティに関する注意事項 Firefox2 Firefox3

Firefoxでは、rowsに指定した値よりも、約1行分高く表示されます。

フォーマッタについて

本部品へのフォーカスの有無によって、以下のようにフォーマットされます。

- フォーカスがない場合
  labelProviderプロパティで指定されたオブジェクトによってフォーマットされたvalueプロパティの値が表示されます。
- フォーカスがある場合
  valueプロパティに指定された値が表示されます。

フォーマッタの指定方法は、“フォーマッタの指定方法”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-TextArea	• サイズ • カラー • フォント • テキスト(whiteSpace、textIndentを除く) • ボーダー • パディング

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	
onChange	部品がフォーカスを失い、かつ値が変更されたときに呼ばれます。	
onValueChange	値が入力され、かつ値が変更されたときに呼ばれます。	ValueChangeEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.1.7 Select

Selectは、単一選択および複数選択が可能なリストボックスを表示する部品です。

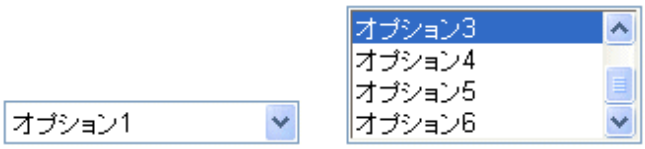
- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

注意

Internet Explorer 6での注意事項 IE6

- Internet Explorer 6では、Window部品の中では利用できません。
- Internet Explorer 6では、z-indexの指定は無効です。例えば、 タグでz-indexを定義して、Selectの上に重ねて表示しようとしても、Selectが一番上に表示されてしまいます。z-indexは、要素を重ねて表示させる場合の順序を指定するものです。

表示例



記述形式

```
<div rcf:type="Select" ... ></div>
```

または

```
<span rcf:type="Select" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
options	Array	選択項目を表す値を指定します。 配列のメンバには、StringまたはObjectを指定できます。 ・ Stringの場合 ラベル(label)と値(value)が一意になります。 ・ Objectの場合 ラベルと値を別々に指定できます。詳細は、“optionsプロパティのObjectの配列指定”を参照してください。 ・ 要素がnullまたは空文字列の場合 エラーとなります。	可	[]	値、バインド(注5)	可	可(注3)
multiple	Boolean	複数選択または単一選択を指定します。 ・ true:複数選択 ・ false:単一選択	可	false	値	不可	不可
selectedIndex	Number	単一選択の場合 ・ 選択されている項目のインデックス(先頭は0)を指定します。 ・ 選択状態の初期値を指定できます。 ・ -1が設定された場合には、選択が解除されます。 ・ 無効な値を指定した場合、エラー(RCF12600)となります。 複数選択の場合 ・ 最後に選択された項目のインデックスを指定します。 ・ 部品の初期化後に本プロパティを変更しようとするとエラー(RCF11002)となります。“multipleと選択状態を表すプロパティの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは-1(選択なし)に設定されます。	可	-1	値、バインド	可(注1)	不可
selectedIndexes	Numberの Array	選択されている項目のインデックス(先頭は0)の配列を指定します。選択順にインデックスが格納されます。 このプロパティにより、選択状態の初期値を指定できます。 何も指定しない場合は、長さ0の配列になります。 無効な値を指定した場合は、エラー(RCF12604)となります。 単一選択(multiple=false)の場合、部品の初期化後に本プロパティを変更しようとすると、エラー(RCF11002)となります。“multipleと選択状態を表すプロパティ	可	[]	値、バインド	可(注2)	可(注2)(注4)

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		の関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは長さ0の配列(選択なし)に設定されます。					
selectedValue	String	単一選択の場合 ・ 選択されている項目の値を指定します。 ・ このプロパティにより選択状態の初期値を指定できます。 ・ 空文字列が指定された場合、選択が解除されます。 ・ 無効な値を指定した場合、エラー(RCF12600)となります。 複数選択の場合 ・ 最後に選択された項目の値を指定します。 ・ 部品の初期化後に本プロパティを変更しようとすると、エラー(RCF11002)となります。“multipleと選択状態を表すプロパティの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティには空文字列(選択なし)が設定されます。	可	""	値、バインド	可 (注1)	不可
selectedValues	Stringの Array	選択されている項目の値の配列を指定します。選択順にインデックスが格納されます。 このプロパティにより選択状態の初期値を指定できます。 何も指定しない場合は、長さ0の配列になります。 無効な値を指定した場合は、エラー(RCF12605)となります。 単一選択の場合、部品の初期化後に本プロパティを変更しようとすると、エラー(RCF11002)となります。“multipleと選択状態を表すプロパティの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは長さ0の配列(選択なし)が設定されます。	可	[]	値、バインド	可 (注2)	可 (注2) (注4)
size	Number	選択リストの高さ(行数)を指定します。 1以上の整数を指定します。1未満の値を指定した場合には、エラーとなります。	可	(自動設定) 単一選択の場合:1 複数選択	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
				の場合:4			
title 	String	optionsプロパティがStringの配列またはObjectの配列で、かつ各Objectがtitle属性でない場合のツールチップで表示されるテキストを指定します。 なお、optionsプロパティがObjectの配列で、かつObjectがtitle属性の場合は、その項目のtitle属性に指定された値がツールチップとして表示されます。 Internet Explorer 6では無効です。	可	""	値、バインド	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効 ・ false:無効 操作できなくなり、イベントも発生しません。	可	true	値、バインド	可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可

注1) 単一選択(multiple=false)の場合にだけ、更新できます。
注2) 複数選択(multiple=true)の場合にだけ、更新できます。
注3) 追加(add)、挿入(insert)、削除(remove)、変更(replace)に対応します。
注4) 追加(add)、削除(remove)に対応します。
注5) Objectの配列の場合は、バインディング式だけ記述できます。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

 注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

optionsプロパティのObjectの配列指定

optionsプロパティにObjectの配列で指定する形式について、以下に示します。

```
{
  label: "項目のラベル",
  value: "項目の値",
  enabled: true|false,
  title: "ツールチップ文字列"
}
```

プロパティの詳細を以下に示します。

プロパティ名	データ型	説明	省略	省略値
label	String	項目のラベルを指定します。	可	valueの値
value	String	項目の値を指定します。 省略時、およびnull、空文字列は、エラーとなります。(RCF12602) options配列の中で重複したvalue値を検出した場合、エラーとなります。(RCF12603)	不可	ー
enabled Firefox2 Firefox3 IE 8 標準	Boolean	項目の選択の許可/禁止を指定します。 ・ true:選択許可 ・ false:選択禁止 FirefoxおよびInternet Explorer 8 標準でだけ有効です。	可	true
title Firefox2 Firefox3 IE 7 IE 8	String	補足情報です。 指定された文字列がツールチップの形で表示されます。 Internet Explorer 6では無効です。	可	ツールチップ非表示

multipleと選択状態を表すプロパティの関係について

selectedIndex、selectedIndexes、selectedValue、およびselectedValuesの4つのプロパティは、multipleの値によって、以下のように扱いが異なります。

通常、以下のプロパティを使用して、値の取得および設定を行ってください。

ー multiple=false(単一選択)の場合

- selectedIndex
- selectedValue

ー multiple=true(複数選択)の場合

- selectedIndexes
- selectedValues

ー **multiple=false(単一選択)の場合**

selectedIndexの値を設定した場合、そのほかのプロパティは以下のように値が変更されます。

selectedIndexの値	変更後の値		
	selectedIndexes	selectedValue	selectedValues
-1	[](長さ0の配列)	""(空文字列)	[](長さ0の配列)
-1以外	selectedIndexの値を持った長さ1の配列	指定されたインデックスの選択項目のvalue	指定されたインデックスの選択項目のvalueを持った長さ1の配列

selectedValueの値を設定した場合、そのほかのプロパティは以下のように値が変更されます。

selectedValueの値	変更後の値		
	selectedIndex	selectedIndexes	selectedValues
""(空文字列)	-1	[](長さ0の配列)	[](長さ0の配列)

selectedValueの値	変更後の値		
	selectedIndex	selectedIndexes	selectedValues
空ではない文字列	指定されたvalueを持つ選択項目のインデックス	指定されたvalueを持つ選択項目のインデックスを持った長さ1の配列	selectedValueの値を持った長さ1の配列

タグにselectedIndexやselectedValueの初期値が指定されていた場合も、上記のルールに従って動作します。ただし、selectedIndexとselectedValueが共に指定されていた場合は、selectedIndexが優先されます。
タグに指定されたselectedIndexesおよびselectedValuesの設定は、無視されます。
selectedIndexesおよびselectedValuesをsetProperty()やモデルバインディングによって変更しようとした場合、エラー(RCF11002)となります。

- **multiple=true(複数選択)の場合**

selectedIndexesの値を設定した場合、そのほかのプロパティは以下のように値が変更されます。

selectedIndexesの値	変更後の値		
	selectedIndex	selectedValue	selectedValues
[](長さ0の配列)	-1	""(空文字列)	[](長さ0の配列)
長さ1以上の配列	selectedIndexesの末尾の要素	selectedIndexesの末尾の要素で指定された選択項目のvalue	selectedIndexesで指定された選択項目のvalueを持った配列

selectedValuesの値を設定した場合、そのほかのプロパティは以下のように値が変更されます。

selectedValuesの値	変更後の値		
	selectedIndex	selectedIndexes	selectedValue
[](長さ0の配列)	-1	[](長さ0の配列)	""(空文字列)
長さ1以上の配列	selectedValuesの末尾のvalueを持つ選択項目のインデックス	selectedIndexesで指定されたvalueを持つ選択項目のインデックスの配列	selectedValuesの末尾の要素

タグにselectedIndexesやselectedValuesの初期値が指定されていた場合も、上記のルールに従って動作します。ただし、selectedIndexesとselectedValuesが共に指定されていた場合は、selectedIndexesが優先されます。
タグに指定されたselectedIndexおよびselectedValueの設定は、無視されます。
selectedIndexおよびselectedValueをsetProperty()やモデルバインディングによって変更しようとした場合、エラー(RCF11002)となります。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-Select	・ width ・ カラー ・ フォント(lineHeightを除く) ・ ボーダー(注)

注) Internet Explorer 6、Internet Explorer 7、およびInternet Explorer 8 互換では無効です。

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	
onChange (注)	現在と異なる項目を選択し、マウスボタンを離したときに呼ばれます。	
onOptionSelected (注)	項目が選択されたときに呼ばれます。	OptionStateChangeEvent
onOptionDeselected (注)	項目の選択が解除されたときに呼ばれます。 単一選択の場合、optiondeselected、optionselectedの順番でイベントが発生します。	

注) マウス、キーボードなどの入力装置によってUI部品を操作した場合に呼ばれるイベントリスナ

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

- **Select部品でのイベントについて**
  単一選択で選択候補のリストが表示されている場合、マウスイベントが正常に通知されません。

  IE 6

  ー dblclickイベントが通知されません。

  ー 部品の外部にマウスを移動させても、mouseoutイベントが通知されません。
  リスト部にマウスを移動させた場合に一度だけmouseoutイベントが通知され、以降、mousemove、mouseover、mouseoutイベントは通知されなくなります。

  ー 部品の外部でマウスボタンを押下した場合にも、mousedown、mouseupイベントが通知されます。

  Firefox2 Firefox3

  ー dblclickイベントが通知されません。

  ー リスト表示ボタン上でのmousedownイベントが通知されません。

- **複数選択操作について**
  キーボード、マウスなどによる複数選択の操作方法は、各ブラウザの<select>要素の操作方法に準じます。
  標準設定では、以下の方法で複数選択が可能です。

  IE 6 IE 7 IE 8
  選択の追加、解除

  ー Ctrlキーを押しながら項目をクリック

  範囲選択

  ー Shiftキーを押しながら方向キーを押下

  ー マウスのドラッグ

  Firefox2 Firefox3
  選択の追加、解除

  ー Ctrlキーを押しながら方向キーでフォーカスを移動し、スペースキーを押下

- Ctrlキーを押しながら項目をクリック

範囲選択

- Shiftキーを押しながら方向キーを押下
- マウスのドラッグ

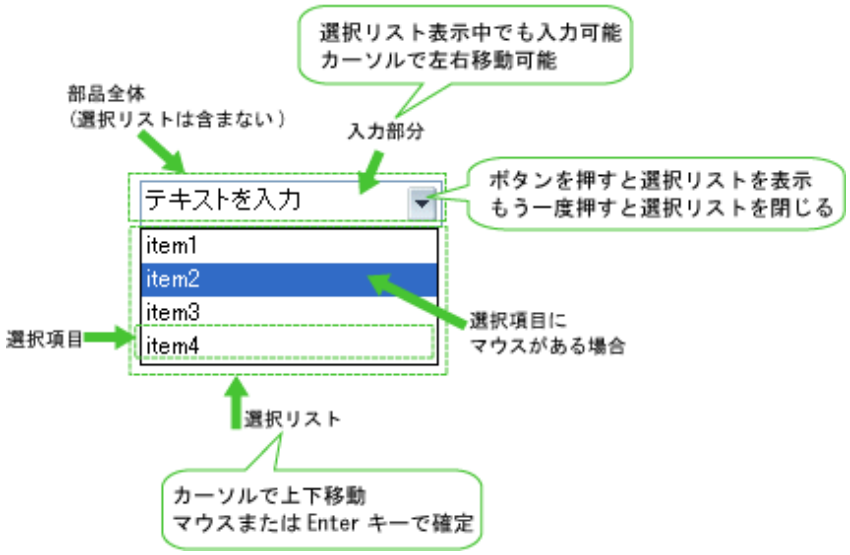
2.1.8 ComboBox

ComboBoxは、コンボボックスを表示する部品です。コンボボックスは、入力フィールドと選択リストからなる部品です。

ポイント

検証を実行するValidationHelper、入力制限を行うLimiterなどの機能付加部品を利用することができます。

表示例



本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

注意

ComboBoxを画面の最下部に配置した場合や、選択リストの高さを設定しないで多数の選択項目を設定した場合など、選択リストを開いたときに画面に収まらないことがあります。
このような場合、ブラウザによってはキー操作でしか選択できなくなるなど操作性が悪くなってしまうため、選択リストが画面内に収まるように部品を配置してください。

記述形式

```
<div rcf:type="ComboBox" ... ></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は、前後に改行コードが挿入されて表示されます。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
list	Array	選択リストに表示する項目の値を指定します。 配列の内容はStringです。	可	[]	値、バインド	可	可
buttonImage	Object	選択リストの表示/非表示を操作するボタンのイメージを指定します。詳細は、“buttonImageプロパティの指定形式”を参照してください。	可	標準のイメージ(注1)	値	不可	不可
fixButtonImageSize	Boolean	選択リストの表示/非表示を操作するボタンのイメージのサイズを固定にするかどうかを指定します。 ・ true: 部品全体のサイズにかかわらず、ボタンのイメージのサイズは固定のままです。 ・ false: 部品全体のサイズに従って、ボタンのイメージのサイズが拡大・縮小されます。 詳細は、“fixButtonImageSizeプロパティの指定時の表示例”を参照してください。	可	false	値	不可	不可
buttonImageSize	Object	選択リストの表示/非表示を操作するボタンのイメージの表示サイズ(高さ、幅)を指定します。単位はピクセルです。(注3) ボタンのイメージが指定サイズと異なる場合は指定サイズに拡大・縮小されます。(注4) オブジェクトの指定方法は“buttonImageSizeプロパティの指定方法”を参照してください。 fixButtonImageSizeがfalseの場合、本プロパティは無視されます。	可	ボタンのイメージのサイズ(注2)	値	不可	不可

注1) 標準のイメージのサイズは、幅15ピクセル、高さ18ピクセルです。

注2) 標準のイメージを使用する場合、ボタンのイメージのサイズは、幅15ピクセル、高さ18ピクセルになります。buttonImageプロパティでボタンのイメージを指定した場合は、指定したボタンのイメージのサイズになります。

注3) 部品全体のサイズ(標準サイズは幅150ピクセル、高さ20ピクセル)よりボタンのイメージのサイズが大きい場合、表示が崩れる場合があります。その場合は、ボタンのイメージのサイズが部品全体のサイズより小さくなるように調整してください。

注4) buttonImageSizeプロパティを指定した場合、選択リストの表示/非表示を操作するボタンのイメージのサイズは、強制的に指定したサイズに拡大・縮小されるため、ボタンのイメージの見映えが異なる場合があります。

TextInputのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.1.2 TextInput”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

ポイント

- maxLengthプロパティを指定している場合
  選択リストの文字列の最大文字数は、maxLengthプロパティの値が上限となります。
- autoEscapeを有効に設定している場合
  選択項目が確定したときに、TextInputと同様の自動脱出の条件が満たされている場合、自動脱出が行われます。
- uppercaseプロパティを有効に設定している場合
  選択リストの文字列が英文字の場合、大文字に変換されて表示されます。

buttonImageプロパティの指定形式

buttonImageプロパティを指定する形式について、以下に示します。

```
{
  base: "URL",
  mouseOver: "URL",
  mouseDown: "URL"
}
```

プロパティの詳細を以下に示します。

プロパティ名	データ型	説明	省略	省略値
base	String	通常表示されるイメージファイルを指定します。	不可	－
mouseOver	String	マウスカーソルが上にあるとき表示されるイメージファイルを指定します。	可	baseの値
mouseDown	String	マウスがクリックされたときに表示される画像ファイルを指定します。	可	baseの値

それぞれのURLには、クエリ文字列およびURLリライティングで用いるセションIDを付加することができます。詳細は、“ユーザーズガイド”を参照してください。

fixButtonImageSizeプロパティの指定時の表示例

fixButtonImageSizeの値によって、ボタンの表示サイズが異なります。

ー fixButtonImageSizeの値がfalseのとき
スタイルプロパティのwidthプロパティとheightプロパティを変更すると、ボタンのイメージのサイズは以下のように変更されます。(単位はピクセル)

- 幅: width/10 (小数点以下切捨て)
- 高さ:height-2

ー fixButtonImageSizeの値がtrueのとき
スタイルプロパティのwidthプロパティとheightプロパティを変更しても、ボタンのイメージのサイズは固定されたままになります。

buttonImageSizeプロパティの指定方法

buttonImageSizeプロパティを指定する形式について、以下に示します。

heightおよびwidthは、それぞれ数値で指定してください。(単位はピクセル)

```
{
    height:  25,    //画像の表示サイズ(高さ)
    width:   20     //画像の表示サイズ(幅)
}
```

heightまたはwidthだけの指定も可能です。省略した場合は、ボタンのイメージのサイズで表示されます。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-ComboBox	・サイズ ・ボーダー
入力部分	input	rcf-ComboBox-input	・カラー ・フォント(注1) ・テキスト(textIndent、textAlign、whiteSpaceを除く)(注2)
選択リスト	list	rcf-ComboBox-list	・カラー ・フォント ・テキスト(textIndent、textAlign、whiteSpaceを除く) ・ボーダー ・サイズのheight
選択項目(マウスカーソルがない場合)	item	rcf-ComboBox-item	・ボーダー
選択項目(マウスカーソルがある場合)	itemHovered	rcf-ComboBox-itemHovered	・カラー

注1) Firefox2 Firefox3
Firefoxの場合は、lineHeightも除きます。

注2) Firefox2 Firefox3
Firefoxの場合は、wordSpacingも除きます。

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onClickButton	ボタン上でマウスがクリックされたときに呼ばれます。	ActionEvent
onDblClickButton	ボタン上でマウスがダブルクリックされたときに呼ばれます。	
onMouseOverButton	マウスがボタン上に重ねられたときに呼ばれます。	
onMouseOutButton	マウスがボタン上から離されたときに呼ばれます。	
onMouseDownButton	ボタン上でマウスが押し下げられたときに呼ばれます。	
onMouseUpButton	ボタン上でマウスが離されたときに呼ばれます。	
onMouseMoveButton	マウスがボタン上で動かされたときに呼ばれます。	
onMouseOverList	マウスが選択リスト上に重ねられたときに呼ばれます。	
onMouseOutList	マウスが選択リスト上から離されたときに呼ばれます。	
onMouseDownList	選択リスト上でマウスが押し下げられたときに呼ばれます。	
onMouseMoveList	マウスが選択リスト上で動かされたときに呼ばれます。	

本部品の入力フィールドのイベントとして、TextInputと同じイベントリスナを使用します。詳細は、“2.1.2 TextInput”の“イベントリスナ”を参照してください。
部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

ポイント

- **onValueChangeについて**
  valuechangeイベントは、選択リストから値を選択したときに、これまで入力されていた値と異なる場合に発生します。入力されている値と同じ値を選択した場合、valuechangeイベントは発生しません。
- **選択リストおよびボタンをクリックした場合のイベントとフォーカスについて**
  マウスで選択リスト(Internet Explorerの場合)またはボタン(Internet ExplorerおよびFirefoxの場合)をクリックした場合、いったん、入力フィールドからフォーカスが外れ、再度、自動的に入力フィールドにフォーカスが戻されます。このとき、onBlurおよびonFocusのイベントが発生し、値が変更されている場合にはchangeイベントもあわせて発生します。
  また、上記のほかにpropertychange、valuechange、mousedownbutton、mousedownlistも、同じタイミングで発生する場合があります。これらのイベントに対するイベントリスナの中では、フォーカス移動処理を行わないでください。前述の入力フィールドにフォーカスを戻す処理との競合により、意図したフォーカス移動が行われない場合があります。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

- **ComboBoxのサイズについて**
  部品全体および選択リストの高さ(height)と幅(width)の単位は、“px”にだけ対応しています。そのほかの単位による指定をした場合、表示が崩れる可能性があります。
- **選択リストからの値の選択**
  マウスでボタンをクリックして選択リストを開いた場合、そのままマウスをドラッグして選択リスト上で離しても選択操作は行われません。選択リストから値を選択するには、選択したい値の上で、再度マウスをクリックしてください。

2.1.9 DateInput

DateInputは、日付を入力および編集する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

ポイント

検証を実行するValidationHelper、入力制限を行うLimiterなどの機能付加部品を利用することができます。

表示例

本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

記述形式

```
<div rcf:type="DateInput" ... ></div>
```

または

```
<span rcf:type="DateInput" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
date	Date	日時データを指定します。APIやバインディングによりdateプロパティが更新された場合は、入力値(valueプロパティの値)も更新されます。	可	null	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
utc	Boolean	世界標準時間(UTC)として扱うかどうかを指定します。 ・ true:世界標準時間(UTC) ・ false:ローカル時間 converterプロパティが設定された場合、この設定は無効となります。 また、表示中に本プロパティを変更することはできません。 “5.1.9 Dateオブジェクトとタイムゾーン”を参照してください。	可	false	値	不可	不可
converter	format関数とparse関数を持つObject	以下の関数を持つオブジェクトを指定します。 ・ JavaScriptのDateオブジェクトを文字列に変換(format関数) ・ 入力テキストをJavaScriptのDateオブジェクトに変換(parse関数) format関数またはparse関数を持たないオブジェクトを指定した場合は、エラーになります。詳細は、“converterプロパティの指定方法”を参照してください。 デフォルトの表示と変換は、以下のとおりです。 表示:“yyyy/MM/dd”形式 変換:Dateオブジェクト 日時は、デフォルトでは、ローカル時間として扱います。utcプロパティがtrueに設定された場合は、世界標準時間(UTC)として扱います。 Dateオブジェクトがnullの場合は、空文字列に変換します。また、入力データが空文字列の場合は、nullに変換します。	可	デフォルトの変換を実行するObject	値	不可	不可
refocus IE 6 IE 7 IE 8	Boolean	入力フィールドからフォーカスが外れて、Dateオブジェクトへの変換に失敗した場合、フォーカスを再度DateInputに戻すかどうかを指定します。 ・ true:フォーカスを戻す ・ false:フォーカスを戻さない Internet Explorerでだけ指定可能です。 Firefoxの場合は、指定しても無視されます。	可	false	値	可	不可

TextInputのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.1.2 TextInput”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

- titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
  また、発生位置はブラウザにより異なります。

  なお、空文字列を指定した場合は、表示されません。
- 入力テキストがJavaScriptのDateオブジェクトに変換されるのは、入力フィールドからフォーカスが外れ、前の値から変更された場合だけです。入力中にdateプロパティの値を参照すると、前回の入力テキストが変換された結果になることがあります。

valueプロパティとdateプロパティの違い

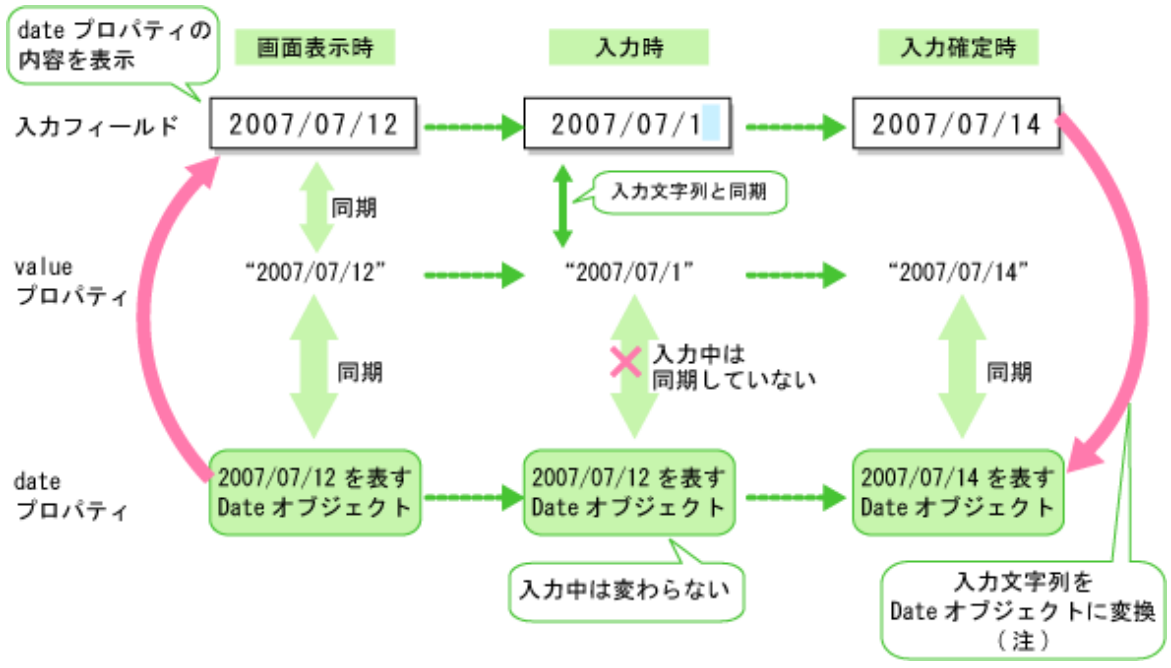
valueプロパティとdateプロパティは、どちらもテキストを入力できるプロパティです。両者の違いについて、以下に示します。

ー 画面表示時
入力フィールドの初期値は、dateプロパティで指定します。dateプロパティが指定されていない、またはnullの場合、入力フィールドの初期値は空になります。
valueプロパティは、dateプロパティを文字列にした値に初期化されます。

ー 入力時
入力フィールドに値を入力しているとき、valueプロパティはその値と同期します。dateプロパティは、入力中のデータとは同期せずに、そのままの値が保持されます。

ー 入力確定時(フォーカスが外れたとき)
dateプロパティはDateオブジェクトに変換した値に更新されます。
変換に失敗した場合は、dateプロパティはnullとなります。

図2.3 valueプロパティとdateプロパティ



注) 変換に失敗した場合は、valueプロパティとdateプロパティは同期しません。以下のようになります。

ー valueプロパティ:入力文字列

ー dateプロパティ:null

valueプロパティ(入力テキスト)とdateプロパティが同期しているかどうかは、isSyncメソッドで確認できます。詳細は、JavaScript APIを参照してください。

初期化時の処理順番

画面の初期化時に、dateプロパティの値が表示されるときの処理順序を以下に示します。

1. converterプロパティに基づいて、dateプロパティの値が文字列に変換され、valueプロパティに設定されます。
2. 文字列が大文字に変換されます。(uppercaseプロパティがtrueの場合)
3. フォーマッタ処理が実行されます。(labelProviderプロパティが指定されている場合)

入力確定時の処理の流れ

入力確定時の処理順序を以下に示します。

1. 文字列が大文字に変換されます。(uppercaseプロパティがtrueの場合)
2. converterプロパティで指定したフォーマッタに従って、文字列からDateオブジェクトに変換されます。

   成功した場合: dateプロパティに変換された値が設定され、dateparsesuccessイベント送出
   失敗した場合: dateプロパティにnullが設定され、dateparseerrorイベント送出
   変換に失敗した場合の処理について以下に示します。

   - onDateParseErrorイベントリスナで、エラー処理を記述することができます。
     例えば、アラートを表示する場合、以下のように記述します。

     ```
     <span rcf:type="DateInput" ... rcf:onDateParseError="alert('入力が間違っています')"></span>
     ```

   - IE 6 IE 7 IE 8
     Internet Explorerでrefocusプロパティにtrueが設定されていると、dateparseerrorが発生した場合、そのDateInputにフォーカスを戻すことができます。

   なお、dateparsesuccessまたはdateparseerrorイベントが発生する前に、dateプロパティに値が設定されるため、以前の値から変更した場合は、datechangeイベントが先に発生します。

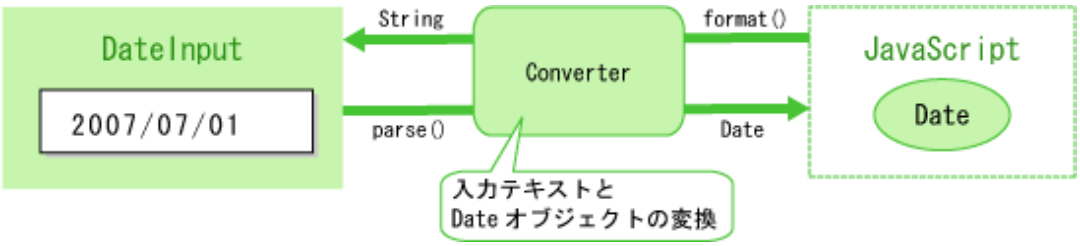
converterプロパティの指定方法

converterプロパティは、Dateオブジェクトと文字列の変換を行うオブジェクトを指定します。これにより入力テキストとDateオブジェクトの変換方法を変更することができます。

文字列からDateオブジェクトへの変換およびDateオブジェクトから文字列の変換は、可逆である必要があります。
指定するオブジェクトは、文字列に変換するformat、および入力テキストからDateを生成するparseを実装したオブジェクトである必要があります。

converterプロパティで指定されたformat関数とparse関数は、以下のように呼び出されます。

図2.4 format関数とparse関数の呼出し



入力テキスト“xxxx年xx月xx日”とJavaScriptのDateオブジェクトを変換する場合の記述例を、以下に示します。
【処理概要】

1. converterプロパティに、format関数とparse関数を持つオブジェクトを指定します。例では“df1”を指定します。
2. format関数は、引数としてDateオブジェクトを受け取り、戻り値として文字列を返します。
3. parse関数は、引数として入力文字列を受け取り、戻り値としてDateオブジェクトを返します。

```
<script type="text/javascript">
//<![CDATA[
  var df1 = {
    format: function(date) {
```

```
      if (date == null) return "";
      return date.getFullYear() + "年"
                                + (date.getMonth()+1) + "月"
                                + date.getDate() + "日";
                        },
    parse: function(value) {
      var reg1 = new RegExp("^¥¥s*([0-9]{4,})年([0-9]{1,2})月([0-9]{1,2})日¥¥s*$");
      var v = value.match(reg1);
      if (v) {
         var year = parseInt(v[1]);
         var month = parseInt(v[2])-1;
         var day = parseInt(v[3]);

         //TODO: year, month, dayの値のチェック

         return new Date(year, month, day);
      }
      throw "Invalid value";
    }
  };
//]]>
</script>

  ...

<div rcf:id="dateInput1" rcf:type="DateInput" rcf:converter="df1"></div>
```

labelProviderプロパティによるフォーマッタの指定

labelProviderプロパティは、フォーカスが当たっていないときに入力フィールドに表示する文字列を変更する場合に指定します。以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var lp1 = {
    getLabel: function(value, date, sync) {
        if (!sync) {
          return "正しい値を入力してください";
        }
        return value;
    }
  };
//]]>
</script>

  ...

<div rcf:id="dateInput1" rcf:type="DateInput" rcf:labelProvider="lp1"></div>
```

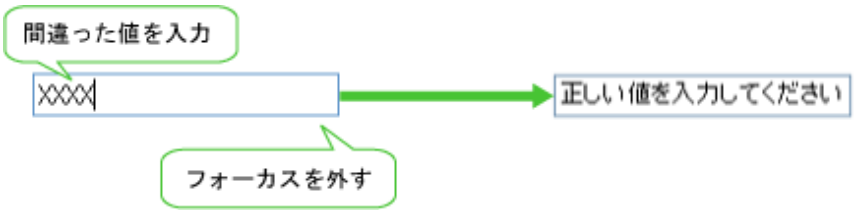
フォーマットを行うオブジェクトには、getLabel関数を用意する必要があります。getLabel関数のインタフェースは以下のとおりです。

メソッド	getLabel(value, date, sync)	
引数	value [String]	valueプロパティの値
	date [Date]	dateプロパティの値
	sync [Boolean]	valueとdateの同期状態
戻り値	表示する文字列	

DateInput部品のlabelProviderプロパティは、getLabel関数を持つオブジェクトを指定します。上記の例では、lp1を指定します。

lp1のgetLabel関数は、同期している場合は入力値を返し、同期していない場合は“正しい値を入力してください”という文字列を返すようになっています。

このように指定すると、上記の例では、以下のようにDateに変換できない値が入力された場合、“正しい値を入力してください”という文字が表示されるようになります。



注意

converterとlabelProviderの違い
converterは、入力テキストとDateオブジェクトの変換方法を指定することができます。これにより、デフォルトの変換方法を変更することができます。
一方、labelProviderは、フォーカスがない場合に表示する文字列を変更するときに指定します。これにより、実際の入力テキストとフォーカスがない場合の表示を変えることができます。

refocusプロパティに関する注意事項
refocusプロパティは、入力テキストからDateオブジェクトへの変換に失敗したときに、フォーカスをDateInputに戻す機能です。ただし、以下の場合はフォーカスが戻らないことがあります。

- ブラウザのアドレスバーやツールバーなど、ページ表示域以外にフォーカスを移動した場合
  例えば、アドレスバーをクリックしたり、アドレスバーのドロップダウンリストを表示したりすると、フォーカスがDateInputから外れることがあります。

スタイルプロパティ

本部品のスタイルプロパティは、TextInputと同じです。詳細は、“2.1.2 TextInput”の“スタイルプロパティ”を参照してください。ただし、クラス名は“rcf-DateInput”となります。
詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onDateChange	dateプロパティの値が変更されたときに呼ばれます。	ValueChangeEvent
onDateParseSuccess	入力テキストからDateオブジェクトの変換が成功したときに呼ばれます。	DateParseEvent
onDateParseError	入力テキストからDateオブジェクトの変換に失敗したときに呼ばれます。	

TextInputのイベントリスナもあります。詳細は、“2.1.2 TextInput”の“イベントリスナ”を参照してください。
部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

デフォルトの変換におけるDateParseEventのerrorプロパティ

converterプロパティを設定していないデフォルトの変換の場合、変換に失敗した場合に呼ばれるonDateParseErrorで渡されるDateParseEventのerrorプロパティには、以下のメッセージを含んだErrorオブジェクトが設定されます。

メッセージ番号はErrorオブジェクトのrcf.messageIDプロパティ、エラーメッセージはErrorオブジェクトのmessageプロパティから取得できます。メッセージの詳細は、“ユーザーズガイド”を参照してください。

- RCF12900
- RCF12901

- RCF12902
- RCF12903

なお、converterを指定した場合、DateParseEventのerrorプロパティには、そのconverterのparse関数で発生したエラーが設定されます。

JavaScript API

■isSyncメソッド

メソッド	isSync()	
引数	なし	
戻り値	[Boolean]	true:同期している場合 false:同期していない場合
例外	なし	
説明	valueプロパティ(入力テキスト)とdateプロパティの値の同期状態を返します。	

■parseメソッド

メソッド	parse()	
引数	なし	
戻り値	[Boolean]	true:プロパティの設定に成功しました。 false:プロパティの設定に失敗しました。
例外	なし	
説明	入力テキストをDateオブジェクトに変換し、dateプロパティに設定します。 通常は、入力確定時に実行しているDateオブジェクトへの変換を、このAPIにより実行することができます。 変換に失敗した場合は、dateプロパティにnullが設定されます。 変換結果のイベント(dateparseerror、dateparsesuccess)も発生します。 uppercaseプロパティをtrueに設定している場合、このメソッドを実行すると大文字化の処理が実行され、入力テキストが変更されます。	

■clearメソッド

メソッド	clear()
引数	なし
戻り値	なし
例外	なし
説明	dateプロパティおよびvalueプロパティを省略値に戻し、入力テキストを空にします。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.1.10 NumberInput

NumberInputは、整数および実数の入力フィールドを表示する部品です。

- JavaScript API
- 補足事項

ポイント

検証を実行するValidationHelper、入力制限を行うLimiterなどの機能付加部品を利用することができます。

表示例

本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

記述形式

```
<div rcf:type="NumberInput" ... ></div>
```

または

```
<span rcf:type="NumberInput" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
number	Number	数値データを指定します。APIやバインディングによりnumberプロパティが更新された場合は、入力値(valueプロパティの値)も更新されます。	可	NaN	値、バインド	可	不可
converter	format関数とparse関数を持つObject	以下の関数を持つオブジェクトを指定します。 ・ JavaScriptのnumberを文字列に変換(format関数) ・ 入力テキストをJavaScriptのnumberに変換(parse関数) format関数またはparse関数を持たないオブジェクトを指定した	可	デフォルトの変換を実行するObject	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		場合は、エラーになります。詳細は、“converterプロパティの指定方法”を参照してください。 デフォルトの変換は、以下のとおりです。 ・ number→テキスト JavaScript の Number.toString(10) で 10 進数の文字列に変換 NaNの場合は、空文字列に変換 ・ テキスト→number 実数表記(指数表記を含む)を、JavaScript の parseFloat()でnumberオブジェクトに変換 実数表記でない場合は、エラー 空文字列の場合は、NaNに変換					
refocus IE 6 IE 7 IE 8	Boolean	入力フィールドからフォーカスが外れて、numberオブジェクトへの変換に失敗した場合、フォーカスを再度戻すかどうかを指定します。 ・ true:フォーカスを戻す ・ false:フォーカスを戻さない Internet Explorerでだけ指定可能です。 Firefoxの場合は、指定しても無視されます。	可	false	値	可	不可

TextInputのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.1.2 TextInput”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

- titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
  また、発生位置はブラウザにより異なります。

  なお、空文字列を指定した場合は、表示されません。
- 入力テキストがJavaScriptのnumberオブジェクトに変換されるのは、入力フィールドからフォーカスが外れ、前の値から変更された場合だけです。入力中にnumberプロパティの値を参照すると、前回の入力テキストが変換された結果になることがあります。

ポイント

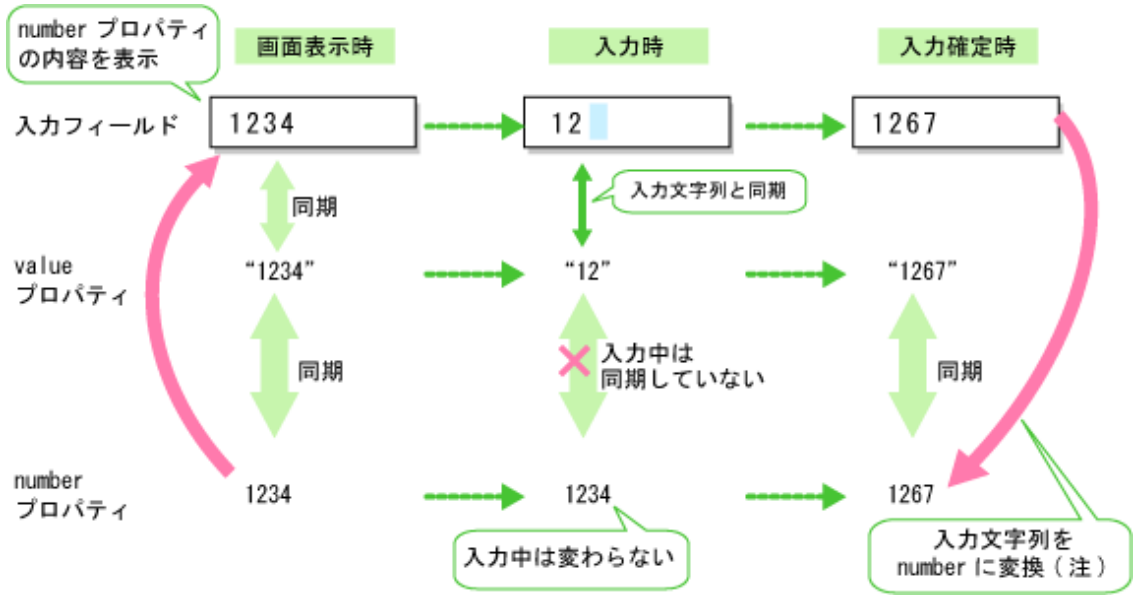
有効桁数は、JavaScriptおよびブラウザと同じ値になります。

valueプロパティとnumberプロパティの違い

valueプロパティとnumberプロパティは、どちらもテキストを入力できるプロパティです。両者の違いについて以下に示します。

- 画面表示時
  入力フィールドの初期値は、numberプロパティで指定します。numberプロパティが指定されていない場合、入力フィールドの初期値は空になります。
  valueプロパティの値は、numberプロパティを文字列にした値に初期化されます。
- 入力時
  入力フィールドに値を入力しているとき、valueプロパティはその値と同期します。numberプロパティは入力中のデータとは同期せずに、そのままの値が保持されます。
- 入力確定時(フォーカスが外れたとき)
  numberプロパティは入力テキストをnumberに変換した値に更新されます。変換に失敗した場合は、numberプロパティはNaNとなります。

図2.5 valueプロパティとnumberプロパティ



注) 変換に失敗した場合は、valueプロパティとnumberプロパティは同期しません。以下のようになります。

- valueプロパティ:入力文字列
- numberプロパティ:NaN

valueプロパティ(入力テキスト)とnumberプロパティが同期しているかどうかは、isSyncメソッドで確認できます。詳細は、JavaScript APIを参照してください。

初期化時の処理順番

画面の初期化時に、valueプロパティの値が表示されるときの処理順序を以下に示します。

1. converterプロパティに基づいて、numberプロパティの値が文字列に変換され、valueプロパティに設定されます。
2. 文字列が大文字に変換されます。(uppercaseプロパティがtrueの場合)
3. フォーマッタ処理が実行されます。(labelProviderプロパティが指定されている場合)

入力確定時の処理の流れ

入力確定時の処理順序を以下に示します。

1. 文字列が大文字に変換されます。(uppercaseプロパティがtrueの場合)

2. converterプロパティで指定したフォーマッタに従って、numberから文字列に変換されます。

   成功した場合:numberプロパティに変換した値を設定し、numberparsesuccessイベントを送出
   失敗した場合:numberプロパティにNaNを設定し、numberparseerrorイベントを送出
   変換に失敗した場合の処理について、以下に示します。

   - onNumberParseErrorイベントリスナでエラー処理を記述することができます。
     例えば、アラートを表示する場合、以下のように記述します。

```
<span rcf:type="NumberInput" ... rcf:onNumberParseError="alert('入力が間違っています')"></span>
```

   - IE 6 IE 7 IE 8
     Internet Explorerでrefocusプロパティにtrueが設定されていると、numberparseerrorが発生した場合、そのNumberInputにフォーカスを戻すことができます。

   なお、numberparsesuccessまたはnumberparseerrorイベントが発生する前に、プロパティに値が設定されるため、以前の値から変更した場合は、numberchangeイベントが先に発生します。

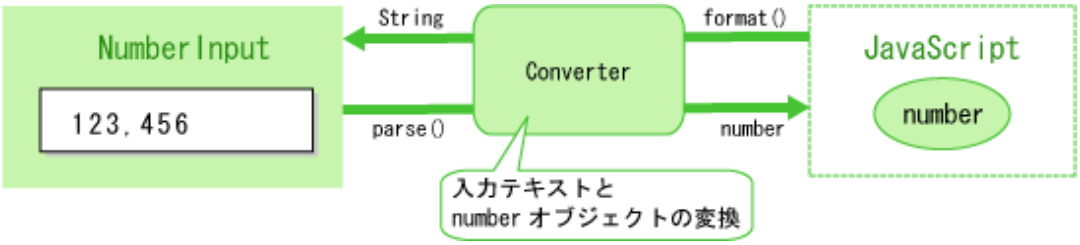
converterプロパティの指定方法

converterプロパティは、numberと文字列の変換を行うオブジェクトを指定します。これにより、入力テキストとnumberの変換方法を変更することができます。

文字列からnumberへの変換およびnumberから文字列の変換は、可逆である必要があります。
指定するオブジェクトは、numberを文字列に変換するformat、および入力テキストからnumberを生成するparseを実装したオブジェクトである必要があります。

converterプロパティで指定されたformat関数とparse関数は、以下のように呼び出されます。

図2.6 format関数とparse関数の呼出し



正の実数を小数点以下3桁表示するフィールドに入力されたテキストと、JavaScriptのnumberオブジェクトを変換する場合の記述例を以下に示します。
【処理概要】

1. converterプロパティに、format関数とparse関数を持つオブジェクトを指定します。例では“nf1”を指定します。
2. format関数は、引数としてnumberを受け取り、戻り値として文字列を返します。
3. parse関数は、引数として入力文字列を受け取り、戻り値としてnumberを返します。

```
<script type="text/javascript">
//<![CDATA[
  // 正の実数を小数点以下3桁で表示する。
  var nf1 = {
    format: function(num) {
      if (isNaN(num)) return "";
      var val = num * 1000;
      val = Math.floor(val);
      val = val / 1000;
      return val.toString(10);
    },

    parse: function(value) {
      var reg1 = new RegExp("^¥¥s*([0-9]+(.[0-9]{1,3})?)[0-9]*¥¥s*$");
      var v = value.match(reg1);
      if (v) {
```

```
            return parseFloat(v[0]);
        }
        throw "invalid value";
      }
    };
//]]>
</script>

  ...

<div rcf:id="numberInput1" rcf:type="NumberInput" rcf:converter ="nf1"></div>
```

labelProviderプロパティによるフォーマッタの指定

labelProviderプロパティは、フォーカスが当たっていないときに入力フィールドに表示する文字列を変更する場合に指定します。以下に例を示します。

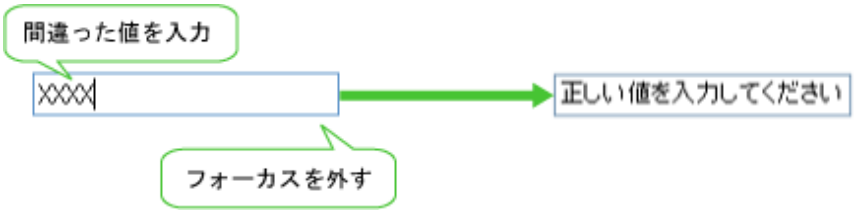
```
<script type="text/javascript">
//<![CDATA[
  var lp1 = {
    getLabel: function(value, number, sync) {
      if (!sync) {
        return "正しい値を入力してください";
      }
    return value;
    }
  };
//]]>
</script>

  ...

<div rcf:id="numberInput1" rcf:type="NumberInput" rcf:labelProvider="lp1"></div>
```

フォーマットを行うオブジェクトには、getLabel関数を用意する必要があります。getLabel関数のインタフェースは以下のとおりです。

メソッド	getLabel(value, number, sync)	
引数	value [String]	valueプロパティの値
	number [Number]	numberプロパティの値
	sync [Boolean]	valueとnumberの同期状態
戻り値	表示する文字列	

NumberInput部品のlabelProviderプロパティは、getLabel関数を持つオブジェクトを指定します。上記の例では、lp1を指定しています。

lp1のgetLabel関数は、同期している場合は入力値を返し、同期していない場合は“正しい値を入力してください”という文字列を返すようになっています。
このように指定すると、上記の例では、以下のようにnumberに変換できない値が入力された場合、“正しい値を入力してください”という文字が表示されるようになります。



注意

converterとlabelProviderの違い
converterは、入力テキストとnumberの変換方法を指定することができます。これにより、デフォルトの変換方法を変更することができます。
一方、labelProviderは、フォーカスがない場合に表示する文字列を変更するときに指定します。これにより、実際の入力テキストとフォーカスがない場合の表示を変えることができます。

refocusプロパティに関する注意事項
refocusプロパティは、入力テキストからNumberオブジェクトへの変換に失敗したときに、フォーカスをNumberInputに戻す機能です。ただし、以下の場合はフォーカスが戻らないことがあります。

- ブラウザのアドレスバーやツールバーなど、ページ表示域以外にフォーカスを移動した場合
  例えば、アドレスバーをクリックしたり、アドレスバーのドロップダウンリストを表示したりすると、フォーカスがNumberInputから外れることがあります。

スタイルプロパティ

本部品のスタイルプロパティは、TextInputと同じです。詳細は、“2.1.2 TextInput”の“スタイルプロパティ”を参照してください。ただし、クラス名は“rcf-NumberInput”となります。
詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onNumberChange	numberプロパティの値が変更されたときに呼ばれます。	ValueChangeEvent
onNumberParseSuccess	入力テキストからnumberへの変換が成功したときに呼ばれます。	NumberParseEvent
onNumberParseError	入力テキストからnumberオブジェクトへの変換に失敗したときに呼ばれます。	

TextInputのイベントリスナもあります。詳細は、“2.1.2 TextInput”の“イベントリスナ”を参照してください。
部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

デフォルトの変換におけるNumberParseEventのerrorプロパティ

converterプロパティを設定していないデフォルトの変換の場合、変換に失敗した場合に呼ばれるonNumberParseErrorに渡されるNumberParseEventのerrorプロパティには、以下のメッセージが含まれたErrorオブジェクトが設定されます。
メッセージ番号はErrorオブジェクトのrcf.messageIDプロパティ、エラーメッセージはErrorオブジェクトのmessageプロパティから取得できます。メッセージの詳細は、“ユーザーズガイド”を参照してください。

- RCF13000

なお、converterを指定した場合、NumberParseEventのerrorプロパティには、そのconverterのparse関数で発生したエラーが設定されます。

JavaScript API

■isSyncメソッド

メソッド	isSync()	
引数	なし	
戻り値	[Boolean]	true:同期している場合 false:同期していない場合
例外	なし	

説明	valueプロパティ(入力テキスト)とnumberプロパティの値の同期状態を返します。

■parseメソッド

メソッド	parse()	
引数	なし	
戻り値	[Boolean]	true:プロパティの設定に成功しました。 false:プロパティの設定に失敗しました。
例外	なし	
説明	入力テキストをnumberに変換し、numberプロパティに設定します。 通常は、入力確定時に実行しているnumberへの変換を、このAPIにより実行することができます。 変換に失敗した場合は、numberプロパティにNaNが設定されます。 変換結果のイベント(numberparseerror、numberparsesuccess)も発生します。 uppercaseプロパティをtrueに設定している場合、このメソッドを実行すると大文字化の処理が実行され、入力テキストが変更されます。	

■clearメソッド

メソッド	clear()
引数	なし
戻り値	なし
例外	なし
説明	numberプロパティおよびvalueプロパティを省略値に戻し、入力テキストを空にします。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

NumberInputのデフォルトのconverterでは、JavaScriptのparseFloat()を利用しているため、入力値によっては誤差が生じる場合があります。詳細は、“5.2.7 Number型のデータに関する注意事項”を参照してください。

2.1.11 MaskedTextInput IE 6 IE 7 IE 8

MaskedTextInputは、穴埋め方式の入力フィールドを表示する部品です。

注意

MaskedTextInputを利用できるブラウザ IE 6 IE 7 IE 8

MaskedTextInputは、Internet Explorerでだけ利用可能な部品です。ほかのブラウザでは、TextInputとして表示されます。ただし、以下の点に注意が必要です。

- typeプロパティの値は、“MaskedTextInput”になります。

- スタイルプロパティのクラス名は、“rcf-MaskedTextInput”です。
- AutoCompleterの対象には指定できません。
- Internet Explorerでは、TextInput固有の機能は使用できません。

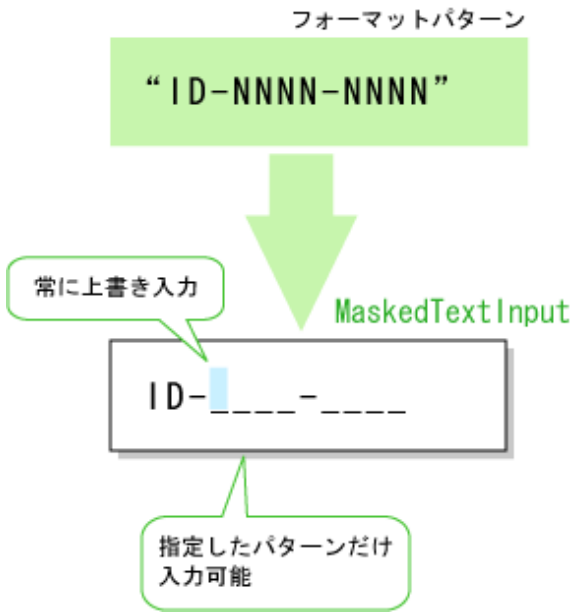
また、Internet Explorerでも、入力フィールドに以下の制約が追加されます。

- IMEを使った入力はできません。(IMEは常に無効です。)
- 常に上書きモードでの入力になります。
- 入力フィールドに対する文字列のカットやペーストはできません。
- ドラッグ アンド ドロップはできません。
- 右クリックしても、コンテキストメニューは表示されません。

ポイント

ValidationHelperを利用して検証を実行するなど、機能付加部品を利用できます。

表示例



カーソルキーでの左右移動が可能です。ただし、移動するのは入力可能な部分だけで、区切り文字の部分にカーソルは移動しません。

本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

記述形式

```
<div rcf:type="MaskedTextInput" ... ></div>
```

または

```
<span rcf:type="MaskedTextInput" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
format	String	フォーマットパターンを指定します。デフォルトは何も入力できません。また、初期値だけが指定可能であり、表示中にフォーマットパターンを変更することはできません。 詳細は、“フォーマットパターンについて”を参照してください。	可	""	値	不可	不可
value	String	テキストを指定します。	可	""	値、バインド	可	不可
tabIndex	Number	タブの順番を指定します。	可	0	値	可	不可

TextInputのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は“2.1.2 TextInput”の“プロパティ”を参照してください。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。

また、発生位置はブラウザにより異なります。
なお、空文字列を指定した場合は、表示されません。

ポイント

自動脱出機能を有効にするには、以下の条件を満たす必要があります。

- autoEscapeプロパティにtrueが指定されていること
- 本部品がFocusManagerに登録されていること
- 本部品がFocusManagerの対象となっており、FocusManagerに次のフォーカスの移動先が指定されていること
- 指定されたフォーマットパターンに、文字がすべて入力されていること
- カーソルが入力可能な部分の最後にあること

フォーマットパターンについて

フォーマットパターンは、入力部分と区切り文字で構成されます。
例えば、“平成NN年NN月NN日”と記述した場合、“N”の部分が入力部分であり、それ以外の、“平成”、“年”、“月”、“日”はあらかじめ決められた区切り文字となります。

フォーマットパターンの入力部分として、指定できる記号を以下に示します。

- A:アルファベット(A-Zおよびa～z)または半角空白が入力可能
- N:数字(0～9)または半角空白が入力可能
- X:アルファベット、数字、および半角空白が入力可能

上記以外の文字は、区切り文字として扱われます。
ただし、区切り文字に上記の文字を使用する場合は“%X”のように“%”を指定します。“%”に続く1文字(A、N、X以外も可)は区切り文字として扱われ、“%”自身は区切り文字には含まれません。
また、“%”自体を区切り文字として使用する場合は“%%”と指定します。文字列の末尾が“%”の場合は、“%”は無視されます。

valueプロパティについて

valueプロパティを指定するときには、フォーマットパターンに従った文字列を指定してください。フォーマットパターンと異なる文字列を指定した場合、フォーマットパターンに初期化されます。

フォーマットパターンに従った文字列とは、以下の条件を満たしているものです。

- フォーマットパターンに前方一致する文字列であること
- フォーマットパターンの入力部分には、入力可能な文字または空白が指定されていること
- 文字列の長さがフォーマットパターンの文字列の長さ以下であること

フォーマットパターンが“ID-NN-AA”の場合の例を、以下に示します。

- 正しい例

valueプロパティの値	備考
ID-1	
ID- 1-A	入力部分に空白を指定した例
ID-12-AB	

- 間違った例

valueプロパティの値	備考
1	区切り文字がなく、前方一致してない
ID-A	フォーマットパターンの“N”の部分にアルファベットが指定されている
ID-12-ABC	フォーマットパターンの文字列よりも長い

また、valueプロパティを取得したときに、フォーマットパターンで入力部分として指定した部分に文字が入力されていない場合、その部分の文字は半角空白になっています。

スタイルプロパティ

本部品のスタイルプロパティは、TextInputと同じです。詳細は、“2.1.2 TextInput”の“スタイルプロパティ”を参照してください。ただし、クラス名は“rcf-MaskedTextInput”となります。
詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onWrongKeyPress	フォーマットパターンに対して、誤ったキーが押されたときに呼ばれます。	WrongKeyPressEvent

名前	説明	イベントオブジェクト
	例えば、入力部分にA(アルファベット)と指定したところに、数値が入力されようとしたときに呼ばれます。	
onInvalidValueError	valueプロパティに、フォーマットパターンと異なった文字列が指定されたときに呼ばれます。 例えば、以下の場合に呼ばれます。 ・ 初期値として指定したvalueプロパティが、フォーマットパターンと異なっている場合 ・ モデル側でvalueプロパティがバインディングしている値を変更したときに、フォーマットパターンと異なる値が指定された場合 このイベントが発生したあと、valueプロパティはフォーマットパターンに初期化されます。	InvalidValueErrorEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。
TextInputのイベントリスナもあります。(onChangeを除きます。)詳細は、“2.1.2 TextInput”の“イベントリスナ”を参照してください。

InvalidValueErrorEventのerrorプロパティ

valueプロパティにフォーマットパターンと異なった文字列が設定されていた場合、onInvalidValueErrorが呼ばれます。InvalidValueErrorEventのerrorプロパティには、以下のメッセージが含まれたErrorオブジェクトが設定されます。メッセージの詳細は、“ユーザーズガイド”を参照してください。

－ RCF13100

なお、メッセージ番号はErrorオブジェクトのrcf.messageIDプロパティ、エラーメッセージはErrorオブジェクトのmessageプロパティから取得できます。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

MaskedTextInputがフォーカスを得たとき、区切り文字以外の最初に入力できる位置にカーソルが表示されます。

2.1.12 MaskedDateInput IE 6 IE 7 IE 8

MaskedDateInputは、穴埋めで日付と時間を入力する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

MaskedDateInputを利用できるブラウザ IE 6 IE 7 IE 8

MaskedDateInputは、Internet Explorerでだけ利用できます。ほかのブラウザでは、DateInputとして動作します。ただし、以下の点に注意が必要です。

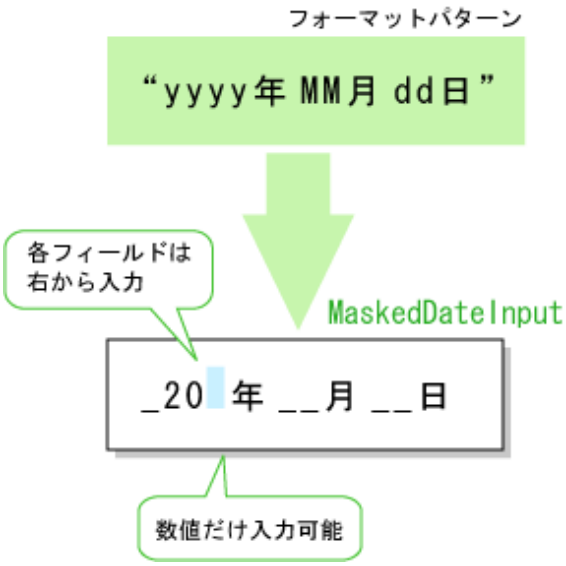
- typeプロパティの値は、“MaskedDateInput”になります。

- スタイルプロパティのクラス名は、“rcf-MaskedDateInput”です。
- Internet Explorerでは、DateInput固有の機能は使用できません。

また、Internet Explorerでも、入力フィールドに以下の制約が追加されます。

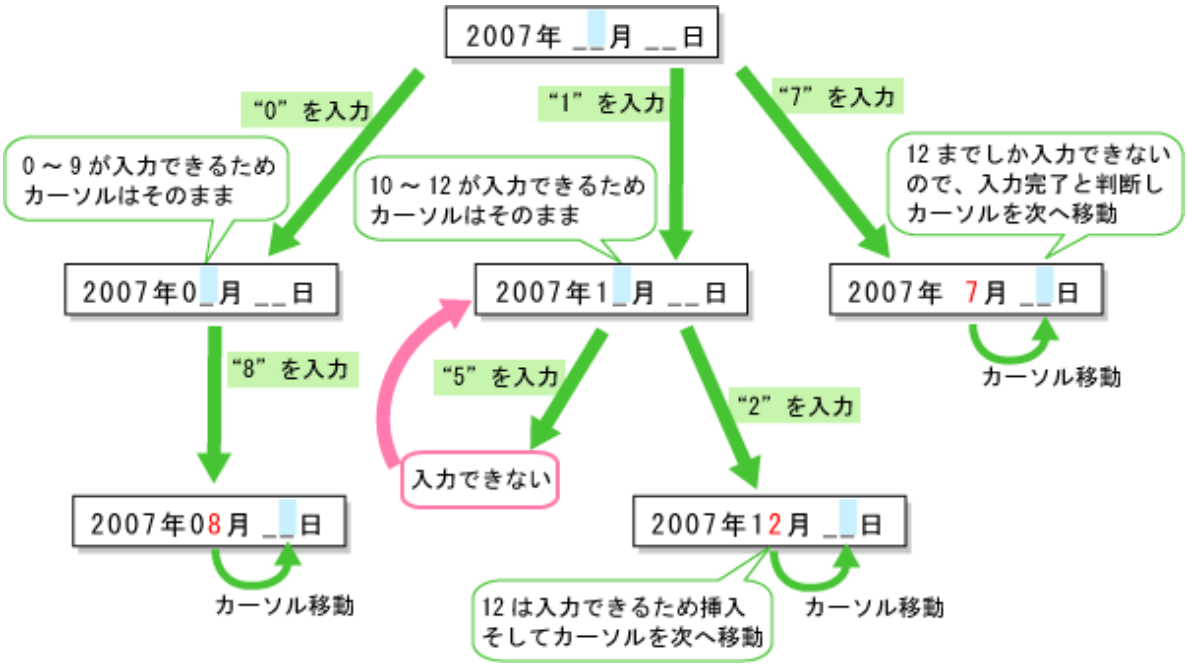
- IMEを使った入力はできません。IMEは常に無効です。
- 入力フィールドに対する文字列のカットやペーストはできません。
- 現在入力中のフィールドが未入力や修正中で有効範囲外の値になっている状態でも、カーソルキーやマウスでの移動は可能です。このとき、フィールドの入力状態はそのまま残ります。カーソルキーやマウスでのフィールド間移動では、year、monthなどのプロパティの値は変更されません。

表示例



本部品では、Ctrlキー + z が使用できます。詳細は、“5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し”を参照してください。

図2.7 入力によるカーソル移動の例



記述形式

```
<div rcf:type="MaskedDateInput" ... ></div>
```

または

```
<span rcf:type="MaskedDateInput" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
format	String	フォーマットパターンを指定します。初期値だけが指定可能であり、表示中にフォーマットパターンを変更することはできません。 詳細は“フォーマットパターンについて”を参照してください。	可	デフォルトのフォーマット	値	不可	不可
year	Number	年の値を指定します。 ・ フォーマットパターンがyyyyの場合:1～9999 ・ フォーマットパターンがyyの場合 :twoDigitYearStart の値 ～ twoDigitYearStart+99 ・ フォーマットに年を含まない場合:100～9999 ・ -1で更新した場合:本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。 ・ 範囲外の値で更新した場合:エラー	可	-1	値、バインド	可	不可
month	Number	月の値-1を指定します。(0-11) -1で更新した場合は、本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。 範囲外の値で更新した場合は、エラーとなります。	可	-1	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
day	Number	日の値を指定します。 -1で更新した場合は、本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。 範囲外の値で更新した場合は、エラーとなります。	可	-1	値、バインド	可	不可
hour	Number	時の値を指定します。 -1で更新した場合は、本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。 範囲外の値で更新した場合は、エラーとなります。	可	-1	値、バインド	可	不可
minute	Number	分の値を指定します。 -1で更新した場合は、本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。 範囲外の値で更新した場合は、エラーとなります。	可	-1	値、バインド	可	不可
second	Number	秒の値を指定します。 -1で更新した場合は、本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。 範囲外の値で更新した場合は、エラーとなります。	可	-1	値、バインド	可	不可
date	Date	日時データを指定します。 フォーカスが外れて入力が確定したときに、すべての入力フィールドに有効な値が設定されている場合、その値を元にDateオブジェクトが生成され、設定されます。 本プロパティを更新した場合、utcに従って取得した年月日時分秒の各値が、上記のyear、month、day、hour、minute、およびsecondの各プロパティに反映されます。また、それぞれの値は、それぞれのプロパティの有効範囲内である必要があります。 nullで更新した場合は、本プロパティを更新するだけで、ほかのプロパティや表示への影響はありません。	可	null	値、バインド	可	不可
utc	Boolean	日付をUTCとして扱うかどうかを指定します。 ・ true:世界標準時間(UTC) ・ false:ローカル時間 “5.1.9 Dateオブジェクトとタイムゾーン”を参照してください。	可	false	値	不可	不可
twoDigitYearStart	Number	西暦を2桁で記述した場合、本プロパティで指定した年から始まる100年間の範囲で解釈されます。 [例]1950と指定した場合、以下のように解釈されます。	可	1930	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ 50～99:1900年代 ・ 00～49:2000年代					
value	String	入力テキストの値を指定します。	可	""	バインド	不可	不可
refocus	Boolean	入力フィールドからフォーカスが外れ、Dateオブジェクトへの変換が失敗した場合、フォーカスを再度MaskedDateInputに戻すかどうかを指定します。 ・ true:フォーカスを戻す ・ false:フォーカスを戻さない すべての入力フィールドが空の場合、フォーカスは戻しません。	可	false	値	可	不可
tabIndex	Number	タブの順番を指定します。	可	0	値	可	不可

TextInputのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は“2.1.2 TextInput”の“プロパティ”を参照してください。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

ポイント

自動脱出機能を有効にするには、以下の条件を満たす必要があります。

- autoEscapeプロパティにtrueが指定されていること
- 本部品がFocusManagerに登録されていること
- 本部品がFocusManagerの対象となっており、FocusManagerに次のフォーカスの移動先が指定されていること
- 指定されたフォーマットパターンに、文字がすべて入力されていること
- カーソルが入力可能な部分の最後にあること

フォーマットパターンについて

フォーマットパターンには、以下の文字が設定できます。

入力文字	説明	入力方法
y	年(西暦)を表します。	yyyy(4桁入力)またはyy(2桁入力)
M	月を表します。	MM
d	日を表します。	dd
H	時を表します。	HH
m	分を表します。	mm

入力文字	説明	入力方法
s	秒を表します。	ss

そのほかの文字は区切り文字としてみなされます。デフォルトでは、以下のフォーマットパターンが定義されています。

－ yyyy/MM/dd

フォーマットパターン内のフォーマット文字のある部分がそれぞれ数値入力フィールドとなります。上記フォーマット文字を区切り文字と使用する場合は、“%y”のように“%”を指定します。“%”に続く1文字(フォーマット文字以外も可)は区切り文字として扱われ、“%”自身は区切り文字には含まれません。
また、“%”自体を区切り文字として使用する場合は“%%”と指定します。文字列の末尾が“%”の場合は、“%”は無視されます。

不正なフォーマットについて以下に示します。不正なフォーマットが指定された場合、エラーとなります。

－ フォーマットパターンに上記フォーマット文字が存在していない場合

－ 各数値入力フィールドに区切り文字がない場合

－ フォーマットが重複している場合

－ フォーマットパターンの桁数が正しくない場合

フォーマットパターンの例を以下に示します。

フォーマットパターン	表示形式(数値部分が入力フィールド)
yyyy年MM月dd日	2006年10月10日
HH時mm分ss秒	15時30分25秒
%year=yyyy %Month=MM	year=2006 Month=10
日付:"yy/MM/dd	日付:’06/10/10
yyyyMMdd	区切り文字がないので設定不可
yyyy年MM月yyyy年	フォーマットが重複しているため設定不可

各フィールドに入力できる数値を以下に示します。範囲外の数値、および数字以外の文字は入力できません。
例えば、月フィールドに50と入力しようとすると、5を入力した時点でカーソルが次のフィールドに移動します。

フィールド名	入力できる内容
年フィールド(yy or yyyy)	1～9999(yyyyの場合)、0～99(yyの場合)
月フィールド(MM)	1～12
日フィールド(dd)	1～31(注)
時フィールド(HH)	0～23
分フィールド(mm)	0～59
秒フィールド(ss)	0～59 うるう秒には対応しません。

注) 年・月フィールドの値により変わります。

－ 1、3、5、7、8、10、12月:1～31

－ 4、6、9、11月:1～30

－ 2月:1～29(うるう年)、1～28(うるう年以外)

月が指定されていない場合は1～31、年が指定されていない場合はうるう年とみなされます。

フォーマットパターンに含まれない単位について

フォーマットパターンは、年月日時分秒のすべての単位を含む必要はありません。フォーマットパターンに含まれない単位に関しては、dateプロパティの設定および入力値のチェックにおいて以下の値が使用されます。

- year、month、day、hour、minute、secondの各プロパティに設定された値(-1以外の場合)
- 前項のプロパティ値が-1の場合
  - 年、月、日は、システム時間の年、月、日の値
  - 時、分、秒は、0

フィールド値の確定について

フィールド値の確定について、以下に示します。

- スペースキーを押したとき、入力中のフィールドの値が正しい場合は、値が確定されます。
- フィールドの末尾に数字をキー入力したとき、以下のどちらかの場合は値が確定されます。
  - すべての桁を入力し終え、かつ値が正しい場合
    例:月フィールドに“12”と入力
  - それ以上数字を入力すると、最大値を越える場合
    例:月フィールドに“9”と入力
- カーソルキーやマウスでフィールド間を移動した場合や、末尾以外に数字を入力した場合には、値の確定は行われません。
- フォーカスが部品から外れたとき、各フィールドに正しい値が入力されていた場合は、それぞれ入力された値で確定されます。
- 値が確定した場合には、対応するyear、month、day、hour、minute、secondのどれかのプロパティに、その値が設定されます。
- 値が確定した場合、次のフィールドにカーソルが移動します。(フォーカスを外した場合を除く)
- 値が確定した場合、上位桁が埋められていなければ、0がパディングされます。
- 入力フィールドの文字を削除して空にした場合、対応する year、month、day、hour、minute、secondのプロパティの値は、-1に設定されます。

初期値の指定方法

初期値を指定する方法は以下の2つがあります。なお、valueプロパティでは初期値を設定することはできません。

- dateプロパティによる指定
  dateプロパティで各フィールドの値をまとめて指定します。
- year、month、day、hour、minute、secondプロパティによる指定
  各フィールドの値を個々のプロパティで指定します。

上記の両方を指定した場合は、dateプロパティから取得した年月日時分秒の値に対して、各プロパティの値が上書きされます。

日フィールドおよびdayプロパティの値チェック

キー入力で日フィールドを入力する場合は、年と月の値によって入力可能な範囲が1-28、1-29、1-30、1-31と変わります。年が未設定の場合はうるう年、月が未設定の場合は31日までとみなされます。

dayプロパティをAPIで更新する場合は、年と月の値によらず、1-31の範囲を有効とします。

また、状況に応じて入力可能範囲が変動するのは日フィールドだけです。年・月のフィールドを入力する際には、日フィールドの値は考慮されません。
(例:日に31を入力したあと、月の値を2に変更するのは可能です。)

スタイルプロパティ

本部品のスタイルプロパティは、TextInputと同じです。詳細は、“2.1.2 TextInput”の“スタイルプロパティ”を参照してください。ただし、クラス名は“rcf-MaskedDateInput”となります。
詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onWrongKeyPress	フォーマットパターンに対して、誤ったキーが押されたときに呼ばれます。例えば、月の入力部分(MM)にA(ア	WrongKeyPressEvent

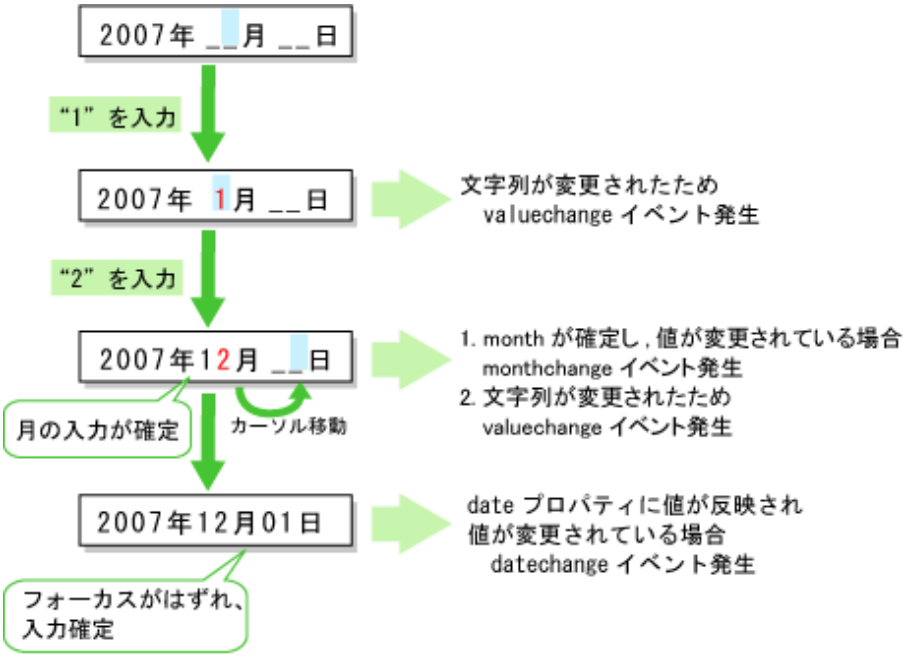
名前	説明	イベントオブジェクト
	ルファベット)を入力しようとしたり、1のあとに3を入力しようとしたりしたときに呼ばれます。	
onYearChange	yearプロパティが変更されたときに呼ばれます。	ValueChangeEvent
onMonthChange	monthプロパティが変更されたときに呼ばれます。	
onDayChange	dayプロパティが変更されたときに呼ばれます。	
onHourChange	hourプロパティが変更されたときに呼ばれます。	
onMinuteChange	minuteプロパティが変更されたときに呼ばれます。	
onSecondChange	secondプロパティが変更されたときに呼ばれます。	
onDateChange	dateプロパティが変更された場合に呼ばれます。	

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。
TextInputのイベントリスナもあります。(onChangeを除きます。)詳細は、“2.1.2 TextInput”の“イベントリスナ”を参照してください。

各プロパティの変更イベントについて

year、month、day、hour、minute、secondプロパティは、各フィールドで入力が確定したときにプロパティが変更されます。
例えば、monthプロパティの場合、以下のように月フィールドからカーソルが移動した時点で入力が確定し、イベントが発生します。

図2.8 monthプロパティの変更イベント



- valuechangeイベントは、文字が入力された時点で文字列が変更されていた場合に発生します。
- monthプロパティなどは、各フィールドで入力が確定した場合に値が設定されます。
  このときに値が変更されていれば、changeイベントが発生します。
  例えば、monthプロパティの場合は、monthchangeイベントが発生します。monthchangeイベントは、valuechangeイベントより先に発生します。
- フォーカスが外れて入力が確定したとき、dateプロパティに入力値が反映されます。このときに値が変更されていれば、datechangeイベントが発生します。

初期状態でのイベント発生

MaskedDateInputでは、初期処理においてyear、month、day、hour、minute、second、およびdateの初期値から、valueの値を求めて設定します。この際に、valueに対するpropertychangeイベントが発生します。

JavaScript API

■isSyncメソッド

メソッド	isSync(name)	
引数	name [String]	プロパティ名 以下のプロパティ名を指定できます。 date、year、month、day、hour、minute、second
戻り値	[Boolean]	true:同期している場合 false:同期していない場合
例外	なし	
説明	入力テキストと指定プロパティの値の同期状態を返します。 dateがnullの場合は、すべての入力フィールドが空の場合に同期しているとみなされます。 year、month、day、hour、minute、secondが-1の場合は、対応する入力フィールドが空の場合に同期しているとみなされます。	

■clearメソッド

メソッド	clear()
引数	なし
戻り値	なし
例外	なし
説明	year、month、day、hour、minute、second、dateの各プロパティを、省略値で再初期化します。 表示文字列の各フィールドを空にし、valueプロパティに設定します。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

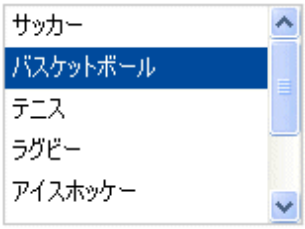
MaskedDateInputがフォーカスを得たとき、入力フィールドの先頭にカーソルが位置づけられます。フィールドの先頭に値が設定されている場合は、それが選択された状態になります。

2.1.13 SelectList

SelectListは、単一選択および複数選択が可能なリストを表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

表示例



記述形式

```
<div rcf:type="SelectList" ... ></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は前後に改行コードが挿入されて表示されます。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
options	Array	選択項目を表す値のリストを指定します。表示されるラベルは、labelProviderプロパティの指定に従います。	可	[]	値、バインド(注3)	可	可
labelProvider	getLabel関数を持つオブジェクト	各選択項目の文字列を確定する関数を指定します。options配列のメンバに応じた文字列変換ロジックを定義します。getLabel関数には、options配列の個々の要素が渡されます。 nullを指定した場合、デフォルトlabelProviderが使用されます。詳細は、“デフォルトlabelProvider”を参照してください。 以下の場合はエラーとなります。 ・ nullおよびObject以外の値が指定された場合 ・ getLabel関数を持たないObjectが指定された場合	可	null(デフォルトlabelProvider)	値	不可	不可
multiple	Boolean	複数選択または単一選択を指定します。 ・ true:複数選択 ・ false:単一選択	可	false	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
selectedIndex	Number	単一選択(multiple=false)の場合 ・ 選択されている項目のインデックス(先頭は0)を指定します。 ・ 選択状態の初期値を指定できます。 ・ -1を指定した場合は、選択が解除されます。 ・ 無効な値を指定した場合は、エラー(RCF13302)となります。 複数選択(multiple=true)の場合 ・ 最後に選択された項目のインデックスを指定します。 ・ 部品の初期化後に本プロパティを変更しようとすると、エラー(RCF11002)となります。“multipleとselectedIndex、selectedIndexesの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは-1(選択なし)に設定されます。	可	-1	値、バインド	可 (注1)	不可
selectedIndexes	Numberの Array	選択されている項目のインデックス(先頭は0)配列を指定します。 選択順にインデックスが格納されます。 このプロパティにより選択状態の初期値を指定できます。 何も選択されていない場合は、長さ0の配列となります。 無効な値が指定された場合は、エラー(RCF13302)となります。 単一選択(multiple=false)の場合、部品の初期化後に本プロパティを変更しようとするとエラー(RCF11002)となります。“multipleとselectedIndex、selectedIndexesの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは長さ0の配列(選択なし)に設定されます。	可	[]	値、バインド	可 (注2)	可 (注2)
title	String	optionsプロパティがStringまたはObjectの配列で、かつ各Objectがtitle属性でない場合、ツールチップで表示されるテキストを指定します。 なお、optionsプロパティがObjectの配列で、かつObjectがtitle属性の場合は、その値がツールチップとして表示されます。	可	""	値	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・ true:有効	可	true	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ false:無効 操作できなくなりイベントも発生しません。					
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可
renderer	render関数を持つオブジェクト	ラベルや項目のインデックスを入力値として各項目の表示用ノードを出力する関数を指定します。 nullが指定された場合には、ラベルの値をそのまま表示します。 以下の場合は、エラー(RCF13303)となります。 ・ nullおよびObject以外の値が指定された場合 ・ render関数を持たないObjectが指定された場合	可	null	値	不可	不可

注1) 単一選択(multiple=false)の場合にだけ更新可能
注2) 複数選択(multiple=true)の場合にだけ更新可能
注3) Objectの配列の場合は、バインディング式だけ記述可能

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

ポイント

- 単一選択の場合
  - 常に1つだけ選択されている状態です。初期値は、selectedIndexプロパティで指定します。指定がない場合、一番上の項目が選択された状態となります。
  - マウスをクリックして、選択項目を変更できます。
  - ↑↓カーソルキーで、選択項目を移動できます。
- 複数選択の場合
  - 初期値は、selectedIndexプロパティで指定します。指定がない場合、すべて非選択状態となります。
  - マウスをクリックすると、選択項目を追加できます。再度クリックすると、非選択状態に戻ります。
  - ↑↓カーソルキーで、マウスオーバーの項目を移動できます。
  - スペースキーで、マウスオーバーの項目の選択/非選択状態を変更できます。

デフォルトlabelProvider

項目リストの配列(options)のメンバ型によって、以下のように動作が異なります。

メンバ型	説明
String型	引数を加工せずそのまま返却します。
Object型	オブジェクトのlabelプロパティの値をラベル文字列とします。labelプロパティ以外は無視します。(注)

注) Selectのように、個々の項目にenabled/disabledを指定することはできません。

optionsプロパティにObjectの配列を利用する場合の定義例を以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  /**
   * 各オブジェクトには以下のように labelプロパティを定義します。
   * titleプロパティが存在する場合、その項目の補足情報にその値が使用されます。
   *   {
   *    label: "[項目のラベル]",
   *    title: "[項目の補足情報]"
   * }
   **/

   // foo, barというラベルを持つ2つの項目を作成する場合
   var obj = {
     userOptions : [
      { label:"foo", title:"fooタイトル" },
      { label:"bar", title:"barタイトル" }
     ]
   };
//]]>
</script>

<!-- モデル定義 -->
<div rcf:type="Model" rcf:id="model1" rcf:object="obj"></div>
<!-- SelectList定義 -->
<div rcf:type="SelectList" rcf:options="{model1.userOptions}"></div>
```

labelProviderプロパティ

labelProviderプロパティには、getLabel関数を持つオブジェクトを指定します。getLabel関数は、リスト項目の文字列を生成するユーザロジックを記述します。

項目リストの配列(options)メンバがObject(プロパティ:nameとtel)の場合の記述例を、以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  // labelProviderの定義
  var userLabelProvider = {
    getLabel: function(item) {
      return "名前は" + item.name + "です。電話は" + item.tel + "です。";
    }
  };
  // optionsの定義
  var obj = {
    userOptions : [
      { name:"Taro YAMADA",   tel: "xxx-xxx-xxxx"},
      { name:"Hanako TANAKA", tel: "yyy-yyy-yyyy"}
    ]
  };
//]]>
</script>
```

```
<!-- モデル定義 -->
<div rcf:type="Model" rcf:id="model1" rcf:object="obj"></div>
<!-- SelectList定義 -->
<div rcf:type="SelectList" rcf:options="{model1.userOptions}" rcf:labelProvider="userLabelProvider"></div>
```

multipleとselectedIndex、selectedIndexesの関係について

multipleの値によって、selectedIndexおよびselectedIndexesの扱いは以下のようになります。

－ multiple=false(単一選択)の場合

部品の初期化時に、selectedIndexの値を基準にして、selectedIndexesの値が変更されます。
selectedIndexesの初期値設定は無視されます。

selectedIndexの値	変更後のselectedIndexesの値
-1	[](長さ0の配列)
-1以外	selectedIndexの値を持った長さ1の配列

タグに初期値が指定されていた場合も、上記のルールに従って動作します。

タグに指定されたselectedIndexesの設定は無視されます。

selectedIndexesをsetProperty()やモデルバインディングによって変更しようとした場合、エラー(RCF11002)となります。

－ multiple=true(複数選択)の場合

部品の初期化時に、selectedIndexesの値を基準にして、selectedIndexの値が変更されます。
selectedIndexの初期値設定は無視されます。

selectedIndexesの値	変更後のselectedIndexの値
[](長さ0の配列)	-1
長さ1以上の配列	selectedIndexesの末尾の要素

タグに初期値が指定されていた場合も、上記のルールに従って動作します。

タグに指定されたselectedIndexの設定は無視されます。

selectedIndexをsetProperty()やモデルバインディングによって変更しようとした場合、エラー(RCF11002)となります。

rendererプロパティ

rendererプロパティには、render関数を持つオブジェクトを指定します。render関数には、リスト項目を表示する領域のDOMノードと、そのほかの項目情報がパラメタとして渡されます。render関数内のユーザロジックでDOMノードを操作することで、リスト項目の表示を制御できます。

render関数のパラメタを以下に示します。

順	パラメタ名	型	説明
1	node	DOM Node	項目リストを表示する領域のDOMノードを指定します。
2	item	Object	options配列のメンバを指定します。
3	label	String	itemをlabelProviderで変換した文字列です。
4	index	Number	options配列の中のindexを指定します。

rendererプロパティに渡すオブジェクトの例を以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  var user_Renderer = {
    /**
     * param node 項目を表示する領域のDOMノード
```

```
     * param item(Object型またはString型) options配列のn番目要素
     * param label(String型) options配列のn番目要素をlabelProviderにより変換した結果の文字列
     * param index(Number型)  n番目
     */
    render: function(node, item, label, index) {
      //
      //以下ユーザ定義ロジック
      //
      node.innerHTML = '<img src="img.png" />' ;
      node.appendChild(document.createTextNode(label));
    }
  };
//]]>
</script>

<div rcf:type="SelectList" rcf:renderer="user_Renderer"></div>
```

スタイルプロパティ

部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-SelectList	• サイズ • カラー • フォント(lineHeightを除く) • テキスト(textIndent、textAlign、whiteSpaceを除く) • ボーダー • パディング

詳細は、“2.9 スタイルプロパティ”を参照してください。

選択項目のスタイル

プレフィックス	クラス	スタイルプロパティ	説明	デフォルト
optionSelected (選択項目)	rcf-SelectList-optionSelected	backgroundColor	選択項目の背景色を指定します。	#004E98
		color	選択項目のフォント色を指定します。	#FFFFFF (白)
optionHovered (マウスオーバー項目)	rcf-SelectList-optionHovered	backgroundColor	マウスオーバー項目の背景色を指定します。	#316AC5
		color	マウスオーバー項目のフォント色を指定します。	#FFFFFF (白)

表示とスタイルプロパティの関係

本部品のスタイルプロパティは、CheckListと同じです。ただし、チェックボックスの表示はありません。詳細は、“2.1.14 CheckList”の“スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	

名前	説明	イベントオブジェクト
onChange (注)	部品の選択状態が変更されたときに呼ばれます。	
onOptionSelected (注)	項目が選択されたときに呼ばれます。	OptionStateChangeEvent
onOptionDeselected (注)	項目の選択が解除されたときに呼ばれます。 単一選択の場合、optiondeselected、optionselectedの順番でイベントが発生します。	

注) マウス、キーボードなどの入力装置によってUI部品を操作した場合に呼ばれるイベントリスナ

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

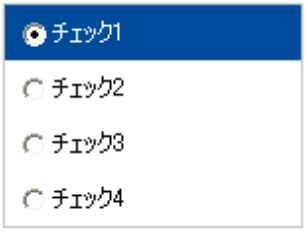
2.1.14 CheckList

CheckListは、チェックボタンをリスト表示する部品です。チェックボタンは、単一選択および複数選択できます。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

表示例

図2.9 単一選択モード



単一選択モードの場合は、ラジオボタンが表示されます。

図2.10 複数選択モード

複数選択モードの場合は、チェックボックスが表示されます。

記述形式

```
<div rcf:type="CheckList" ... ></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は前後に改行コードが挿入されて表示されます。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
options	Array	表示される項目のリストを指定します。 表示されるラベルは、labelProviderプロパティの指定に従います。	可	[]	値、バインド(注3)	可	可
labelProvider	getLabel関数を持つObject	各選択項目の文字列を確定する関数を指定します。options配列のメンバに応じた文字列変換ロジックを定義します。getLabel関数には、options配列の個々の要素が渡されます。 nullを指定した場合、デフォルトlabelProviderが使用されます。詳細は、“デフォルトlabelProvider”を参照してください。 以下の場合はエラーとなります。 ・ nullおよびObject以外の値が指定された場合 ・ getLabel関数を持たないObjectが指定された場合	可	null(デフォルトlabelProvider)	値	不可	不可
multiple	Boolean	複数選択または単一選択を指定します。 ・ true:複数選択 ・ false:単一選択	可	false	値	不可	不可
selectedIndex	Number	単一選択の場合 ・ 選択されている項目のインデックス(先頭は0)を指定します。 ・ 選択状態の初期値を指定できます。 ・ -1を指定した場合、選択が解除されます。	可	-1	値、バインド	可(注1)	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・無効な値が指定された場合は、エラー(RCF13402)となります。 複数選択の場合 ・最後に選択された項目のインデックスを指定します。 ・部品の初期化後に本プロパティを変更しようとすると、エラー(RCF11002)となります。“multipleとselectedIndex、selectedIndexesの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは-1(選択なし)に設定されます。					
selectedIndexes	NumberのArray	選択されている項目のインデックス(先頭は0)配列を指定します。選択順にインデックスが格納されます。 選択状態の初期値を指定できます。 何も選択されていない場合は、長さ0の配列となります。 無効な値が指定された場合は、エラー(RCF13402)となります。 単一選択(multiple=false)の場合、部品の初期化後に本プロパティを変更しようとするとエラー(RCF11002)となります。“multipleとselectedIndex、selectedIndexesの関係について”を参照してください。 optionsプロパティの値が変更された場合、本プロパティは長さ0の配列(選択なし)に設定されます。	可	[]	値、バインド	可(注2)	可(注2)
title	String	optionsプロパティがStringまたはObjectの配列で、かつ各Objectがtitle属性でない場合に、ツールチップで表示されるテキストを指定します。 なお、optionsプロパティがObjectの配列で、かつObjectがtitle属性の場合は、その値が項目のツールチップとして表示されます。	可	""	値	可	不可
enabled	Boolean	本部品の有効/無効を指定します。 ・true:有効 ・false:無効	可	true	値、バインド	可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。	可	0	値	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		FocusManagerによるフォーカス移動には関係しません。					

注1) 単一選択(multiple=false)の場合にだけ更新可能
注2) 複数選択(multiple=true)の場合にだけ更新可能
注3) Objectの配列の場合は、バインディング式だけ記述可能

部品共通のプロパティも指定できます。詳細は、"2.8.1 画面部品共通プロパティ"を参照してください。

注意

titleプロパティに長い文字列を指定すると、自動的に改行または省略されることがあります。
また、発生位置はブラウザにより異なります。

なお、空文字列を指定した場合は、表示されません。

ポイント

- 選択状態の初期値は、selectedIndexプロパティで指定します。指定がない場合、すべて非選択状態となります。
- 単一選択の場合
  ↑↓カーソルキーで、選択項目を移動できます。
- 複数選択の場合
  ↑↓カーソルキーで、フォーカスを当てる項目を移動できます。

スタイルプロパティ

部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-CheckList	• サイズ • カラー • フォント(lineHeightを除く) • テキスト(textIndent、textAlign、whiteSpaceを除く) • ボーダー • パディング

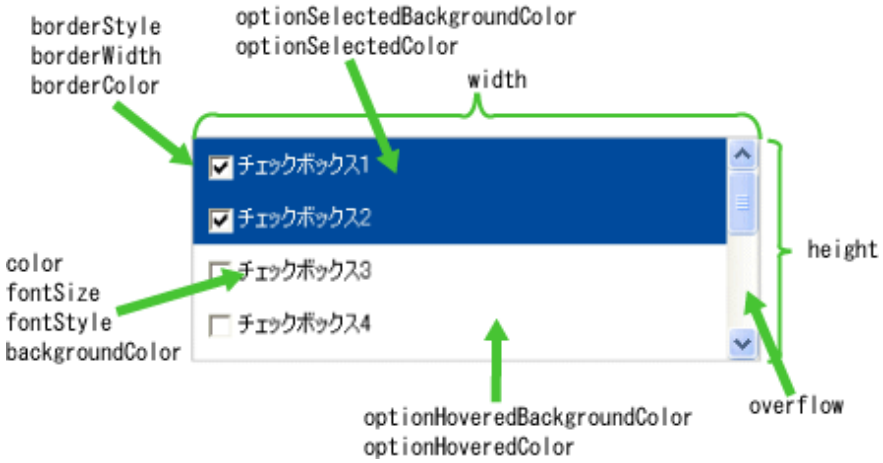
詳細は、"2.9 スタイルプロパティ"を参照してください。

選択項目のスタイル

プレフィックス	クラス	スタイルプロパティ	説明	デフォルト
optionSelected(選択項目)	rcf-CheckList-optionSelected	backgroundColor	選択項目の背景色を指定します。	#004E98
		color	選択項目のフォント色を指定します。	#FFFFFF(白)
optionHovered(マウスオーバー項目)	rcf-CheckList-optionHovered	backgroundColor	マウスオーバー項目の背景色を指定します。	#316AC5

プレフィックス	クラス	スタイルプロパティ	説明	デフォルト
		color	マウスオーバー項目のフォント色を指定します。	#FFFFFF(白)

図2.11 表示とスタイルプロパティの関係



表示領域よりも選択項目のサイズが大きくなった場合、以下のように表示されます。

- 横方向に大きくなった場合:
  表示領域を超えた部分は隠されます。(横スクロールバーは表示されません。)
- 縦方向に大きくなった場合:
  スクロールバーの操作によって全体を表示することができます。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	
onChange (注)	部品の選択状態が変更されたときに呼ばれます。	OptionStateChangeEvent
onOptionSelected (注)	項目が選択されたときに呼ばれます。	
onOptionDeselected (注)	項目の選択が解除されたときに呼ばれます。 単一選択の場合、optiondeselected、optionselectedの順番でイベントが発生します。	

注) マウス、キーボードなどの入力装置によってUI部品を操作した場合に呼ばれるイベントリスナ

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

- **Tabキーによるフォーカス移動について** IE6 IE7 IE8

  Internet Explorerで単一選択の場合、Tabキーによるフォーカス移動を行うと、部品内のラジオボタンにフォーカスが移動します。このとき、blurイベントが発生します。
  次の部品にフォーカスを移動するには、もう一度Tabキーを押下してください。

- **部品上のラジオボタン、チェックボックスをクリックした場合の動作について**

  CheckList上のラジオボタンまたはチェックボックスをクリックした場合、CheckList部品に対してフォーカスを設定します。
  すでにCheckListにフォーカスが設定されている状態でこの操作を行った場合、CheckList部品からフォーカスが外れてラジオボタンやチェックボックスに移動したあと、CheckList部品に再設定されます。このとき、blur→focusの順で、イベントが発生します。
  またInternet Explorerの場合、CheckList部品の左上隅が表示される位置にブラウザがスクロールします。

2.2 コンテナ部品

コンテナ部品は、複数の画面部品を配置することができる部品です。

ここでは、コンテナ部品の設定内容、および設定方法について説明します。

注意

- コンテナ部品内に、子要素としてHTML要素およびUI部品を配置する場合、Ajaxフレームワークの動作定義によって基準位置が変わります。詳細は、“1.3.7 コンテナ部品内のレイアウト”を参照してください。
- Ajaxフレームワークの動作定義でbasePoint_inContainerがtrueの場合、コンテナ部品には必ずスタイルプロパティでサイズを指定してください。
  コンテナ部品にサイズを指定しないでボーダーやパディングを指定すると、UI部品およびHTML要素の表示がずれたり、FragmentContainerのフラグメントHTMLが正しく表示されない場合があります。
  Ajaxフレームワークの動作定義の詳細は、“ユーザーズガイド”を参照してください。

2.2.1 ViewContainer

ViewContainerは、画面情報(コンテナ)を指定する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

ViewContainerは、子要素にHTML要素を持つことができ、HTML要素をまとめて部品として扱うことができます。ViewStackまたはTabPanelで、切り替える画面の単位を指定するために使用します。

記述形式

```
<div rcf:type="ViewContainer" ... >
    HTML要素
</div>
```

ポイント

子要素には、HTML要素およびUI部品を指定できます。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
label	String	タブパネル子要素として利用する場合のタブ名を指定します。	可	""	値、バインド	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-ViewContainer	• サイズ • カラー • ボーダー • パディング • オーバーフロー

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

ViewContainerは、ViewStackやTabPanelと組み合わせて使用する部品です。
ViewContainerを単体で使用した場合、通常の 要素と同様に描画されます。

2.2.2 Panel

Panelは、タイトル部とボディ部でできている画面情報(コンテナ)を指定する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

表示例

記述形式

```
<div rcf:type="Panel" ... >
    HTML要素
</div>
```

ポイント

子要素には、HTML要素およびUI部品を指定できます。子要素に定義されたHTML要素は、ボディ部の内容になります。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
label	String	タイトル部のラベルテキストを指定します。	可	""	値、バインド	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-Panel	• サイズ(注) • カラー • ボーダー • パディング
タイトル部	title	rcf-Panel-title	• サイズ(widthを除く) • カラー • フォント • テキスト(white-spaceを除く) • ボーダー • パディング • マージン
ボディ部	body	rcf-Panel-body	• カラー

パーツ名	プレフィックス	クラス名	使用可能なスタイル
			• フォント • テキスト • ボーダー • パディング • オーバーフロー

注) widthのデフォルトは300pxです。

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

部品全体のイベントリスナ

部品共通のイベントリスナを利用できます。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

タイトル部のイベントリスナ

名前	説明	イベントオブジェクト
onClickTitle	タイトル部でマウスによりクリックされたときに呼ばれます。	ActionEvent
onDblClickTitle	タイトル部でマウスによりダブルクリックされたときに呼ばれます。	
onMouseDownTitle	タイトル部でマウスボタンが押し下げられたときに呼ばれます。	
onMouseUpTitle	タイトル部でマウスボタンが離されたときに呼ばれます。	
onMouseOverTitle	マウスがタイトル部の上に重ねられたときに呼ばれます。	
onMouseOutTitle	マウスがタイトル部の上から離れたときに呼ばれます。	
onMouseMoveTitle	マウスがタイトル部の上で動かされたときに呼ばれます。	

ボディ部のイベントリスナ

名前	説明	イベントオブジェクト
onClickBody	ボディ部でマウスによりクリックされたときに呼ばれます。	ActionEvent
onDblClickBody	ボディ部でマウスによりダブルクリックされたときに呼ばれます。	
onMouseDownBody	ボディ部でマウスボタンが押し下げられたときに呼ばれます。	
onMouseUpBody	ボディ部でマウスボタンが離されたときに呼ばれます。	
onMouseOverBody	マウスがボディ部の上に重ねられたときに呼ばれます。	
onMouseOutBody	マウスがボディ部の上から離れたときに呼ばれます。	
onMouseMoveBody	マウスがボディ部の上で動かされたときに呼ばれます。	
onKeyPressBody	ボディ部内のHTML要素上でキーが押されて離されたときに呼ばれます。	
onKeyDownBody	ボディ部内のHTML要素上でキーが押し下げされたときに呼ばれます。	
onKeyUpBody	ボディ部内のHTML要素上でキーが離されたときに呼ばれます。	

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

Panelのサイズ

Panelのサイズについては、以下の注意事項があります。

- 幅(width)と高さ(height)は、スタイルプロパティで指定してください。CSSで指定した場合、表示が崩れる可能性があります。
- 幅(width)と高さ(height)を設定する際の単位は、“px”にだけ対応しています。そのほかの単位による指定およびパーセント値(%)による指定をした場合、表示が崩れる可能性があります。
- ボディ部に100px×100pxの要素を配置できるサイズがPanelの最小サイズとなります。これより小さいサイズに変更しようとした場合、最小サイズに設定されます。

フォーカスの設定

- Firefox2 Firefox3
  部品にフォーカスを当てることはできません。
- IE 6 IE 7 IE 8
  部品、またはボディ部をクリックすることにより、部品全体やボディ部にフォーカスが設定されます。フォーカスが設定されている状態でキーを入力すると、対応するキーイベントが発生します。

その他

- IE 6
  Panel部品の内部に表示する内容(タイトル部含む)のサイズが部品全体のサイズを超える場合、部品の初期化後にPanelのサイズが強制的に拡張されます。表示する内容が収まるように部品全体のサイズを設定するか、初期化後にサイズを再設定してください。
- Firefox2 Firefox3
  ボディ部のoverflowに“auto”を指定し、Panelのサイズを指定しなかった場合、ボディ部に不要なスクロールバーが表示されることがあります。Panelのサイズを指定することにより、回避してください。

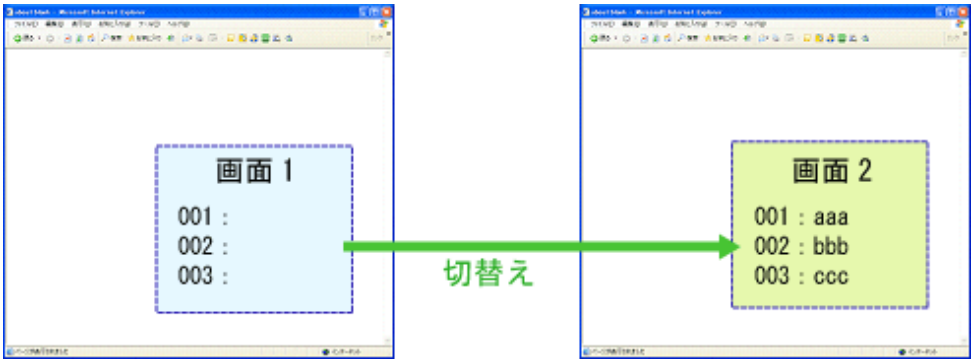
2.2.3 ViewStack

ViewStackは、切替え対象の画面情報(コンテナ)をグループ化する部品です。画面の同じ位置で表示を切り替える場合に利用します。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

コンテナには、ViewContainerまたはFragmentContainerを利用できます。各画面の情報(HTMLコンテンツ)は、コンテナの子要素として記述します。

図2.12 動作イメージ



記述形式

```
<div rcf:type="ViewStack" ... >
  <div rcf:type="ViewContainer">
    HTML要素
  </div>
  <div rcf:type="FragmentContainer" rcf:src="fragment.jsp">
  </div>
  ...
</div>
```

ポイント

- 子要素は、1個以上記述してください。
- 子要素には、ViewContainerまたはFragmentContainerを記述します。
- 子要素にViewContainerおよびFragmentContainer以外のものを記述しても、表示することはできません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
selectedIndex	Number	選択されている画面のインデックスを指定します。 インデックスは、ViewStack子要素として定義されているコンテナの出現順に、0から1ずつ増分で自動採番されます。 存在しないインデックスを指定した場合は、すべてのコンテナが不可視(visibility:hidden; display:none)になります。	可	0	値、バインド	可	不可
selectedIndexLock	Boolean	画面切替えを抑止するかどうかを指定します。 ・ true: 抑止する ・ false: 抑止しない trueを指定した場合、以下による画面切替えは行われません。	可	false	値	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		• selectedIndexプロパティの変更 • selectNextメソッド • selectPreviousメソッド • setSelectedIndexメソッド					

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-ViewStack	• サイズ • カラー • ボーダー • パディング • オーバーフロー

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onSelectedIndexChange	選択されている画面のインデックスが変更されたときに呼ばれます。	SelectedIndexChangeEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

■selectNextメソッド

メソッド	selectNext()
戻り値	なし
例外	なし
説明	次の画面に切り替えます。最後の画面の場合は切り替わりません。

■selectPreviousメソッド

メソッド	selectPrevious()
戻り値	なし
例外	なし
説明	前の画面に切り替えます。最初の画面の場合は切り替わりません。

■setSelectedIndexメソッド

メソッド	setSelectedIndex(index)	
引数	index [Number]	画面のインデックス

戻り値	なし
例外	なし
説明	指定された番号の画面に切り替えます。指定された番号の画面がない場合は、切り替わりません。

■setSelectedContainerIdメソッド

メソッド	setSelectedContainerId(id)	
引数	id [String]	ViewContainerまたはFragmentContainerの部品ID
戻り値	なし	
例外	なし	
説明	指定されたViewContainerまたはFragmentContainerの内容で、ViewStackの表示を切り替えます。なお、FragmentContainerの部品IDを指定する場合は、対象となるFragmentContainerが有効化された状態である必要があります。	

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

ViewStack部品の内部に表示する内容(タイトル部含む)のサイズが、部品全体のサイズを超える場合、部品の描画が崩れる可能性があります。
表示する内容が収まるように、部品全体のサイズを設定してください。

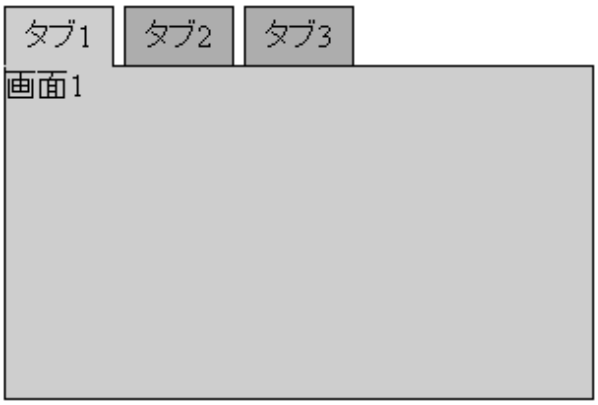
2.2.4 TabPanel

TabPanelは、タブによる切替え対象の画面情報(コンテナ)をグループ化する部品です。タブで画面の表示を切り替える場合に利用します。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

タブやタブ内のフォーム部品にフォーカスがある場合、キー操作でタブを切り替えることができます。また、タブ切替えキーをカスタマイズすることもできます。

表示例



記述形式

```
<div rcf:type="TabPanel" ... >
  <div rcf:type="ViewContainer" rcf:label="タブ1">
    HTML要素
  </div>
  <div rcf:type="FragmentContainer" rcf:label="タブ2">
  </div>
  ...
</div>
```

ポイント

- 子要素は1個以上記述してください。
- 子要素には、切り替える画面単位にコンテナ要素を記述します。コンテナ要素として、ViewContainerとFragmentContainerが利用できます。
- 子要素にViewContainerおよびFragmentContainer以外のものを記述しても、表示することはできません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
selectedIndex	Number	選択されている画面のインデックスを指定します。 インデックスは、TabPanel子要素として定義されているコンテナの出現順に、0から1ずつ増分で自動採番されます。 存在しないインデックスを指定した場合は、すべてのコンテナが不可視(visibility:hidden; display:none)になります。	可	0	値、バインド	可	不可
tabPosition	String	タブ表示位置を指定します。	可	TOP_LEFT	値	不可	不可
nextKey	String	次のタブに切り替えるキーを指定します。	可	Ctrl+→カーソ	値	不可	不可

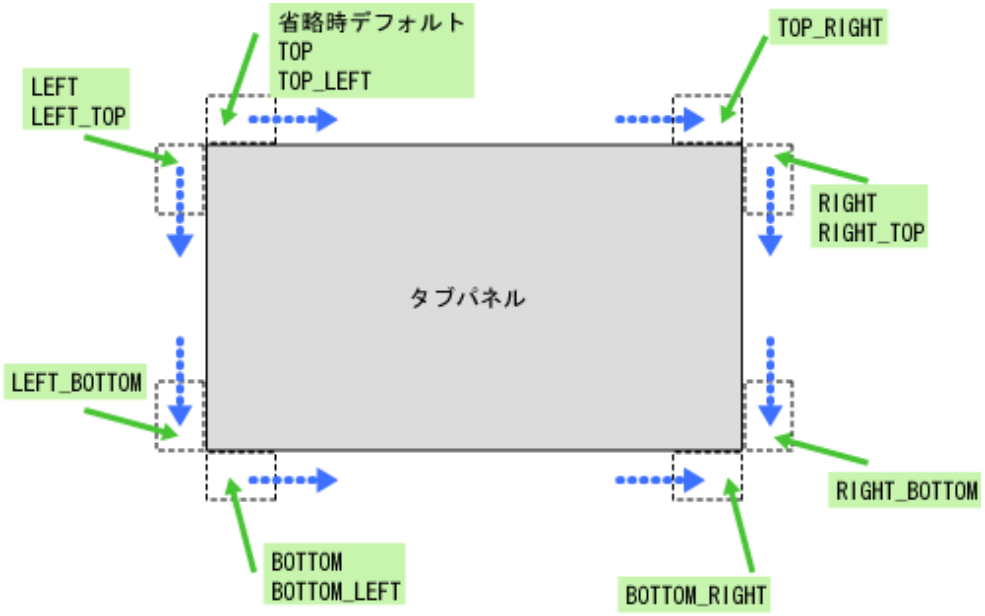
名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		“キーの指定”を参照してください。		ルキー("39+CTRL")			
previousKey	String	前のタブに切り替えるキーを指定します。 “キーの指定”を参照してください。	可	Ctrl+←カーソルキー("37+CTRL")	値	不可	不可
tabSeparator	Boolean	タブの間に隙間をあけるかどうかを指定します。 ・ true:隙間あり ・ false:隙間なし	可	true	値	不可	不可
tabRenderer	render関数を持つオブジェクト	タブ内部のレンダリングを行うオブジェクトを指定します。	可	null	値	不可	不可
tabIndex	Number	TABキーで移動するフォーカスの順番を指定します。 フォーカスは、タブ部に対して設定されます。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には、関係しません。	可	0	値	可	不可
selectedIndexLock	Boolean	画面切替えを抑止するかどうかを指定します。 ・ true: 抑止する ・ false: 抑止しない trueを指定した場合、以下による画面切替えは行われません。 ・ マウス操作 ・ selectedIndexプロパティの変更 ・ nextKeyプロパティに設定されているキー ・ previousKeyプロパティに設定されているキー ・ selectNextメソッド ・ selectPreviousメソッド	可	false	値	可	不可

ViewContainerのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.2.1 ViewContainer”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

タブの基準位置の指定

tabPositionプロパティで、1番目タブの表示位置と2番目以降のタブの追加方向を指定できます。

- 大文字/小文字を区別しません。
- 青色の破線矢印は、2番目以降のタブの追加方向を示します。
- タブ内のラベル文字は、タブ表示位置にかかわらず、左から右になります。



キーの指定

nextKeyプロパティ、previousKeyプロパティでは、選択するタブを切り替えるキーを指定します。
キーには、“+”を区切り文字として、以下を指定できます。

- 数字または別名:キーコード
- ALT:Altキー
- CTRL:Ctrlキー
- SHIFT:Shiftキー

キーコードは必須で、最初に指定する必要があります。

指定例を以下に示します。

指定例	内容
13	Enterキー
13 + SHIFT	Enterキー + Shiftキー
13 + CTRL + SHIFT	Enterキー + Ctrlキー + Shiftキー
SHIFT	キーコードが指定されていなので、設定不可
CTRL+13	キーコードが最初に指定されていないので、設定不可

キーコードには、数字の直接指定のほか、以下の別名を指定することができます。
別名による指定は、その別名が表すキーの実際のキーコードを指定した場合と同様に動作します。
例えば、“ENTER+SHIFT”は、“13+SHIFT”と同じ値として扱います。

別名	対象となるキー	実際のキーコード
BACKSPACE	Backspaceキー	8
TAB	Tabキー	9
ENTER	Enterキー	13
ESC	Escキー	27
SPACE	スペースキー	32
PAGEUP	PageUpキー	33
PAGEDOWN	PageDownキー	34
END	Endキー	35
HOME	Homeキー	36
LEFT	←(左カーソルキー)	37
UP	↑(上カーソルキー)	38
RIGHT	→(右カーソルキー)	39
DOWN	↓(下カーソルキー)	40
INSERT	Insertキー	45
DELETE	Deleteキー	46
F1	F1キー	112
F2	F2キー	113
F3	F3キー	114
F4	F4キー	115
F5	F5キー	116
F6	F6キー	117
F7	F7キー	118
F8	F8キー	119
F9	F9キー	120
F10	F10キー	121
F11	F11キー	122
F12	F12キー	123

tabRendererプロパティ

tabRendererプロパティには、render関数を持つオブジェクトを指定します。

render関数は、TabPanelの初期描画時、および選択タブ変更時に、個々のタブについて呼び出されます。
render関数にタブをレンダリングするためのユーザロジックを記述することより、個々のタブに表示する文字やスタイルをカスタマイズしたり画像を貼り付けたりすることができます。

TabPanelのタブ部分は、TDノードにより実装されています。render関数には、レンダリング対象のタブに対応するTDのDOMノードおよび補足情報がパラメタとして渡されます。
渡されるTDのDOMノードには、子要素としてタブに表示する文字列(TabPanelの子要素のコンテナ部品のlabel属性の値)が設定されています。

render関数のパラメタを以下に示します。

順	パラメタ名:型	説明
1	node: Node	レンダリング対象タブの内部構造を扱うTD DOMノードが渡されます。

順	パラメタ名:型	説明
		このTDノード配下を参照および変更できます。ただし、このTDノードより上の階層のノードは参照および更新できません。
2	x: Number	タブのインデックスが渡されます。開始値は0です。
3	side: String	タブエリアの表示位置が渡されます。 ("TOP" | "RIGHT" | "LEFT" | "BOTTOM")
4	selected: Number	選択されているタブのインデックスが渡されます。(xとselectedが同じ値の場合は、渡されたタブが選択されています。)

tabRenderer の設定例を以下に示します。(タブの表示内容を“N番目のタブ”に変更する例)

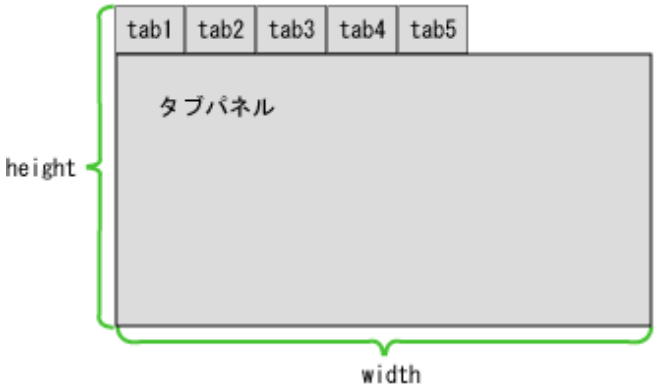
```
<script type="text/javascript">
//<![CDATA[
  var tr1 = {
    render: function(node, x, side, selected) {
                    var text = document.createTextNode((x+1) + "番目のタブ");
      node.innerHTML = "";
                      node.appendChild(text);
    }
  };
//]]>
</script>

<div rcf:type="TabPanel" rcf:tabRenderer="tr1">
...
</div>
```

スタイルプロパティ

幅と高さ

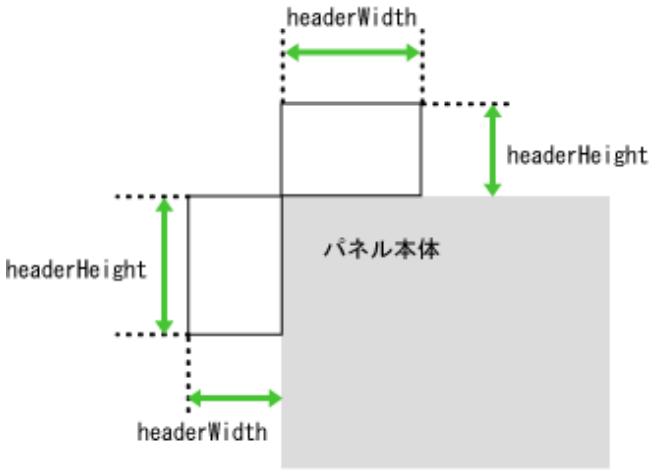
TabPanelの幅と高さは、heightプロパティおよびwidthプロパティで指定します。各プロパティは、タブの高さ(幅)を含みます。

デフォルトでは、widthが100%、heightがautoとなっています。
heightがautoの場合、タブパネルの高さは、パネルに表示されるViewContainerまたはFragmentContainerの高さに合わせて変動します。



タブのサイズ指定

タブの位置に関わらず、headerWidthプロパティおよびheaderHeightプロパティの省略値はautoです。

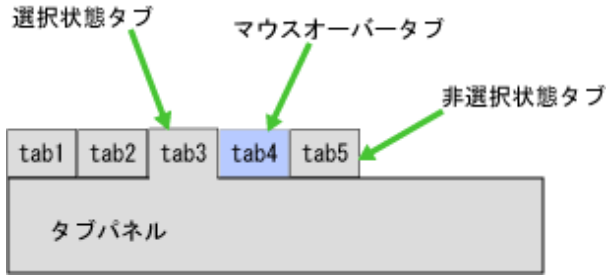


タブを左右に配置する場合、タブの幅はタブに表示する最長の文字列に依存します。

タブを上下に配置する場合、タブの幅は表示する文字列に依存して個々に異なります。

全体の装飾プロパティ

以下の図に示します。



スタイルプロパティ

部品名	プレフィックス	クラス名	スタイル分類	説明
全体	なし	rcf-TabPanel	• サイズ	部品全体のサイズ
ヘッダ	header	rcf-TabPanel-header	• サイズ	タブ部分のサイズ タブ位置により heightまたはwidth が指定可能
ボディ	body	rcf-TabPanel-body	• カラー • ボーダー • パディング	ボディ部分
タブ	tab	rcf-TabPanel-tab	• カラー • フォント • テキスト • ボーダー • パディング	個々のタブのスタイル
選択状態タブ	tabSelected	rcf-TabPanel-tabSelected	• カラー • フォント • テキスト	選択されているタブのスタイル

部品名	プレフィックス	クラス名	スタイル分類	説明
			• ボーダー • パディング	
マウスオーバータブ	tabHovered	rcf-TabPanel-tabHovered	• カラー • フォント • テキスト • ボーダー • パディング	マウス位置にあるタブのスタイル
セパレータ	tabSeparator	rcf-TabPanel-tabSeparator	• サイズ • ボーダー	タブ間の幅を指定 TOP/BOTTOMではwidth LEFT/RIGHTではheight
ブランク	tabBlank	rcf-TabPanel-tabBlank	• ボーダー • パディング	タブのある辺のタブ部分以外のスタイル

詳細は、“2.9 スタイルプロパティ”を参照してください。

選択タブとマウスオーバータブが重なった場合

選択タブとマウスオーバータブが重なった場合は、タブのスタイルクラスに、tabSelected、tabHoveredの順に宣言します。

CSSでだけ指定可能なスタイル

tab、tabSelected、tabHovered、tabSeparator、tabBlankは、CSSでだけ指定できます。

カスタマイズ上の注意

ボディとタブのデフォルトのスタイル設定は、タブの位置に依存して、上下左右で異なるため、カスタマイズの際は注意が必要です。デフォルトのスタイル設定は、以下のとおりです。

ー ボディ

- タブのある辺: border-style: none
- タブのない3辺: border-style: solid

ー タブ

- 4辺とも border-color: #000
  ただし、タブが選択状態でかつマウスがタブ上にない場合は、ボディに接する辺のボーダーの色がタブのbackground-colorに合わせられます。
  カスタマイズで背景色を変えた場合も同様です。

イベントリスナ

タブ部分に関しては、以下のイベントリスナが使用できます。

名前	説明	イベントオブジェクト
onClickHeader	タブ部をマウスでクリックしたときに呼ばれます。	ActionEvent
onDblClickHeader	タブ部をマウスでダブルクリックしたときに呼ばれます。	
onMouseDownHeader	タブ部をマウスで押したときに呼ばれます。	
onMouseUpHeader	タブ部でマウスを離したときに呼ばれます。	

名前	説明	イベントオブジェクト
onMouseOverHeader	タブ部の上にマウスを重ねたときに呼ばれます。	
onMouseOutHeader	タブ部の上からマウスが外れたときに呼ばれます。	
onMouseMoveHeader	タブ部の上でマウスを動かしたときに呼ばれます。	
onKeyPressHeader	タブ部の上でキーを押して離したときに呼ばれます。	
onKeyDownHeader	タブ部の上でキーを押したときに呼ばれます。	
onKeyUpHeader	タブ部の上でキーを離したときに呼ばれます。	
onSelectedIndexChange	選択している画面のインデックスが変更されたときに呼ばれます。	SelectedIndexChangeEvent

パネル本体部に関しては、部品共通のイベントリスナが使用できます。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■selectNextメソッド

メソッド	selectNext()
戻り値	なし
例外	なし
説明	次のタブに切り替えます。最後のタブの場合は切り替わりません。

■selectPreviousメソッド

メソッド	selectPrevious()
戻り値	なし
例外	なし
説明	前のタブに切り替えます。最初のタブの場合は切り替わりません。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

TabPanelのサイズについて

- スタイルのカスタマイズにおいて、サイズの単位は“px”にだけ対応しています。ボーダーの幅やマージン、パディングに関しても同様です。そのほかの単位による指定をした場合、表示が崩れる可能性があります。
- TabPanel部品の内部に表示する内容(タイトル部含む)のサイズが、部品全体のサイズを超える場合、部品の描画が崩れる可能性があります。
  表示する内容が収まるように、部品全体のサイズを設定してください。

フォーカスについて

TABキーまたはFocusManagerで、TabPanelにフォーカスを移動させた場合は、ヘッダ部分にフォーカスが当たった状態になります。また、nextKeyおよびpreviousKeyで指定したキーでタブを切り替えたあとも、ヘッダ部分にフォーカスが当たります。

ヘッダ部分へのフォーカスは、個々のタブ単位ではなく、ヘッダ部分全体に対してフォーカスが当たった状態になります。

ボディ内に記述されている部品にフォーカスが当たっている場合、nextKeyおよびpreviousKeyによるタブ切替えは可能ですが、TABキーやFocusManagerでのフォーカス移動は、TabPanelではなく、フォーカスが直接当たっている部品が対象になります。

タブ切替えキー(nextKey、previousKey)の設定について

- グローバルイベント制御を行っているキーがタブ切替えキーに指定された場合、予期しない動作をする場合があります。タブ切替えキーには、それらと干渉しないキーを設定してください。
- ブラウザのショートカットキーなどと一致するキーがタブ切替えキーに指定された場合、タブ切替え処理が優先され、元々の処理がキャンセルされる場合があります。タブ切替えキーには、有効にしたいショートカットなどと干渉しないキーを設定してください。

  例)
  “66+CTRL(Ctrl + B)”をタブ切替えキーに指定した場合、機能付加対象の部品でCtrl + Bを入力しても、以下のブラウザの標準動作が行われなくなります。

  - IE 6 IE 7 IE 8
    お気に入りの整理
  - Firefox2 Firefox3
    ブックマークの表示
- 数字や文字をタブ切替えキーに割り当てる場合は、CTRLまたはALTと組み合わせて、指定してください。数字や文字のキーを単独でタブ切替えキーに割り当てた場合、IMEが有効になっているとキーコードを判別できないため、タブの切替えが正常に行われません。

<h1>～<h6>タグおよび<p>タグについて Firefox2 Firefox3 IE 8 標準

ViewContainerのHTML要素の先頭またはFragmentContainerのフラグメントHTMLの先頭に、<h1>～<h6>タグおよび<p>タグがあると、描画が崩れる場合があります。

2.2.5 FragmentContainer

FragmentContainerは、画面情報(コンテナ)を別ファイルで指定する部品です。

ViewContainerは子要素として同一ファイル内にHTML要素を記述するのに対して、FragmentContainerは別ファイルに記述します。

FragmentContainer内のHTML(以降、フラグメントHTMLと呼びます)は、activateメソッドを呼び出したタイミングで非同期通信によるデータの取得処理が開始されます。ここでは、activateメソッドまたはreloadメソッドを実行してFragmentContainer内の部品が利用可能な状態となることを有効化(可視化)と呼び、また、activateメソッドの実行以前の状態、またはunloadメソッドにより部品が利用不可の状態になることを無効化と呼びます。なお、フラグメントHTMLは、単体のHTMLとしてDTDに適合していません。
取得後のフラグメントHTMLは、FragmentContainerを記述したHTML(以降、コンテナHTMLと呼びます)に挿入されて、データが反映されます。

用途の例を以下に示します。

- 初期化処理の高速化
  TabPanelやViewStackと組み合わせて、コンテナの内容を遅延読込みさせます。
- 分散開発
  開発資産(ファイル)が分割されるので、分散して開発できます。

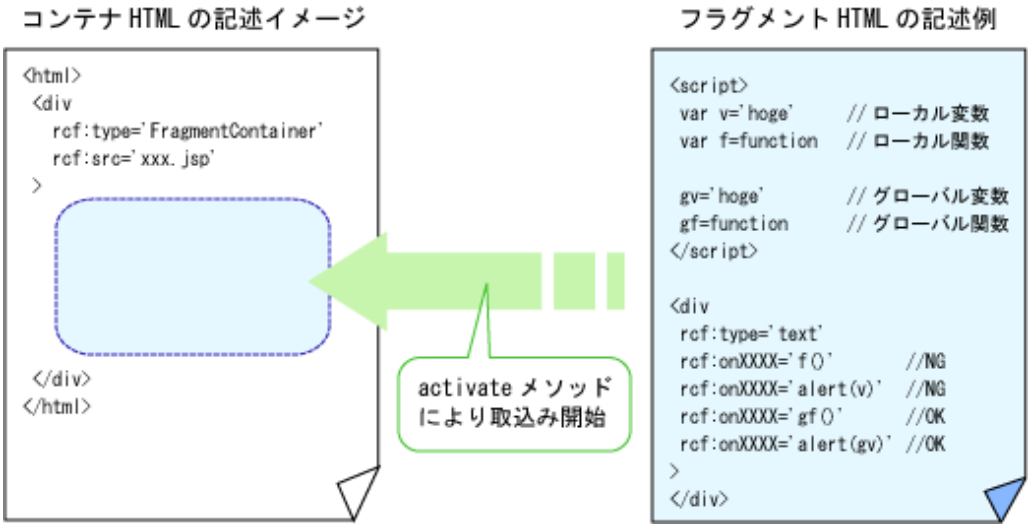
記述形式

```
<div rcf:type="FragmentContainer" rcf:src="fragment.jsp" ... ></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

図2.13 FragmentContainerの利用イメージ



フラグメントHTMLは、通常のHTMLと同様に、UI部品、<script>タグを利用したJavaScript変数、JavaScript関数などを利用できます。

コンテナHTMLとフラグメントHTML間の変数名および関数名の参照可否を、以下に示します。

		参照先(TO)			
		コンテナHTML内		フラグメントHTML内	
		グローバル	ローカル	グローバル	ローカル
参照元 (FROM)	コンテナHTML内	可	可	可	不可
	フラグメントHTML内	可	可	可	不可

- フラグメントHTMLのイベントハンドラ(rcf:onXXXXや通常のonXXX)は、フラグメントHTML内の変数名(関数名)のローカル変数(ローカル関数)を参照できません。この場合、未定義変数(関数)を参照した場合と同じ動作になります。
  - ローカル変数(関数)の例
    var i = 1 (var f = func(..){..})
  - グローバル変数(関数)の例
    i = 1 (f = func(..){..})
- フラグメントHTML内のグローバル変数(関数)は、コンテナHTMLからも参照できます。このため、複数のフラグメントHTMLを利用する場合、名前の管理に注意してください。
  フラグメントHTML、コンテナHTMLで変数名(関数名)が一意になるようにしてください。
- rcf:widthおよびrcf:heightが指定された場合、activate前から画面領域として指定されたサイズを確保します。
- フラグメントHTMLの構造をJSPで作成することもできます。ただし、JSP拡張タグおよびJSPスクリプトレットと混在させることはできません。混在した場合の動作は保証されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
src	String	フラグメントHTMLを取得するURLを指定します。 ・ HTTPメソッドはGET固定です。 ・ コンテナ側HTMLとフラグメントHTMLを提供するサーバは必ず同じにします。 ・ URLに、日本語などの2バイト文字を含めることはできません。 ・ URLには、クエリ文字列およびURLリライティングで用いるセションIDを付加することができます。詳細は、“ユーザーズガイド”を参照してください。	不可	ー	値	不可	不可
showStatus	Boolean	activate前のFragmentContainerの状態(初期化直後、ロード中、エラーなど)の表示/非表示を指定します。 ・ true:表示する ・ false:表示しない	可	true	値	不可	不可
statusIconImage	Object	正常系activate前の状態、および異常系状態のアイコンイメージを指定します。詳細は、“statusIconImageプロパティ”を参照してください。	可	標準の画像	値	不可	不可
timeout	Number	フラグメントHTMLの取得開始から取得完了までの最大待ち時間を指定します。単位は、ミリ秒です。 1より小さい数値を指定した場合は、エラーとなります。 タイムアウトの場合は、FragmentErrorEventが発生します。	可	60000(60秒)	値	不可	不可

ViewContainerのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.2.1 ViewContainer”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

statusIconImageプロパティ

statusIconImageプロパティは、以下のプロパティを持つ連想配列で指定します。

プロパティ名	データ型	説明
normal	String	画像のURL文字列(正常系activate前に表示するアイコン) このプロパティが存在しない場合、デフォルトのアイコンを利用します。

プロパティ名	データ型	説明
error	String	画像のURL文字列(request/activateでエラーが起こった場合のアイコン) このプロパティが存在しない場合、デフォルトのアイコンを利用します。

それぞれのURLには、クエリ文字列およびURLリライティングで用いるセションIDを付加することができます。詳細は、“ユーザーズガイド”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-FragmentContainer	• サイズ • カラー • ボーダー • パディング • オーバーフロー

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onStateChange	フラグメントHTMLのダウンロード中や、有効化(可視化)処理中など、FragmentContainerの状態が変化したときに呼ばれます。	FragmentStateChangeEvent
onFragmentError	フラグメントHTMLダウンロード処理や、有効化(可視化)処理が異常終了したときに呼ばれます。	FragmentErrorEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。なお、activateが完了する前は、画面部品共通イベントリスナは、利用できません。

JavaScript API

■activateメソッド

メソッド	activate()
引数	なし
戻り値	なし
説明	フラグメントHTMLの有効化(可視化)処理を行います。同じFragmentContainerに対する2度目以降のactivate呼出しは、何もせず復帰します。“FragmentContainerの正常系状態とAPIの関係”を参照してください。

■requestメソッド

メソッド	request()
引数	なし
戻り値	なし
説明	srcプロパティで指定されたフラグメントHTMLの取得を開始します。このメソッドは、取得開始(HTTPリクエスト送信)と同時に復帰します。同じFragmentContainerに対する2

	度目以降のrequestメソッド呼出しは、何もせず復帰します。“FragmentContainerの正常系状態とAPIの関係”を参照してください。

■isActivateCalledメソッド

メソッド	isActivateCalled()	
引数	なし	
戻り値	[Boolean]	true:呼出しあり false:呼出しなし
説明	過去にactivateメソッド呼出しを行ったかどうかを返します。	

■isRequestCalledメソッド

メソッド	isRequestCalled()	
引数	なし	
戻り値	[Boolean]	true:取得要求あり false:取得要求なし
説明	フラグメントHTMLの取得要求を出したかどうかを返します。(requestメソッド、activateメソッドを呼ぶとtrueになります。)	

■unloadメソッド

メソッド	unload()
引数	なし
戻り値	なし
説明	対象となるFragmentContainerをアンロードし、無効化します。 本メソッドを利用する場合、フラグメントHTML上の部品のIDは省略できません。

■reloadメソッド

メソッド	reload()
引数	なし
戻り値	なし
説明	対象となるFragmentContainerをリロードし、有効化します。 なお、一度もactivateされていないFragmentContainerは、activateを実行し、有効化します。 本メソッドを利用する場合、フラグメントHTML上の部品のIDは省略できません。

■replaceSrcメソッド

メソッド	replaceSrc(fragmentHTML)	
引数	fragmentHTML [String]	フラグメントHTML
戻り値	なし	
説明	引数で指定されたフラグメントHTMLの取得を開始します。引数を指定しない場合はエラーとなります。(RCF13903) なお、本メソッドは、取得開始(HTTPリクエスト送信)と同時に復帰します。“FragmentContainerの正常系状態とAPIの関係”を参照してください。	

■getChildItemListメソッド

メソッド	getChildItemList()	
引数	なし	
戻り値	[Array]	対象となるFragmentContainer上のフォーム部品、テーブル部品、カレンダ部品、ツリー部品のオブジェクトのリスト
説明	対象となるFragmentContainer上のフォーム部品、テーブル部品、カレンダ部品、ツリー部品のリストを返します。ただし、リストには子部品より下の階層の部品は含まれません。 なお、本メソッドは、FragmentContainerが有効化された状態の場合に利用できます。	

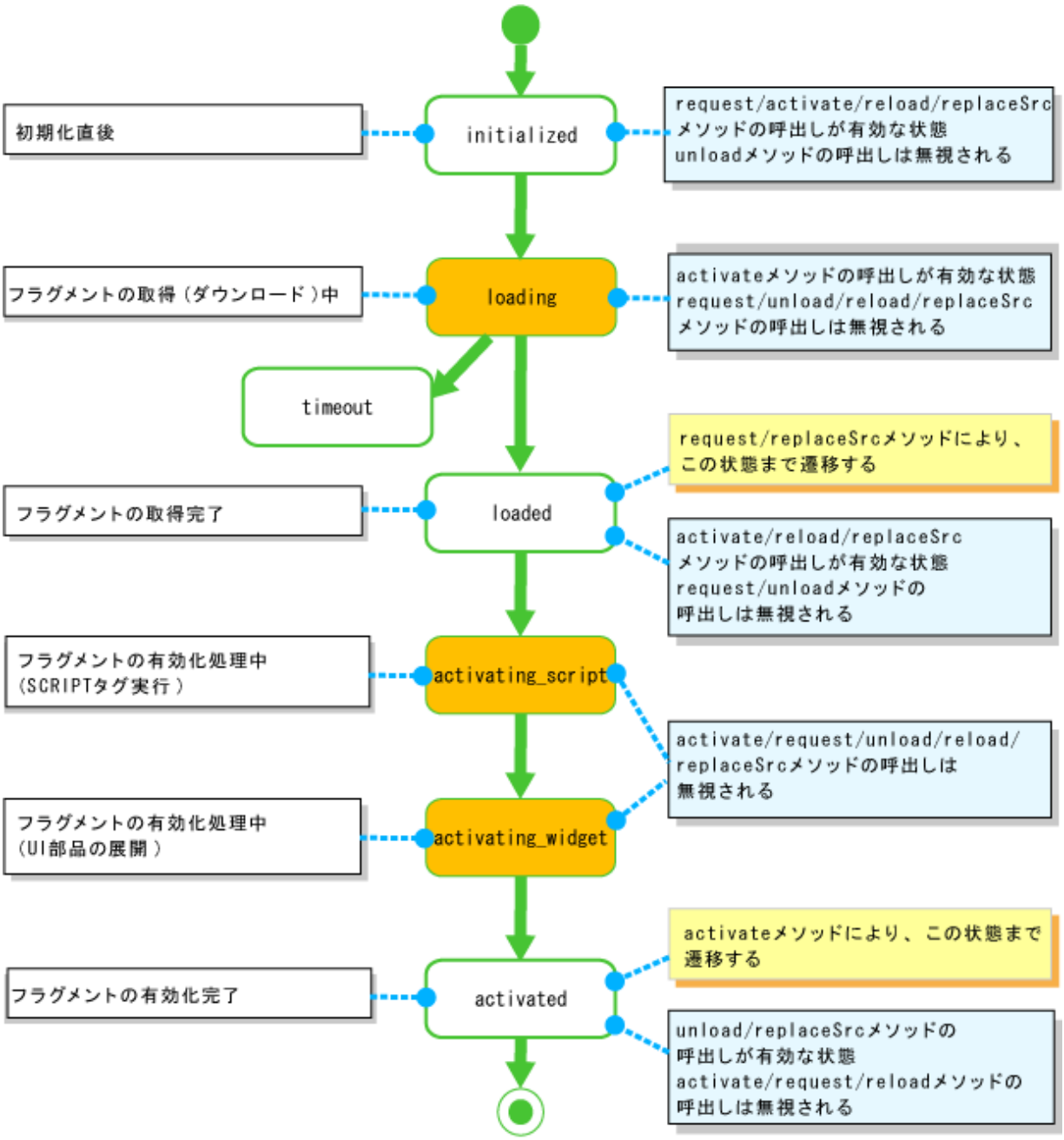
■getAllChildItemListメソッド

メソッド	getAllChildItemList()	
引数	なし	
戻り値	[Array]	対象となるFragmentContainer上にあるすべての部品のオブジェクトのリスト
説明	対象となるFragmentContainer上にあるすべての部品のリストを返します。ただし、リストには子部品より下の階層の部品は含まれません。 なお、本メソッドは、FragmentContainerが有効化された状態の場合に利用できます。	

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

FragmentContainerの正常系状態とAPIの関係

FragmentContainerの正常系状態とAPIの関係を以下の図に示します。



上記の状態が遷移するとき、FragmentContainerはFragmentStateChangeEventを送出します。

オレンジ色の状態は、エラーとなる可能性がある状態です。これらの状態でエラーとなった場合は、このコンテナはエラー状態になります。また、timeout状態に遷移したときも、同じくエラー状態になります。

FragmentContainerは、正常な状態からエラー状態に遷移するとき、FragmentErrorEventを送出します。また、rcf:showStatus=true(省略値)の場合は、インジケータアイコンが、エラーアイコンに変わります。

一度エラー状態になったコンテナは、正常状態に戻せません。

注意

activateについて

FragmentContainerのactivate関数を実行すると、非同期で有効化処理が行われます。そのため、以下のように2つのFragmentContainerのactivate関数を連続して実行した場合、fc1の有効化が先に完了することは保証されません。

```
<script type="text/javascript">
//<![CDATA[
   ...
   fc1.activate();
   fc2.activate();
   ...

//]]>
</script>
```

フラグメントHTMLに記述されている部品の呼出し

FragmentContainerの有効化が完了する前に、フラグメントHTMLに記述された部品のAPIや部品に対するイベントの登録を行うと、部品が存在していないため、エラーになります。イベントの登録は、必ずFragmentContainerの有効化の完了(activated)のあとに行ってください。

以下は、フラグメントHTMLに記述された部品に対してイベント定義を行う例です。

```
<head>
...
<script type="text/javascript">
//<![CDATA[
    //FragmentContainerのフラグメントHTMLに含まれる部品のイベント定義
    var fragment1events = {
      button1: {...}
    };

    //FragmentContainer1の状態変化に対するイベントリスナ
    function fcStateChange(stateChangeEvent) {
      switch (stateChangeEvent.state) {
        case 'activated':
          //↓コンテナ内部品の初期化(activate)が完了した場合の処理
            //↓フラグメントHTMLに含まれる部品のイベント定義を登録
          rcf.event.EventRegistrar.registerEvents(fragment1events,'fragment1events');
          break;
      }
    }

    // FragmentContainer1のイベント定義
    var fragmentContainerEvent = {
      //↓FragmentContainer1の状態変化に対するイベントリスナ登録
      fragmentContainer1      : { statechange: fcStateChange }
    };

    RCF.addInitializedListener(function(eventObject) {
      //FragmentContainer1のイベント定義を登録
      rcf.event.EventRegistrar.registerEvents(fragmentContainerEvent, 'fragmentContainer');

      //FragmentContainer1のactivate
      fragmentContainer1.activate();
    });

//]]>
</script>
...
</head>
<body>
...
  // FragmentContainer1
<div rcf:type="FragmentContainer" rcf:src="sample-fc.jsp" rcf:id="fragmentContainer1"></div>
...
</body>
```

上記の例では、RCF.addInitializedListenerでFragmentContainer1のactivateを行う前に、FragmentContainer1のイベント定義(fragmentContainerEvent)を登録しています。FragmentContainer1のイベント定義では、statuschangeイベントに対するイベントリスナ

(fcStatusChange)が定義されています。fcStatusChange関数では、FragmentContainerの状態が有効化完了(activated)になったら、FragmentContainer1のフラグメントHTMLに含まれる部品のイベント定義を登録しています。

アンロード・リロード IE 6 IE 7 IE 8

アンロード・リロードの機能は、Internet Explorerでだけ利用できます。

アンロード

FragmentContainerのアンロードは、有効化されているFragmentContainerに対して、FragmentContainer上の部品を削除することで無効化し、フラグメントHTMLの内容を表示しないようにします。

リロード

FragmentContainerのリロードは、いったんアンロードされて無効化されているFragmentContainerに対して、部品を再構築することで有効化し、フラグメントHTMLの内容を表示できるようにします。

アンロード・リロードの機能を使うことにより、例えば、複数のタブパネルに、それぞれFragmentContainerが配置され、そのFragmentContainer上に画面部品、機能部品、および一般HTML部品を配置するコンテンツがある場合、選択されているタブパネル(FragmentContainer上の部品)だけを有効化し、当面必要のないタブパネル(FragmentContainer上の部品)は無効化することができるようになります。
このように操作に必要のないFragmentContainerを無効化し、操作に必要なFragmentContainerだけを有効化することで、全体としてのメモリ使用量を抑えることができます。

以下は、TabPanelのタブ上に定義されているFragmentContainerをアンロード・リロードする例です。

```
<head>
...
<script type="text/javascript">
//<![CDATA[
  var current_index = 0;
  var counter_view  = 0;
  var tabPanelIndex = { selectedIndex : counter_view };
  var eventTabPanel = {
    tabPanel1 : { selectedindexchange : reload_fragmentContainer }
  };
  function reload_fragmentContainer(eventObject){
    var next_index = eventObject.index;  // 選択されたTabPanelのindex
    var fc_prev = null;
    var fc_next = null;
    if ( current_index == 0 ) {
        fc_prev = fc0;
    } else if ( current_index == 1 ) {
        fc_prev = fc1;
    }
    if ( next_index == 0 ) {
        fc_next = fc0;
    } else if ( next_index == 1 ) {
        fc_next = fc1;
    }
    current_index = next_index;
    fc_prev.unload();  // 現在のFragmentContainerのアンロード
    fc_next.reload();  // 選択されたFragmentContainerのリロード
  }
  RCF.addInitializedListener(function() {
    rcf.event.EventRegistrar.registerEvents(eventTabPanel, 'myevent');
    fc0.activate();
  });
//]]>
</script>
...
</head>
<body>
  <div rcf:type="Model" rcf:id="modelTabPanel" rcf:object="tabPanelIndex"></div>
```

```
  <div rcf:type="TabPanel" rcf:selectedIndex="{modelTabPanel.selectedIndex}" rcf:id="tabPanel1" rcf:width="auto">
    <div rcf:type="FragmentContainer" rcf:id="fc0" rc:src="fc0.html"></div>
    <div rcf:type="FragmentContainer" rcf:id="fc1" rcf:src="fc1.html"></div>
  </div>
</body>
</html>
```

- FragmentContainerのデータ更新について
  FragmentContainer上のUI部品のデータ内容を更新した場合、アンロードを実施したあとでも前回のデータの更新内容を保持しておくためには、サーバ側で変更内容を保持しておくか、またはアンロードされない上位のコンテナHTMLにユーザデータをモデルオブジェクトとしてバインディングしておき、変更内容を保持しておく必要があります。これは、リロードした際に部品を再構築するには、そのユーザデータの変更内容が反映されている必要があるためです。また、このとき、必要に応じて各UI部品のAPIで、データ値を設定するようにしてください。

- アンロード・リロードの適用について
  FragmentContainerのアンロード時には不要となったFragmentContainer上の部品を削除し、また、リロード時には選択されたFragmentContainer上の部品を再構築するため、各FragmentContainerのフラグメントHTMLの内容により、アンロード・リロードのための時間がかかります。したがって、クライアントの環境、利用するコンテンツなどの利用環境に合わせて本機能の適用の可否を判断する必要があります。以下の長所および短所を考慮し、適用の可否を検討してください。

方式	長所	短所
activate	クライアントの処理性能が高く、メモリが十分に搭載されている場合は、FragmentContainerの切替えが早くなります。 理由:各FragmentContainerを一度activateして有効化しておくと、それ以降はFragmentContainerの有効化に時間かからないためです。	クライアントに処理性能が低く、メモリが十分搭載されていない場合、操作性能に影響する場合があります。 理由:多くの部品を配置するFragmentContainerを切り替えて利用する場合、各FragmentContainerをactivateするごとに、その分リソースの使用量が増加するためです。
アンロード・リロード	activate方式に比べて、リソースの使用量が少なくてすみます。 理由:多くの部品を配置するFragmentContainerを切り替えて利用する場合、当面利用しないFragmentContainerを無効化し、すぐに利用するFragmentContainerだけを有効化すればよいので、全体としてのメモリ使用量を少なく抑えることができるためです。	FragmentContainerの切替えが遅くなる場合があります。 理由:リロード時にFragmentContainerを有効化するために毎回部品を再構築するので、部品数が多い場合やデータ量の大きなテーブルなどを利用している場合、初期化に時間がかかってしまうためです。

注意

- アンロード・リロードの対象となるFragmentContainerのフラグメントHTML上の部品のIDは、省略できません。
  省略した場合、アンロードは行われず、エラーメッセージが表示されます。

- アンロード時には、アンロード対象のFragmentContainer上の部品をすべて削除します。アンロード対象のFragmentContainer上の部品としてFragmentContainerがある場合、このFragmentContainerも削除します。

- アンロード時には、対象となるFragmentContainer上でユーザが独自に定義したJavascriptオブジェクトは破棄できません。アンロード時にnullを代入するなどして、オブジェクトを破棄してください。

- HTML部品の<FORM>タグは、FragmentContainer上の部品として利用できません。

- FragmentContainer上のUI部品に対してイベントリスナを設定している場合には、アンロードを実行する前に、イベントリスナの設定を外してください。

フラグメントHTMLのリプレース IE 6 IE 7 IE 8

フラグメントHTMLのリプレースの機能は、Internet Explorerでだけ利用できます。

フラグメントHTMLのリプレース機能により、任意のフラグメントHTMLを読み込むことができるので、アンロード・リロード機能と組み合わせることでFragmentContainerを効率良く利用することができます。例えば、当面不要になった現在のFragmentContainerをアンロードし、本機能により別のフラグメントHTMLを取得後、FragmentContainerをリロードして有効化するような操作を繰り返すことができます。

以下は、アンロード/リプレースの各ボタンを押すことにより、現在のFragmentContainerをアンロードし、別のフラグメントHTMLを読み込んだあと、リロードにより有効化する例です。

```
<script type="text/javascript">
//<![CDATA[
  var replaceSrc_flg = false;

  var fcloaded = function(eventObject) { // イベントを受け取ってリロード
    if (replaceSrc_flg && eventObject.state == 'loaded' ) {
          fc.reload();
          replaceSrc_flg = false;
    }
  };
  var events = {
    statechange: fcloaded
  };
  var eventmap = {
    fc: events
  };
  function unload_fragment(){   //  アンロード
    fc.unload();
  };
  function replace_fragment(){  //   別のフラグメントHTMLを読み込む
    replaceSrc_flg = true;
    fc.replaceSrc("FCnew.html");
  };
  RCF.addInitializedListener(function() {
    rcf.event.EventRegistrar.registerEvents(eventmap, 'map');
    fc.activate();
  });
//]]>
</script>
</head>
<body>
    <div rcf:type="Button" rcf:label="アンロード" rcf:onclick="unload_fragment()"></div>
    <div rcf:type="Button" rcf:label="リプレース" rcf:onclick="replace_fragment()"></div>
    <hr>
    <div rcf:id="fc" rcf:type="FragmentContainer" rcf:src="FCbase.html"></div>
</body>
```

補足事項

- FragmentContainerは、ViewStackやTabPanelと組み合わせて利用することができます。
- FragmentContainerを単体で使用した場合、activate完了後は、通常の タグと同様に表示されます。
- フラグメントHTMLは、HTMLのボディに追加されます。そのため、HTMLのヘッダに記述する要素を、フラグメントHTMLに記述することはできません。

2.2.6 Window

Windowは、ブラウザ内で移動可能な内部ウィンドウを指定する部品です。

- 表示例

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

注意

本部品は、デフォルトでは不可視です。可視化するには、showメソッドまたはshowWindowプロパティを利用します。詳細は、“5.1.10 Window、PopupCalendar部品を自動的に表示させる場合”を参照してください。

表示例

本部品で指定可能な項目を以下に示します。

- ウィンドウ自体の表示/非表示
- ドラッグでの移動
- ウィンドウサイズの変更
- スクロールバーの表示/非表示
- [閉じる]ボタンの表示/非表示

記述形式

```
<div rcf:type="Window" rcf:top="100px" rcf:left="200px" ... >
    HTML要素
</div>
```

ポイント

子要素には、HTML要素およびUI部品を指定できます。子要素に定義されたHTML要素は、ボディ部の内容になります。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
closeButton	Boolean	右上の閉じるボタンを表示するかどうかを指定します。 ・ true:表示する ・ false:表示しない	可	false	値	不可	不可
mode(注)	Number	画面モードを指定します。 ・ 0:モードレス ・ 1:モーダル ・ 2:モーダル(Window外クリック時に閉じる)	可	0	値	不可	不可
resizable	Boolean	ウィンドウサイズの変更可否を指定します。 ・ true:変更可 Window右下角をドラッグすることで、枠サイズを変更可能 ・ false:変更不可	可	false	値	不可	不可
closeButtonImage	Object	ボタンのイメージを指定するオブジェクトを指定します。オブジェクト構造は、ComboBoxのbuttonImageプロパティの指定形式と同じです。	可	標準の画像	値	不可	不可
label	String	タイトル部のラベルテキストを指定します。	可	""	値、バインド	可	不可
showWindow	Boolean	Windowの表示/非表示を切り替えます。動作はshowメソッド/hideメソッドに相当します。 ・ true: 表示 ・ false: 非表示	可	false	値	可	不可

注) mode=1または2の場合、以下の注意事項があります。

- ブラウザ共通
  - Windowが閉じられるまで、Windowの領域外でのマウス操作(クリックなど)は無効となります。
  - tabキーによる部品間のフォーカス移動は無効となります。Window表示中のフォーカスの移動には、FocusManagerを使用してください。
  - グローバルキーイベントはモーダルウィンドウの表示中も発生し、イベントリスナが実行されます。
  - mode=1または2のWindowを複数同時に表示する場合は、zIndexの値をすでに開いているモーダルウィンドウより2以上大きい値にしてください。また、最後に表示したものから順に閉じるようにしてください。順序が守られなかった場合、エラーとなります。
- Firefox2 Firefox3
  HTMLの<body>タグ内に、overflow属性を記述しないでください。Windowが操作できなくなります。
- IE6
  ウィンドウを表示した場合、ウィンドウ外部のSelect部品、HTMLのselect要素が表示されなくなります。(背景色で塗りつぶされた状態になります。)ウィンドウを閉じると、表示は元に戻ります。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	使用可能なスタイル
部品全体	なし	rcf-Window	• サイズ(注1) • カラー • ボーダー • パディング • top(Window左上X座標) (注2) • left(Window左上Y座標) (注2) • zIndex (注3)
タイトル部	title	rcf-Window-title	• サイズ(widthを除く) • カラー • ボーダー • パディング
ボディ部	body	rcf-Window-body	• カラー • フォント • テキスト • ボーダー • パディング • オーバーフロー
タイトルラベル部	titleLabel	rcf-Window-titleLabel	• カラー • フォント • テキスト(word-spacing、letter-spacing、white-spaceを除く) (注4)
閉じるボタン部	closeButton	rcf-Window-closeButton	• サイズ

注1) widthのデフォルトは300pxです。
注2) top、leftのデフォルトは共に0pxです。
注3) zIndexのデフォルトは、モーダルおよびwindow外クリックで閉じる場合が20000、モードレスの場合が10000です。
注4) white-spaceの値はnowrapで、変更できません。

詳細は、“2.9 スタイルプロパティ”を参照してください。

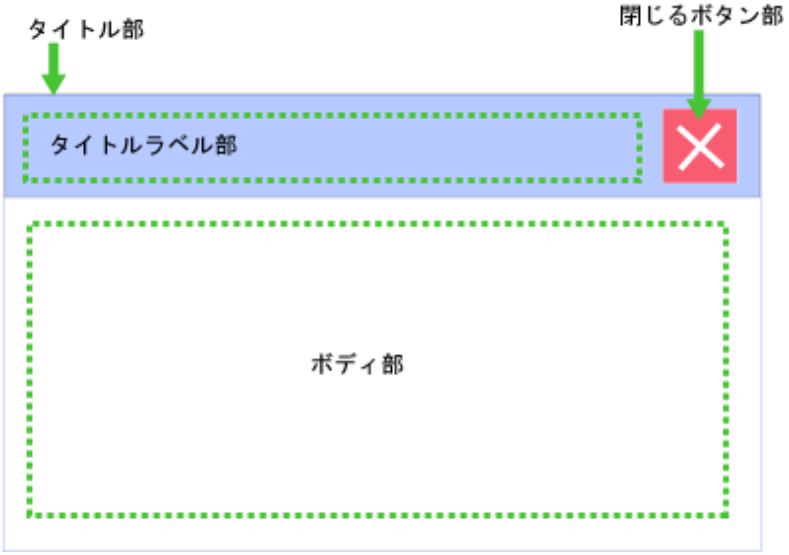
ポイント

APIでWindow位置を変更する場合は、setStyleメソッドを利用します。

例

```
window1.setStyle("top", "100px");
window1.setStyle("left", "100px");
```

図2.14 Windowの部品構成



イベントリスナ

Panelのイベントリスナを利用できます。詳細は、“2.2.2 Panel”の“イベントリスナ”を参照してください。
Windowのパーツごとの部品共通のイベントリスナを利用できます。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

タイトルラベル部のイベントリスナ

名前	説明	イベントオブジェクト
onClickTitleLabel	タイトルラベル部でマウスによりクリックされたときに呼ばれます。	ActionEvent
onDblClickTitleLabel	タイトルラベル部でマウスによりダブルクリックされたときに呼ばれます。	
onMouseDownTitleLabel	タイトルラベル部でマウスボタンが押し下げられたときに呼ばれます。	
onMouseUpTitleLabel	タイトルラベル部でマウスボタンが離されたときに呼ばれます。	
onMouseOverTitleLabel	マウスがタイトルラベル部の上に重ねられたときに呼ばれます。	
onMouseOutTitleLabel	マウスがタイトルラベル部の上から離れたときに呼ばれます。	
onMouseMoveTitleLabel	マウスがタイトルラベル部の上で動かされたときに呼ばれます。	

閉じるボタン部のイベントリスナ

名前	説明	イベントオブジェクト
onClickCloseButton	閉じるボタン部でマウスによりクリックされたときに呼ばれます。	ActionEvent
onMouseDownCloseButton	閉じるボタン部でマウスボタンが押し下げられたときに呼ばれます。	
onMouseUpCloseButton	閉じるボタン部でマウスボタンが離されたときに呼ばれます。	

名前	説明	イベントオブジェクト
onMouseOverCloseButton	マウスが閉じるボタン部の上に重ねられたときに呼ばれます。	
onMouseOutCloseButton	マウスが閉じるボタン部の上から離れたときに呼ばれます。	
onMouseMoveCloseButton	マウスが閉じるボタン部の上で動かされたときに呼ばれます。	

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

ポイント

Windowは、デフォルトでは不可視です。
本部品の表示/非表示は、showWindowプロパティ、または画面部品共通APIのshowメソッドおよびhideメソッドを利用してください。showメソッドおよびhideメソッドの詳細は、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

Windowのサイズ
Windowのサイズについては、以下の注意事項があります。

- 幅(width)と高さ(height)を設定する際の単位は、“px”にだけ対応しています。そのほかの単位による指定およびパーセント値(%)による指定をした場合、表示が崩れる可能性があります。
- ボディ部に100px×100pxの要素を配置できるサイズがWindowの最小サイズとなります。これより小さいサイズに変更しようとした場合、最小サイズに設定されます。
- IE 6
  Window部品の内部に表示する内容(タイトル部含む)のサイズが部品全体のサイズを超える場合、部品の初期化中にWindowのサイズが強制的に拡張されます。表示する内容が収まるように部品全体のサイズを設定するか、初期化後にサイズを再設定してください。
- Firefox2 Firefox3
  Windowのサイズを指定しなかった場合、ボディ部に不要なスクロールバーが表示されることがあります。Windowのサイズを指定することにより、回避してください。

Windowの配置

- topとleftは、スタイルプロパティで指定してください。
  CSSでの指定は無効となります。
- Windowは、画面の左端より左、上端より上には移動できません。
  ドラッグやAPIによってそのような位置に移動させようとした場合、画面の左端、上端に配置されます。

zIndex
Windowの最前面の描画は、zIndexの仕組みを利用します。複数のWindow部品がある場合や、ほかの部品でzIndexを指定している場合などは、zIndexの値が大きいものから前面に表示されます。
したがって、複数のWindow部品を同時に表示する場合には、通常、あとから表示するWindowのzIndexを、すでに開いているWindowのzIndexより大きい値に設定します。

その他

- Windowの初期化が完了する前にshowメソッドで表示すると、描画が崩れる場合があります。
  初期化完了と同時にWindowを表示したい場合は、showWindowプロパティをtrueにするか、タイムアウトを設定してshowメソッドを

呼ぶ関数を作成し、RCF.addInitializedListener()でイベントリスナを登録してください。
詳細は、“5.1.10 Window、PopupCalendar部品を自動的に表示させる場合”を参照してください。

- 本部品は、 タグのstyle属性(position、top、leftなど)が利用できない部品の1つです。詳細は、“1.3.5 UI部品の タグおよびタグで利用できる属性”を参照してください。

2.3 テーブル部品

テーブル部品は、テーブルの表示や編集にかかわる部品です。

ここでは、テーブル部品の設定内容、および設定方法について説明します。

ポイント

TableViewおよびTableEditは、比較的シンプルな構造のテーブルに向いています。テーブルの編集が必要な場合はTableEdit、編集をしない場合はTableViewを使用してください。

TableEditよりも複雑な構造のテーブルを編集する場合は、DataGridを使用してください。例えば、以下のような機能が使用できます。

- プルダウン
- チェックボックス
- ツリー
- 画像配置

2.3.1 TableView

TableViewは、2次元のデータを表形式で表示する部品です。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

表示例

	品名	価格	販売開始日
1	AAA	1000	2006/06/08
2	BBB	2000	2006/06/09
3	CCC	3000	2006/06/10
4	DDD	4000	2006/06/11
5	EEE	5000	2006/06/12
6	FFF	6000	2006/06/13
7	GGG	6000	2006/06/14
8	HHH	5000	2006/06/15
9	III	4000	2006/06/16
10	JJJ	3000	2006/06/17
11	KKK	2000	2006/06/18
12	LLL	1000	2006/06/19

記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
...
<div rcf:type="TableView" rcf:data="model1" ... >
  <div rcf:type="ViewColumn" ... ></div>
  ...
</div>
```

ポイント

- 子要素には、テーブルの定義情報であるViewColumnを記述します。
- 本部品は、前後に改行コードが挿入されて表示されます。
- Modelの仕様については、“3.1.1 Model”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
data	Array	Objectの配列を指定します。個々のObjectは、テーブルの各行を構成します。“dataプロパティと表示データの指定方法”を参照してください。	可	[]	バインド	可	可
showColumnHeader	Boolean	列のヘッダの有無を指定します。 ・ true:表示する ・ false:表示しない マウスで列幅を変更できません。列のヘッダをクリックしてソートしたり、列を選択したりできません。	可	true	値	可(注)	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
showRowHeader	Boolean	行のヘッダの有無を指定します。 ・ true:表示する ・ false:表示しない 行のヘッダをクリックして行を選択できません。	可	false	値	可(注)	不可
showDummyColumn	Boolean	ダミーの列の有無を指定します。表示領域を埋めるために使用します。 ・ true:表示する ・ false:表示しない	可	true	値	可	不可
columnWidthResizable	Boolean	列幅を変更可能にするかどうかを指定します。 ・ true:変更可 ・ false:変更不可	可	true	値	可	不可
defaultColumnWidth	Number	デフォルトの列幅を指定します。単位はピクセルです。 最小値は10です。10未満の数値を指定した場合、エラーとなります。	可	80	値	不可	不可
selectedRows	Array	選択されている行の行インデックスの配列を指定します。 属性はバインドだけが指定できます。バインディングする場合、モデルデータの初期値として、[]を指定する必要があります。[]以外を指定した場合は、エラーとなります。 選択行の指定および解除は、JavaScript APIのselectRowおよびdeselectRowで行うことができます。	可	[]	バインド	不可	不可
multipleSelect	Boolean	行の複数選択または単一選択を指定します。 ・ true:複数選択 ・ false:単一選択	可	false	値	不可	不可
tabIndex	Number	TABキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可

注) IE 6 IE 7 IE 8
Internet Explorerでだけ更新可能(Firefoxは更新不可)

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

ポイント

- 行のソートについて
  通常は、設定した行列データのインデックス順に表示されます。列ヘッダの各列の表示領域をクリックすると、その列のデータをキーに表示データ全体がソートされます。また、行のソートを行うと、dataプロパティの内容もソートされます。
  モデルとバインディングしている場合、モデルに指定しているオブジェクトの中身もソートされます。
  1回目のクリックでは昇順にソートされ、2回目は降順にソートされます。3回目以降は交互に昇順、降順とソートされます。行のフォーカス状態と選択状態はソート時に解除されます。

- ダミー列について
  テーブルを構成する行および列を合計した表示領域の大きさより、テーブル全体の表示領域が大きい場合、その隙間を埋めるためのダミー列を表示することができます。
  ダミー列はテーブルの右端に配置されます。

dataプロパティと表示データの指定方法

dataプロパティは、テーブルに表示するObjectの配列を指定します。以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    scores: [
      { id: 'ID0001',  name: 'Andy',  score:  90 },
      { id: 'ID0002',  name: 'Bob',   score:  80 },
      { id: 'ID0003',  name: 'Cindy',  score: 100 },
        /*  :  */
      { id: 'ID0040',  name: 'Yolanda', score:  90 }
    ]
  };
//]]>
</script>
...
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="TableView" rcf:data="{model1.scores}" ... >
  <div rcf:type="ViewColumn" rcf:name="column1" rcf:propertyName="id" rcf:label="ID"></div>
  <div rcf:type="ViewColumn" rcf:name="column2" rcf:propertyName="name" rcf:label="名前"></div>
  <div rcf:type="ViewColumn" rcf:name="column3" rcf:propertyName="score" rcf:label="スコア"></div>
</div>
```

dataプロパティに指定する配列は、同じプロパティを持つオブジェクトの配列を指定します。

上記の例の場合、model1にバインディングされたmodelDataのscoresを指定しています。scoresは、id、name、scoreプロパティを持つオブジェクトの配列です。

配列に含まれるオブジェクトのプロパティ名は任意です。オブジェクトのあるプロパティの値を列として表示したい場合は、ViewColumnのpropertyNameプロパティに、そのプロパティ名を指定します。上記の例の場合は、3つのViewColumnのpropertyNameプロパティに、それぞれid、name、scoreが指定されており、以下のように表示されます。

データ型とセルの内容

Stringおよびundefined以外のデータ型の場合、JavaScriptのString関数により文字列に変換された内容が表示されます。undefinedの場合は、何も表示されません。
表示内容を変更したい場合は、ViewColumnのrendererを利用してください。

表2.1 各データ型の表示内容

データ型	表示内容
String	文字列
Number	10進数での数値 NaNの場合は、NaN
Boolean	trueまたはfalse
Object(nullを除く)	JavaScriptにより文字列化した内容 例: [object Object]
Array	JavaScriptにより文字列化した内容 例: 1,2,3,4... (各要素をカンマによってつなげた内容)
Date	JavaScriptにより文字列化した内容 例: Wed Mar 21 14:30:54 UTC+0900 2007
null	null
undefined	何も表示されません

データ型を文字列化した内容は、JavaScriptの処理系に依存するため、ブラウザによって表示が異なる場合があります。

TableViewの子要素にViewColumnが指定されていない場合の動作

TableViewの子要素にViewColumnが指定されていない場合、TableViewはdataプロパティに指定された配列の最初の要素の内容に基づいて表示されます。列の順番は不定であり、列ヘッダ部にはプロパティ名が表示されます。

行のソート

行のソートを実行した場合、デフォルトでは、以下の動作をする比較関数によりソートが実行されます。各行の比較関数を変更したい場合は、ViewColumnのcomparatorを利用してください。

データ型	比較方法
Number	数値として比較します。
Date	1970年1月1日午前0時からのミリ秒で比較します。

データ型	比較方法
上記以外	JavaScriptのString関数により文字列に変換し、文字列で比較します。

注意

行のソートを行うと、dataプロパティの内容もソートされます。
dataプロパティがモデルとバインディングしている場合、モデルのデータもソートされます。

ソートイメージの表示

ソート方向を表すソートイメージ(▲: 昇順、▼: 降順)は、以下のように表示されます。

- replaceItemAtメソッドの実行時:
  ソートイメージが残る
- insertItemAtメソッドの実行時:
  ソートイメージが消える
- removeItemAtメソッドの実行時:
  ソートイメージが消える
- addItemメソッドの実行時:
  ソートイメージが残る
- setData、setPropertyメソッドの実行時(テーブル全体の更新):
  ソートイメージが消える
- setData、setPropertyメソッドの実行時(テーブルの部分更新):
  ソートイメージが残る

列幅の変更

マウスにより列幅変更を行う場合、マウスによるドラッグが有効なのはブラウザ内だけです。

ドラッグしたまま、ブラウザ外までマウスを移動し、そこでマウスボタンを離してドロップを行っても検知できないため、TableViewはドラッグされたままの状態になります。
その場合は、ブラウザ内で再度クリックすると、列幅変更が実行されます。

TableViewのフォーカス

TableViewには、以下の方法でフォーカスを当てることができます。フォーカスはTableView全体に当たり、フォーカスがある状態ではキーボードによる操作が可能になります。

- ブラウザが提供するTabキーおよびTab+SHIFTキーによるフォーカス移動
- FocusManagerによるフォーカス移動
- TableViewをマウスでクリック

フォーカスがある場合、TableViewの外枠にフォーカスがあることを示すアウトラインが表示されます。また、フォーカスが当たったとき、TableView内では、以下の行にフォーカスが当たり、キーボードによりフォーカス行を移動させることができます。

- Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、TableViewにフォーカスが当たった場合
  以前フォーカスを失ったときのフォーカス行に、フォーカスが当たります。初めてフォーカスが当たった場合は、最初の行にフォーカスが当たります。
- マウスでクリックされたことによりTableViewにフォーカスが当たった場合
  行がクリックされた場合は、その行にフォーカスが当たります。それ以外をクリックした場合は、Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、TableViewにフォーカスが当たった場合と同様になります。

注意

- Internet Explorerの場合、TableView内のスクロールバーをクリックすると、TableViewからフォーカスが外れます。

ー TableViewのデータが空であった場合、行にフォーカスは当たりません。

マウス操作

以下の表に示します。

操作		処理	
		単一選択モード	複数選択モード
行でクリック	なし	クリックした行を選択します。 ダミー列のセルをクリックしても選択されません。	ほかの選択状態を解除し、クリックした行だけ選択します。 ダミー列のセルをクリックしても機能しません。
	+Shift		直前に選択/解除した行からクリックした行までを範囲選択します。 直前に選択された行がなければ、その行だけを選択範囲とします。 ダミー列のセルをクリックしても機能しません。
	+Ctrl		クリックした行の選択状態を切り替えます。 ダミー列のセルをクリックしても機能しません。
列ヘッダでクリック	なし	クリックした列を元にソートを実行します。	ほかの選択状態を解除し、クリックした列を元にソートを実行します。
	+Shift		
	+Ctrl		
行ヘッダでクリック	なし	クリックした行を選択します。	ほかの選択状態を解除し、クリックした行を選択します。
	+Shift		直前に選択した行からクリックした行までを範囲選択します。直前に選択された行がなければ、その行だけを選択範囲とします。
	+Ctrl		クリックした行の選択/解除を切り替えます。

キーボード操作

キーボード操作は、TableViewにフォーカスが当たっている場合に有効になります。

操作		処理	
		単一選択モード	複数選択モード
スペース	なし	フォーカス行を選択します。	フォーカス行を選択します。
	+Shift		直前に選択/選択解除した行からフォーカスのある行までを範囲選択します。
	+Ctrl		フォーカスのある行の選択状態を切り替えます。
↑		フォーカスを上の行に移動します。	
↓		フォーカスを下の行に移動します。	
←		スクロールバーが表示されている場合、左側にスクロールします。	
→		スクロールバーが表示されている場合、右側にスクロールします。	
Home		フォーカスを先頭行に移動します。	
End		フォーカスを最終行に移動します。	
PageUp		スクロールバーが表示されている場合、上にスクロールします。	
PageDown		スクロールバーが表示されている場合、下にスクロールします。	

注意

Firefox3

列ヘッダおよび行ヘッダを表示している場合、部品内でスクロールを行うと、データ部のスクロール移動に対し、ヘッダ部の移動が遅れることがあります。

スタイルプロパティ

テーブルのスタイル

パーツ名	プレフィックス	クラス名	指定可能なスタイル
部品全体	なし	rcf-TableView	• サイズ
列ヘッダ	ccell	rcf-TableView-ccell	• フォント(lineHeightを除く) • セル
セル	cell	rcf-TableView-cell	• カラー • フォント(lineHeightを除く) • ボーダー(borderWidthを除く) • セル

ポイント

ー テーブルの幅と高さの指定について

- テーブルの幅(width)と高さ(height)は、“px”指定だけ対応しています。
  そのほかの単位およびパーセント値(%)による指定をした場合、動作が不定になります。
- テーブルの幅と高さを小さくすると、スクロールバーが表示されなくなるなど、表示が崩れる場合があります。幅、高さ共に100px未満の値を設定した場合、100pxで表示されます。
- 列ヘッダのセルの内容が自動改行により複数行で表示される場合、列ヘッダの高さが自動的に調整されるようになっています。これにより、テーブルの高さが指定されたものよりも大きくなる場合があります。
- テーブルのデフォルトの幅と高さは、それぞれ400px(幅)、300px(高さ)です。

ー セルの内容の表示について
各セルの表示は、デフォルトでは改行しないようになっています。自動改行を行いたい場合は、列ヘッダ部はccellWhiteSpaceで、セルはcellWhiteSpaceで、normalを指定してください。CSSでも指定することができます。

ー ボーダーについて
列ヘッダおよびセルのボーダーに関する注意事項は以下のとおりです。

- ボーダーの幅は、1pxに固定されています。変更することはできません。
  そのため、ボーダーのスタイルに二重線などを指定した場合、実線と区別がつかないことがあります。
- デフォルトでは表示の見栄えを良くするために、左と上のボーダーの色は白になっています。

ー 行ヘッダの左右と、列ヘッダの下端領域について
行ヘッダの左右、および列ヘッダの下端(下部スクロールバーの左)の領域の色は、カスタマイズできません。

操作によるスタイル変更

操作によるスタイルは、CSSでだけ設定することができます。

列ヘッダのセルにマウスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
ccellHovered	rcf-TableView-ccellHovered	・ カラー

行にフォーカスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
rowFocused	rcf-TableView-rowFocused	・ カラー

行が選択された場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
rowSelected	rcf-TableView-rowSelected	・ カラー

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

本部品全体のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onDataChange	dataプロパティが変更されたときに呼ばれます。	DataChangeEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

列ヘッダ部のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onClickColumn	列ヘッダ部をマウスでクリックしたときに呼ばれます。	ItemEvent
onDblClickColumn	列ヘッダ部をマウスでダブルクリックしたときに呼ばれます。	
onMouseDownColumn	マウスが列ヘッダ部の上で押し下げられたときに呼ばれます。	
onMouseUpColumn	マウスが列ヘッダ部の上で離されたときに呼ばれます。	
onMouseOverColumn	列ヘッダ部の上にマウスを重ねたときに呼ばれます。	
onMouseOutColumn	列ヘッダ部の上からマウスが外れたときに呼ばれます。	
onMouseMoveColumn	列ヘッダ部の上でマウスを動かしたときに呼ばれます。	

行ヘッダ部のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onClickRow	行ヘッダ部をマウスでクリックされたときに呼ばれます。	ItemEvent
onDblClickRow	行ヘッダ部をマウスでダブルクリックされたときに呼ばれます。	
onMouseDownRow	マウスが行ヘッダ部の上で押し下げられたときに呼ばれます。	
onMouseUpRow	マウスが行ヘッダ部の上で離されたときに呼ばれます。	
onMouseOverRow	マウスが行ヘッダ部の上に重ねられたときに呼ばれます。	
onMouseOutRow	マウスが行ヘッダ部の上から外れたときに呼ばれます。	
onMouseMoveRow	マウスが行ヘッダ部の上で動いたときに呼ばれます。	

名前	説明	イベントオブジェクト
onFocusRow	行にフォーカスが当たったときに呼ばれます。	
onBlurRow	行からフォーカスが外れたときに呼ばれます。	
onSelectRow	行が選択されたときに呼ばれます。	
onDeselectRow	行の選択が解除されたときに呼ばれます。	

セルのイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onClickCell	セルがクリックされたときに呼ばれます。	ItemEvent
onDblClickCell	セルがダブルクリックされたときに呼ばれます。	
onMouseDownCell	マウスがセルの上で押し下げられたときに呼ばれます。	
onMouseUpCell	マウスがセルの上で離されたときに呼ばれます。	
onMouseOverCell	マウスがセルの上に重ねられたときに呼ばれます。	
onMouseOutCell	マウスがセルの上から外れたときに呼ばれます。	
onMouseMoveCell	マウスがセルの上で動いたときに呼ばれます。	

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■selectRowメソッド

メソッド	selectRow(row)	
引数	row [Number]	対象行のインデックスを指定します。 省略時は、現在フォーカスがある行を指定したとみなされます。
戻り値	[Boolean]	true:行の選択に成功しました。 false:行の選択に失敗しました。
例外	なし	
説明	指定した行インデックスに対応する行を選択状態にします。multipleSelectプロパティがfalseの場合(単一選択の場合)、指定した行の選択に成功すると、それ以外の選択状態の行は選択状態を解除されます。 指定した行の選択に失敗した場合、選択状態の解除は行われません。 指定した行インデックスが範囲外の値の場合、およびrowを省略したときにフォーカスしている行がない場合には、選択に失敗します。 対象行が選択状態の場合でも成功しますが、その場合はselectrowイベントは発生しません。	

■deselectRowメソッド

メソッド	deselectRow(row)	
引数	row [Number]	対象行のインデックスを指定します。 省略時は、現在フォーカスがある行を指定したとみなされます。
戻り値	[Boolean]	true:行の選択解除に成功しました。 false:行の選択解除に失敗しました。
例外	なし	
説明	指定した行インデックスに対応する行の選択状態を解除します。(非選択状態) 以下の場合には、選択解除に失敗します。	

	・ 指定した行インデックスが範囲外の値の場合 ・ フォーカスしている行がない場合 解除に成功した場合、deselectrowイベントが発生します。 対象行が選択状態でない場合も成功しますが、その場合はdeselectrowイベントは発生しません。

■sortメソッド

メソッド	sort(column, asc)	
引数	column [String]	対象列の列名(ViewColumnのname属性に指定した値)を指定します。 省略できません。
	asc [Boolean]	true:昇順 false:降順 省略した場合、昇順でソートされます。
戻り値	[Boolean]	true:ソートに成功しました。 false:ソートに失敗しました。(ViewColumnの指定でsortableがfalseの場合、ソートに失敗します。)
例外	なし	
説明	指定した列名の列データに基づいてソートを実行します。モデルで指定した配列データのソートも実行されます。 columnが省略された場合、または存在していない列名を指定した場合は、ソートに失敗します。 ソートに成功した場合、datachangeイベントが発生します。	

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

ポイント

テーブル部品のモデルデータ更新における画面更新の動作は、以下のようになります。

- テーブル全体を更新するケース
  - setPropertyを使用したデータ更新
  - データプロバイダを使用した行の挿入や削除
  - ソート
- 部分的に更新するケース
  - データプロバイダを使用した行の追加や置換

データプロバイダの使用方法については、“ユーザーズガイド”の“データプロバイダ”を参照してください。

2.3.2 TableEdit

TableEditは、2次元のデータを表形式で表示し、編集することもできる部品です。

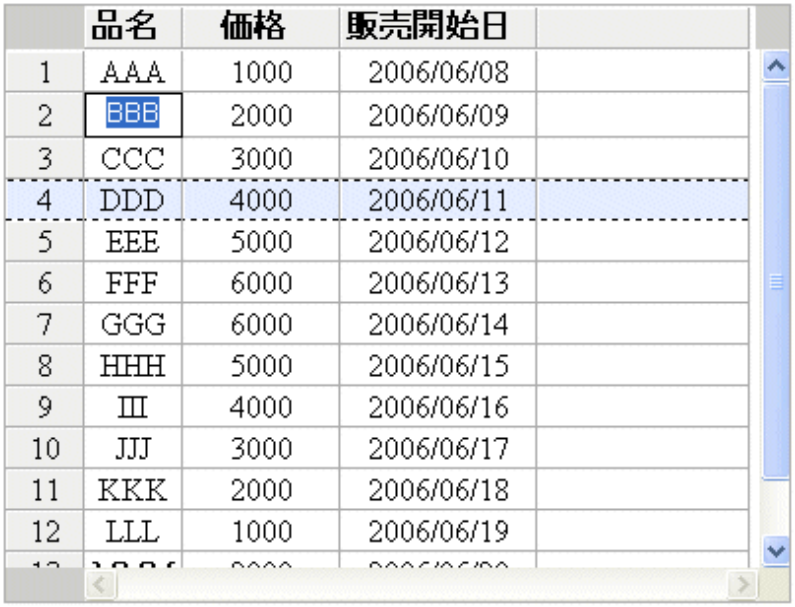
- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

- 補足事項

注意

本部品では、行のソートはできません。

表示例



記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
...
<div rcf:type="TableEdit" rcf:data="model1" ... >
  <div rcf:type="ViewColumn" ... ></div>
  ...
</div>
```

ポイント

- 子要素には、テーブルの定義情報であるViewColumnを記述します。
- 本部品は、前後に改行コードが挿入されて表示されます。
- Modelの仕様については、“3.1.1 Model”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
editable	Boolean	セルの内容を編集可能にするかどうかを指定します。 ・ true:編集可	可	true	値	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ false:編集不可 ViewColumnのeditableプロパティの値に関わらず、すべてのセルが編集不可になります。					
selectedCell	Object	選択されているセルのインデックスオブジェクトを返します。 row(行インデックス番号。先頭は0)およびcolumn(列名)の2つのプロパティでセルの位置を表します。 属性は、バインドだけ指定できます。バインディングする場合、モデルデータの初期値として、nullを指定する必要があります。null以外を指定した場合は、エラーとなります。	可	null	バインド	不可	不可
innerTabMove	Boolean	Tabキーによるセルのフォーカス移動操作の有効/無効を指定します。 ・ true:有効にする ・ false:無効にする	可	true	値	可	不可

TableViewのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.3.1 TableView”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

ポイント

- 値の入力について
  セルをマウスでダブルクリックしたとき、またはセルにフォーカスがある状態でEnterキーを押したとき、入力フィールドが表示され、文字を入力することができます。キーを押した場合、選択セルの内容はクリアされ、押されたときの文字はセルに入力されます。編集できるデータ型は、string、number、dateです。
- ダミー列について
  テーブルを構成する列を合計した表示領域の大きさより、テーブル全体の表示領域が大きい場合、その隙間を埋めるためのダミー列を表示することができます。
  ダミー列はテーブルの右端に配置されます。ダミー列にあるセルは、編集できません。

データ型とセルの内容

TableEditでは、データと表示内容は、以下の優先順位によって決定されます。

1. ViewColumnにrendererが指定されており、かつrendererのrender関数の戻り値にvalueを含んでいる場合
   ⇒render関数の戻り値で指定されたvalueの内容で表示されます。valueの内容は、表示時だけ有効であり、編集中は有効になりません。
2. ViewColumnの子要素に、DateInputまたはNumberInputが指定されている場合
   ⇒DateInputまたはNumberInputのconverterのformatの戻り値の内容が表示されます。
3. 上記以外の場合
   ⇒デフォルトの表示内容になります。

デフォルトの各データ型の表示内容を、以下の表に示します。

表2.2 各データ型の表示内容

データ型	表示内容
String	文字列

データ型	表示内容
Number	10進数での数値 NaNの場合は、NaN
Boolean	trueまたはfalse
Object(nullを除く)	JavaScriptにより文字列化した内容 例: [object Object]
Array	JavaScriptにより文字列化した内容 例: 1,2,3,4... (各要素をカンマによってつなげた内容)
Date	JavaScriptにより文字列化した内容 例: Wed Mar 21 14:30:54 UTC+0900 2007
null	null
undefined	何も表示されない

データ型を文字列化した内容はJavaScriptの処理系に依存するため、ブラウザにより表示が異なる場合があります。

DateInputを指定した場合のデフォルトの表示内容を、以下の表に示します。
DateInputにconverterを指定しなかった場合、デフォルトのconverterの内容により表示されます。

データ型	表示内容
Date	yyyy/MM/dd 形式 例: 2007/12/24
それ以外	何も表示されない

converterを指定した場合は、そのconverterのformat関数の戻り値が表示されます。

NumberInputを指定した場合のデフォルトの表示内容を、以下の表に示します。
NumberInputにconverterを指定しなかった場合、デフォルトのconverterの内容により表示されます。

データ型	表示内容
Number	10進数での数値 NaNの場合は、何も表示されない
それ以外	何も表示されない

converterを指定した場合は、そのconverterのformat関数の戻り値が表示されます。

TableEditの子要素にViewColumnが指定されていない場合の動作

TableEditの子要素にViewColumnが指定されていない場合、TableEditはdataプロパティに指定された配列の最初の要素の内容に基づいて表示されます。列の順番は不定であり、列ヘッダ部にはプロパティ名が表示されます。
また、どのセルも編集できません。

値の編集について

TableEditでは、String、Number、およびDateの値を編集することができます。値を編集する場合は、ViewColumnで適切な編集用入力部品を指定する必要があります。それぞれのデータ型に対応した入力部品は、以下のとおりです。

データ型	入力部品
String	TextInput
Number	NumberInput
Date	DateInput

String、Number、およびDate以外の値は、表示することはできます。それらのデータを表示する場合は、ViewColumnのeditable属性をfalseにしてください。データ型と入力部品が一致していない場合は、編集時にエラー(RCF14201)になります。

また、NumberInputおよびDateInputによる編集では、不正な値が入力された場合、デフォルトでは以下のとおりになります。

- NumberInput:NaN
- DateInput:null

入力された値の変換方法をカスタマイズしたい場合は、converterプロパティで指定してください。

null、undefined、NaNの編集

編集しようとしているデータの値が、null、undefined、またはNaNであった場合、各入力部品によって編集可否が異なります。

データ	TextInput	NumberInput	DateInput
null	不可(エラー)	不可(エラー)	可
undefined	可	可	可
NaN	不可(エラー)	可	不可(エラー)

行の選択について

行ヘッダ部をクリックすることで、行の選択ができます。

行の選択を行う場合は、行ヘッダ部が表示されている必要がありますので、showRowHeaderプロパティにtrueを指定してください。

列幅変更の注意事項

TableViewの列幅の変更と同様です。

TableEditのフォーカス

TableEditには、以下の方法でフォーカスを当てることができます。フォーカスがある状態ではキーボードによる操作が可能になります。

- ブラウザが提供するTabキーおよびTab+SHIFTキーによるフォーカス移動
- FocusManagerによるフォーカス移動
- TableEditをマウスでクリック

フォーカスがある場合、TableEditの外枠にフォーカスがあることを示すアウトラインが表示されます。また、フォーカスが当たったとき、TableEdit内では、以下の行にフォーカスが当たり、キーボードによりセルに対するフォーカスを移動させることができます。

- Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、TableEditにフォーカスが当たった場合
  以前フォーカスを失ったときにフォーカスがあったセルに、フォーカスが当たります。初めてフォーカスが当たった場合は、最初のセルにフォーカスが当たります。
- マウスでクリックされたことによりTableEditにフォーカスが当たった場合
  セルがクリックされた場合は、そのセルにフォーカスが当たります。それ以外をクリックした場合は、Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、TableEditにフォーカスが当たった場合と同様になります。

注意

- Internet Explorerの場合、TableEdit内のスクロールバーをクリックすると、TableEditからフォーカスが外れます。
- セルを編集状態にした場合、TableEditからフォーカスが外れ、編集する入力部品にフォーカスが移ります。
- TableEditのデータが空であった場合、セルにフォーカスは当たりません。

マウス操作

以下の表に示します。

操作	処理	
	単一選択モード	複数選択モード
セルでクリック	クリックしたセルを選択します。 ダミー列のセルは選択できません。	

操作		処理	
		単一選択モード	複数選択モード
セルでダブルクリック		セルを編集状態に変更します。 ダミー列のセルは選択できません。	
行ヘッダでクリック	なし	クリックした行を選択します。	ほかの選択状態を解除し、クリックした行を選択します。
	+Shift		直前に選択/解除した行からクリックした行までを範囲選択します。
	+Ctrl		クリックした行の選択/解除を切り替えます。

キーボード操作

キーボード操作は、TableEditにフォーカスが当たっている場合に、有効になります。

操作		処理	
		単一選択モード	複数選択モード
スペース		フォーカスセルを選択します。	
↑		フォーカスを上のセルに移動します。	
↓		フォーカスを下のセルに移動します。	
←		フォーカスを左のセルに移動します。	
→		フォーカスを右のセルに移動します。	
Enter		セルの編集状態を切り替えます。 編集状態でない場合:フォーカスセルが編集可能であれば、編集状態に変更します。 編集状態の場合:入力内容を確定し、編集状態を解除します。	
ESC		編集状態の場合、入力内容を破棄し、編集状態を解除します。	
Tab	なし	フォーカスを右のセルに移動します。右端のセルの場合は、次の行の左端のセルに移動します。 最終セルの場合、ブラウザが提供するTabキーによるフォーカス移動が有効になり、TableEditからフォーカスが移動します。(注) また、innerTabMoveプロパティにfalseが設定されている場合、TableEditでは処理をしません。常にブラウザが提供するTabキーによるフォーカス移動が有効になり、TableEditからフォーカスが移動します。(注)	
	+Shift	フォーカスを左のセルに移動します。左端のセルの場合は、前の行の右端のセルに移動します。 ブラウザが提供するTab＋SHIFTキーによるフォーカス移動が有効になり、TableEditからフォーカスが移動します。(注) また、innerTabMoveプロパティにfalseが設定されている場合、TableEditでは処理をしません。常にブラウザが提供するTab+SHIFTキーによるフォーカス移動が有効になり、TableEditからフォーカスが移動します。(注)	
Home		フォーカスを先頭セルに移動します。	
End		フォーカスを最終セルに移動します。	
PageUp		スクロールバーが表示されている場合、上にスクロールします。	
PageDown		スクロールバーが表示されている場合、下にスクロールします。	

注) TableEditがFocusManagerによるフォーカス制御対象となっている場合、FocusManagerの設定が優先されます。

注意

TableEditをFocusManagerによるフォーカス制御対象とする場合

TableEditをFocusManagerによるフォーカス制御の対象とする場合には、以下の注意事項があります。

- FocusManagerのデフォルトのフォーカス移動キーはEnterキーになっています。TableEditではEnterキーはセルの編集状態の切替えを行いますので、フォーカス移動キーには、TableEditの操作キーと干渉しないキー(スペース、↑、↓、→、←、Enter、ESC、Home、End、PageUp、PageDown以外のキー)を設定してください。
- FocusManagerでフォーカス移動キーにTabキーを指定したい場合、TableEditのinnerTabMoveプロパティにfalseを指定してください。これにより、TableEditではTabキーによる処理を行わなくなるため、干渉しないキーとなります。

部品内でのスクロールについて Firefox3

列ヘッダおよび行ヘッダを表示している場合、部品内でスクロールを行うと、データ部のスクロール移動に対し、ヘッダ部の移動が遅れることがあります。

スタイルプロパティ

テーブルのスタイル

パーツ名	プレフィックス	クラス名	指定可能なスタイル
部品全体	なし	rcf-TableEdit	• サイズ
列ヘッダ	ccell	rcf-TableEdit-ccell	• フォント(lineHeightを除く) • セル
セル	cell	rcf-TableEdit-cell	• カラー • フォント(lineHeightを除く) • ボーダー(borderWidthを除く) • セル

ポイント

- テーブルの幅と高さについて
  - テーブルの幅(width)と高さ(height)は、“px”指定だけ対応しています。
    そのほかの単位およびパーセント値(%)による指定をした場合、動作が不定になります。
  - テーブルの幅と高さを小さくすると、スクロールバーが表示されなくなるなど、表示が崩れる場合があります。幅、高さ共に100px未満の値を設定した場合、100pxで表示されます。
  - 列ヘッダのセルの内容が自動改行により複数行で表示される場合、列ヘッダの高さが自動的に調整されるようになっています。これにより、テーブルの高さが指定されたものよりも大きくなる場合があります。
  - テーブルのデフォルトの幅と高さは、それぞれ400px(幅)、300px(高さ)です。
- セルの内容の表示について
  各セルの表示は、デフォルトでは、改行しないようになっています。自動改行を行いたい場合は、列ヘッダ部はccellWhiteSpaceで、セルはcellWhiteSpaceで、normalを指定してください。CSSでも指定することができます。
- ボーダーについて
  列ヘッダおよびセルのボーダーに関する注意事項は、以下のとおりです。
  - ボーダーの幅は、1pxに固定されています。変更することはできません。
    そのため、ボーダーのスタイルに二重線などを指定した場合、実線と区別がつかないことがあります。
  - デフォルトでは、表示の見栄えを良くするために、左と上のボーダーの色は白になっています。

ー 行ヘッダの左右と、列ヘッダの下端領域について
行ヘッダの左右、および列ヘッダの下端(下部スクロールバーの左)の領域の色は、カスタマイズできません。

操作によるスタイル変更

操作によるスタイルは、CSSでだけ設定できます。

行ヘッダにフォーカスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
rowFocused	rcf-TableEdit-rowFocused	・ カラー

行ヘッダにより行が選択された場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
rowSelected	rcf-TableEdit-rowSelected	・ カラー

セルにフォーカスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
cellFocused	rcf-TableEdit-cellFocused	・ カラー

セルが選択された場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
cellSelected	rcf-TableEdit-cellSelected	・ カラー

セルを編集しているときのスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
cellEditing	rcf-TableEdit-cellEditing	・ カラー

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

本部品全体のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onDataChange	dataプロパティが変更されたときに呼ばれます。	DataChangeEvent

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

列ヘッダ部のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onClickColumn	列ヘッダ部をマウスでクリックされたときに呼ばれます。	ItemEvent
onDblClickColumn	列ヘッダ部をマウスでダブルクリックされたときに呼ばれます。	
onMouseDownColumn	マウスが列ヘッダ部の上で押し下げられたときに呼ばれます。	
onMouseUpColumn	マウスが列ヘッダ部の上で離されたときに呼ばれます。	
onMouseOverColumn	マウスが列ヘッダ部の上に重ねられたときに呼ばれます。	

名前	説明	イベントオブジェクト
onMouseOutColumn	マウスが列ヘッダ部の上から外れたときに呼ばれます。	
onMouseMoveColumn	マウスが列ヘッダ部の上で動いたときに呼ばれます。	

行ヘッダ部のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onClickRow	行ヘッダ部をマウスでクリックされたときに呼ばれます。	ItemEvent
onDblClickRow	行ヘッダ部をマウスでダブルクリックされたときに呼ばれます。	
onMouseDownRow	マウスが行ヘッダ部の上で押し下げられたときに呼ばれます。	
onMouseUpRow	マウスが行ヘッダ部の上で離されたときに呼ばれます。	
onMouseOverRow	マウスが行ヘッダ部の上に重ねられたときに呼ばれます。	
onMouseOutRow	マウスが行ヘッダ部の上から外れたときに呼ばれます。	
onMouseMoveRow	マウスが行ヘッダ部の上で動いたときに呼ばれます。	
onSelectRow	行が選択状態になったときに呼ばれます。	
onDeselectRow	行の選択状態が解除されたときに呼ばれます。	

セルのイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onClickCell	セルがクリックされたときに呼ばれます。	ItemEvent
onDblClickCell	セルがダブルクリックされたときに呼ばれます。	
onMouseDownCell	マウスがセルの上で押し下げられたときに呼ばれます。	
onMouseUpCell	マウスがセルの上で離されたときに呼ばれます。	
onMouseOverCell	マウスがセルの上に重ねられたときに呼ばれます。	
onMouseOutCell	マウスがセルの上から外れたときに呼ばれます。	
onMouseMoveCell	マウスがセルの上で動いたときに呼ばれます。	
onFocusCell	セルにフォーカスが当たったときに呼ばれます。	
onBlurCell	セルからフォーカスが外れたときに呼ばれます。	
onSelectCell	セルが選択状態のときに呼ばれます。	
onDeselectCell	セルの選択状態が解除されたときに呼ばれます。	

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■selectRowメソッド

メソッド	selectRow(row)	
引数	row [Number]	対象行のインデックスを指定します。 省略時は現在フォーカスがある行を指定したとみなされます。

戻り値	[Boolean]	true:行の選択に成功しました。 false:行の選択に失敗しました。
例外	なし	
説明	指定した行インデックスに対応する行を選択状態にします。 multipleSelectプロパティがfalseの場合(単一選択の場合)、指定した行の選択に成功すると、それ以外の選択状態の行、セルは選択状態が解除されます。 指定した行の選択に失敗した場合には、選択状態の解除は行われません。 指定した行インデックスが範囲外の値の場合や、rowを省略したときにフォーカスしている行がない場合には、選択に失敗します。 対象行が選択状態の場合でも成功しますが、この場合、selectrowイベントは発生しません。	

■deselectRowメソッド

メソッド	deselectRow(row)	
引数	row [Number]	対象行のインデックスを指定します。 省略時は現在フォーカスがある行を指定したとみなされます。
戻り値	[Boolean]	true:行の選択解除に成功しました。 false:行の選択解除に失敗しました。
例外	なし	
説明	指定した行インデックスに対応する行の選択状態を解除します。(非選択状態) 以下の場合は、選択状態の解除に失敗します。 ・ 指定した行インデックスが範囲外の値の場合 ・ フォーカスしている行がない場合 解除に成功した場合、deselectrowイベントが発生します。 対象行が選択状態でない場合も成功しますが、その場合はdeselectrowイベントは発生しません。	

■selectCellメソッド

メソッド	selectCell(row, column)	
引数	row [Number]	対象セルがある行のインデックスを指定します。 省略時は、現在フォーカスがある行を指定したとみなされます。また、この場合は、columnも省略する必要があります。
	column [String]	対象セルがある列の列名(ViewColumnのname属性に指定した値)を指定します。 省略時は現在フォーカスがある列を指定したとみなされます。
戻り値	[Boolean]	true:セルの選択に成功しました。 false:セルの選択に失敗しました。
例外	なし	
説明	指定した行インデックスおよび列名に対応するセルを選択状態にします。 以下の場合は選択に失敗します。 ・ 指定した行インデックスが範囲外の値の場合 ・ 列名に対応する列がない場合 ・ 引数省略時にフォーカスしているセルがない場合 セルの選択に成功した場合、selectcellイベントが発生します。 対象セルがすでに選択状態でも成功しますが、その場合はselectcellイベントは発生しません。	

■deselectCellメソッド

メソッド	deselectCell()
引数	なし
戻り値	なし
例外	なし
説明	現在選択されているセルを非選択にします。 選択されているセルがない場合は、何もしません。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

ポイント

テーブル部品のモデルデータ更新における画面更新の動作は、以下のようになります。

- テーブル全体を更新するケース
  - setPropertyを使用したデータ更新
  - データプロバイダを使用した行の挿入や削除
  - ソート
- 部分的に更新するケース
  - データプロバイダを使用した行の追加や置換

データプロバイダの使用方法については、“ユーザーズガイド”の“データプロバイダ”を参照してください。

補足事項

IE 6 IE 7 IE 8

セルの編集によるdatachangeイベントでは、RCF.setFocus()などにより、ほかの部品にフォーカスを移動することはできません。

2.3.3 DataGrid IE 6 IE 7 IE 8

DataGridは、2次元のデータを表形式で表示し、編集することもできる部品です。また、テーブル内にチェックボックスやツリーなどが表示でき、TableEditよりも多彩な編集ができます。

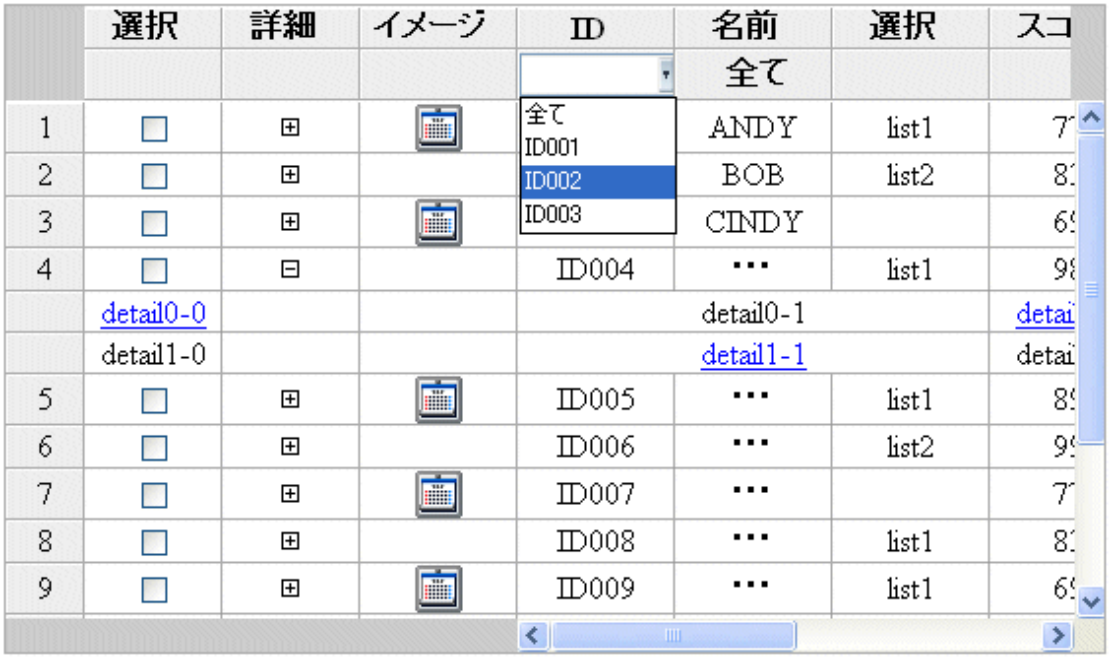
- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

注意

IE 6 IE 7 IE 8

DataGridは、Internet Explorerでだけ利用可能な部品です。

表示例



上記テーブルの“ID”セルの下には、フィルタ機能(選択された項目のデータ行だけを表示する機能)で使用するコンボボックスが設定されています。以降、このコンボボックスをフィルタ用コンボボックスと呼びます。

ポイント

DataGridでは、表示領域の大きさによって、スクロールバーの表示/非表示を自動で切り替えます。スクロールバーの表示/非表示については、フレームワークでの特別な設定は必要ありません。
スクロールバーは、テーブル全体の表示領域の内側に表示されます。

記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnGrid" ... ></div>
  ...
</div>
```

ポイント

- 子要素には、テーブルの定義情報であるViewColumnGridを記述します。(ViewColumnCheck、ViewColumnTree、ViewColumnSelect、およびViewColumnImageも使用できます。)
- 本部品は、前後に改行コードが挿入されて表示されます。
- Modelの仕様については、“3.1.1 Model”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
showRowNum	Number	表示するデータの行数を指定します。	可	全データ表示	値	不可	不可
showRowIndex	Number	データの表示開始位置を指定します。 表示開始位置には、dataプロパティで指定した配列のインデックス番号を指定します。	可	0	値	可	不可
filterCount	Number	フィルタ用コンボボックスの選択リストに追加する項目数を指定します。指定する項目数は、“全て”を除いた項目数です。(設定範囲: 0-100)値によって履歴の動作が変わります。詳細は、“フィルタ用コンボボックスの動作”を参照してください。	可	5	値	不可	不可
focusTransparent	Boolean	フォーカスの背景を透過にするかどうかを指定します。 ・ true: 透過にします。 ・ false: 透過にしません。 透過にしない場合は、CSSのスタイルで指定した値を使用します。	可	false	値	不可	不可
linkPrefix	String	アンカーリンク対象データのプレフィックスを設定します。 指定しない場合は、アンカーリンク機能は無効となります。	可	指定なし	値	不可	不可
filterList	Object	フィルタ用コンボボックスの選択項目をオブジェクトで指定します。	可	[]	バインド	可	可
dispRowsOneByOne	Boolean	データを1行ずつ順次表示するか、表示対象のすべての行を一括で表示するかを指定します。 ・ true: 1行ずつ順次表示します。 ・ false: 表示対象行を一括表示します。	可	false	値	不可	不可
dispRowsInterval	Number	dispRowsOneByOneをtrueにした場合の各行の表示間隔をミリ秒単位で指定します。(設定範囲: 1-1000)	可	50	値	不可	不可
defaultFilterLabel	String	フィルタに表示するデフォルトのラベルを指定します。	可	全て	値	不可	不可
selectedRenderer	Render関数を持つオブジェクト	render関数は、選択された行の表示カスタマイズを指示するオブジェクトを返します。	可	null	値	不可	不可
keepSortImage	String	ソートイメージの表示方法を指定します。 ・ manual: ソートイメージを常に表示します。	可	compatible	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		• compatible: TableViewと同様の表示になります。詳細は、“2.3.1 TableView”の“ソートイメージの表示”を参照してください。 なお、ページ遷移の実行時は、ソートイメージが残ります。					
focusMode	Number	DataGridにフォーカスが当たっている場合のフォーカスの動作を指定します。 • -1: 行単位で、フォーカスおよび選択を同時に行います。 • 0: TableEditと同様に、セル単位で、フォーカスと選択を別々に行います。 • 1: 0を設定した場合の動作に加え、セルへのフォーカスと同時に選択状態となります。	可	-1	値	不可	不可
cursorMode(注)	Number	セルが編集状態のときのカーソルキーによる動作を指定します。 • 0: TableEditと同様に、以下の動作となります。 ー ←キー: セル内でキャレットを左に移動します。 ー →キー: セル内でキャレットを右に移動します。 ー ↑キー: 何もしません。 ー ↓キー: 何もしません。 • 1: 編集状態を解除し、フォーカスを移動します。 ー ←キー: 入力内容を確定し、フォーカスを左に移動します。 ー →キー: 入力内容を確定し、フォーカスを右に移動します。 ー ↑キー: 入力内容を確定し、フォーカスを上に移動します。 ー ↓キー: 入力内容を確定し、フォーカスを下に移動します。	可	0	値	不可	不可
clickMode(注)	Number	編集可能なセルを編集状態とする条件を指定します。 • 0: ダブルクリックにより、セルを編集状態とします。 • 1: シングルクリックおよびダブルクリックによりセルを編集状態とします。	可	0	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
escapeDirection(注)	String	編集状態のセルでEnterキーによって入力内容を決定したとき、同時にカーソルが移動する方向を指定します。 ・ down: 下方向にカーソルを移動します。 ・ right: 右方向にカーソルを移動します。 ・ 指定なし: カーソルを移動しません。	可	指定なし	値	不可	不可
defaultColumnWidth	String	デフォルトの列幅を指定します。単位はピクセル(px)またはエム(em)です。 指定には、数値だけを指定する方法と、数値の後ろに単位を指定する方法があります。 数値だけを指定した場合、単位はピクセルとして扱われます。 px、em以外の単位を設定した場合は、設定した値は無効となり、デフォルトの幅と同じ80(px)となります。 emを指定した場合、列ごとにセルのfontFamily、fontSizeを基準とし、各列の幅を設定します。 ピクセルとして指定した場合の最小値は10ピクセルです。指定した値が10ピクセル未満の場合は、エラーとなります。	可	80	値	不可	不可

注) focusModeが0および1の場合にだけ、設定が有効になります。

TableViewのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.3.1 TableView”の“プロパティ”を参照してください。

TableEditのプロパティも指定できます。ただし、focusModeプロパティが-1の場合は、selectedCell、innerTabMoveの各プロパティは利用できません。詳細は、“2.3.2 TableEdit”の“プロパティ”を参照してください。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

ポイント

DataGridでも、TableViewと同様にデータのソートができます。詳細は、“2.3.1 TableView”を参照してください。
DataGridでは、以下の留意事項があります。

- ソートの対象は、全データ(ページ遷移による非表示データを含む)です。
- 列ヘッダが複数行で行データが1行の場合は、下位の列ヘッダをクリックしたときにソートされます。

データ型とセルの内容

DataGridでは、データと表示内容は、以下の優先順位によって決定されます。

1. 以下のrendererまたはselectedRendererが指定されており、かつrender関数の戻り値にvalueを含んでいる場合
   - ViewColumnGridのrenderer
   - DataGridまたはViewColumnGridのselectedRenderer

   ⇒render関数の戻り値で指定されたvalueの内容で表示されます。valueの内容は、表示時だけ有効であり、編集中は有効になりません。

2. ViewColumnGridの子要素に、DateInputまたはNumberInputが指定されている場合

   ⇒DateInputまたはNumberInputのconverterのformatの戻り値の内容が表示されます。

3. データの内容がlinkPrefixプロパティの対象データとなる場合

   ⇒DateInputまたはNumberInputのconverterのformatの戻り値の内容が表示されます。

4. 上記以外の場合

   ⇒デフォルトの表示内容になります。
   デフォルトの各データ型の表示内容はTableEditに準拠します。詳細は、“2.3.2 TableEdit”の“表2.2 各データ型の表示内容”を参照してください。

DataGridの子要素が何も指定されていない場合の動作

DataGridの子要素が何も指定されていない場合、DataGridはdataプロパティに指定された配列の最初の要素の内容に基づいて表示されます。列の順番は不定であり、列ヘッダ部にはプロパティ名が表示されます。
また、どのセルも編集できません。

値の編集について

動作はTableEditに準拠します。
詳細は、“2.3.2 TableEdit”の“値の編集について”を参照してください。(TableEditはDataGrid、ViewColumnはViewColumnGridに読み替えてください。)

null、undefined、NaNの編集

動作はTableEditに準拠します。詳細は、“2.3.2 TableEdit”の“null、undefined、NaNの編集”を参照してください。

行の選択について

focusModeプロパティが-1の場合は、TableViewおよびTableEditと異なり、行ヘッダ部をクリックすることで行の選択はできません。

列幅の変更

TableViewの列幅の変更と同様です。詳細は、“2.3.1 TableView”の“列幅の変更”を参照してください。

フィルタ用コンボボックスのヘッダ行で操作を行うと、うまく操作できない場合があります。
その場合は、フィルタ用コンボボックスのヘッダ行の上の行で操作を行ってください。

DataGridのフォーカス

DataGridには、以下の方法でフォーカスを当てることができます。フォーカスはDataGrid全体に当たり、フォーカスがある状態ではキーボードによる操作が可能になります。

- ブラウザが提供するTabキーおよびTab+SHIFTキーによるフォーカス移動
- FocusManagerによるフォーカス移動
- DataGridをマウスでクリック

フォーカスがある場合、DataGridの外枠にフォーカスがあることを示すアウトラインが表示されます。

また、フォーカスが当たったとき、DataGrid内では、以下の行またはセルにフォーカスが当たり、キーボードによりフォーカス行を移動させることができます。

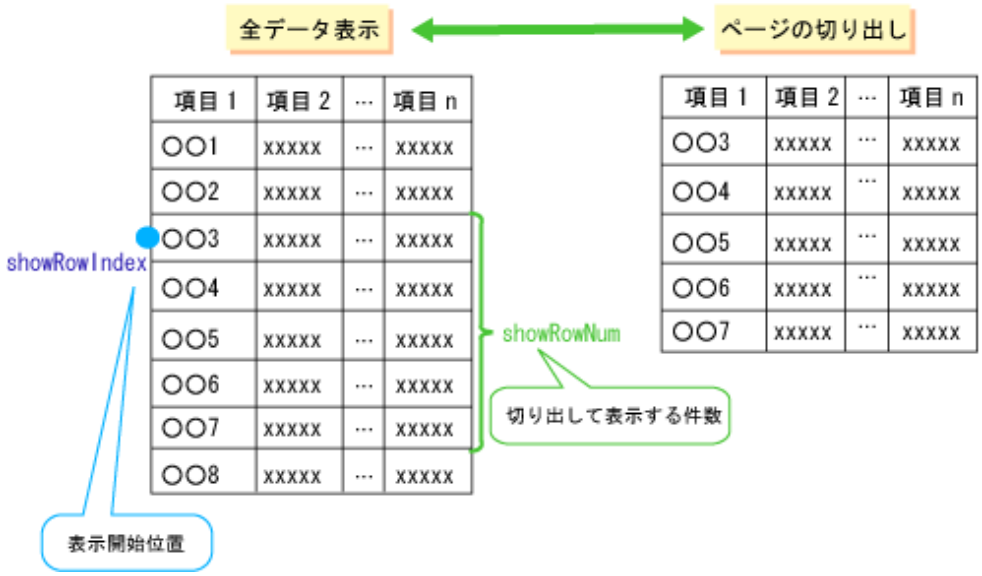
- Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、DataGridにフォーカスが当たった場合
  以前フォーカスを失ったときのフォーカス行またはセルに、フォーカスが当たります。初めてフォーカスが当たった場合は、最初の行またはセルにフォーカスが当たります。
- マウスでクリックされたことによりDataGridにフォーカスが当たった場合
  行またはセルがクリックされた場合は、その行またはセルにフォーカスが当たります。それ以外をクリックした場合は、Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、DataGridにフォーカスが当たった場合と同様になります。

DataGridにフォーカスがある場合、Tabキーによりフォーカス行またはセルが移動します。

ページ遷移について

showRowNumプロパティとshowRowIndexプロパティを使用して、1ページに表示するデータの件数を指定できます。
showRowIndexプロパティで表示開始位置を、showRowNumプロパティで切り出して表示する行数を、設定します。

また、全データから指定した件数のデータへ、表示の切替え(ページ遷移)ができます。



データの表示位置の変更方法

テーブルのデータの表示位置は、テーブル部品のオブジェクトからshowRowIndexプロパティのデータプロバイダを取得することで変更できます。

以下にプロパティの変更方法を示します。

```
<script type="text/javascript">
//<![CDATA[
    function selectCheckBox() {
      // showRowIndexプロパティに対するデータプロバイダを取得
       var showRowIndex = DataGrid.getDataProvider("showRowIndex");
      // 取得したデータプロバイダを利用してプロパティの値を変更
       showRowIndex.setData(100);
    }
//]]>
</script>
```

データプロバイダについては、“付録B データプロバイダ”を参照してください。

アンカーリンクを配置するためのdataプロパティ例

以下に例を示します。

```
...
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    scores: [
      { id: '_ID0001',  name: 'Andy', score: 77 },
      { id: 'ID0002',  name: '_Bob',    score: 83 },
      { id: 'ID0003',  name: 'Cindy',  score: 69 }
    ...
    ]
  };
//]]>
</script>
...
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.scores}" rcf:linkPrefix="_">
...
</div>
...
```

上記を指定した場合のアンカーリンクの表示は以下のようになります。

ID	名前	スコア	
ID0001	ANDY	77	
ID0002	BOB	83	
ID0003	CINDY	69	
ID0004	DED	98	
ID0005	EARTH	89	
ID0006	BANDY	31	

フィルタ用コンボボックスの動作

フィルタ用コンボボックスの先頭は、defaultFilterLabelプロパティで指定した値(省略時は“全て”、インデックス=0)になります。

filterCountプロパティに0以外を指定した場合、選択した項目または入力した値が履歴として残り、選択リストが更新されます。

最後に選択または入力した値は、選択リストの2番目(インデックス=1)となり、入力した項目と選択リストの数がfilterCountを超えた場合、最も古い項目(選択リストの最後にあった項目)が履歴から消されます。

filterCountプロパティに0を指定した場合、入力した値は無効となり、履歴には反映されず、選択した項目の順番だけが履歴に残ります。

filterListの指定方法

DataGridにおいて指定するfilterListが示すオブジェクトのデータには、以下のように初期選択位置、および選択項目を指定します。

形式

```
{
名称1:['初期選択位置', '選択項目1', '選択項目2', ... ],
名称2:['初期選択位置', '選択項目1', '選択項目2', ... ],
...
}
```

指定項目の説明

指定項目	説明
名称	ViewColumnGridのnameに設定したカラムの名前を指定します。
初期選択位置	フィルタ用コンボボックスの初期表示時の選択位置を指定します。 初期表示時に最初の選択項目を指定する場合は“1”になります。 “0”を指定した場合は、defaultFilterLabelプロパティの設定値(省略時は“全て”)が表示されます。 本指定が選択位置として認識できない値の場合、初期選択位置の指定が省略されたものと見なされ、先頭の項目も含めて指定された値がリストに表示されます。このとき、初期選択位置はdefaultFilterLabelプロパティの設定値(省略時は“全て”)となります。
選択項目n	フィルタ用コンボボックスに表示する選択項目を指定します。 初期選択位置以外の指定は、すべて選択項目として表示します。

フィルタ用コンボボックスを表示しないカラムがある場合でも、空の配列 ([]) を指定する必要があります。

以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    scores: [
      { id: 'ID0001',  name: 'Andy',  score: 90 },
      { id: 'ID0002',  name: 'Bob',   score: 80 },
      { id: 'ID0003',  name: 'Cindy',  score: 100 },
      { id: 'ID0004',  name: '...',  score: 90 },
      { id: 'ID0005',  name: '...',  score: 80 }
    ],
    flist: {
        column1: ['ID0001', 'ID0002', 'ID0003'],
        column2: ['2', 'Andy', 'Bob'],
        column3: []
    }
  };
//]]>
</script>

...
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:id="grid1" rcf:data="{model1.scores}" rcf:filterList="{model1.flist}">
<div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="id" rcf:label="ID" rcf:showFilter="true"></div>
<div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name" rcf:label="名前" rcf:showFilter="true"></
div>
<div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="score" rcf:label="スコア" rcf:showFilter="false">
  <div rcf:type="NumberInput"></div>
</div>
</div>
```

上記プロパティの設定により表示されるDataGridのコンボボックスのイメージを、以下に示します。

selectedRendererプロパティ

以下のプロパティを持ったオブジェクトを返すことで、選択行の状態をカスタマイズできます。

プロパティ名	データ型	説明
color	String	選択行の文字の色 選択行に対して、フォーカス、選択、マウスオーバーなどのイベントが発生した場合は、操作によるスタイルが優先されます。
backgroundColor	String	選択行の背景色 選択行に対して、フォーカス、選択、マウスオーバーなどのイベントが発生した場合、操作によるスタイルが優先されます。
borderColor	String	選択行のボーダーの色
borderStyle	String	選択行のボーダーの種類
fontFamily	String	選択行の文字のフォント名
fontSize	String	選択行の文字のフォントサイズ(ピクセル数)
fontWeight	String	選択行の文字のウェイト
textAlign	String	選択行のテキストの行揃え
verticalAlign	String	選択行の縦のデフォルト表示位置
value	String	実際に表示するセルの内容 ここで設定した値は、バインディングしているモデルには反映されません。 また、ここで設定した値は、行のすべてのセルに対して反映されます。
textDecoration	String	選択行の文字装飾

DataGrid部品にselectedRendererプロパティを指定した場合、行単位で選択状態を設定することができます。
DataGrid部品のrender関数では、行ごとにrender関数の呼出しが行われるため、列名(column)およびデータの値(value)は返却しません。
ViewColumnGrid部品にselectedRendererプロパティが設定されている場合、当該列については、ViewColumnGrid部品の設定が優先されます。

マウス操作(focusModeプロパティ: -1)

focusModeプロパティが-1の場合、以下の表のようになります。

操作		処理	
		単一選択モード	複数選択モード
入力可能なセルでシングルクリック		クリックした行を選択します。	クリックした行をフォーカス状態にします。
入力可能なセルでダブルクリック		クリックした行を選択し、セルを編集状態に変更します。	クリックした行をフォーカス状態にします。 セルを編集状態に変更します。
チェックボックスのあるセルでシングルクリック	なし	クリックした行を選択します。 また、活性状態で表示されているチェックボックスがあれば、チェックボックスの選択/解除を切り替えます。	活性状態で表示されているチェックボックスがあれば、チェックボックスの選択/解除を切り替えます。 また、クリックした行の選択/解除を切り替えます。
	+ Shift		活性状態で表示されているチェックボックスがあれば、チェックボックスの選択/解除を切り替えます。現在のフォーカス位置の選択状態を、クリックした位置まで適用します。そのとき、操作以前の選択状態は保持されません。

操作		処理	
		単一選択モード	複数選択モード
チェックボックスのあるセルでダブルクリック		ー	活性状態で表示されているチェックボックスがあれば、チェックボックスの選択/解除を切り替えます。 また、クリックした行の選択/解除を切り替えます。
アンカーリンクのあるセルでシングルクリック		onClickLinkのイベントが呼び出されます。 クリックした行を選択します。	onClickLinkのイベントが呼び出されます。 クリックした行をフォーカス状態にします。
アンカーリンクのあるセルでダブルクリック		onClickLinkのイベントが呼び出されます。 クリックした行を選択します。	onClickLinkのイベントが呼び出されます。 クリックした行をフォーカス状態にします。
ツリーの展開ボタンがあるセルでシングルクリック		展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 クリックした行を選択します。	展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 クリックした行をフォーカス状態にします。
ツリーの展開ボタンがあるセルでダブルクリック		展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 クリックした行を選択します。	展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 クリックした行をフォーカス状態にします。
画像ボタンがあるセルでシングルクリック		onClickImageのイベントが呼び出されます。 クリックした行を選択します。	onClickImageのイベントが呼び出されます。 クリックした行をフォーカス状態にします。
画像ボタンがあるセルでダブルクリック		onClickImageのイベントが呼び出されます。 クリックした行を選択します。	onClickImageのイベントが呼び出されます。 クリックした行をフォーカス状態にします。
プルダウンがあるセルでシングルクリック		クリック後、プルダウンが表示されます。 クリックした行を選択します。 再度クリックすると、リストが展開されます。	クリック後、プルダウンが表示されます。 クリックした行をフォーカス状態にします。 再度クリックすると、リストが展開されます。
プルダウンがあるセルでダブルクリック		クリック後、リストが展開されます。 クリックした行を選択します。	クリック後、リストが展開されます。 クリックした行をフォーカス状態にします。
上記以外のセルでシングルクリック		クリックした行を選択します。	クリックした行をフォーカス状態にします。
上記以外のセルでダブルクリック		クリックした行を選択します。 onDblClickCellのイベントが呼び出されます。	クリックした行をフォーカス状態にします。 onDblClickCellのイベントが呼び出されます。
列ヘッダでシ	なし	クリックした列を元にソートを実行します。	ほかの選択状態を解除し、クリックした列を元にソートを実行します。 ただし、ViewColumnCheck部品が配
	+ Shift		
	+ Ctrl		

操作		処理	
		単一選択モード	複数選択モード
ングルクリック			置されている場合、選択状態は保持されます。
ダミーセルでシングルクリック		操作できません。	
ダミーセルでダブルクリック		操作できません。	
プルダウン初期表示のテキストをクリック		クリック後、リストが展開されます。再度クリックすると、リスト上のクリックした項目を選択します。	
コンボボックス初期表示のテキストをクリック		フィルタ用コンボボックスに表示を切り替えます。	
コンボボックスのフィルタのボタン部分(▼)をクリック		選択リストを展開して表示します。	
コンボボックスの入力部分をクリック		キーボードから文字入力が可能となります。	
コンボボックスの選択項目をクリック		選択した文字列が入力部分に表示され、選択した文字列を利用者に通知します。 利用者への通知については、“フィルタ用コンボボックスに入力および選択された文字列の通知”を参照してください。 “全て”を選択した場合、入力項目は空になります。	

行ヘッダをクリックしても、選択状態は変化しません。(TableViewでの処理と異なります。)

マウス操作(focusModeプロパティ: 0、1)

focusModeプロパティが0または1の場合、以下の表のようになります。

操作		処理	
		単一選択モード	複数選択モード
入力可能なセルでシングルクリック		クリックしたセルを選択します。 ダミーセルは選択できません。	
入力可能なセルでダブルクリック		クリックしたセルを編集状態にします。 ダミー列のセルは選択できません。	
行ヘッダでクリック	なし	クリックした行を選択します。	クリックした行を選択します。
	+Shift		直前に選択した行からクリックした行までを範囲選択します。
	+Ctrl		クリックした行の選択/選択解除を切り替えます。
チェックボックスのあるセルでシングルクリック	なし	クリックしたセルを選択します。 また、活性状態で表示されているチェックボックスがあれば、チェックボックスの選択/解除を切り替えます。	
	+ Shift		
チェックボックスのあるセルでダブルクリック		ー	

操作		処理	
		単一選択モード	複数選択モード
アンカーリンクのあるセルでシングルクリック		onClickLinkのイベントが呼び出されます。 クリックしたセルを選択します。	
アンカーリンクのあるセルでダブルクリック		onClickLinkのイベントが呼び出されます。 クリックしたセルを選択します。	
ツリーの展開ボタンがあるセルでシングルクリック		展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 クリックしたセルを選択します。	
ツリーの展開ボタンがあるセルでダブルクリック		展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 クリックしたセルを選択します。	
画像ボタンがあるセルでシングルクリック		onClickImageのイベントが呼び出されます。 クリックしたセルを選択します。	
画像ボタンがあるセルでダブルクリック		onClickImageのイベントが呼び出されます。 クリックしたセルを選択します。	
プルダウンがあるセルでシングルクリック		クリック後、プルダウンが表示されます。 クリックしたセルを選択します。 再度クリックすると、リストが展開されます。	
プルダウンがあるセルでダブルクリック		クリック後、リストが展開されます。 クリックしたセルを選択します。	
上記以外のセルでシングルクリック		クリックしたセルを選択します。	
上記以外のセルでダブルクリック		クリックしたセルを選択します。 onDblClickCellのイベントが呼び出されます。	
列ヘッダでシングルクリック	なし	クリックした列を元にソートを実行します。	
	+ Shift		
	+ Ctrl		

上記以外の操作については、focusModeプロパティが-1の場合と同様です。

キーボード操作(focusModeプロパティ: -1)

focusModeプロパティが-1の場合、以下の表のようになります。

操作		処理	
		単一選択モード	複数選択モード
スペース	なし	－	フォーカス行の選択/解除を切り替えます。 チェックボックスの選択/解除を切り替えます。
	+Shift	－	チェックボックスの選択/解除を切り替えます。 直前に選択していた行位置の選択状態を現在の選択位置まで適用します。 そのとき、操作以前の選択状態は保持しません。
↑	なし	選択行を上の行に移動します。	フォーカスを上の行に移動します。

操作		処理	
		単一選択モード	複数選択モード
	+Shift	選択行を上の行に移動します。	直前に選択していた行位置の選択状態を上方向に適用します。 操作開始行が選択状態のとき、チェックボックスを選択状態にします。 操作開始行が非選択状態のとき、チェックボックスを非選択状態にします。 選択操作以前の選択状態は保持しません。
↓	なし	選択行を下の行に移動します。	フォーカスを下の行に移動します。
	+Shift	選択行を下の行に移動します。	直前に選択していた行位置の選択状態を下方向に適用します。 操作開始行が選択状態のとき、チェックボックスを選択状態にします。 操作開始行が非選択状態のとき、チェックボックスを非選択状態にします。 選択操作以前の選択状態は保持しません。
←		スクロールバーが表示されている場合、左側にスクロールします。	
→		スクロールバーが表示されている場合、右側にスクロールします。	
Enter		編集状態の場合、入力内容を確定し、編集状態を解除します。フィルタ用コンボボックスに入力した文字列を利用者に通知します。 利用者への通知については、“フィルタ用コンボボックスに入力および選択された文字列の通知”を参照してください。	
ESC		編集状態の場合、入力内容を破棄し、編集状態を解除します。	
Tab	なし	選択行を下の行に移動します。 最終行の場合、ブラウザが提供するTabキーによるフォーカス移動が有効になり、DataGridからフォーカスが移動します。(注) プルダウンが表示されているセルで押下した場合、プルダウンを閉じます。	フォーカスを下の行に移動します。 最終行の場合、ブラウザが提供するTabキーによるフォーカス移動が有効になり、DataGridからフォーカスが移動します。(注) プルダウンが表示されているセルで押下した場合、プルダウンを閉じます。
	+Shift	選択行を上の行に移動します。 先頭行の場合、ブラウザが提供するTab+Shiftキーによるフォーカス移動が有効になり、DataGridからフォーカスが移動します。(注)	フォーカスを上の行に移動します。 先頭行の場合、ブラウザが提供するTab+Shiftキーによるフォーカス移動が有効になり、DataGridからフォーカスが移動します。(注)
Home		先頭行を選択状態にします。	フォーカスを先頭行に移動します。
End		最終行を選択状態にします。	フォーカスを最終行に移動します。
PageUp		スクロールバーが表示されている場合、上にスクロールします。	
PageDown		スクロールバーが表示されている場合、下にスクロールします。	

注) DataGridがFocusManagerによるフォーカス制御対象となっている場合、FocusManagerの設定が優先されます。

注意

DataGridをFocusManagerによるフォーカス制御対象とする場合

DataGridをFocusManagerによるフォーカス制御の対象とする場合には、以下の注意事項があります。

- FocusManagerのデフォルトのフォーカス移動キーはEnterキーになっています。DataGridではEnterキーはセルの編集状態の切替えを行いますので、フォーカス移動キーには、DataGridの操作キーと干渉しないキー(スペース、↑、↓、→、←、Enter、ESC、Home、End、PageUp、PageDown以外のキー)を設定してください。

キーボード操作(focusModeプロパティ: 0)

focusModeプロパティが0の場合、以下の表のようになります。

操作		処理	
		単一選択モード	複数選択モード
スペース	なし	フォーカスセルを選択します。	
	+Shift	フォーカスを上のセルに移動します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
↑		フォーカスを上のセルに移動します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
↓		フォーカスを下のセルに移動します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
←		フォーカスを左のセルに移動します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
→		フォーカスを右のセルに移動します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
Enter		セルの編集状態を切り替えます。 編集状態でない場合、フォーカスセルが編集可能であれば編集状態に変更します。 編集状態の場合、入力内容を確定し、編集状態を解除します。編集状態を解除したあとの動作は、escapeDirectionプロパティの設定に従います。 チェックボックスがあるセルにフォーカスがある場合は、チェックボックスの状態により、チェックボックスの選択/解除を切り替えます。 ツリーの展開ボタンにフォーカスがある場合は、ツリーの展開ボタンの状態により、onOpenDetailまたはonCloseDetailのイベントが呼び出されます。 プルダウンがあるセルにフォーカスがある場合は、プルダウンの状態により、プルダウンの表示/非表示を切り替えます。	
ESC		編集状態の場合、入力内容を破棄し、編集を解除します。	
Tab	なし	フォーカスを右のセルに移動します。右端のセルの場合は、次の行の左端のセルに移動します。 セルが編集状態のときは何もしません。 プルダウンが表示されているセルで押下した場合、プルダウンを閉じます。	
	+Shift	フォーカスを左のセルに移動します。左端のセルの場合は、前の行の右端のセルに移動します。 セルが編集状態のときは何もしません。	
Home		フォーカスを先頭セルに移動します。	
End		フォーカスを最終セルに移動します。	

操作	処理	
	単一選択モード	複数選択モード
PageUp	スクロールバーが表示されている場合、上にスクロールします。	
PageDown	スクロールバーが表示されている場合、下にスクロールします。	

キーボード操作:(focusModeプロパティ: 1)

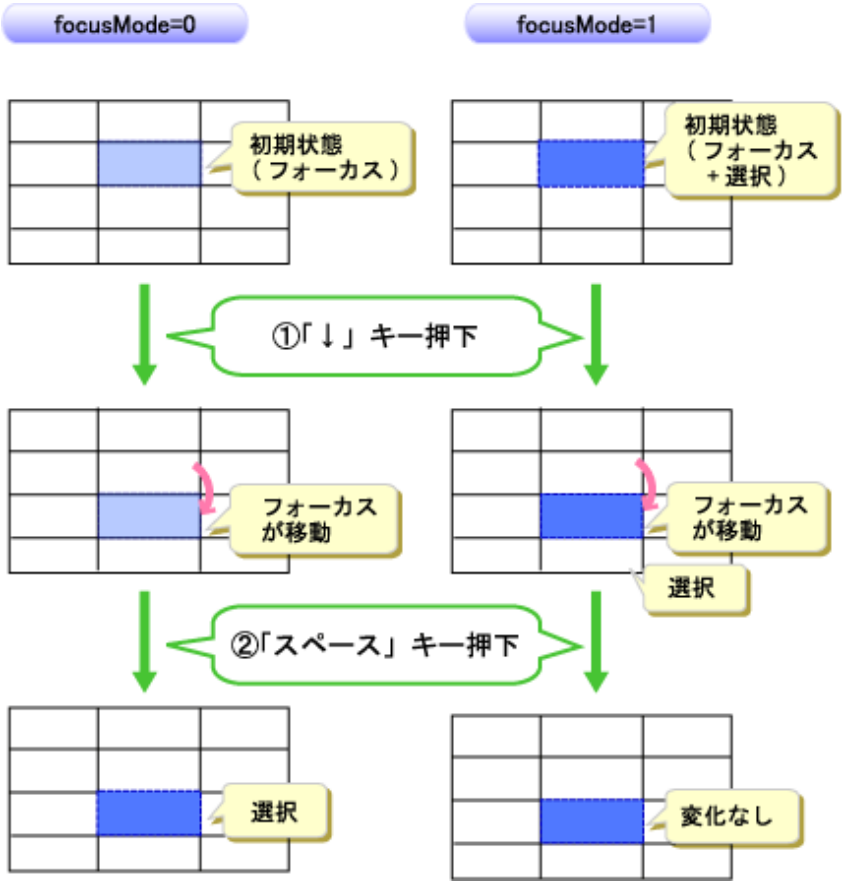
focusModeプロパティが1の場合、以下の表のようになります。

操作		処理	
		単一選択モード	複数選択モード
スペース		―	
↑		上のセルを選択します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
↓		下のセルを選択します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
←		左のセルを選択します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
→		右のセルを選択します。 セルが編集状態のときは、cursorModeプロパティの設定に従います。	
Tab	なし	右のセルを選択します。右端のセルの場合は、次の行の左端のセルを選択します。 セルが編集状態の場合は何もしません。 プルダウンが表示されているセルで押下した場合、プルダウンを閉じます。	
	+Shift	左のセルを選択します。左端のセルの場合は、前の行の右端のセルを選択します。 セルが編集状態の場合は何もしません。	
	+Ctrl	―	
Home		先頭セルを選択します。	
End		最終セルを選択します。	

上記以外の操作については、focusModeプロパティが0の場合と同様です。

focusModeプロパティとキーボード操作について

focusModeプロパティ(0および1)の設定によるカーソルキーおよびスペースキーの動作の違いを、以下の図に示します。



注意

focusModeプロパティが1の場合、focusableプロパティまたはselectableプロパティのどちらかにfalseが設定されていると、フォーカスおよび選択がされません。

マウス操作については、以下のとおりです。

- focusModeプロパティが-1の場合は、行操作を主体とする動作となります。
- focusModeプロパティが0の場合は、TableEditと同様にセル操作を主体とする動作となります。
- focusModeプロパティが1の場合は、セル操作を主体とする動作となり、かつセルをクリックすることでフォーカスと同時に選択する動作となります。

スタイルプロパティ

テーブルのスタイル

パーツ名	プレフィックス	クラス名	指定可能なスタイル
部品全体	なし	rcf-DataGrid	• サイズ
列ヘッダ	ccell	rcf-DataGrid-ccell	• フォント(lineHeightを除く) • セル
フィルタ行(初期表示時)	fcell	rcf-DataGrid-fcell	• フォント(lineHeightを除く) • セル
セル	cell	rcf-DataGrid-cell	• カラー • フォント(lineHeightを除く)

パーツ名	プレフィックス	クラス名	指定可能なスタイル
			• ボーダー(borderWidthを除く) • セル(注)
アンカーリンク	link	rcf-DataGrid-link	• カラー • フォント(lineHeightを除く) • ボーダー(borderWidthを除く) • セル(注)
ツリー展開時の詳細データ表示セル	detail	rcf-DataGrid-detail	• カラー • フォント(lineHeightを除く) • ボーダー(borderWidthを除く) • セル(注)

注) DataGridでは、textDecorationを指定することができます。
textDecorationでは、文字装飾を指定します。CSSのtext-decorationプロパティの値を指定できます。
line-throughを指定すると、取り消し線が設定できます。Internet Explorerでは、blinkは指定できません。

ポイント

テーブルの幅と高さの指定について

- テーブルの幅と高さを小さくすると、スクロールバーが表示されなくなるなど、表示が崩れる場合があります。幅、高さ共に100px未満の値を設定した場合、100pxで表示されます。

フィルタ用コンボボックスのスタイル

フィルタ用コンボボックスで指定できるスタイルは、ComboBox部品のスタイルプロパティのうち、以下のクラス名です。

- rcf-ComboBox-input
- rcf-ComboBox-list
- rcf-ComboBox-item
- rcf-ComboBox-itemHovered

rcf-ComboBoxの指定には、対応しません。指定した場合の動作は不定となります。

フィルタ用コンボボックスのスタイルを設定するためのCSSの定義例

```
<style type="text/css">
  .myClass .rcf-DataGrid .rcf-ComboBox-input {
    background-color: Menu;
    color: MenuText;
  }
  .myClass .rcf-DataGrid .rcf-ComboBox-list {
    font-family: monospace;
    background-color: InfoBackground;
    color: InfoText;
  }
</style>
...
<div rcf:type="DataGrid" rcf:id="grid1" rcf:data="{model1.scores}" rcf:styleClass="myClass">
...
</div>
...
```

アンカーリンク

アンカーリンクのスタイルを指定しない場合は、青文字、アンダースコアでアンカーリンクの文字列が表示されます。

アンカーリンクのスタイルを設定するためのスタイルプロパティの定義例

```
<div rcf:type="DataGrid" rcf:id="grid1" rcf:data="{model1.scores}"
   rcf:linkTextDecoration="line-through" rcf:linkColor="blue">
...
</div>
```

アンカーリンクのスタイルを設定するためのCSSの定義例

```
<style type="text/css">
  .myClass .rcf-DataGrid-link {
    color: blue;
    text-decoration: line-through;
  }
</style>
...
<div rcf:type="DataGrid" rcf:id="grid1" rcf:data="{model1.scores}" rcf:styleClass="myClass"></div>
...
```

操作によるスタイル変更

操作によるスタイルは、CSSでだけ設定できます。

列ヘッダのセルにマウスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
ccellHovered	rcf-DataGrid-ccellHovered	・ カラー(背景色、文字色)

行にフォーカスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
rowFocused	rcf-DataGrid-rowFocused	・ カラー(背景色、文字色)

フォーカスを点線だけ(背景色なし)に設定する場合は、focusTransparentプロパティで指定してください。

行が選択された場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
rowSelected	rcf-DataGrid-rowSelected	・ カラー(背景色、文字色)

セルを編集しているときのスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
cellEditing	rcf-DataGrid-cellEditing	・ カラー(背景色、文字色)

以下のスタイルは、focusModeプロパティが0または1の場合に有効となります。

セルにフォーカスがある場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
cellFocused	rcf-DataGrid-cellFocused	・ カラー(背景色、文字色)

セルが選択された場合のスタイルを以下に示します。

プレフィックス	クラス名	指定可能なスタイル
cellSelected	rcf-DataGrid-cellSelected	・ カラー(背景色、文字色)

詳細は、“2.9 スタイルプロパティ”を参照してください。

注意

CSSでセルおよびアンカーリンクのスタイルを定義する場合の注意事項

CSSでDataGridのセルおよびアンカーリンクのスタイルを指定する場合は、下記の順序で定義してください。

1. セルのスタイルを指定
2. アンカーリンクのスタイルを指定

スタイルプロパティの優先順位

スタイルプロパティで指定した操作によるスタイルの優先順位を、以下に示します。

1. rcf-DataGrid-cellSelected
2. rcf-DataGrid-cellFocused

スタイルの優先順位

ViewColumnGridおよびDataGridのスタイルを設定した場合の優先順位を、以下に示します。

列ヘッダの場合

1. スタイルプロパティで設定したViewColumGridのスタイル
2. スタイルプロパティで設定したDataGridのスタイル
3. CSSで設定したViewColumnGridのスタイル
4. CSSで設定したDataGridのスタイル

セルまたは行の場合

1. ViewColumnGridのselectedRendererで指定したスタイル
2. DataGridのselectedRendererで指定したスタイル
3. スタイルプロパティで設定したDataGridの操作に関するスタイル
4. ViewColumnGridのrendererで指定したスタイル
5. スタイルプロパティで設定したViewColumGridのスタイル
6. スタイルプロパティで設定したDataGridのスタイル
7. CSSで設定したViewColumnGridのスタイル
8. CSSで設定したDataGridのスタイル

イベントリスナ

フィルタ用コンボボックスのイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onSelectFilter	フィルタ機能により、項目が選択された場合に呼ばれます。 Enterキーが押下された場合に呼ばれます。	SelectFilterEvent

フィルタ用コンボボックスに入力および選択された文字列の通知

フィルタ用コンボボックスに入力および選択された文字列は、利用者にイベントで通知されます。通知されるイベントについては、“A.2.18 SelectFilterEvent”を参照してください。

行のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onFocusRow	行にフォーカスが当たったときに呼ばれます。	ItemEvent
onBlurRow	行からフォーカスが外れたときに呼ばれます。	

セルのイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onSetCheckBox	1つの行のチェックボックスがONになった場合に呼ばれます。	CheckItemChangeEvent
onResetCheckBox	1つの行のチェックボックスがOFFになった場合に呼ばれます。	
onSetAllCheckBox	setAllCheckBoxメソッドによってすべての行のチェックボックスがONになった場合に呼ばれます。	
onResetAllCheckBox	resetAllCheckBoxメソッドによってすべての行のチェックボックスがOFFになった場合に呼ばれます。	
onSetAllDisplayCheckBox	setAllDisplayCheckBoxメソッドによって画面上に表示されているすべての行のチェックボックスがONになった場合に呼ばれます。	
onResetAllDisplayCheckBox	resetAllDisplayCheckBoxメソッドによって画面上に表示されているすべての行のチェックボックスがOFFになった場合に呼ばれます。	
onSelectedPulldownChange	選択されているプルダウンの項目が変更されたときに呼ばれます。	SelectedPulldownChangeEvent
onOpenDetail	行の詳細データを展開したときに呼ばれます。 一括展開のJavaScript APIを用いて展開処理を行った場合、このイベントは呼び出されません。	ItemEvent
onCloseDetail	行の詳細データを折りたたんだときに呼ばれます。 一括折りたたみのJavaScript APIを用いて折りたたみ処理を行った場合、このイベントは呼び出されません。	
onClickLink	アンカーリンクをクリックした場合に呼ばれます。	
onClickImage	画像をクリックした場合に呼ばれます。	
onClickDetailLink	ツリーの詳細データ内にあるアンカーリンクをクリックした場合に呼ばれます。	DetailItemEvent
onClickDetailCell	ツリーの詳細データのセルをクリックした場合に呼ばれます。	

onOpenDetailイベントリスナについて

上記のonOpenDetailイベントリスナにより詳細データの展開通知を受け取った場合、フレームワークの呼出し元は、JavaScript APIのshowDetail()メソッドを呼び出すことで詳細データを指定する必要があります。

TableEditのイベントリスナも使用できます。詳細は、“2.3.2 TableEdit”の“イベントリスナ”を参照してください。ただし、focusModeプロパティが-1の場合は、以下のイベントリスナは使用できません。

- onFocusCell
- onBlurCell
- onSelectCell
- onDeselectCell

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■getRowCountメソッド

メソッド	getRowCount()	
引数	なし	
戻り値	[Number]	テーブルの行数
例外	なし	
説明	テーブルの全行数を返します。	

■setAllCheckBoxメソッド

メソッド	setAllCheckBox()	
引数	なし	
戻り値	[Boolean]	true :チェックボックスの全選択に成功しました。 false :チェックボックスの全選択に失敗しました。
例外	なし	
説明	テーブル内のすべてのチェックボックスを選択状態にします。 チェックボックスが配置されていない場合は、何も処理を行わず、falseを返却します。	

■resetAllCheckBoxメソッド

メソッド	resetAllCheckBox()	
引数	なし	
戻り値	[Boolean]	true :チェックボックスの全選択解除に成功しました。 false :チェックボックスの全選択解除に失敗しました。
例外	なし	
説明	テーブル内のすべてのチェックボックスを選択解除状態にします。 チェックボックスが配置されていない場合は、何も処理を行わず、falseを返却します。	

■setAllDisplayCheckBoxメソッド

メソッド	setAllDisplayCheckBox()	
引数	なし	
戻り値	[Boolean]	true :チェックボックスの全選択に成功しました。 false :チェックボックスの全選択に失敗しました。
例外	なし	
説明	表示されているテーブル内のすべてのチェックボックスを選択状態にします。 チェックボックスが配置されていない場合は、何も処理を行わず、falseを返却します。	

■resetAllDisplayCheckBoxメソッド

メソッド	resetAllDisplayCheckBox()	
引数	なし	
戻り値	[Boolean]	true :チェックボックスの全選択解除に成功しました。 false :チェックボックスの全選択解除に失敗しました。
例外	なし	

説明	表示されているテーブル内のすべてのチェックボックスを選択解除状態にします。 チェックボックスが配置されていない場合は、何も処理を行わず、falseを返却します。

■getCheckedRowメソッド

メソッド	getCheckedRow()	
引数	なし	
戻り値	[Array]	チェックがONとなっている行のインデックス番号からなる配列
例外	なし	
説明	テーブル内のチェックボックスがONの状態である行を調べ、そのインデックスからなる配列を返します。 チェックボックスが配置されていない場合は、空配列を返します。 ソートしたデータに対して本メソッドを呼び出した場合は、ソート後の行に対応するインデックス番号を返します。 データの先頭からのインデックス番号を返します。	

JavaScript APIとチェックボックスの関係を以下に示します。

・ sortメソッドを呼び出した場合
ソート後のチェックボックスの状態を保持します。

■openDetailAllメソッド

メソッド	openDetailAll(data,id)	
引数	data [Array]	詳細データとして表示するデータをオブジェクトで指定します。
	id [String]	詳細データを表示するセルのid(ViewColumnGroup)を指定します。
戻り値	[Boolean]	true :ツリーの一括展開に成功しました。 false :ツリーの一括展開に失敗しました。
例外	なし	
説明	表示されているすべての行に対する詳細データを展開します。(詳細データが存在しない行の展開処理は行いません。) すべての行が展開された状態で本APIを呼び出しても、エラーにはなりません。 処理が成功した場合もonOpenDetailイベントは発生しません。順次表示中に本APIを呼び出した場合は、falseを返却し、展開処理は行いません。	

詳細データ(全データ)の指定方法

表示行に対する詳細データを表示する場合は、以下のようなデータオブジェクトを作成し、JavaScript APIのopenDetailAllメソッドを呼び出してください。

```
var detailData = {
  data: [
   [
    { detail1: 'det0-1', detail2: 'det0-2', detail3: 'det0-3'},    //詳細データの1行目
    { detail1: 'det1-1', detail2: 'det1-2', detail3: 'det1-3'},    //詳細データの2行目
    { detail1: 'det2-1', detail2: 'det2-2', detail3: 'det2-3'}     //詳細データの3行目
   ],
   [
    { detail1: 'det0-1', detail2: 'det0-2', detail3: 'det0-3'},    //詳細データの1行目
    { detail1: 'det1-1', detail2: 'det1-2', detail3: 'det1-3'}     //詳細データの2行目
   ],
...
  ]
};
```

■closeDetailAllメソッド

メソッド	closeDetailAll()	
引数	なし	
戻り値	[Boolean]	true :ツリーの一括折りたたみに成功しました。 false :ツリーの一括折りたたみに失敗しました。
例外	なし	
説明	展開表示されている詳細データをすべて折りたたみます。 すべての行が折りたたまれた状態で本APIを呼び出しても、エラーにはなりません。 処理が成功した場合も、onCloseDetailイベントは発生しません。順次表示中に本APIを呼び出した場合は、falseを返却し、折りたたみ処理は行いません。	

■showDetailメソッド

メソッド	showDetail(event,data,id)	
引数	event [Object]	ツリー展開時のイベント(ItemEvent)を指定します。
	data [Array]	詳細データとして表示するデータをオブジェクトで指定します。
	id [String]	詳細データを表示するセルのid(ViewColumnGroup)を指定します。
戻り値	[Boolean]	true :詳細データの表示に成功しました。 false :詳細データの表示に失敗しました。
例外	なし	
説明	指定された行の詳細データを表示します。	

詳細データ(1件)の指定方法

ある特定の行に対して詳細データを表示する場合は、以下のようなデータオブジェクトを作成し、JavaScript APIのshowDetailメソッドを呼び出してください。

```
var detailData = {
  data: [
    { detail1: 'det0-1', detail2: 'det0-2', detail3: 'det0-3'},    //詳細データの1行目
    { detail1: 'det1-1', detail2: 'det1-2', detail3: 'det1-3'},    //詳細データの2行目
    { detail1: 'det2-1', detail2: 'det2-2', detail3: 'det2-3'},    //詳細データの3行目
    ...
  ]
};
```

■closeDetailメソッド

メソッド	closeDetail(rowIndex)	
引数	rowIndex [Number]	ツリーを折りたたむ表示行のインデックスを指定します。
戻り値	[Boolean]	true :ツリーの折りたたみに成功しました。 false :ツリーの折りたたみに失敗しました。
例外	なし	
説明	指定された表示行のツリーを折りたたみます。	

■isOpenedDetailメソッド

メソッド	isOpenedDetail(rowIndex)

引数	rowIndex [Number]	ツリーの展開状態を取得する表示行インデックスを指定します。
戻り値	[Boolean]または Undefined	true :ツリーが展開されています。 false :ツリーが折りたたまれています。 undefined :ツリーがありません。
例外	なし	
説明	指定された表示行のツリーの展開状態を返します。	

■getFilterListメソッド

メソッド	getFilterList()	
引数	なし	
戻り値	[Array]	フィルタ用コンボボックスで現在選択されている選択項目の配列
例外	なし	
説明	フィルタ用コンボボックスで現在選択されている選択項目の配列を返します。	

getFilterListは、以下の配列を返却します。

```
[
  { column: 'カラム名', keyword: '選択項目名', index: 選択項目のインデックス},
  { column: 'カラム名', keyword: '選択項目名', index: 選択項目のインデックス},
  { column: 'カラム名', keyword: '選択項目名', index: 選択項目のインデックス},
  ...
]
```

■getFocusedRowDisplayIndexメソッド

メソッド	getFocusedRowDisplayIndex()	
引数	なし	
戻り値	[Number]	フォーカスのインデックス
例外	なし	
説明	表示されているテーブル内におけるフォーカス行のインデックスを返します。	

■getFocusedRowIndexメソッド

メソッド	getFocusedRowIndex()	
引数	なし	
戻り値	[Number]	フォーカスのインデックス
例外	なし	
説明	テーブル内におけるフォーカス行に対応するデータのインデックスを返します。	

■selectRowメソッド

メソッド	selectRow(row)	
引数	row [Number]	対象行のインデックスを指定します。 省略時は、現在フォーカスがある行を指定したとみなされます。
戻り値	[Boolean]	true :行の選択に成功しました。 false :行の選択に失敗しました。
例外	なし	
説明	指定した行インデックスに対応する行を選択状態にします。multipleSelectプロパティがtrueであり(複数選択)、活性状態のチェックボックスがある場合は、チェックボックスも選	

	択状態にします。multipleSelectプロパティがfalseの場合(単一選択の場合)、指定した行の選択に成功すると、それ以外の選択状態の行は選択状態を解除されます。 指定した行の選択に失敗した場合、選択状態の解除は行われません。 指定した行インデックスが範囲外の値の場合、およびrowを省略したときにフォーカスしている行がない場合には、選択に失敗します。 対象行が選択状態の場合でも成功しますが、その場合はselectrowイベントは発生しません。

■deselectRowメソッド

メソッド	deselectRow(row)	
引数	row [Number]	対象行のインデックスを指定します。 省略時は、現在フォーカスがある行を指定したとみなされます。
戻り値	[Boolean]	true :行の選択解除に成功しました。 false :行の選択解除に失敗しました。
例外	なし	
説明	指定した行インデックスに対応する行の選択状態を解除します。multipleSelectプロパティがtrueであり(複数選択)、活性状態のチェックボックスがある場合は、チェックボックスも選択解除状態にします。(非選択状態) 以下の場合には、選択解除に失敗します。 ・ 指定した行インデックスが範囲外の値の場合 ・ フォーカスしている行がない場合 解除に成功した場合、deselectrowイベントが発生します。 対象行が選択状態でない場合も成功しますが、その場合はdeselectrowイベントは発生しません。	

■sortメソッド

メソッド	sort(column, asc)	
引数	column [String]	対象列の列名(ViewColumnGridのname属性に指定した値)を指定します。 省略できません。
	asc [Boolean]	true :昇順 false :降順 省略した場合、昇順でソートされます。
戻り値	[Boolean]	true :ソートに成功しました。 false :ソートに失敗しました。(ViewColumnGridの指定でsortableがfalseの場合、ソートに失敗します。)
例外	なし	
説明	指定した列名の列データに基づいてソートを実行します。モデルで指定した配列データのソートも実行されます。 columnが省略された場合、または存在していない列名を指定した場合は、ソートに失敗します。 ソートに成功した場合、datachangeイベントが発生します。	

■changeSortImageメソッド

メソッド	changeSortImage(index, sort)	
引数	index [Number]	ソートイメージを変更するカラムのインデックスを指定します。

	sort [String]	ソート順またはソートイメージの削除を指定します。 asc :昇順(▲) desc :降順(▼) delete: 表示されているソートイメージの削除 省略した場合、昇順を表すソートイメージが変更されます。 上記以外の値が指定された場合は、ソートイメージの変更は行いません。 deleteの指定時、すでに非表示の場合は何もしません。
戻り値	[Boolean]	true :ソートイメージの変更に成功しました。 false :ソートイメージの変更に失敗しました。
例外	なし	
説明	インデックスで指定された列のソートイメージを変更します。	

■isUpdatingメソッド

メソッド	isUpdating()	
引数	なし	
戻り値	[Boolean]	true :順次表示の処理が動作中です。 false :順次表示の処理は動作していません。
例外	なし	
説明	dispRowsOneByOneプロパティが有効のとき、順次表示の処理中であるかどうかを返します。	

■selectCellメソッド (注)

メソッド	selectCell(row, column)	
引数	Row [Number]	対象セルがある行のインデックスを指定します。 省略時は、現在フォーカスがある行を指定したとみなされます。また、この場合は、columnも省略する必要があります。
	Column [String]	対象セルがある列の列名(ViewColumnGridのname属性に指定した値)を指定します。 省略時は現在フォーカスがある列を指定したとみなされます。
戻り値	[Boolean]	true:セルの選択に成功しました。 false:セルの選択に失敗しました。
例外	なし	
説明	指定した行インデックスおよび列名に対応するセルを選択状態にします。 以下の場合は選択に失敗します。 ・ 指定した行インデックスが範囲外の値の場合 ・ 列名に対応する列がない場合 ・ 引数省略時にフォーカスしているセルがない場合 セルの選択に成功した場合、selectcellイベントが発生します。 対象セルがすでに選択状態でも成功しますが、その場合はselectcellイベントは発生しません。	

■deselectCellメソッド (注)

メソッド	deselectCell()
引数	なし
戻り値	なし
例外	なし

説明	現在選択されているセルを非選択にします。 選択されているセルがない場合は、何もしません。

注) focusModeプロパティの設定状況により、利用できるメソッドが以下のように異なります。

- focusModeプロパティが-1の場合
  以下のメソッドが利用できません。
  - selectCellメソッド
  - deselectCellメソッド

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

ポイント

テーブル部品のモデルデータ更新における画面更新の動作は、以下のようになります。

- テーブル全体を更新するケース
  - setPropertyを使用したデータ更新
  - データプロバイダを使用した行の挿入や削除
  - ソート
- 部分的に更新するケース
  - データプロバイダを使用した行の追加や置換

データプロバイダの使用方法については、“ユーザーズガイド”の“データプロバイダ”を参照してください。

2.3.4 ViewColumn

ViewColumnは、TableViewおよびTableEditのテーブルの列を定義する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

記述形式

```
<div rcf:type="TableEdit" ... >
  <div rcf:type="ViewColumn" rcf:name="xxx" ... >
    <div rcf:type="TextInput" ... ></div>
  </div>
  ...
</div>
```

ポイント

子要素には、TextInput、NumberInput、およびDateInputが指定できます。詳細は、“編集用TextInput、NumberInput、DateInputの制限について”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
renderer	render関数を持つオブジェクト	render関数は、セルの表示カスタマイズを指示するオブジェクトを返します。 オブジェクトの内容に従って、セルの内容文字列、ボーダー、背景色、文字の色などを変更できます。詳細は、“rendererプロパティ”を参照してください。	可	null	値	不可	不可
name	String	カラムの名前を指定します。TableViewまたはTableEditにあるViewColumn内で、一意の値を指定してください。一意でない値を指定した場合、エラーになります。	不可	ー	値	不可	不可
propertyName	String	モデルに指定した配列データに含まれるオブジェクトの持つプロパティのうち、ViewColumnが参照するプロパティ名を指定します。	可	nameプロパティの値	値	不可	不可
label	String	列ヘッダ部に表示する文字列を指定します。	可	nameプロパティの値	値	不可	不可
columnWidth	Number	列幅を指定します。単位はピクセルです。 最小値は10です。10未満の数値を指定した場合、エラーとなります。 指定しない場合は、TableViewまたはTableEditのdefaultColumnWidthプロパティの値となります。 TableViewまたはTableEditで複数行表示する場合は、個々に設定された列幅のうち、最大の値が有効になります。	可	指定なし	値	不可	不可
editable	Boolean	列のセルの内容を編集可能にするかどうかを指定します。TableViewの場合、本プロパティは無視されます。 ・ true: 編集可 ・ false: 編集不可	可	true	値	不可	不可
sortable	Boolean	列のセルの内容に基づいてソートを可能にするかどうかを指定します。TableEditの場合、本プロパティは無視されます。 ・ true: ソート可 ・ false: ソート不可	可	true	値	不可	不可
comparator	compare関数を持ったオブジェクト	ソートのときに使用するcompare関数を指定します。詳細は“comparatorプロパティ”を参照してください。	可	null	値	不可	不可
focusable	Boolean	セルのフォーカスを可能にするかどうかを指定します。TableViewの場合、本プロパティは無視されます。 ・ true:フォーカス可	可	true	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ false:フォーカス不可					
selectable	Boolean	セルの選択を可能にするかどうかを指定します。 ・ true: その列のセルが選択可能 ・ false: その列のセルが選択不可 TableViewの場合は無視されます。	可	true	値	不可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

propertyNameでのパス名の指定

propertyNameには、オブジェクトのプロパティだけではなく、パス名も指定できます。

以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var obj1 = {
    data1: [
     { name: 'AAA',
       data: {
              price: 1000,
              date: '20060608'}
     },
     { name: 'AAA',
       data: {
              price: 1000,
              date: '20060608'}
     }
   ...
     ]
  };
//]]>
</script>

 ...

<div rcf:type="Model" rcf:id="model1" rcf:object="obj1"></div>
<div rcf:type="TableView" rcf:data="{model1.data1}">
    <div rcf:type="ViewColumn" rcf:name="name" rcf:propertyName="name" rcf:label="品名"></div>
    <div rcf:type="ViewColumn" rcf:name="price" rcf:propertyName="data.price" rcf:label="価格">
      <div rcf:type="NumberInput"></div>
    </div>
    <div rcf:type="ViewColumn" rcf:name="date" rcf:propertyName="data.date" rcf:label="販売開始日"></div>
</div>
```

propertyNameで指定したプロパティが存在しない場合、以下のような動作となります。

- TableViewの場合
  空のセルが表示されます。

- TableEdit、DataGridの場合
  空のセルが表示されます。
  そのセルが編集可能の場合、値を入力すると以下のように動作します。

  - プロパティ名またはパス名の最後が存在していなかった場合
    新たなプロパティとして、入力値を設定します。

- パス名で途中のオブジェクトが存在していなかった場合(例えば、data.priceというパス式で、dataが存在していなかった場合)
  入力値は破棄されます。

rendererプロパティ

rendererプロパティには、render関数を持つオブジェクト(以降、セルレンダラと呼びます)を指定します。
render関数には、カスタマイズ対象のセルの位置を示すインデックスと、TableEditまたはTableViewとバインディングしているオブジェクトが渡されます。

ポイント

ここでのカスタマイズは、TableEdit、TableView、ViewColumnのスタイルプロパティで定義した設定よりも優先されます。

以下のプロパティを持ったオブジェクトを返すことで、セルの表示をカスタマイズできます。

プロパティ名	データ型	説明
color	String	セルの文字の色 セルやセルを含む行に対して、フォーカス、選択、マウスオーバーなどのイベントが発生した場合は、操作によるスタイルが優先されます。
backgroundColor	String	セルの背景色 セルやセルを含む行に対して、フォーカス、選択、マウスオーバーなどのイベントが発生した場合、操作によるスタイルが優先されます。
borderColor	String	セルのボーダーの色
borderStyle	String	セルのボーダーの種類
fontFamily	String	セルの文字のフォント名
fontSize	String	セルの文字のフォントサイズ(ピクセル数)
fontWeight	String	セルの文字のウェイト
textAlign	String	セルのテキストの行揃え
verticalAlign	String	セル内容の縦のデフォルト表示位置
value	String	実際に表示するセルの内容 ここで設定した値はバインディングしているモデルには反映されません。

セルレンダラの定義例を以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  var cellRenderer = {
    /**
     * @param row    行インデックス
     * @param column 列名
     * @param value データの値
     * @param data   TableEdit/TableViewがバインディングしているオブジェクト(配列)
     * @return セルの表示をカスタマイズするプロパティを設定したオブジェクト。
     *         カスタマイズしない場合はnullを返す。
     */
    render: function(row, column, value, data) {
      // 奇数行の場合は背景色を薄い緑にする
      if (row % 2 == 1) {
        return { backgroundColor: "#ddffff" };
      }
      return null;
```

```
    }
  };
//]]>
</script>
...
<div rcf:type="ViewColumn" rcf:renderer="cellRenderer" ... ></div>
...
```

セルレンダラのrender関数は、セルの部分更新のタイミングで呼び出されます。
部分更新で発生するイベントに対する描画の範囲を以下に示します。

イベント	描画の範囲
TableView、TableEditの表示時	・ 全セル
モデルの内容が変更されたとき	・ 特定のプロパティが変更:そのプロパティに対応するセル ・ 要素が追加:その要素に対応する行が追加され、その行の範囲のセル ・ 行の入れ替え:入れ替わりの対象となった行に含まれるセル ・ 行の削除:全セル
TableViewでソートが行われたとき	・ 全セル
マウスオーバー、フォーカス、選択など状態更新時	・ 状態が戻るときに関連するセル

comparatorプロパティ

TableViewの内容をソートするときに、使用するcompare関数を持ったオブジェクトを指定します。compare関数は2つの引数を持ち、その2つの引数aと引数bを比較して、以下の値を返します。

- a < b なら、0よりも小さい値
- a = b なら、0
- a > b なら、0よりも大きな値

comparatorプロパティに指定するオブジェクトの定義例を以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  var priceComparator = {
    /**
     * 各要素のpriceプロパティの値を比較する
     * @param a 価格1。priceプロパティが未定義の場合 null
     * @param b 価格2。priceプロパティが未定義の場合 null
     * @return a < b なら -1、a = b なら 0、a > b なら 1を返す
     */
    compare: function(a, b) {
      /* データの性質によっては、プロパティが未定義だったり、値がNaNだったりと
           異常データも考慮した比較が必要だが、ここでは簡単に単純な比較だけ行っている */
      if (a < b) return -1;
      if (a > b) return 1;
      return 0;
    }
  };
//]]>
</script>
...
<div rcf:type="ViewColumn" rcf:comparator="priceComparator" ... ></div>
...
```

編集用TextInput、NumberInput、DateInputの制限について

ViewColumnでは、TextInput、NumberInput、DateInputを子要素に定義して、入力時の動作をカスタマイズすることができます。これによって、セルの編集に使う子要素の動作を列ごとに設定できます。ただし、以下に示すプロパティだけが指定できます。また、指定された内容は編集中にだけ有効です。

TextInput

プロパティ		補足説明
プロパティ	maxLength	特になし
	imeMode IE6 IE7 IE8	Internet Explorerの場合にだけ有効
スタイルプロパティ	カラー	特になし
	フォント(lineHeightを除く)	特になし
	テキスト(textIndent、whiteSpaceを除く)	Firefoxの場合、wordSpacingも除く

NumberInput

プロパティ		補足説明
プロパティ	maxLength	特になし
	imeMode IE6 IE7 IE8	Internet Explorerの場合にだけ有効
	converter	特になし
スタイルプロパティ	カラー	特になし
	フォント(lineHeightを除く)	特になし
	テキスト(textIndent、whiteSpaceを除く)	Firefoxの場合、wordSpacingも除く

DateInput

プロパティ		補足説明
プロパティ	maxLength	特になし
	imeMode IE6 IE7 IE8	Internet Explorerの場合にだけ有効
	converter	特になし
スタイルプロパティ	カラー	特になし
	フォント(lineHeightを除く)	特になし
	テキスト(textIndent、whiteSpaceを除く)	Firefoxの場合、wordSpacingも除く

子要素に何も指定していない場合、以下のTextInputが指定されたとみなされます。

```
<div rcf:type="TextInput"></div>
```

編集用入力部品に指定できる機能付加部品について IE6 IE7 IE8

編集用入力部品に指定できる機能付加部品は、以下のとおりです。これら以外の機能付加部品は指定できません。

- Limiter (Internet Explorerだけ有効)
- NumeralOnlyLimiter (Internet Explorerだけ有効)

ー EnableCharTypeLimiter (Internet Explorerだけ有効)

スタイルプロパティ

本部品のスタイルプロパティでは、定義する列のスタイルを指定することができます。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
列ヘッダ	ccell	• TableViewの場合 rcf-TableView-ccell* • TableEditの場合 rcf-TableEdit-ccell* * は列番号(0～)を表します。	• フォント(lineHeightを除く) • セル
セル	cell	• TableViewの場合 rcf-TableView-cell* • TableEditの場合 rcf-TableEdit-ccell* * は列番号(0～)を表します。	• カラー • フォント(lineHeightを除く) • ボーダー(borderWidthを除く) • セル

詳細は、“2.9 スタイルプロパティ”を参照してください。

クラス名の列番号

CSSによりスタイルを設定する場合、クラス名の最後に列番号を追加します。
列番号は、以下のとおりになります。

ー ViewColumnGroupによる段組を行わない場合
列番号は左から順番に0、1、2 ... となります。

ー ViewColumnGroupにより段組を行った場合
以下のように、ViewColumnで定義した順番になります。

CSSで列のスタイルを定義するときの注意事項

CSSでTableView(TableEdit)の全体に対するスタイルとViewColumnの列のスタイルを指定する場合は、定義は以下の順番で記述してください。

1. TableView(TableEdit)の列ヘッダまたはセル全体に対するスタイル
2. ViewColumnの列ヘッダまたはセルの固有の列に対するスタイル

セルのスタイルの場合の例

```
.myStyle .rcf-TableView-cell {          ←セル全体に対するスタイルを
   background-color: yellow;              先に記述する
}

.myStyle .rcf-TableView-cell0 {
  background-color: blue;
}

.myStyle .rcf-TableView-cell2 {

}

 ...
```

定義の順番を逆にすると、セル全体に対するスタイルが優先されてしまうことがあります。

スタイルの優先順位

ViewColumnおよびTableView(TableEdit)のスタイルを設定した場合の優先順位は、以下のとおりになります。

列ヘッダの場合

1. スタイルプロパティで設定したViewColumnの列ヘッダのスタイル
2. スタイルプロパティで設定したTableView(TableEdit)の列ヘッダのスタイル
3. CSSで設定したViewColumnの列ヘッダのスタイル
4. CSSで設定したTableView(TableEdit)の列ヘッダのスタイル

セルの場合

1. ViewColumnのrendererで指定したスタイル
2. スタイルプロパティで設定したViewColumnのセルのスタイル
3. スタイルプロパティで設定したTableView(TableEdit)のセルのスタイル
4. CSSで設定したViewColumnのセルのスタイル
5. CSSで設定したTableView(TableEdit)のセルのスタイル

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.3.5 ViewColumnGrid IE 6 IE 7 IE 8

ViewColumnGridは、DataGridのテーブルの列を定義する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

注意

IE 6 IE 7 IE 8

ViewColumnGridは、Internet Explorerでだけ利用可能な部品です。

記述形式

```
<div rcf:type="DataGrid" ... >
  <div rcf:type="ViewColumnGrid" rcf:name="xxx" ... ></div>
  ...
</div>
```

ポイント

子要素には、TextInput、NumberInput、およびDateInputが指定できます。詳細は、“ViewColumn”の“編集用TextInput、NumberInput、DateInputの制限について”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
name	String	カラムの名前を指定します。DataGridにあるViewColumnGrid、ViewColumnCheck、ViewColumnTree、	不可	ー	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		ViewColumnSelect、ViewColumnImage内で、一意の値を指定してください。一意でない値を指定した場合、エラーになります。					
columnSpan	Number	セルの結合をする場合、結合するカラム数を指定します。	可	1	値	不可	不可
column Width	String	列幅を指定します。単位はピクセル(px)またはエム(em)です。 指定には、数値だけを指定する方法と、数値の後ろに単位を指定する方法があります。 数値だけを指定した場合、単位はピクセルとして扱われます。 px、em以外の単位を指定した場合には、指定した値は無効となり、DataGridのdefaultColumnWidthプロパティの値となります。 emを指定した場合、セルのfontFamily、fontSizeを基準とし、幅を設定します。 ピクセルとして指定した場合の最小値は10ピクセルです。指定した値が10ピクセル未満の場合は、エラーとなります。 複数行にわたる列ヘッダを定義する場合は、最初に定義された列幅の指定が優先されます。 列が結合されている場合には、以降の行において、最初に定義されている結合されていない列幅の指定が有効になり、以降の列幅は最初の列幅の定義を元に自動的に調整されます。 指定しない場合は、DataGridのdefaultColumnWidthプロパティの値となります。	可	指定なし	値	不可	不可
showFilter	Boolean	フィルタ用コンボボックスの表示/非表示を指定します。 ・ true: フィルタ用コンボボックスを表示します。 ・ false: フィルタ用コンボボックスを表示しません。	可	false	値	不可	不可
fixable	Boolean	列を固定表示するかどうかを指定します。 最後尾列に指定することはできません。 ・ true: 固定する ・ false: 固定しない	可	false	値	不可	不可
sortitem	Number	結合されたセルに対するソート対象を指定します。	可	0	値	不可	不可
renderer	Render関数を持つオ	render関数は、セルの表示カスタマイズを指示するオブジェクトを返します。 オブジェクトの内容にtextDecorationの	可	null	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
	ブジェクト	line-throughを指定することでセル単位の取り消し線の設定ができます。					
selected Renderer	Render関数を持つオブジェクト	render関数は、選択されたセルの表示カスタマイズを指示するオブジェクトを返します。	可	null	値	不可	不可

注) DataGridのfocusModeプロパティが0または1の場合にだけ、設定が有効となります。

ViewColumnのプロパティも指定できます。ただし、DataGridのfocusModeプロパティが-1の場合は、focusable、selectableの各プロパティは利用できません。詳細は“2.3.4 ViewColumn”の“プロパティ”を参照してください。なお、comparatorプロパティに指定した関数には、プレフィックス付きのデータを渡します。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

columnSpanプロパティを使用した列結合方法

結合する列の種類によって、方法が変わります。

ー 列ヘッダだけ、またはデータ行だけを結合する場合
列ヘッダとデータを別のViewColumnGroupで指定します。以下は、列ヘッダだけを結合する場合の例です。このとき、結合されている列ヘッダに対応するViewColumnGridのcolumnWidthプロパティに、セルの列幅をそれぞれ指定できます。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    data: [
      { name1: 'data1-1',  name2: 'data2-1',  name3: 'data3-1'},
      { name1: 'data1-2',  name2: 'data2-2',  name3: 'data3-2'},
      { name1: 'data1-3',  name2: 'data2-3',  name3: 'data3-3'},
      { name1: 'data1-4',  name2: 'data2-4',  name3: 'data3-4'},
      { name1: 'data1-5',  name2: 'data2-5',  name3: 'data3-5'}
    ]
  };
//]]>
</script>

<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}" ... >
  <div rcf:type="ViewColumnGroup" rcf:dispType="column">
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:label="ラベル2" rcf:columnSpan="2"></div>
  </div>
  <div rcf:type="ViewColumnGroup" rcf:dispType="data">
    <div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="name1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column4" rcf:propertyName="name2" rcf:columnWidth="80"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column5" rcf:propertyName="name3" rcf:columnWidth="80"></div>
  </div>
</div>
```

	ラベル1	ラベル2		
1	data1-1	data2-1	data3-1	
2	data1-2	data2-2	data3-2	
3	data1-3	data2-3	data3-3	
4	data1-4	data2-4	data3-4	
5	data1-5	data2-5	data3-5	

ー 列ヘッダとデータ行の両方を結合する場合(複数行表示)
以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    data: [
      {name1:'data1-1', name2:'data2-1', name3:'data3-1', name4:'data4-1', name5:'data5-1'},
      {name1:'data1-2', name2:'data2-2', name3:'data3-2', name4:'data4-2', name5:'data5-2'},
      {name1:'data1-3', name2:'data2-3', name3:'data3-3', name4:'data4-3', name5:'data5-3'},
      {name1:'data1-4', name2:'data2-4', name3:'data3-4', name4:'data4-4', name5:'data5-4'},
      {name1:'data1-5', name2:'data2-5', name3:'data3-5', name4:'data4-5', name5:'data5-5'}
    ]
  };
//]]>
</script>

<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}" ... >
  <div rcf:type="ViewColumnGroup">
    <div rcf:type="ViewColumnGrid" rcf:name="col1" rcf:propertyName="name1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="col2" rcf:propertyName="name2" rcf:label="ラベル2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="col3" rcf:propertyName="name3" rcf:label="ラベル3"></div>
  </div>
  <div rcf:type="ViewColumnGroup">
    <div rcf:type="ViewColumnGrid" rcf:name="col4" rcf:propertyName="name4" rcf:label="ラベル4"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="col5" rcf:propertyName="name5" rcf:label="ラベル5"
      rcf:columnSpan="2"></div>
  </div>
</div>
```

	ラベル1	ラベル2	ラベル3	
	ラベル4	ラベル5		
1	data1-1	data2-1	data3-1	
	data4-1	data5-1		
2	data1-2	data2-2	data3-2	
	data4-2	data5-2		
3	data1-3	data2-3	data3-3	
	data4-3	data5-3		
4	data1-4	data2-4	data3-4	
	data4-4	data5-4		
5	data1-5	data2-5	data3-5	
	data4-5	data5-5		

テーブルにフィルタ用コンボボックスを表示するための指定方法

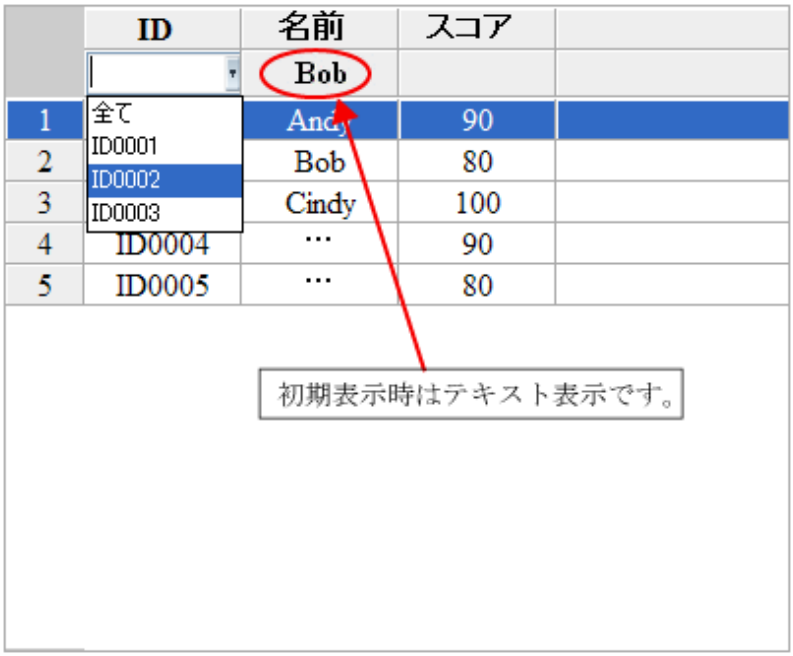
フィルタ用コンボボックスの表示および非表示は、showFilterプロパティで列ごとに指定することができます。

以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    scores: [
      { id: 'ID0001',  name: 'Andy',  score: 90 },
      { id: 'ID0002',  name: 'Bob',   score: 80 },
      { id: 'ID0003',  name: 'Cindy',  score: 100 },
      { id: 'ID0004',  name: '...',  score: 90 },
      { id: 'ID0005',  name: '...',  score: 80 }
    ],
    flist: {
        column1: ['ID0001', 'ID0002', 'ID0003'],
        column2: ['2', 'Andy', 'Bob'],
        column3: []
    }
  };

  // イベント定義
  eventMap = {
    grid1:{
      selectfilter: selectFilter,
      ...
    }
  };

  // イベント登録
  RCF.addInitializedListener(
    function(eventObject) {
      rcf.event.EventRegistrar.registerEvents(eventMap, "eventMap");
    }
  );

  // フィルタ用コンボボックスのリストが選択された場合に呼び出される関数
  function selectFilter() {
    // ここでフィルタリングしてください。
```

```
    ...
    // ここですべてのフィルタ用コンボボックスの選択内容を取得してください。
    var array = getFilterList();
    // ここでデータオブジェクトを作成してください。
    var changeScores = new Array();
    ...
    // ここでDataGridのdataプロパティを更新してください。
    grid1.getDataProvider("data").setData(changeScores);
  }

//]]>
</script>
...
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:id="grid1" rcf:data="{model1.scores}"
   rcf:filterList="{model1.flist}" rcf:filterCount="5">
<div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="id" rcf:label="ID" rcf:showFilter="true">
</div>
<div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name" rcf:label="名前" rcf:showFilter="true">
</div>
<div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="score"
   rcf:label="スコア" rcf:showFilter="false">
    <div rcf:type="NumberInput"></div>
</div>
</div>
```

以下は、フィルタ用コンボボックスの表示例です。



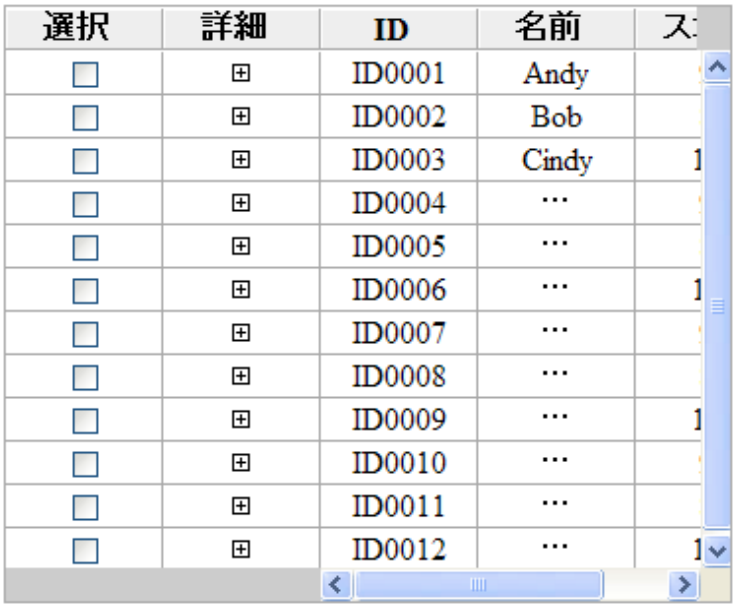
fixableプロパティ指定方法

以下にプロパティの指定方法を示します。

```
<div rcf:type="DataGrid" ... >
  <div rcf:type="ViewColumnCheck" rcf:name="check" rcf:propertyName="check" rcf:label="選択"></div>
  <div rcf:type="ViewColumnTree" rcf:name="detail"  rcf:label="詳細" rcf:fixable="true"></div>
  <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="name1" rcf:label="ID"></div>
  <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name2" rcf:label="名前"></div>
  <div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="name3" rcf:label="スコア"></div>
```

```
...
</div>
```

以下は、“詳細”列を固定列に指定した場合の固定列指定機能の表示例です。



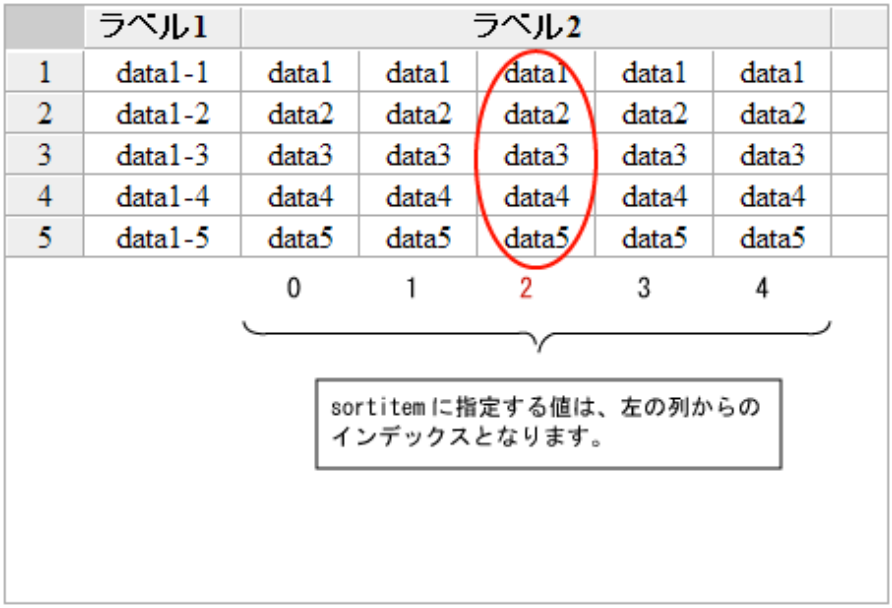
複数の列を固定列に指定した場合、最後に指定された列が固定列となります。

sortitemプロパティの指定方法

列ヘッダを結合した場合について説明します。
sortitemに2を指定した場合、図中の赤で囲まれた箇所がソートされます。

```
...
<div rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnGroup" rcf:dispType="column">
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:label="ラベル2"
      rcf:columnSpan="5" rcf:sortitem="2"></div>
  </div>
...
</div>
```

上記の例でのソートの対象データを以下に示します。



rendererおよびselectedRendererプロパティ

以下のプロパティを持ったオブジェクトを返すことで、セルの表示をカスタマイズできます。

プロパティ名	データ型	説明
color	String	セルの文字の色 セルやセルを含む行に対して、フォーカス、選択、マウスオーバーなどのイベントが発生した場合は、操作によるスタイルが優先されます。
backgroundColor	String	セルの背景色 セルやセルを含む行に対して、フォーカス、選択、マウスオーバーなどのイベントが発生した場合、操作によるスタイルが優先されます。
borderColor	String	セルのボーダーの色
borderStyle	String	セルのボーダーの種類
fontFamily	String	セルの文字のフォント名
fontSize	String	セルの文字のフォントサイズ(ピクセル数)
fontWeight	String	セルの文字のウェイト
textAlign	String	セルのテキストの行揃え
verticalAlign	String	セル内容の縦のデフォルト表示位置
value	String	実際に表示するセルの内容 ここで設定した値は、バインディングしているモデルには反映されません。
textDecoration	String	セルの文字装飾 line-throughを設定すると、セル単位で取り消し線が設定できます。

ポイント

取り消し線(textDecoration)の設定には、以下の2つの方法があります。

ー スタイルプロパティを使用して、列単位の取り消し線の設定が可能です。

ー rendererプロパティを使用して、セル単位の取り消し線の設定が可能です。

スタイルプロパティ

本部品のスタイルプロパティでは、定義する列のスタイルを指定することができます。

パーツ名	プレフィックス	クラス名	使用可能なスタイル
列ヘッダ	ccell	rcf-DataGrid-ccell* * は列番号(0～)を表します。	・ フォント(lineHeightを除く) ・ セル
セル	cell	rcf-DataGrid-cell* * は列番号(0～)を表します。	・ カラー ・ フォント(lineHeightを除く) ・ ボーダー(borderWidthを除く) ・ セル(注)
ツリー展開時の詳細データ表示セル	detail	rcf-DataGrid-detail* * は列番号(0～)を表します。	・ カラー ・ フォント(lineHeightを除く) ・ ボーダー(borderWidthを除く) ・ セル(注)

注) DataGridでは、textDecorationを指定することができます。
textDecorationでは、文字装飾を指定します。CSSのtext-decorationプロパティの値を指定できます。
line-throughを指定すると、取り消し線が設定できます。Internet Explorerでは、blinkは指定できません。

ポイント

取り消し線(textDecoration)の設定には、以下の2つの方法があります。

- スタイルプロパティを使用して、列単位の取り消し線の設定が可能です。
- renderおよびselectedRendererプロパティを使用して、セル単位の取り消し線の設定が可能です。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.3.6 ViewColumnGroup

ViewColumnGroupは、複数のViewColumnまたはViewColumnGridなどをグループ化する部品です。テーブルのヘッダやデータを複数行表示する場合に利用します。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

ポイント

ダミーセルの表示について

2つ以上のViewColumnGroupが定義されていて、それぞれのViewColumnGroupで定義しているViewColumnまたはViewColumnGridの数が異なる場合、数が少ないほうにダミーのViewColumnまたはViewColumnGridが自動的に補われてダミーのセルが表示されます。

記述形式

```
<div rcf:type="TableEdit" ... >
  <div rcf:type="ViewColumnGroup">
    <div rcf:type="ViewColumn" ... ></div>
    ...
  </div>
  ...
</div>
```

ポイント

- 子要素には、ViewColumn、ViewColumnGrid、ViewColumnCheck、ViewColumnTree、ViewColumnSelect、およびViewColumnImageが指定できます。
- TableEditまたはTableView直下のViewColumn、およびDataGrid直下のViewColumnGridは、見えないViewColumnGroupで囲まれているものとして扱われます。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
dispType	String	列ヘッダおよびデータ行の表示の有無を、以下の文字列で指定します。 ・column: 列ヘッダの表示形式を指定 ・data: データ行の表示形式を指定 ・both: 列ヘッダ、データ行の表示形式を指定 ・detail: 詳細データの表示形式を指定 DataGridで使用します。TableView、TableEditでは使用できません。	可	both	値	不可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

dispTypeプロパティ指定方法

以下にdispTypeプロパティの指定方法と表示イメージを示します。

列ヘッダが複数行でデータ行を一行表示する場合

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    data: [
```

```
      { name1: 'data1-1',  name2: 'data2-1',  name3: 'data3-1' },
      { name1: 'data1-2',  name2: 'data2-2',  name3: 'data3-2' },
      { name1: 'data1-3',  name2: 'data2-3',  name3: 'data3-3' },
      { name1: 'data1-4',  name2: 'data2-4',  name3: 'data3-4' },
      { name1: 'data1-5',  name2: 'data2-5',  name3: 'data3-5' }
    ]
  };
//]]>
</script>
  :
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}" ... >
  <div rcf:type="ViewColumnGroup" rcf:dispType="both">
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="name1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name2" rcf:label="ラベル2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="name3" rcf:label="ラベル3"></div>
  </div>
  <div rcf:type="ViewColumnGroup" rcf:dispType="column">
    <div rcf:type="ViewColumnGrid" rcf:name="column4" rcf:label="ラベル4"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column5" rcf:label="ラベル5"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column6" rcf:label="ラベル6"></div>
  </div>
</div>
```

	ラベル1	ラベル2	ラベル3	
	ラベル4	ラベル5	ラベル6	
1	data1-1	data2-1	data3-1	
2	data1-2	data2-2	data3-2	
3	data1-3	data2-3	data3-3	
4	data1-4	data2-4	data3-4	
5	data1-5	data2-5	data3-5	

列ヘッダおよび行ヘッダを複数行表示する場合

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    data: [
      { name1:'data1-1', name2:'data2-1', name3:'data3-1', name4: 'data4-1', name5:'data5-1', name6: 'data6-1' },
      { name1:'data1-2', name2:'data2-2', name3:'data3-2', name4: 'data4-2', name5:'data5-2', name6: 'data6-2' },
      { name1:'data1-3', name2:'data2-3', name3:'data3-3', name4: 'data4-3', name5:'data5-3', name6: 'data6-3' },
      { name1:'data1-4', name2:'data2-4', name3:'data3-4', name4: 'data4-4', name5:'data5-4', name6: 'data6-4' },
      { name1:'data1-5', name2:'data2-5', name3:'data3-5', name4: 'data4-5', name5:'data5-5', name6: 'data6-5' }
    ]
  };
//]]>
</script>
 ...
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
```

```
<div rcf:type="DataGrid" rcf:data="{model1.data}" ... >
  <div rcf:type="ViewColumnGroup" rcf:dispType="both">
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="name1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name2" rcf:label="ラベル2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="name3" rcf:label="ラベル3"></div>
  </div>
  <div rcf:type="ViewColumnGroup" rcf:dispType="both">
    <div rcf:type="ViewColumnGrid" rcf:name="column4" rcf:propertyName="name4" rcf:label="ラベル4"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column5" rcf:propertyName="name5" rcf:label="ラベル5"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column6" rcf:propertyName="name6" rcf:label="ラベル6"></div>
  </div>
</div>
```

	ラベル1 ラベル4	ラベル2 ラベル5	ラベル3 ラベル6	
1	data1-1	data2-1	data3-1	
	data4-1	data5-1	data6-1	
2	data1-2	data2-2	data3-2	
	data4-2	data5-2	data6-2	
3	data1-3	data2-3	data3-3	
	data4-3	data5-3	data6-3	
4	data1-4	data2-4	data3-4	
	data4-4	data5-4	data6-4	
5	data1-5	data2-5	data3-5	
	data4-5	data5-5	data6-5	

詳細データのセルを定義する場合

詳細データセルの形式を、ViewColumnGroupで囲まれたViewColumnGridにより指定します。

ViewColumnGroupのdispTypeプロパティに“detail”を指定することで、ViewColumnGroup配下に指定したViewColumnGridを詳細データセルとして扱います。

スタイルプロパティ

固有のスタイルプロパティはありません。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.3.7 ViewColumnCheck IE 6 IE 7 IE 8

ViewColumnCheckは、DataGridのテーブルで指定した列にチェックボックスを配置する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ

- JavaScript API

注意

IE 6 IE 7 IE 8

ViewColumnCheckは、Internet Explorerでだけ利用可能な部品です。

記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnCheck" rcf:name="checkbox1" rcf:propertyName="check" rcf:label="削除"></div>
  ...
</div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
name	String	カラムの名前を指定します。 DataGridにあるViewColumnGrid、ViewColumnCheck、ViewColumnTree、ViewColumnSelect、ViewColumnImage内で、一意の値を指定してください。一意でない値を指定した場合、エラーになります。	不可	ー	値	不可	不可
propertyName	String	DataGridのオブジェクトを含むdataプロパティのオブジェクトで、チェックボックスの状態を示すプロパティ名を指定します。	可	ー	バインド	可	可
label	String	列ヘッダ部に表示する文字列を指定します。	可	nameプロパティの値	値	不可	不可
columnWidth	String	列幅を指定します。単位はピクセル(px)またはエム(em)です。 指定には、数値だけを指定する方法と、数値の後ろに単位を指定する方法があります。 数値だけを指定した場合、単位はピクセルとして扱われます。 px、em以外の単位を指定した場合には、指定した値は無効となり、DataGridのdefaultColumnWidthプロパティの値となります。	可	指定なし	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		emを指定した場合、セルのfontFamily、fontSizeを基準とし、幅を設定します。 ピクセルとして指定した場合の最小値は10ピクセルです。指定した値が10ピクセル未満の場合は、エラーとなります。 指定しない場合は、DataGridのdefaultColumnWidthプロパティの値となります。					

注意

- 配置列に対して、編集はできません。
- 複数のチェックボックスを設定することはできません。複数の設定を行った場合は、エラーになります。
- ソートが行われた場合でも、チェックボックスの状態は保持されます。

ViewColumnGridのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.3.5 ViewColumnGrid ”の“プロパティ”を参照してください。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

propertyNameプロパティの指定値

ViewColumnCheckのpropertyNameプロパティに指定したデータオブジェクトの値には、チェックボックスの初期状態を表す以下の数値を指定します。

チェックボックスの状態は、0から7までの数値で指定します。数値の意味は、以下のとおりです。

- ビット番号0 (LSB: Least Significant Bit):チェックボックスの選択状態
- ビット番号1:活性状態(使用可否)
- ビット番号2 (MSB: Most Significant Bit):表示状態

設定値 (括弧内の数字はビット番号)				説明
	表示 (2)	活性 (1)	選択 (0)	
0～7以外	-	-	-	チェックボックスを表示しません。
0～3	0	0/1	0/1	
4	1	0	0	チェックボックスを使用不可 + 非選択状態で表示します。
5	1	0	1	チェックボックスを使用不可 + 選択状態で表示します。
6	1	1	0	チェックボックスを使用可 + 非選択状態で表示します。
7	1	1	1	チェックボックスを使用可 + 選択状態で表示します。

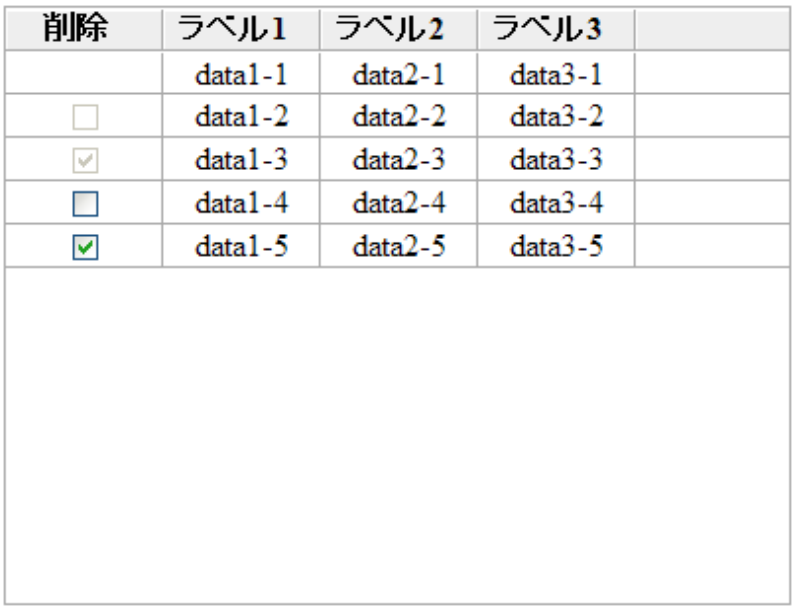
ソート順は、ViewColumn部品と同様にcomparatorプロパティで指定できます。comparatorプロパティの指定がない場合、上表の設定値を数値として比較し、ソートします。(“0～7以外”は0とみなします)。

ViewColumnCheckの指定方法

以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    data: [
      { check: 3, name1: 'data1-1',  name2: 'data2-1',  name3: 'data3-1'},
      { check: 4, name1: 'data1-2',  name2: 'data2-2',  name3: 'data3-2'},
      { check: 5, name1: 'data1-3',  name2: 'data2-3',  name3: 'data3-3'},
      { check: 6, name1: 'data1-4',  name2: 'data2-4',  name3: 'data3-4'},
      { check: 7, name1: 'data1-5',  name2: 'data2-5',  name3: 'data3-5'}
    ]
  };
  var checkComparator = {
    compare: function(a, b) {
      /* ここでは簡単に単純な比較だけ行っている。
         活性、非活性にかかわらず、選択されていないチェックボックスが先、
         選択されているチェックボックスが後になるように比較する。
         表示されていないチェックボックスは最後とする。 */
      if ((a & 0x4) == 0 && (b & 0x4) == 0) return 0; /* 非表示なら等しいとみなす */
      if ((a & 0x4) == (b & 0x4)) { /* 表示同士なら選択状態で比較する */
        if ((a & 0x1) > (b & 0x1)) return 1;
        if ((a & 0x1) < (b & 0x1)) return -1;
        return 0;
      }
      if ((a & 0x4) < (b & 0x4)) { /* 表示は非表示より先 */
        return 1;
      }
      else {
        return -1;
      }
    }
  };
//]]>
</script>

<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnGroup">
    <div rcf:type="ViewColumnCheck" rcf:name="checkbox1" rcf:propertyName="check"
       rcf:label="削除" rcf:comparator="checkComparator"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="name1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name2" rcf:label="ラベル2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="name3" rcf:label="ラベル3"></div>
  </div>
</div>
```

チェックボックスを表示する場合のイメージは以下のようになります。



スタイルプロパティ

ViewColumnGridのスタイルプロパティと同様です。“2.3.5 ViewColumnGrid ”の“スタイルプロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.3.8 ViewColumnTree IE 6 IE 7 IE 8

ViewColumnTreeは、DataGridのテーブルで指定した列にツリー展開用のボタンを配置する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

注意

IE 6 IE 7 IE 8

ViewColumnTreeは、Internet Explorerでだけ利用可能な部品です。

記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:id="grid1" rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnTree" rcf:name="detail1" rcf:label="詳細"></div>
  ...
</div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
name	String	カラムの名前を指定します。 DataGridにあるViewColumnGrid、ViewColumnCheck、ViewColumnTree、ViewColumnSelect、ViewColumnImage内で、一意の値を指定してください。一意でない値を指定した場合、エラーになります。	不可	ー	値	不可	不可
label	String	列ヘッダ部に表示する文字列を指定します。	可	nameプロパティの値	値	不可	不可
columnWidth	String	列幅を指定します。単位はピクセル(px)またはエム(em)です。 指定には、数値だけを指定する方法と、数値の後ろに単位を指定する方法があります。 数値だけを指定した場合、単位はピクセルとして扱われます。 px、em以外の単位を指定した場合には、指定した値は無効となり、DataGridのdefaultColumnWidthプロパティの値となります。 emを指定した場合、セルのfontFamily、fontSizeを基準とし、幅を設定します。 ピクセルとして指定した場合の最小値は10ピクセルです。指定した値が10ピクセル未満の場合は、エラーとなります。 指定しない場合は、DataGridのdefaultColumnWidthプロパティの値となります。	可	指定なし	値	不可	不可
focusable(注)	Boolean	セル(ツリーの展開ボタンがあるセル)のフォーカスを可能にするかどうかを指定します。 ・ true: フォーカス可 ・ false: フォーカス不可	可	true	値	不可	不可
selectable(注)	Boolean	セル(ツリーの展開ボタンがあるセル)の選択を可能にするかどうかを指定します。 ・ true: その列のセルが選択可能	可	true	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		• false: その列のセルが選択不可					

注) DataGridのfocusModeプロパティが0または1の場合にだけ、設定が有効となります。

注意

- 配置列に対して、編集はできません。
- 配置列に対して、ソートはできません。
- 複数のツリーを設定することはできません。複数の設定を行った場合は、エラーになります。
- ツリーの展開は、ViewColumnGroupに詳細データ(detail)を追加することにより可能になります。

ViewColumnGridのプロパティも指定できます。ただし、propertyNameプロパティは利用できません。詳細は、“2.3.5 ViewColumnGrid ”の“プロパティ”を参照してください。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

ツリー展開ボタンと詳細データの指定方法

以下にツリー展開ボタンおよび詳細データの指定方法を示します。

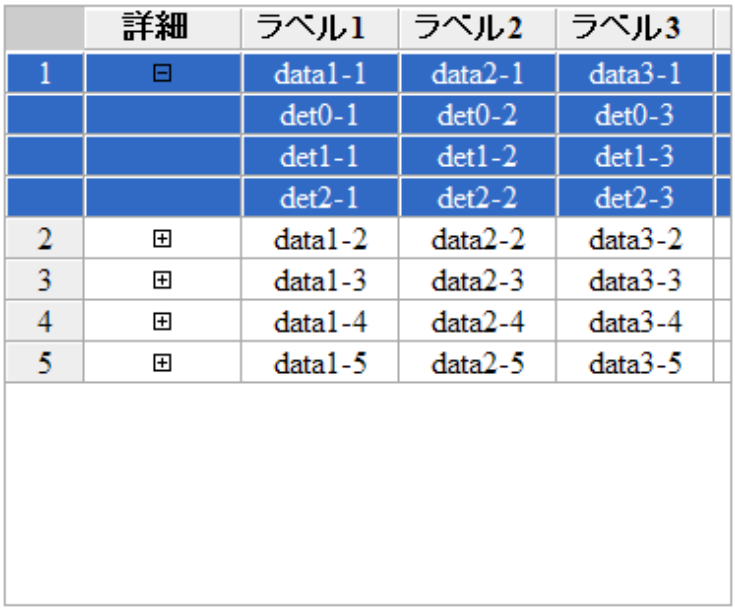
詳細データの指定と表示は、DataGridのonOpenDetailイベントリスナで設定します。

フレームワークの呼出し元で、onOpenDetailイベントで展開通知を受け取った際に表示する詳細データを指定して、JavaScript APIのshowDetailメソッドを呼び出すことにより、詳細データがツリー展開されます。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    data: [
      { name1: 'data1-1',  name2: 'data2-1',  name3: 'data3-1'},
      { name1: 'data1-2',  name2: 'data2-2',  name3: 'data3-2'},
      { name1: 'data1-3',  name2: 'data2-3',  name3: 'data3-3'},
      { name1: 'data1-4',  name2: 'data2-4',  name3: 'data3-4'},
      { name1: 'data1-5',  name2: 'data2-5',  name3: 'data3-5'}
    ]
  };

  // イベント登録
  eventMap = {
    grid1:{
      opendetail: createDetail,
      ...
    }
  };

  // イベント登録
  RCF.addInitializedListener(
    function(eventObject) {
      rcf.event.EventRegistrar.registerEvents(eventMap, "eventMap");
    }
  );

// 詳細展開用ボタンがクリックされた場合に呼び出される関数
  function createDetail(eventObj) {
    // ここで詳細データのオブジェクトを作成してください。
    var detailData1 = {
      data : [
        {dcolumn1: 'det0-1', dcolumn2: 'det0-2', dcolumn3: 'det0-3'},
        {dcolumn1: 'det1-1', dcolumn2: 'det1-2', dcolumn3: 'det1-3'},
```

```
        {dcolumn1: 'det2-1', dcolumn2: 'det2-2', dcolumn3: 'det2-3'}
      ]
    };
    // ここでJavaScriptAPIを呼び出して詳細データを表示してください。
    grid1.showDetail( eventObj, detailData1, "tree1");
  }

  // ユーザ定義関数
  function createDetailAll(eventObj) {
    // ここで詳細データのオブジェクトを作成してください。
    var allData = new Array();
    ...
    // ここでJavaScriptAPIを呼び出して詳細データを表示してください。
    grid1.openDetailAll( allData, "tree2");
  }
//]]>
</script>

<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:id="grid1" rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnGroup">
    <div rcf:type="ViewColumnTree" rcf:name="detail1" rcf:label="詳細"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="name1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="name2" rcf:label="ラベル2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column3" rcf:propertyName="name3" rcf:label="ラベル3"></div>
  </div>
  <div rcf:id="tree1" rcf:type="ViewColumnGroup" rcf:dispType="detail">
    <div rcf:type="ViewColumnGrid" rcf:name="dummy"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="dcolumn1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="dcolumn2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="dcolumn3"></div>
  </div>
  <div rcf:id="tree2" rcf:type="ViewColumnGroup" rcf:dispType="detail">
    <div rcf:type="ViewColumnGrid" rcf:name="dcolumn4"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="dcolumn5" rcf:columnSpan="2"></div>
  </div>
</div>

  <div rcf:type="Button" rcf:onClick="createDetailAll()">全展開</div>
```

詳細データを表示した例を以下に示します。



スタイルプロパティ

ViewColumnGridのスタイルプロパティと同様です。“2.3.5 ViewColumnGrid ”の“スタイルプロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.3.9 ViewColumnSelect IE 6 IE 7 IE 8

ViewColumnSelectは、DataGridのテーブルで指定した列にプルダウン(リストボックス)を配置する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

IE 6 IE 7 IE 8

ViewColumnSelectは、Internet Explorerでだけ利用可能な部品です。

記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}">
  <div rcf:type="ViewColumnSelect" rcf:name="list1" rcf:propertyName="list" rcf:label="選択"></div>
  ...
</div>
```

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
name	String	カラムの名前を指定します。DataGridにあるViewColumnGrid、ViewColumnCheck、ViewColumnTree、ViewColumnSelect、ViewColumnImage内で、一意の値を指定してください。一意でない値を指定した場合、エラーになります。	不可	―	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
propertyName	String	DataGridのオブジェクトを含むdataプロパティのオブジェクトで、プルダウンの選択項目を示すプロパティ名を指定します。 セルのプルダウンを無効にする場合は、dataプロパティの該当オブジェクトにnullを指定します。	可	―	バインド	可	可
label	String	列ヘッダ部に表示する文字列を指定します。	可	nameプロパティの値	値	不可	不可

注意

- 配置列に対して、編集はできません。
- 配置列に対して、ソートはできません。

ViewColumnGridのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.3.5 ViewColumnGrid ”の“プロパティ”を参照してください。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

propertyNameの指定方法

ViewColumnSelectにおいて、指定するpropertyNameが示すオブジェクトのデータには、以下のように初期選択位置および選択項目を指定します。

形式

```
['初期選択位置', '選択項目1', '選択項目2', ... ]
```

指定項目の説明

指定項目	説明
初期選択位置	プルダウンの初期表示時の選択位置の先頭(0)からのインデックスを指定します。 本指定がインデックスとして認識できない値のとき、初期選択位置は、先頭(0)のリスト位置とし、指定された値をプルダウンの選択項目とします。
選択項目n	プルダウンに表示する選択項目を指定します。 初期選択位置以外の指定は、すべて選択項目として表示します。

ViewColumnSelectの指定方法

以下に例を示します。

```
...
<script type="text/javascript">

//<![CDATA[
  var modelData = {
    data: [
      {list: ['list1','list2','list3'],   name1:'data1-1', name2:'data2-1', name3:'data3-1'},
      {list: ['1','list1','list2','list3'], name1:'data1-2', name2:'data2-2', name3:'data3-2'},
      {list: null,                name1:'data1-3', name2:'data2-3', name3:'data3-3'},
      {list: ['0','list1','list2'],       name1:'data1-4', name2:'data2-4', name3:'data3-4'},
```

```
      {list: ['2','list1','list2','list3'], name1:'data1-5', name2:'data2-5', name3:'data3-5'}
    ]
  };
//]]>

</script>
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}">
    <div rcf:type="ViewColumnSelect" rcf:name="list1" rcf:propertyName="list" rcf:label="選択"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="name1" rcf:propertyName="name1" rcf:label="ラベル1"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="name2" rcf:propertyName="name2" rcf:label="ラベル2"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="name3" rcf:propertyName="name3" rcf:label="ラベル3"></div>
</div>
```

上記プロパティの設定により表示されるDataGridプルダウンのイメージを、以下に示します。

スタイルプロパティ

ViewColumnGridのスタイルプロパティと同様です。“2.3.5 ViewColumnGrid ”の“スタイルプロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.3.10 ViewColumnImage IE 6 IE 7 IE 8

ViewColumnImageは、DataGridのテーブルで指定した列に画像ファイルを配置する部品です。

- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

注意

IE 6 IE 7 IE 8

ViewColumnImageは、Internet Explorerでだけ利用可能な部品です。

記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.data}">
    ...
    <div rcf:type="ViewColumnImage" rcf:name="img1" rcf:propertyName="img" rcf:label="イメージ"></div>
    ...
</div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
name	String	カラムの名前を指定します。 DataGridにあるViewColumnGrid、ViewColumnCheck、ViewColumnTree、ViewColumnSelect、ViewColumnImage内で、一意の値を指定してください。一意でない値を指定した場合、エラーになります。	不可	ー	値	不可	不可
propertyName	String	モデルに指定した配列データに含まれるオブジェクトの持つプロパティのうち、表示画像のURLが指定されているプロパティ名を指定します。 URLには、クエリ文字列およびURLリライティングで用いるセションIDを付加することができます。詳細は、“ユーザーズガイド”を参照してください。 セル単位で画像を非表示にする場合は、該当オブジェクトにnullを指定します。	可	ー	バインド	可	可
label	String	列ヘッダ部に表示する文字列を指定します。	可	nameプロパティの値	値	不可	不可
fixedImageSize	Object	配置する画像ファイルのサイズ(高さ、幅)を指定します。単位はピクセルです。同一のViewColumnImage内でサイズ(高さ、幅)が異なる画像ファイルを表示する場合は、画像ファイルは同じ	可	指定した画像ファイルの	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		サイズ(高さ、幅)にしてください。 画像ファイルが指定サイズと異なる場合は、指定サイズに拡大・縮小されます。(注) オブジェクトの指定方法は“fixedImageSizeプロパティの指定方法”を参照してください。		サイズ			

注) fixedImageSizeプロパティを指定した場合、ViewColumnImage内で表示する画像は強制的に指定したサイズに拡大・縮小されるため、画像の見映えが異なる場合があります。

注意

- 配置列に対して、編集はできません。
- 配置列に対して、ソートはできません。

ViewColumnGridのプロパティも指定できます。ここで説明を省略しているプロパティの詳細は、“2.3.5 ViewColumnGrid ”の“プロパティ”を参照してください。

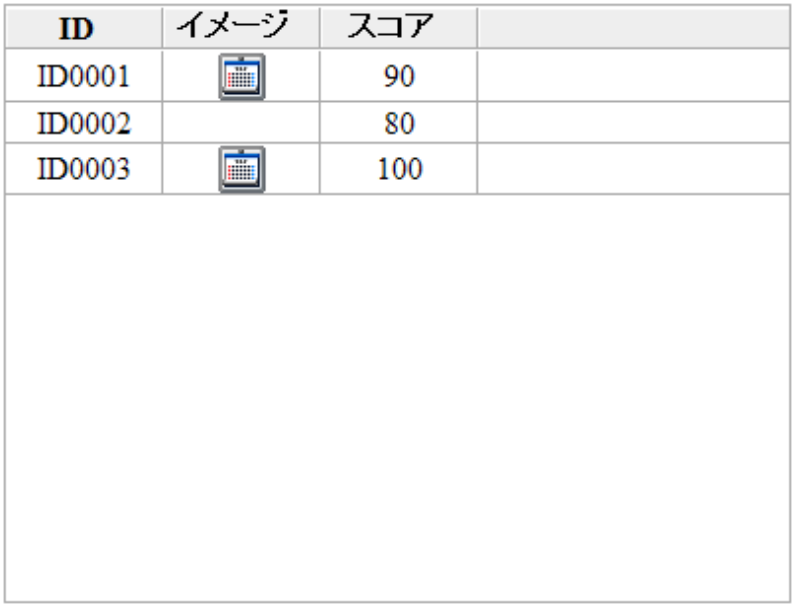
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

ViewColumnImageの指定方法

以下に例を示します。

```
...
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    scores: [
      { id: 'ID0001',  img: 'http://xxx/xxx/img/xxx.gif', score: 90 },
      { id: 'ID0002',  img: null,                        score: 80 },
      { id: 'ID0003',  img: 'http://xxx/xxx/img/xxx.gif',  score: 100 }
    ]
  };
//]]>
</script>
...
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="DataGrid" rcf:data="{model1.scores}">
    <div rcf:type="ViewColumnGrid" rcf:name="column1" rcf:propertyName="id" rcf:label="ID"></div>
    <div rcf:type="ViewColumnImage" rcf:name="img1" rcf:propertyName="img" rcf:label="イメージ"></div>
    <div rcf:type="ViewColumnGrid" rcf:name="column2" rcf:propertyName="score" rcf:label="スコア">
        <div rcf:type="NumberInput"></div>
    </div>
</div>
...
```

上記の指定を行った場合の画像の表示は、以下のようになります。



fixedImageSizeプロパティの指定方法

fixedImageSizeプロパティを指定する形式について、以下に示します。

heightおよびwidthは、それぞれ数値で指定してください。(単位はピクセル)

```
{
    height:  25,    //画像の表示サイズ(高さ)
    width:   20     //画像の表示サイズ(幅)
}
```

heightまたはwidthだけの指定も可能です。省略した場合は、指定した画像ファイルのサイズで表示されます。

スタイルプロパティ

ViewColumnGridのスタイルプロパティと同様です。“2.3.5 ViewColumnGrid ”の“スタイルプロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

2.4 カレンダ部品

カレンダ部品は、カレンダを表示する部品です。画面に直接カレンダを表示したり、ポップアップでカレンダを表示させることができます。

ここでは、カレンダ部品の設定内容、および設定方法について説明します。

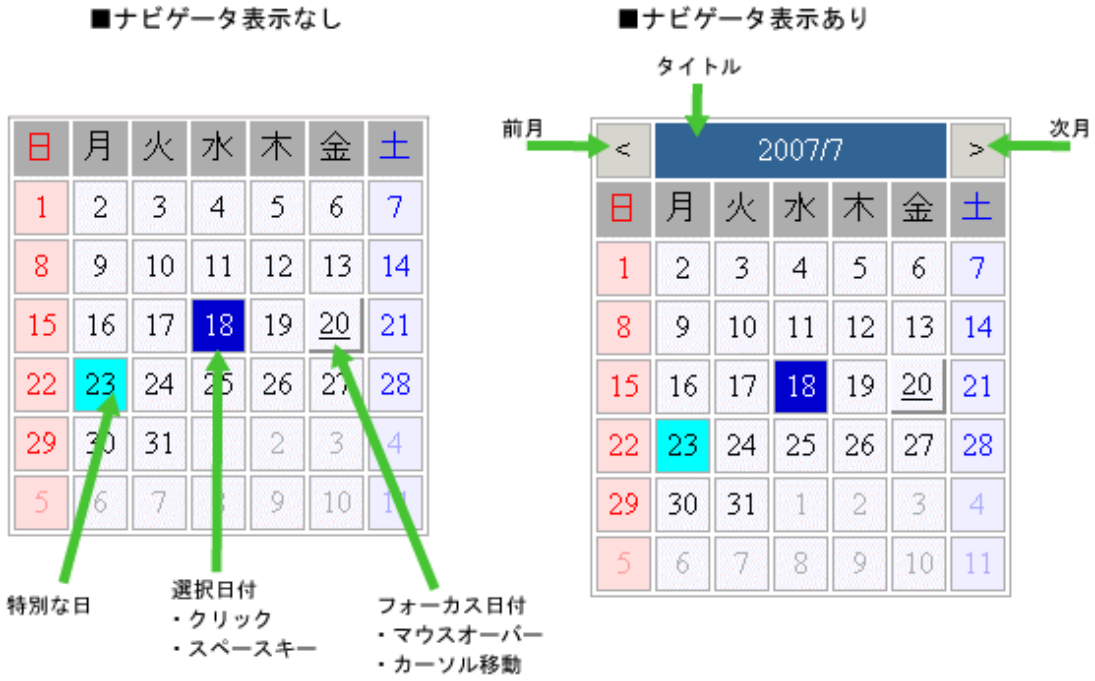
2.4.1 Calendar

Calendarは、カレンダを表示する部品です。表示されたカレンダから日付を指定することもできます。

- 表示例
- 記述形式
- プロパティ

- スタイルプロパティ
- イベントリスナ
- JavaScript API

表示例



記述形式

```
<div rcf:type="Calendar" ... ></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は、前後に改行コードが挿入されて表示されます。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
year	Number	カレンダ表示年を指定します。(1以上)	可	システム時計の今年	値	可	不可
month	Number	カレンダ表示月(0-11)を指定します。	可	システム時計の今月	値	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
firstDayOfWeek	Number	カレンダ左端カラムの曜日を指定します。値は、日曜日始まりの0から6です。	可	0(日曜日)	値	不可	不可
selectedDates	Date型のArray	選択日付の配列を指定します。 Dateオブジェクトの配列です。	可	選択日付なし(長さ0の配列)	バインド	可	可
daysOfWeek	Stringの配列	曜日として表示する文字列の配列を指定します。日曜日始まりで、7つの曜日文字列を指定します。 要素数が7以外の場合は、エラーとなります。 タグ内に記述する場合は、半角セミコロンで区切って指定します。 例:"S;M;T;W;T;F;S"	可	["日","月","火","水","木","金","土"]	値	不可	不可
holidays	Object型のArray	祭日の情報を指定します。	可	祭日なし(長さ0の配列)	バインド	可	可
specialDates	Object型のArray	背景色やフォント色など、ユーザが自由に定義できる特別な日付を指定します。	可	特別な日なし(長さ0の配列)	バインド	可	可
dateToolTips	Object型のArray	個々の日付のツールチップ文字列を指定します。	可	ツールチップなし(長さ0の配列)	バインド	可	可
naviType	String	ナビゲータ表示を指定します。 ・ ALL: 前月ボタン、次月ボタン、タイトルを表示 ・ TITLE: タイトルだけ表示 ・ NONE:ナビゲータ非表示	可	ALL	値	不可	不可
naviButtonRenderer	render関数を持つオブジェクト	前月/次月ボタンをカスタマイズします。	可	null	値	不可	不可
naviTitleLabelProvider	getLabel関数を持つオブジェクト	タイトル文字列をカスタマイズするオブジェクトを指定します。	可	null	値	不可	不可
dayOfWeek	render関数を持つオ	曜日セルをカスタマイズします。	可	null	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
CellRenderer	ブジェクト						
dateCellRenderer	render関数を持つオブジェクト	日付セルをカスタマイズします。	可	null	値	不可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可
selectable	Boolean	日付の選択可否を指定します。 ・ true:選択可能 ・ false:選択不可	可	true	値	可	不可
selectMode	String	単一選択/複数選択モードを指定します。 ・ SINGLE:単一選択 ・ MULTI:複数選択	可	SINGLE	値	不可	不可
utc	Boolean	日付をUTCとして扱うかどうかを指定します。 ・ true:世界標準時間(UTC) ・ false:ローカル時間 “5.1.9 Dateオブジェクトとタイムゾーン”を参照してください。	可	false	値	不可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

holidaysプロパティ

オブジェクトのプロパティについて以下に示します。

プロパティ名	データ型	説明	省略	省略値
date	Date	対象となる日付のDateオブジェクトを指定します。	不可	-
type	String	日付の指定方法を指定します。 ・ normal : 指定した年月日だけ ・ everyYear : 毎年(dateの年は無視される)	可	normal
message	String	指定した場合は、ツールチップが表示されます。	可	-

holidaysの値を更新しても、画面は即時には更新されません。次にカレンダの表示年月を更新する際に、画面表示に反映されます。

初期値は、モデルとのバインディングにより指定してください。
RCF.addInitializedListenerにより設定した場合は、表示月を変更したタイミングで画面表示に反映されます。初期画面には反映されません。

日付が固定でない場合の設定

第n月曜や毎月末に日付を設定したい場合、または不定期の場合、月日の値が固定でないため、typeに“everyYear”を指定してholidays、specialDates、dateToolTipsの日付を設定することができません。この場合、以下の2つの対応方法があります。

ー アプリケーションで必要な表示年月の範囲が限定されている場合、typeに“normal”を指定して、必要な範囲のすべての年について個々に設定します。
例えば、holidaysに成人の日を設定する場合、2007～2010の範囲を対象とするならば、holidaysに2007/1/8、2008/1/14、2009/1/12、2010/1/11の計4オブジェクトを設定します。

ー 初期表示で必要な範囲の値だけを設定しておき、onBeforeDisplayChangeEventのリスナーの設定によって、表示月がその範囲を越えたら、新たにプロパティを設定し直します。
例えば、初期表示が2007年8月の場合、holidaysの初期値として2007年の祝日のデータだけを設定しておき、リスナーの中で表示年が2008年に変わったら、2008年の祝日のデータを計算し、holidaysプロパティを更新します。

specialDatesプロパティ

オブジェクトのプロパティについて以下に示します。

プロパティ名	データ型	説明	省略	省略値
date	Date	対象となる日付のDateオブジェクトを指定します。	不可	-
type	String	日付の指定方法を指定します。 ・ normal : 指定した年月日だけ ・ everyYear : 毎年(dateの年は無視される)	可	normal
styleNumber	Number	0～9の数値を指定します。 日付セルにrcf-Calendar-special#(# は 0～9)のクラスが設定され、それに従ったスタイルで表示されます。	可	0
message	String	指定した場合は、ツールチップが表示されます。	可	-

specialDatesの値を更新しても、画面は即時には更新されません。次にカレンダの表示年月を更新する際に、画面表示に反映されます。
初期値は、モデルとのバインディングにより指定してください。
RCF.addInitializedListener により設定した場合は、表示月を変更したタイミングで画面表示に反映されます。初期画面には反映されません。

日付が固定でない場合の設定については、“holidaysプロパティ”の記事を参照してください。

dateToolTipsプロパティ

オブジェクトのプロパティについて以下に示します。

プロパティ名	データ型	説明	省略	省略値
date	Date	対象となる日付のDateオブジェクトを指定します。	不可	-
type	String	日付の指定方法を指定します。 ・ normal : 指定した年月日だけ ・ everyYear : 毎年(dateの年は無視される)	可	normal
message	String	ツールチップの文字列を指定します。同一日にツールチップが複数指定された場合、つなげて表示されます。	不可	-

dateToolTipsの値を更新しても、画面は即時には更新されません。次にカレンダの表示年月を更新する際に、画面表示に反映されます。
初期値は、モデルとのバインディングにより指定してください。
RCF.addInitializedListener により設定した場合は、表示月を変更したタイミングで画面表示に反映されます。初期画面には反映されません。

日付が固定でない場合の設定については、“holidaysプロパティ”の記事を参照してください。

naviButtonRendererプロパティ

render関数を持つオブジェクトを指定します。render関数は、レンダリング対象のTD DOMノードをパラメタとして渡されます。render関数内のユーザロジックでナビゲータの前月/次月のボタンに表示する文字をカスタマイズしたり、画像を貼り付けたりできます。カレンダ初期化時、および表示月を変更したときに呼び出されます。

render関数内でセルのスタイルを直接変更した場合、そのあとで前月・次月を表示すると、以前の設定が残ることがあります。render関数内で必要に応じて再設定してください。

render関数のパラメタを順番に示します。

順	パラメタ名:型	説明
1	node: Node	レンダリング対象のTD DOMノードを指定します。曜日に応じてクラス設定され、子要素として“<”または“>”のテキスト要素を持つTDノードが渡されます。
2	buttonType: String	ナビゲータボタンの種類を表す文字列を指定します。 "next" : 次月, "previous" : 前月

naviTitleLabelProviderプロパティ

naviTitleLabelProviderには、getLabel関数を持つオブジェクトを指定します。

getLabel関数には、タイトルに表示する年および月がパラメタとして渡されます。getLabel関数は文字列を返却し、その値がタイトルとして表示されます。getLabel関数内のユーザロジックにより、ナビゲーションタイトルに表示する文字列を制御できます。カレンダ初期化時、および表示月切替えのタイミングで呼び出されます。

getLabel関数のパラメタは、以下のとおりです。

順	パラメタ名:型	説明
1	year: Number	カレンダに表示する年を指定します。
2	month: Number	カレンダに表示する月-1を指定します。 (0...11)

dayOfWeekCellRendererプロパティ

dayOfWeekCellRendererプロパティには、render関数を持つオブジェクトを指定します。render関数は、レンダリング対象の曜日セルに対応するTD DOMノードと、そのほかの補足情報がパラメタとして渡されます。render関数内のユーザロジックでDOMノードを操作することにより、曜日セル内部の表示を制御できます。カレンダ初期化時と、表示月の切替えのタイミングで呼び出されます。

render関数内でセルのスタイルを直接変更した場合、そのあとで前月・次月を表示すると、以前の設定が残ることがあります。render関数内で必要に応じて再設定してください。

render関数のパラメタを順番に示します。

順	パラメタ名:型	説明
1	node: Node	レンダリング対象のTD DOMノードを指定します。 曜日に応じてクラス設定され、子要素として日付のテキスト要素を持ったTDノードが渡されます。
2	day: Number	曜日を指定します。 (日曜:0, ... 土曜:6)
3	x: Number	カレンダ内の横座標(0-6)を指定します。(左端が0)

dateCellRendererプロパティ

dateCellRendererプロパティには、render関数を持つオブジェクトを指定します。render関数は、レンダリング対象のTD DOMノードと、そのほかの補足情報がパラメタとして渡されます。render関数内のユーザロジックでDOMノードを操作することにより、日付セル内部の表示を制御できます。カレンダ初期化時と、表示月の切替えのタイミングで呼び出されます。

render関数内でセルのスタイルを直接変更した場合、そのあとで前月・次月を表示すると、以前の設定が残ることがあります。render関数内で必要に応じて再設定してください。

render関数のパラメタを順番に示します。

順	パラメタ名:型	説明
1	node: Node	レンダリング対象のTD DOMノードを指定します。 曜日や祭日などに応じてクラスが設定され、子要素として日付のテキスト要素を持ったTDノードが渡されます。
2	date: Date	レンダリング対象の日付を指定します。
3	day: Number	曜日を指定します。 (日曜:0, ... 土曜:6)
4	x: Number	カレンダ内の横座標(0-6)を指定します。(左端が0)
5	y: Number	カレンダ内の縦座標(0-6)を指定します。(ナビゲータ・曜日行を除いて、日付部分の上から順に 0,1,2...)

マウス操作

以下の表に示します。

操作	処理
ナビゲータの前月ボタンをクリック	前の月を表示します。
ナビゲータの次月ボタンをクリック	次の月を表示します。
日付のセルをクリック	クリックした日付の選択状態を切り替えます。
日付のセルをダブルクリック	クリックした日付を選択された状態にします。
日付のセル上にマウスを移動	マウスの位置するセルの日付をフォーカス日付にします。

キー操作

以下の表に示します。

操作		処理
スペース		フォーカス日付の選択状態を切り替えます。
←		フォーカス日付を前日に移動します。(表示されている範囲内での移動)
	+ SHIFT	前の月を表示します。
→		フォーカス日付を翌日に移動します。(表示されている範囲内での移動)
	+ SHIFT	次の月を表示します。
↑		フォーカス日付を上(前の週)に移動します。(表示されている範囲内での移動)
↓		フォーカス日付を下(次の週)に移動します。(表示されている範囲内での移動)
Home		フォーカス日付を表示月の1日に移動します。
End		フォーカス日付を表示月の最終日に移動します。
PageUp		前の月を表示します。
PageDown		次の月を表示します。

スタイルプロパティ

本部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	指定可能なスタイル
全体(外枠)	なし	rcf-Calendar	・ サイズ ・ ボーダー

パーツ名	プレフィックス	クラス名	指定可能なスタイル
セル(全体共通)	cell	rcf-Calendar-cell	・サイズ ・カラー ・フォント ・ボーダー ・セル
ナビゲーションタイトル	naviTitle	rcf-Calendar-naviTitle	・サイズ(widthを除く) ・カラー ・フォント ・ボーダー ・セル
ナビゲーションボタン	naviButton	rcf-Calendar-naviButton	・サイズ(widthを除く) ・カラー ・フォント ・ボーダー ・セル
曜日ヘッダ(共通)	dayOfWeek	rcf-Calendar-dayOfWeek	・サイズ(widthを除く) ・カラー ・フォント ・ボーダー ・セル
曜日ヘッダ(個別)	dayOfWeek0 ～ dayOfWeek6	rcf-Calendar-dayOfWeek0 ～ rcf-Calendar-dayOfWeek6	・カラー ・フォント ・ボーダー ・セル
日付(共通・当月)	date	rcf-Calendar-date	・カラー ・フォント ・ボーダー ・セル
日付(曜日ごと・当月)	date0 ～ date6	rcf-Calendar-date0 ～ rcf-Calendar-date6	・カラー ・フォント ・ボーダー ・セル
日付(共通・他の月)	otherMonth	rcf-Calendar-otherMonth	・カラー ・フォント ・ボーダー ・セル

パーツ名	プレフィックス	クラス名	指定可能なスタイル
日付(曜日ごと・他の月)	otherMonth0 ～ otherMonth6	rcf-Calendar-otherMonth0 ～ rcf-Calendar-otherMonth6	• カラー • フォント • ボーダー • セル
フォーカス日付	focus	rcf-Calendar-focus	• カラー • フォント • ボーダー • セル
選択日付	selected	rcf-Calendar-selected	• カラー • フォント • ボーダー • セル
休日	holiday	rcf-Calendar-holiday	• カラー • フォント • ボーダー • セル
特別日	special0 ～ special9	rcf-Calendar-special0 ～ rcf-Calendar-special9	• カラー • フォント • ボーダー • セル

詳細は、“2.9 スタイルプロパティ”を参照してください。

注意

- カレンダの各セルは、tableのTD要素で表示されます。
- 日付(当月)、日付(他の月)、曜日ヘッダは、共通のスタイル設定のほかに、曜日ごとにスタイルを変更することができます。この場合は、曜日を表す数字をプレフィックス/クラス名に後置します。
  (例:日曜日0...土曜日6)
- 日付セルのスタイルは、デフォルトでは、以下の優先順位を持ちます。(上の方が優先度高)
  1. selected
  2. focus
  3. special0..9
  4. holiday
  5. date0..6 / otherMonth0..6
  6. date /otherMonth
  7. cell
- 全体(外枠)以外のスタイルの設定は、CSSでだけ可能です。CSSで指定する際は、優先度の低いものから順に指定してください。

・ CSSで指定したスタイルの方が、デフォルトのスタイルより優先されます。cellやdateなど優先度が低いものだけをCSSで定義すると、選択日やフォーカスなどのデフォルトのスタイル指定が無効になる場合があります。必要なスタイルには、優先順位に従って、すべてを定義するようにしてください。

イベントリスナ

名前	説明	イベントオブジェクト
onDisplayMonthChanged	表示月が変更されたときに呼ばれます。	DisplayMonthChangeEvent
onBeforeDisplayMonthChanged	表示月が変更されたとき、カレンダ再表示の前に呼ばれます。	
onDateSelected	日付が選択されたときに呼ばれます。	CalendarSelectionChangeEvent
onDateDeselected	日付が選択状態でなくなったときに呼ばれます。	
onDateDblClicked	日付をダブルクリックしたときに呼ばれます。	
onDateFocused	フォーカス日付が変更されたときに呼ばれます。	

イベントリスナ内では、表示年月の更新をしないでください。
APIで表示月を設定した場合、設定した値が現在表示中の年月と同一だった場合、displaymonthchangedイベントおよびbeforedisplaymonthchangedイベントは発生せず、画面も更新されません。

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■selectメソッド

メソッド	select(date)	
引数	date [Date]	選択する日を表すDateオブジェクト
戻り値	なし	
例外	なし	
説明	selectModeがSINGLEの場合、selectedDatesプロパティに選択日付を設定し、既存の選択日は非選択となります。 selectModeがMULTIの場合、selectedDatesプロパティに選択日付を追加します。 指定日がすでに選択されている場合は、何もしません。 指定した日付がカレンダ上で表示されている場合は、当該セルの選択状態を変更します。また、selectModeがSINGLEの場合は、以前に選択されていた日付がカレンダ上で表示されている場合も、当該セルの選択状態を変更します。	

■deselectメソッド

メソッド	deselect(date)	
引数	date [Date]	非選択にする日を表すDateオブジェクト
戻り値	なし	
例外	なし	
説明	指定日がselectedDatesプロパティに含まれる場合は削除します。 選択されていない日付の場合は何もしません。	

	指定した日付が、カレンダ上で表示されている場合は、当該セルを非選択状態に変更します。

■setNextMonthメソッド

メソッド	setNextMonth()
引数	なし
戻り値	なし
例外	なし
説明	カレンダ表示年月変更(現表示月を基準に、次月)

■setPreviousMonthメソッド

メソッド	setPreviousMonth()
引数	なし
戻り値	なし
例外	なし
説明	カレンダ表示年月変更(現表示月を基準に、前月)

■setThisMonthメソッド

メソッド	setThisMonth()
引数	なし
戻り値	なし
例外	なし
説明	カレンダ表示年月変更(システム時間の月)

■setDisplayMonthメソッド

メソッド	setDisplayMonth(year,month)	
引数	year [Number]	表示する年
	month [Number]	表示する月(0-11)
戻り値	なし	
例外	なし	
説明	カレンダ表示年月を変更します。(指定年月) year/monthの両者を変更する場合などに使用します。(setPropertyを使うと2回再表示されるため)	

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.4.2 PopupCalendar

PopupCalendarは、ポップアップダイアログのカレンダを表示する部品です。CalendarButtonと組み合わせて指定します。ポップアップダイアログから日付を確定させることができます。

- 表示例
- 記述形式
- プロパティ

- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

注意

本部品は、デフォルトでは不可視です。可視化するには、showメソッドを利用します。詳細は、“5.1.10 Window、PopupCalendar部品を自動的に表示させる場合”を参照してください。

表示例

記述形式

```
<div rcf:type="PopupCalendar"></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
selectedDate	Date	選択された日付を指定します。タイムゾーンはカレンダのutcプロパティに依存します。	可	null	値、バインド	可	不可
labelOK	String	OKボタンの文字列を指定します。""(空文字列)を指定すると、ボタンが非表示になります。(注)	可	OK	値	不可	不可
labelCancel	String	キャンセルボタンの文字列を指定します。""(空文字列)を指定すると、ボタンが非表示になります。(注)	可	"キャンセル"	値	不可	不可

注) labelOKおよびlabelCancelの両方に空文字列を指定した場合は、フッター部(OKボタンおよびキャンセルボタンが表示される領域)が非表示になります。

Calendarのプロパティも指定できます。(selectedDates、selectMode、tabIndex、selectableを除きます。) ここで説明を省略しているプロパティの詳細は、“2.4.1 Calendar”の“プロパティ”を参照してください。
Windowのプロパティも指定できます。(mode、resizableを除きます。)詳細は、“2.2.6 Window”の“プロパティ”を参照してください。
部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

スタイルプロパティ

本部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	指定可能なスタイル
OK/キャンセルボタン	button	rcf-PopupCalendar-button	• カラー • フォント(lineHeightを除く)

Calendarのスタイルプロパティも指定できます。詳細は、“2.4.1 Calendar”の“スタイルプロパティ”を参照してください。
ただし、Calendarのスタイルのうち、全体(外枠)に対応するものは、PopupCalendarでは以下のようになります。

- プレフィックス: calendar
- クラス名: rcf-Calendar-calendar

Windowのスタイルプロパティも指定できます。詳細は、“2.2.6 Window”の“スタイルプロパティ”を参照してください。
表示されるウィンドウのサイズは、カレンダ部品のサイズに合わせて調整されます。このため、Windowのスタイルプロパティのうち、全体のサイズ、タイトルのwidthは設定できません。
ボディ部のカラー、フォント、テキストの設定は、できません。
スタイルの設定やレンダラの処理により、フォーカスや選択日など動的に変更される部分のサイズが変わると、ウィンドウに余白やスクロールバーが表示される場合があります。サイズが変動しないように、スタイルを設計してください。

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onDateFixed	日付をダブルクリックするか、OKボタンまたはENTERキーが押されたときに呼ばれます。calendarイベントのあとに呼ばれます。 また、okメソッド実行時に日付が選択されていた場合にも呼ばれます。	CalendarSelectionChangeEvent
onOk	OKボタンまたはENTERキーが押されたときに呼ばれます。calendarイベントのあと、かつdatefixedイベントの前に呼ばれます。 okメソッド実行時は呼ばれません。	ActionEvent

名前	説明	イベントオブジェクト
onCancel	キャンセルボタン、閉じるボタン、またはESCキーが押されたときに呼ばれます。 cancelメソッド、hideメソッド実行時には呼ばれません。	

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■getCalendarメソッド

メソッド	getCalendar()	
引数	なし	
戻り値	[Object]	Calendar部品のインスタンス
説明	内部のCalendarインスタンスを返します。	

■okメソッド

メソッド	ok()
引数	なし
戻り値	なし
説明	OKボタンクリックと同じ処理です。 日付の選択状態の確認は行いません。

■cancelメソッド

メソッド	cancel()
引数	なし
戻り値	なし
説明	キャンセルボタンのクリックと同じ処理です。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

- 本部品は、 タグのstyle属性(position、top、leftなど)が利用できない部品の1つです。詳細は、“1.3.5 UI部品の タグおよびタグで利用できる属性”を参照してください。
- PopupCalendarの初期化が完了する前にshowメソッドで表示を行うと、描画が崩れる場合があります。
  初期化完了と同時にPopupCalendarを表示したい場合は、タイムアウトを設定してshowメソッドを呼ぶ関数を作成し、RCF.addInitializedListener()でイベントリスナを登録してください。
  詳細は、“5.1.10 Window、PopupCalendar部品を自動的に表示させる場合”を参照してください。
- PopupCalendarは、CalendarとWindowの両方の部品を合わせ持っている部品です。それぞれの部品に由来する仕様の詳細は、“2.4.1 Calendar”および“2.2.6 Window”を参照してください。
- hideメソッドによってPopupCalendarを閉じることができます。ただし、その場合はcancelメソッドと異なり、選択日付はポップアップ時の値には戻りません。
- topとleftは、スタイルプロパティで指定してください。
  CSSでの指定は無効となります。

2.4.3 CalendarButton

CalendarButtonは、PopupCalendarを表示するボタン部品です。PopupCalendarと組み合わせて指定します。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

ボタンをクリックすること、またはフォーカス時にEnterキーまたはスペースキーを押すことで、PopupCalendarが表示されます。

表示例

記述形式

```
<div rcf:type="CalendarButton" rcf:target="popupCalendar1" ... ></div>
```

または

```
<span rcf:type="CalendarButton" rcf:target="popupCalendar1" ... ></span>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

本部品は、以下のように表示されます。

- タグの場合:前後に改行コードが挿入されます。
- タグの場合:前後に改行コードは挿入されません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
alt IE 6 IE 7 IE 8 互換	String	画像の代替テキストを指定します。 FirefoxおよびInternet Explorer 8 標準では無効です。	可	Calendar Button	値、バインド	可	不可
enabled	Boolean	ボタンの利用を許可するかどうか指定します。 • true: 許可する	可	true	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		・ false: 許可しない					
target	String	表示するPopupCalendarのIDを指定します。	不可	-	値	不可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。 HTMLのtabindex属性と同様の指定ができます。1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

CalendarButtonの記述例

PopupCalendarやDateInputを組み合わせて表示するための記述例を、以下に示します。

```
 ...
<span rcf:type="DateInput" rcf:date="{model1.date}"></span>
<span rcf:type="CalendarButton" rcf:target="popupCalendar1"></span>
<div rcf:type="PopupCalendar" rcf:id="popupCalendar1" rcf:selectedDate="{model1.date}"></div>
 ...
<div rcf:type="Model" rcf:id="model1" ... ></div>
 ...
```

入力手順を以下に示します。

1. CalendarButtonをクリックします。
   ⇒PopupCalendarが表示されます。
2. PopupCalendarで日付を選択します。
   ⇒model1.dateを介してDateInputに日付が入力されます。

スタイルプロパティ

パーツ名	プレフィックス	クラス名	指定可能なスタイル
部品全体	なし	rcf-CalendarButton	・ ボーダー ・ verticalAlign

verticalAlignについて説明します。

名前	データ型	説明	省略値
verticalAlign	String	行内の上下の表示位置を指定します。 CSSのvertical-alignプロパティの値を指定できます。	baseline

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFocus	部品がフォーカスを得たときに呼ばれます。	ActionEvent
onBlur	部品がフォーカスを失ったときに呼ばれます。	

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

ポイント

dblclickイベントは発生しません。最初のクリックでPopupCalendar(モーダル)が開くためです。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

2.5 ツリー部品

ツリー部品は、ツリー形式で項目リストを表示する部品です。

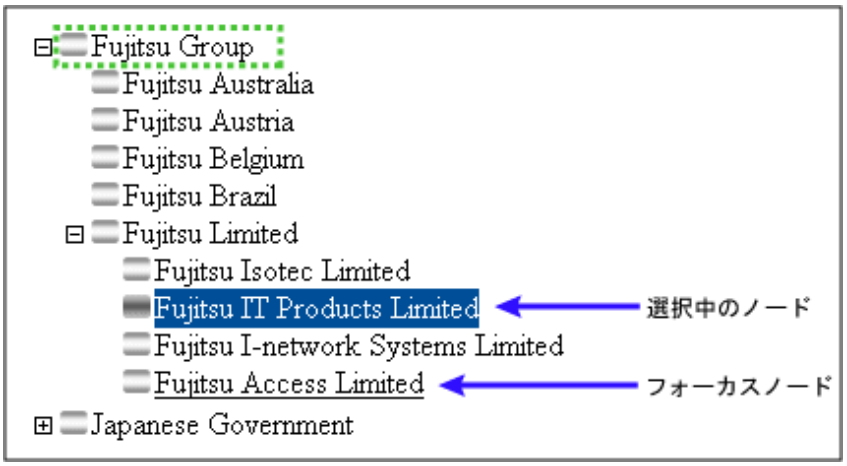
ここでは、ツリー部品の設定内容、および設定方法について説明します。

2.5.1 TreeView

TreeViewは、ツリー形式で項目リストを表示させる部品です。各項目の下位要素を展開、または折りたたんで表示させることができます。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API

表示例



本書内では、破線枠の範囲をノードと呼びます。また、ノード配下のそれぞれの部分を以下のように呼びます。

- [＋]、[－]:開閉アイコン
- 画像:見出し画像
- “Fujitsu Group”部分:ラベル

スタイルの指定を行わない場合、ラベルは以下のように表示されます。

- フォーカスノード:ラベルに下線を表示
- 選択ノード:ラベルを反転表示

記述形式

```
<div rcf:id="model1" rcf:type="TreeModel" rcf:object="modelData"></div>
<div rcf:type="TreeView" rcf:data="{model1.treeData}"></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

- rcf:type="TreeView"の タグ内には、属性にTreeViewのプロパティを指定できます。
- 本部品は、前後に改行コードが挿入されて表示されます。
- TreeModelの仕様については、“3.1.2 TreeModel”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
data	Object	ツリー形式のObjectのルートを指定します。データには、ルート以降の各ノードのデータを記述します。 “dataプロパティと表示データの指定方法”を参照してください。	不可	なし	バインド	可	可
headImage (注)	Object	ノードの選択/非選択時に表示する見出し画像のURLまたはパスを指定します。 詳細は、“headImageプロパティ”を参照してください。	可	標準で添付する画像	値	不可	不可
tabIndex	Number	Tabキーで移動するフォーカスの順番を指定します。HTMLのtabindex属性と同様に、1以上の整数を指定した場合、数字の小さい順にフォーカスが移動します。 FocusManagerによるフォーカス移動には関係しません。	可	0	値	可	不可

注) headImageプロパティで指定する見出し画像の推奨サイズは、高さ12ピクセル、幅14ピクセルです。

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

headImageプロパティ

以下にheadImageのプロパティを示します。

プロパティ名	データ型	説明
normal	String	非選択ノードに表示する見出し画像のURLまたはパスを指定します。空の文字列を指定した場合は、画像は表示されません。省略時は、標準で添付されている画像が表示されます。
selected	String	選択ノードに表示する見出し画像のURLまたはパスを指定します。空の文字列を指定した場合は、画像は表示されません。省略時は、標準で添付されている画像が表示されます。

それぞれのURLには、クエリ文字列およびURLリライティングで用いるセションIDを付加することができます。詳細は、“ユーザーズガイド”を参照してください。

注意

プロパティで指定した文字列に対するエスケープ処理は行いません。フォーム画面などからユーザが入力したデータを本プロパティに設定する場合は、アプリケーション側で < > & " ' の5つの文字をエスケープするようにしてください。

dataプロパティと表示データの指定方法

以下のJavaScriptデータで1つのノードを表現し、1つのルートノードから子ノードのデータを入れ子形式で指定します。

```
{
  name: "fj",           // ノード名
  label: "Fujitsu",     // ラベル表示文字列
  isExpanded: false,    // 子ノードを展開表示している場合は、true
  headImage: {
    normal: "/img/closeFolder.gif",      //非選択時の見出し画像
    selected: "/img/openFolder.gif"      //選択時の見出し画像
  },
  children: []          // 子ノードのリスト
};
```

子ノードを持つ場合は、以下のように記述します。

```
{
  name: "fj",
  label: "Fujitsu",
  isExpanded: true,
  headImage: {
    normal: "/img/closeFolder.gif",
    selected: "/img/openFolder.gif"
  },
  children: [
    {
      name: "jinji",
      label: "人事部",
      headImage: {
        normal: "/img/closeFolder.gif",
        selected: "/img/openFolder.gif"
      }
    }
  ]
};
```

ノードごとに持つプロパティを説明します。

プロパティ名	データ型	説明	省略	省略値
name	String	ノードの名前を指定します。 ノード位置を示すパスを表現する際に使用します。同一ノードの子ノード間では、nameの値は一意でなければなりません。/を含む文字列、および空の文字列は指定できません。	不可	なし
label	String	ノードのラベルに表示する文字列です。	可	nameの値
unselectable	Boolean	ノード(ラベル)を選択不可にするかどうかを指定します。 ・ true: 選択不可 ・ false: 選択可能	可	false
isExpanded	Boolean	子ノードを展開して表示するか、折りたたんで表示するかを指定します。 ・ true: 展開して表示する ・ false: 折りたたんで表示する ルート、または子ノードがないノードでは、本パラメタは無視されます。	可	false
headImage	Object	ノードの選択/非選択時に表示する見出し画像のURLまたはパスを指定します。 詳細は、“headImageプロパティ”を参照してください。	可	プロパティで指定する画像
children	Array	子ノードのデータを保持するObjectの配列です。配列の1要素に1つのノードを割り当てます。省略時は、子ノードがない(追加しない)ことを示します。子ノードのデータをあとで追加する場合は、children: []を指定します。	可	子ノードがないことを意味する

注意

- ルートノードのchildren配列には、必ず1つ以上のノードデータを定義してください。空の配列(children: [])は指定できません。
- ルートノードは表示されません。
- dataプロパティに対する処理については、“JavaScript API”を参照してください。
- 同一ノードの子ノード間でname値の重複があった場合は、エラーとなり、ブラウザのエラーメッセージが表示されます。また、name値が重複したノードを検出した時点で、ノードの描画は中止します。name値に / を含む場合も同様のエラーとなります。
- データプロバイダによりunselectableプロパティをtrueに変更した場合、ノードの選択状態は解除されます。
- labelまたはnameに以下の特殊文字が指定された場合は、ツリー表示時に以下の値に変換します。

元の文字	変換後の文字列
&	&amp;
<	<
>	>
"	"
'	'
空白(半角空白およびタブ)	

dataプロパティの記述例

以下にdataプロパティの記述例を示します。

```
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:rcf="http://ria.fujitsu.com/rcf">
<meta http-equiv="content-type" content="text/html; charset=UTF-8" />
<script type="text/javascript" src="rcf_config.js"></script>
<script type="text/javascript" src="acf/file/rcf/rcf.js"></script>
<script type="text/javascript">
//<![CDATA[
    var modelData = {
      treeData: {
        name: "root",         // ルートノード名
        children: [
          { // 1つめの子ノード
            name: "fj",  // ノード名
            label: "Fujitsu Group",
            isExpanded: true,
            headImage: { normal: "/img/en.gif" },
            children: [
              { // 孫ノード
                name: "fja", // ノード名
                label: "Fujitsu Australia",
                isExpanded: false,
                headImage: {
                    normal: "/img/blueSquare.gif",
                    selected: "/img/redSquare.gif"
                }
              }
            ]
          },
          { // 2つめの子ノード
            name: "jgov", // ノード名
            label: "Japanese Government",
            isExpanded: false,
            headImage: { normal: "/img/en.gif" }
          }
        ]
      }
    };
//]]>
</script>
<body>
<div rcf:id="model1" rcf:type="TreeModel" rcf:object="modelData"></div>
<div rcf:type="TreeView" rcf:data="{model1.treeData}"></div>
</body>
</html>
```

このdataプロパティが指定された場合のツリー表示の例は、以下になります。
ルートノードは表示されません。

ノードパスの表現

選択しているノードの位置を示すパス情報は、以下の形式となります。
(/ の前後に空白は入りません。)

```
/   fj  /  fja
↑   ↑      ↑
(1) (2)    (3)
```

1. ルート
2. 第1階層のノードのname
3. 第2階層のノードのname

パスは、ルートからの絶対パス指定だけが有効です。相対パスは指定できません。

データプロバイダとノードパスを組み合わせることにより、ノードのデータを取得できます。

TreeViewのフォーカス

TreeViewで使用できるフォーカスには、以下の2つがあります。

フォーカス

部品間でフォーカスを移動させる場合に使用します。TabキーおよびShift+Tabキーを用いてフォーカスを移動します。

ラベルフォーカス

TreeView部品内で使用するフォーカスで、↑ ↓のカーソルを使ってラベル間のフォーカスを移動します。

TreeViewには、以下の方法でフォーカスを当てることができます。フォーカスはTreeView全体に当たり、フォーカスがある状態ではキーボードによる操作が可能になります。

- ブラウザが提供するTabキーおよびShift＋Tabキーによるフォーカス移動
- FocusManagerによるフォーカス移動
- TreeViewをマウスでクリック
- RCF.setFocus関数を使用して、ユーザロジックからフォーカス移動

フォーカスがある場合、TreeViewの外枠にフォーカスがあることを示すアウトラインが表示されます。また、フォーカスが当たったとき、TreeView内では、以下の規則に従ってラベルフォーカスが当たり、キーボードによりラベルフォーカスを移動させることができます。

- Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、TreeViewにフォーカスが当たった場合
  以前にラベルフォーカスを失ったときのラベルに、ラベルフォーカスが当たります。初めてラベルフォーカスが当たった場合は、先頭のノードにラベルフォーカスが当たります。
- マウスでクリックしたことによりTreeViewにフォーカスが当たった場合
  ラベルがクリックされた場合は、そのラベルにラベルフォーカスが当たります。
  それ以外をクリックした場合は、Tabキーによるフォーカス移動またはFocusManagerによるフォーカス移動で、TreeViewにフォーカスが当たった場合と同様になります。
- 子ノードのラベルにラベルフォーカスがある状態で、ノードの折りたたみを行った場合は、折りたたみを行ったノードにラベルフォーカスを移動します。
- モデルデータを変更したノードの子ノードにラベルフォーカスがある場合は、モデルデータを変更したノード(TreeDataChangeEventデータで通知するnodePathに該当するノード。ただし、ルートの場合は表示上の最上位ノード)に、ラベルフォーカスが移動します。モデルデータ内で選択中のノード、または選択中のノードの親ノードが変更された場合は、選択状態が解除されます。(データ中のnameプロパティの変更により、選択中のノードパスが消滅した場合は、選択状態が解除されます。)

マウス操作

TreeViewにおけるマウス操作について、以下の表に示します。

操作	処理
開閉アイコンでクリック	子ノードが存在する場合に、展開または折りたたみをします。
ラベル、および見出し画像でクリック	クリックしたラベルを選択します。
ラベル、および見出し画像にマウスを移動	マウスの位置するラベルをラベルフォーカスが当たります。

注意

- ラベルをクリックすると、見出し画像が変化します。
- 開閉アイコンで、子ノードの展開や折りたたみを行います。

キーボード操作

キーボード操作は、TreeViewにフォーカスが当たっている場合に有効になります。

操作	処理
スペース	ラベルフォーカスが当たっているラベルを選択します。
↑	ラベルのラベルフォーカスを上のラベルに移動します。 先頭のノードにラベルフォーカスがある場合は、移動しません。
↓	ラベルのラベルフォーカスを下のラベルに移動します。 最終のノードにラベルフォーカスがある場合は、移動しません。
→	子ノードが存在する場合に、展開します。
←	子ノードが存在する場合に、折りたたみをします。
Home	先頭のノードのラベルに移動します。
End	画面に表示中の最終のノードのラベルに移動します。

注意

- TreeViewの初期フォーカス位置は、先頭のノードです。
- AltやShiftを押しながら同時にほかのキーボード操作を行っても、特別な操作はできません。
- Enterによるキー操作はありません。
- 複数のラベルは選択できません。
- 見出し画像やラベル上にマウスポインタがあっても、↑キー、↓キー、Homeキー、およびEndキーの入力により、ラベルフォーカスは移動します。

スタイルプロパティ

本部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	指定可能なスタイル
全体(外枠)	なし	rcf-TreeView	• サイズ • ボーダー (注1) • オーバーフロー (注2)
開閉アイコン	icon	rcf-TreeView-icon	• サイズ (注3)
見出し画像	headImage	rcf-TreeView-headImage	• サイズ (注3)
ラベル(選択やラベルフォーカスが当たっていないラベル)	label	rcf-TreeView-label	• カラー • フォント (注4)
選択されたラベル (注5)	selectedLabel	rcf-TreeView-selectedLabel	• カラー • フォント (注4)

パーツ名	プレフィックス	クラス名	指定可能なスタイル
ラベルフォーカスが当たっているラベル (注6)	focusedLabel	rcf-TreeView-focusedLabel	• カラー • フォント (注4)

注1) Internet Explorer 6、Internet Explorer 7、およびInternet Explorer 8 互換では、部品にフォーカスを当てるとボーダーの表示が無効になる場合があります。(例:borderWidthに1pxを指定した場合)

注2) デフォルトではスクロールバーは表示されません。ラベルに長い文字列が指定された場合は、サイズのwidthで指定した横幅を超えて、ラベルが1行に表示されます。

注3) 幅(width)と高さ(height)は、スタイルプロパティで指定してください。CSSで指定した値は無視されます。

注4) Internet Explorer 6では、lineHeightは指定できません。

注5) デフォルトは、背景色は#004E98で、フォント色は#FFFFFF(白)です。

注6) デフォルトは、テキストに下線が表示されます。

注意

- 子ノードのインデントの幅は、開閉アイコンと見出し画像のスタイルに応じて自動的に設定されます。
- 選択されたラベルとラベルフォーカスが当たっているラベルが同じ位置のとき、ラベルフォーカスがあるラベルのスタイルが優先されます。ただし、ラベルフォーカスがあるラベルで設定していないスタイルについては、選択されたラベルのスタイルが適用されます。

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

本部品全体のイベントリスナを以下に示します。

名前	説明	イベントオブジェクト
onDataChange	モデルのデータが変更されると発生します。	TreeDataChangeEvent

名前	説明	イベントオブジェクト
onTreeNodeExpand	ノードが展開されると発生します。	TreeNodeEvent
onTreeNodeCollapse	ノードが折りたたまれると発生します。	
onTreeNodeSelect	ノードが選択されると発生します。	

名前	説明	イベントオブジェクト
onFocus	TreeView全体がフォーカスされると発生します。(注)	ActionEvent
onBlur	TreeView全体がフォーカスを失うと発生します。	

注) ノードのラベルにラベルフォーカスが当たっても、onFocusは発生しません。

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

■selectメソッド

メソッド	select(nodePath)

引数	nodePath [String]	ノードのパス名 例:/travel/Asia
戻り値	なし	
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合 ・ nodePathに空文字列が指定された場合 ・ nodePathに存在していないノードのパスが指定された場合 ・ nodePathで指定されたノードが選択不可能である場合	
説明	指定したパス名のノードを選択します。親ノードのツリー表示が折りたたまれている場合は、展開しません。	

■getSelectedNodePathメソッド

メソッド	getSelectedNodePath()	
引数	なし	
戻り値	[String]	選択されたノードのパス ノードとノードの区切り記号は、/です。TreeViewに選択されたノードがない場合は、空文字列を返します。 例:/travel/Asia
例外	なし	
説明	選択されたノードのパスを取得します。	

■getFocusedNodePathメソッド

メソッド	getFocusedNodePath()	
引数	なし	
戻り値	[String]	ラベルフォーカスが当たっているノードのパス ノードとノードの区切り記号は、/です。 例:/travel/Asia
例外	なし	
説明	ラベルフォーカスが当たっているノードのパス名を取得します。	

■getNodeDataProviderメソッド

メソッド	getNodeDataProvider(nodePath)	
引数	nodePath [String]	ノードのパス名
戻り値	[Object]	指定したノードのデータにアクセスするためのDataProvider [ObjectDataProvider]
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合 ・ nodePathに空文字列が指定された場合 ・ nodePathに存在していないノードのパスが指定された場合	
説明	指定したノードのデータにアクセスするためのデータプロバイダを取得します。ノードのデータは、dataプロパティ中の指定ノードとその配下を表示するための部分データとなります。	

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

なお、TreeView部品では、部品共通のJavaScript APIのうち、getStyleおよびsetStyleは利用できません。

2.6 スクレイピング部品

スクレイピング部品は、マッシュアップフレームワークのクライアント通信APIから受信したXMLデータを表示する部品です。複数のサービスを組み合わせたより使いやすい業務画面を作成できます。

ここでは、スクレイピング部品の設定内容、および設定方法について説明します。

2.6.1 ScrapingView

ScrapingViewは、Webアプリケーションからスクレイピングしたコンテンツを表示する部品です。マッシュアッププロキシから取得したコンテンツを表示する場合などに使用します。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

表示例

HTMLドキュメント、およびdata属性に指定したmodelData.scrapingHtmlの値が以下の場合の表示例を示します。

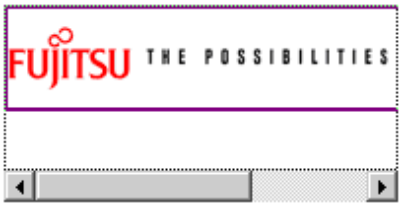
HTMLドキュメント

```
...
<div rcf:type="Model" rcf:id="dataModel" rcf:object="modelData"></div>
<div rcf:type="ScrapingView" rcf:id="scrapingView" rcf:data="{dataModel.scrapingHtml}"
    rcf:width="200" rcf:height="100" rcf:overflow="auto"></div>
...
```

modelData.scrapingHtmlの値

```
<div><p><a href="http://jp.fujitsu.com/">
    <img src="http://interstage.fujitsu.com/jp/imgv3/jp/ja/fujitsu/messagelockup.gif"
     width="300" height="50" alt="富士通" /></a></p>
</div>
```

表示例



記述形式

```
<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"></div>
<div rcf:type="ScrapingView" data="{model1.html}" ... ></div>
```

modelで定義したオブジェクトは、ScrapingViewのdata属性で指定する値と対応をとる必要があります。(“.html”の箇所は、アプリケーションに依存した属性名となります。)

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

- 本部品は、前後に改行コードが挿入されて表示されます。
- ScrapingViewで使用するデータ(dataプロパティで指定したデータ)は、Modelによりモデル定義されたデータとバインディングします。Modelの仕様については、“3.1.1 Model”を参照してください。
- ScrapingViewには、以下の方法でフォーカスを当てることができます。フォーカスはScrapingView全体に当たり、フォーカスがある状態では、キーボードによる操作が可能になります。
  - ブラウザが提供するTabキーおよびTab+SHIFTキーによるフォーカス移動
  - FocusManagerによるフォーカス移動
  - ScrapingViewをマウスでクリック

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
data	String	Webスクレイピングコンテンツを文字列で指定します。	可	""	バインド	可	不可
disableTagNames	String	dataプロパティで指定したコンテンツの中で、無効にするタグ名を正規表現で指定します。	可	(注1)	値	不可	不可
disableAttributeNames	String	dataプロパティで指定したコンテンツの中で、無効にする属性名を正規表現で指定します。	可	(注2)	値	不可	不可

注1) "^(script|style|object|embed|link|applet|iframe|frame|frameset|layer|ilayer|meta)$"

注2) "^(id|class|tabindex|on.*)$"

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。
なお、styleClassプロパティは、dataで指定したHTML部分を囲むスタイルに使用されます。含まれる個々のタグにスタイルを適用する機能はありません。

ScrapingViewの記述例

以下にScrapingViewとMuRequestを組み合わせての使用する例を示します。

```
<script type="text/javascript">
// イベント定義
var myEvent = {
  userButton:{
    click:userTable
  }
};
// 初期処理定義
RCF.addInitializedListener(
  function(eventObject) {
```

```
    // イベント登録
    rcf.event.EventRegistrar.registerEvents(myEvent);
  }
);
var userData = {
  userId:'',
  userTableHTML: ''
};
function userTable(event){
  var reqParams = {
    serviceId:'employeeInfo',  // 登録したサービスID(webスクレイピング)
    type : 'xml'  //結果をXML形式の文字列で取得する
  };
  var option = {
    url:'muf/proxy',
    callback:function(html){ // Webサービスの結果を処理
      //webスクレイピング結果をScrapingViewのdataプロパティに反映
      userDataModel.setProperty('userTableHTML', html);
    }
  };
  var user=new Object();
  user.userId=userDataModel.getProperty('userId');
  MuRequest.send(user, reqParams, option);  // webスクレイピング呼出し
}
</script>
<!-- 機能定義部 -->
<div rcf:id="userDataModel" rcf:type="Model" rcf:object="userData"></div>
<!-- 画面情報定義部 -->
ユーザID:
<span rcf:id="search" rcf:type="TextInput" rcf:value="{userDataModel.userId}"></span>
<div rcf:id="userButton" rcf:type="Button" rcf:value="search">検索</div>
<!-- 結果表示部 -->
<div rcf:id="result" rcf:type="ScrapingView" rcf:data="{userDataModel.userTableHTML}"></div>
```

スタイルプロパティ

本部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	指定可能なスタイル
部品全体	なし	rcf-ScrapingView	• サイズ • カラー • フォント • テキスト • ボーダー • パディング • マージン • ポジション • オーバーフロー

詳細は、“2.9 スタイルプロパティ”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onDataChange	Dataプロパティが変更されたときに呼ばれます。	ValueChangeEvent

部品共通のイベントリスナもあります。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

本部品を使用する場合、以下の注意事項があります。

- スクレイピングしたコンテンツ内に、スクリプトは記述しないでください。スクリプトが含まれているコンテンツを表示したい場合には、disableTagNamesやdisableAttributeNamesを用いて、無効化してください。
  一般的にスクリプトを含む可能性がある、<script>タグや“on*”によるイベント記述属性は、disableTagNamesやdisableAttributeNamesのデフォルト値で無効化されます。スクリプトは、記述する場所にかかわらず、動作は保障されません。
- スクレイピングしたコンテンツ内にスタイルが指定されている場合、その指定のとおりに表示されます。意図しないスタイルが含まれている場合には、disableAttributeNamesに“style”を含めるなどして、コンテンツ内の指定を無効化することができます。
- スクレイピングしたコンテンツ内のフォーカスの移動は、FocusManagerではできません。
- id属性およびclass属性は、disableAttributeNamesのデフォルト値に従って削除されます。
  これらの属性を有効とし、かつスクレイピングしたコンテンツに含まれる、id属性およびclass属性が、Ajaxフレームワークが作成する属性と重複した場合、Ajaxフレームワークとして動作は保障されません。“5.1.3 画面部品表示時のエラー”も参考にしてください。

2.7 メニュー部品

メニュー部品は、メニュー形式で項目リストを表示する部品です。

ここでは、メニュー部品の設定内容、および設定方法について説明します。

2.7.1 ContextMenu

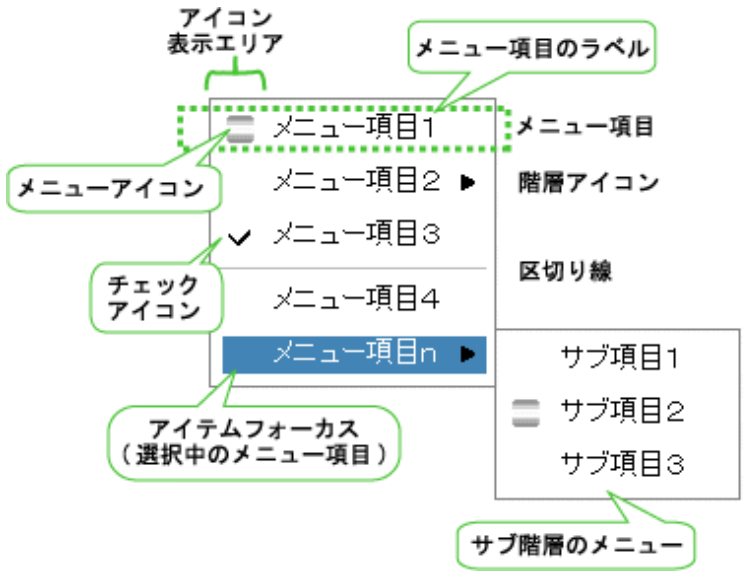
ContextMenuは、マウスの右クリックによりコンテキストメニューを表示する部品です。ContextMenuのメニュー項目の内容は、Ajaxフレームアプリケーションで独自に定義することが可能です。また、表示や非表示のほか、内容を動的に変更することができます。

注意

Webブラウザのデフォルトの ContextMenuの表示と重複しないように、アプリケーション側で制御する必要があります。詳細は“ContextMenuManagerクラス”を参照してください。

- 表示例
- 記述形式
- プロパティ
- スタイルプロパティ
- イベントリスナ
- JavaScript API
- 補足事項

表示例



ContextMenuで表示される部分を、以下のように呼びます。

- メニュー項目(1行分の項目)
- メニューアイコン
- メニュー項目のラベル
- 区切り線
- 階層アイコン
- アイテムフォーカス:メニュー項目上にマウスポインタがあり背景色が変わっている状態
- 選択可状態:メニュー項目から選択できるようになっている状態
- 選択不可状態:メニュー項目から選択できなくなっている状態(グレー表示)

ブラウザの表示領域とContextMenuの表示

マウスの右クリック時のマウスポインタの座標とブラウザの表示領域に応じて、コンテキストメニューの表示座標(サブメニューも含む)を移動させ、表示領域のスクロール処理が発生しないように最大限表示できるようにします。

記述形式

```
<div rcf:id="model1" rcf:type="MenuModel" rcf:object="modelData"></div>
<div rcf:type="ContextMenu" rcf:id="ctxm1" rcf:data="{model1.menuData}"></div>
```

注意

子要素は指定できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

ポイント

MenuModelの仕様については、“3.1.3 MenuModel”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
data	String	メニュー形式のObjectのルートを指定します。データには、ルート以降の各メニュー項目のデータを記述します。 “dataプロパティと表示データの指定方法”を参照してください。	不可	なし	バインド	可	可
timerInterval	Number	子階層のメニューを表示するまでの操作待ち時間をミリ秒で設定します。	可	200	値	不可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

dataプロパティと表示データの指定方法

以下のJavascriptデータで1つのメニュー項目を表現し、1つのルートのメニュー項目から子階層のメニュー項目のデータを入れ子形式で指定します。

```
{
    name:    "item01",          // メニュー項目名
    label:     "項目1",         // メニュー項目のラベルの表示文字列
    isChecked:    false,        // メニュー項目のチェック済みフラグ(true:チェック済み、false:チェック未)
    isSelectable:   true,       // メニュー項目の選択可否(true:選択可、false:選択不可)
    headImage:
        {   normal:  "/img/icon_item01.gif"          // メニューアイコン
        },
    eventType:  "ctxm_item01",     // メニュー項目が選択されたときに送出されるイベントタイプのベース名
    children: []                   // 子階層のメニュー項目のリスト
};
```

子階層のメニュー項目を持つ場合は、以下のように記述します。

```
{
    name:    "item01",
    label:   "項目1",
    isSelectable:  true,
    isChecked:     false,
    headImage:
        {   normal:"/img/icon_item01.gif"
        },
    eventType:     "ctxm_item01",
    children: [
          {
           name:  "input_list1",
           label: "入力候補1",
           eventType: "ctxm_item01"
          },
          {
           name:  "input_list2",
           label: "入力候補2",
           eventType: "ctxm_item01"
          },
          {
           name:  "input_list3",
           label: "入力候補3",
           eventType: "ctxm_item01"
           }
    ] //  children
};
```

メニュー項目ごとに持つプロパティを説明します。

プロパティ名	データ型	説明	省略	省略値
name	String	メニュー項目の名前を指定します。 メニュー項目の位置を示すパスを表現する際に使用します。同一メニュー項目の子階層のメニュー項目では、nameの値は一意でなければなりません。/を含む文字列、および空の文字列は指定できません。 メニュー項目名にrcf-ContextMenu-hlineを指定すると区切り線が表示されます。	不可	なし
label	String	メニュー項目のラベルに表示する文字列です。	可	nameの値
eventType	String	メニュー項目を選択した場合に送出されるイベントタイプ名を指定します。 送出されるイベントタイプ名は、“ContextMenuのID”＋“_”＋eventTypeの形式となります。 このイベントタイプ名に対応するイベントリスナをイベントマップに定義し、rcf.event.EventRegistrar.registerEventsで登録することにより、個々のメニュー項目に対応したユーザ独自のイベントリスナを呼び出すことができます。	可	なし
isChecked	Boolean	チェック済みかどうかを指定します。 ・ true: チェック済み ・ false: チェック未 trueの場合、メニューアイコンより優先してチェックアイコンがアイコン表示エリアに表示されます。	可	false
isSelectable	Boolean	選択可能かどうかを指定します。 ・ true: 選択可能 ・ false: 選択不可	可	true
headImage	Object	メニューアイコンのイメージを指定します。 指定しない場合、メニューアイコンは表示されません。	可	なし
children	Array	子階層のメニュー項目のデータを保持するObjectの配列を指定します。 配列の1要素に1つのメニュー項目を割り当てます。省略時はメニュー項目がないことを示します。データをあとで追加する場合は、children:[]を指定します。	可	なし

注意

- ルート階層のchildren配列には、必ず1つ以上のメニュー項目のデータを定義してください。空の配列(children: [])は指定できません。
- ルート階層は表示されません。
- dataプロパティに対する処理については、“JavaScript API”を参照してください。
- 同一メニュー項目の子階層のメニュー項目間でname値の重複があった場合は、エラーとなり、ブラウザのエラーメッセージが表示されます。また、name値が重複したメニュー項目を検出した時点で、メニュー項目の描画は中止されます。name値に / を含む場合も同様のエラーとなります。
- データプロバイダによりisSelectableプロパティをfalseに変更した場合、該当するメニュー項目は選択不可の状態となります。

ー labelまたはnameに以下の特殊文字が指定された場合は、ContextMenu表示時に以下の値に変換されます。

元の文字	変換後の文字
&	&amp;
<	&lt;
>	&gt;
"	&quot;
'	&#39;
空白(半角空白およびタブ)	

headImageプロパティ

以下にheadImageのプロパティを示します。

プロパティ名	データ型	説明
normal	String	メニューアイコンのURLまたはパスを指定します。 空の文字列を指定した場合、および省略時は、メニューアイコンは表示されません。
unselected	String	選択不可のメニューアイコンのURLまたはパスを指定します。 空の文字列を指定した場合、および省略時は、メニューアイコンは表示されません。

headImageプロパティで指定するメニューアイコンは、14×14ピクセルまたは16×16ピクセルの大きさの透過GIFを推奨します。

それぞれのURLには、クエリ文字列およびURLリライティングで用いるセションIDを付加することができます。詳細は、“ユーザーズガイド”を参照してください。

注意

プロパティで指定した文字列に対するエスケープ処理は行いません。フォーム画面などからユーザが入力したデータを本プロパティに設定する場合は、アプリケーション側で < > & " ' の5つの文字をエスケープするようにしてください。

dataプロパティの記述例

以下にdataプロパティの記述例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {       // モデルデータ名は任意
    menuData: {
        name:     "root",  // ルート階層
        children: [
          { // 1つ目のメニュー項目
            name:     "item01",
            label:      "項目1",
            isChecked:        false,
            isSelectable:       true,
            headImage:
                {   normal:  "/img/icon_img1.gif"
                },
            eventType:      "ctxm_item01"
          },
          {
            name: "rcf-ContextMenu-hline"
          },
          { // 2つ目のメニュー項目
            name:    "item02",
```

```
            label:    "項目2",
            isChecked:    false,
            isSelectable:  true,
            headImage:
                {    normal:  "/img/icon_img3.gif"
                },
            eventType:       "ctxm_item02",
            children: [
                    {
                    name: "list1",
                    label:  "候補1",
                    eventType:  "ctxm_item02"
                    },
                    {
                    name: "list2",
                    label:  "候補2",
                    eventType:  "ctxm_item02"
                    },
                    {
                    name: "list3",
                    label:  "候補3",
                    eventType:  "ctxm_item02"
                    }
            ] // children
          }
        ] // children
    }
};
//]]>
</script>
<body>
<div rcf:id="modelData1" rcf:type="MenuModel" rcf:object="modelData"></div>
<div rcf:id="ctxm1" rcf:type="ContextMenu" rcf:data="{modelData1.menuData}"></div>
</body>
```

このdataプロパティが指定された場合のContextMenuの表示例は、以下になります。

ContextMenuのフォーカス

ContextMenu部品にフォーカスを当てることはできません。メニュー項目上にマウスポインタがある場合は、メニュー項目にはアイテムフォーカスがあたります。

マウス操作

マウスを右クリックしたあと、マウスポインタをContextMenu上で移動して操作できます。

- ContextMenu上のメニュー項目上にマウスポインタを重ねると、アイテムフォーカスが表示されます。当該メニュー項目に子階層のメニュー項目がある場合、子階層のメニュー項目が表示されます。
- ContextMenu上のマウスポインタの上下移動により、上下方向にアイテムフォーカスが移動します。
- ContextMenuのエリアからマウスポインタが外れると、メニュー項目のアイテムフォーカスは解除されます。

操作	処理
ContextMenuのエリア上をクリック	アイテムフォーカス状態のメニュー項目を選択します。
ContextMenuのエリア外をクリック	ContextMenuを非表示にするようにアプリケーション側で制御する必要があります。詳細は、“ContextMenuManagerクラス”を参照してください。

キーボード操作

マウスを右クリックしたあと、キーによるContextMenuの操作ができます。

操作	処理
↑	アイテムフォーカスが上に移動します。
↓	アイテムフォーカスが下に移動します。
→	子階層がある場合は子階層を表示し、アイテムフォーカスを子階層に移動します。
←	子階層を非表示にして親階層に戻ります。
Enter	アイテムフォーカス状態にあるメニュー項目を選択します。
ESC	ContextMenuを非表示にします。

スタイルプロパティ

本部品全体のスタイルプロパティを以下に示します。

パーツ名	プレフィックス	クラス名	指定可能なスタイル
部品全体	なし	rcf-ContextMenu	・背景色(注1) ・ボーダー(注2) ・パディング ・マウスポインタの形状 ・zIndex(注3)
メニュー項目	itemMain	rcf-ContextMenu-itemMain	・パディング
アイテムフォーカスが当たっているメニュー項目	itemFocus	rcf-ContextMenu-itemFocus	・背景色(注4) ・ボーダー
メニュー項目のラベル	label	rcf-ContextMenu-label	・カラー ・フォント(注5) ・パディング(注6)
メニュー項目のラベル(アイテムフォーカスが当たっているラベル)	focusedLabel	rcf-ContextMenu-focusedLabel	・カラー(注7) ・フォント(注5) ・パディング(注6)
メニューアイコン	headImage	rcf-ContextMenu-headImage	・パディング
非選択	unselectable	rcf-ContextMenu-unselectable	・カラー(注8) ・フォント(注5) ・パディング(注6)

注1) デフォルトの背景色は、#FFFFFF(白)です。

注2) デフォルトのボーダー色は、#808080(灰色)です。

注3) zIndexのデフォルトは、30000です。

注4) デフォルトの背景色は、#4682B4(スチールブルー)です。

注5) デフォルトのフォントサイズは14pxです。
Internet Explorer 6では、lineHeightは指定できません。
フォントサイズは一定にしてください。一定にしない場合、表示が崩れる可能性があります。

注6) padding-left、padding-rightが有効となります。
パディングサイズは一定にしてください。一定にしない場合、表示が崩れる可能性があります。

注7) デフォルトのフォント色は、#FFFFFF(白)です。

注8) デフォルトのフォント色は、#C0C0C0(シルバー)です。

詳細は、“2.9 スタイルプロパティ”を参照してください。

マウスポインタの形状

名前	データ型	説明	省略	省略値
cursor	String	マウスポインタの形状を指定します。	可	ブラウザのデフォルトの形状

ポイント

ContextMenuの横幅、および子階層の横幅は、メニュー項目のラベルの最大長に合わせて表示されます。

イベントリスナ

本部品全体のイベントリスナを以下に示します。

アプリケーションでは、以下のイベントリスナの発生をキャッチして、アプリケーション独自の動作を記述することができます。

- 各ContextMenuを表示したとき
- 各ContextMenuを非表示としたとき
- メニュー項目を選択したとき
- モデルのデータが変更されたとき

名前	説明	イベントオブジェクト
onContextMenuShow	ContextMenuを表示したときに呼ばれます。	ContextMenuEvent
onContextMenuHide	ContextMenuを非表示したときに呼ばれます。	
onContextMenuSelect	メニュー項目を選択したときに呼ばれます。	
指定イベントタイプ名(注)	メニュー項目を選択したとき	
onDataChange	モデルのデータが変更されたとき	MenuDataChangeEvent

注) メニュー項目が選択された際に、MenuModelのオブジェクトのeventTypeで指定されたイベントタイプ名を持つイベントが送出されます。
このとき、イベント名は一意性保証のため、“ContextMenuのID” + “_” + eventType の形式となります。

例) ContextMenuのID=“ctxm1”、eventType= item01 の場合
送出されるイベント名=“ctxm1_item01”

eventTypeが省略された場合は、上記イベントタイプ名のイベントは送出されません。

部品共通のイベントリスナもあります。詳細は、“2.8.2 画面部品共通イベントリスナ”を参照してください。

ポイント

dblclickイベントは発生しません。最初のクリックでContextMenuが非表示になるためです。

JavaScript API

ContextMenuのJavascript APIには、ContextMenuクラスとContextMenuManagerクラスがあります。

- ContextMenuクラス
  他部品と同様、部品で使用できるJavascript APIです。
- ContextMenuManagerクラス
  ContextMenuを管理するクラスで、ブラウザの右クリックの制御やContextMenuを表示するための制御などを行います。

ContextMenuクラス

以下に、ContextMenuクラスのJavaScript APIを説明します。

■showメソッド

メソッド	show(eventObject)	
引数	eventObject [Object]	イベントオブジェクト(ActionEventまたはnativeEvent)
戻り値	なし	
例外	なし	
説明	ContextMenuを表示します。すでに表示済みの場合は何もせず復帰します。	

■getNodeDataProviderメソッド

メソッド	getNodeDataProvider(nodePath)	
引数	nodePath [String]	メニュー項目のパス名
戻り値	[Object]	指定したメニュー項目のデータにアクセスするためのDataProvider
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合 ・ nodePathに空文字列が指定された場合 ・ nodePathに存在していないメニュー項目のパスが指定された場合	
説明	指定したメニュー項目のデータにアクセスするためのデータプロバイダを取得します。 メニュー項目のデータは、dataプロパティ中の指定メニュー項目とその配下を表示するための部分データとなります。	

部品共通のJavaScript APIもあります。ただし、部品共通のshowメソッドは使用できません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

ContextMenuManagerクラス

ContextMenuManagerは、ContextMenuの表示に関わる動作を管理するクラスです。
ContextMenuManagerクラスとそのメソッドを使用するには、“rcf.widget.menu.ContextMenuManager.getInstance();”により、オブジェクトを生成してから使用する必要があります。

以下に、ContextMenuManagerクラスのJavascript APIを説明します。

■getInstanceメソッド

メソッド	getInstance()
引数	なし

戻り値	[Object]	ContextMenuManagerのオブジェクト
例外	なし	
説明	ContextMenuManagerのオブジェクトを返します。 本メソッドを利用する場合は、以下のようにクラスを指定します。 ctxm_mgr = rcf.widget.menu.ContextMenuManager.getInstance();	

■getContextMenuListメソッド

メソッド	getContextMenuList()	
引数	なし	
戻り値	[Arrey]	ContextMenuManagerに登録されているContextMenuのオブジェクトの配列
例外	なし	
説明	ContextMenuManagerに登録されているContextMenuのオブジェクトの配列を返します。	

■onClickHideMenuメソッド

メソッド	onClickHideMenu(listener)	
引数	listener [Function]	イベントリスナ
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	マウスのクリックですべてのContextMenuを非表示にするリスナを設定します。	

■hideAllメソッド

メソッド	hideAll()
引数	なし
戻り値	なし
例外	なし
説明	すべてのContextMenuを非表示にします。

■hideメソッド

メソッド	hide(ctxm)	
引数	ctxm [Object]	ContextMenuのオブジェクト
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定されたContextMenuを非表示にします。	

■onContextMenuByElementメソッド

メソッド	onContextMenuByElement(element, listener)	
引数	element [Object]	DOM要素
	listener [Function]	イベントリスナ

戻り値	なし
例外	引数が不正な場合、RCF18850のエラーとなります。
説明	指定された要素上にマウスポインタがある場合、マウスの右クリックで呼ばれるリスナを設定します。

■onContextMenuByIdメソッド

メソッド	onContextMenuById(id, listener)	
引数	id [String]	UI部品の部品ID
	listener [Function]	イベントリスナ
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定されたUI部品の要素上にマウスポインタがある場合、マウスの右クリックで呼ばれるリスナを設定します。	

■resetContextMenuByElementメソッド

メソッド	resetContextMenuByElement(element)	
引数	element [Object]	DOM要素
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	onContextMenuByElementで設定されたリスナを削除します。	

■resetContextMenuByIdメソッド

メソッド	resetContextMenuById(id)	
引数	id [String]	UI部品の部品ID
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	onContextMenuByIdで設定されたリスナを削除します。	

■preventDefaultCtxmByElementメソッド

メソッド	preventDefaultCtxmByElement(element)	
引数	element [String]	対象要素
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定された要素上でのWebブラウザの右クリックの動作を抑止します。	

■preventDefaultCtxmByIdメソッド

メソッド	preventDefaultCtxmById(id)	
引数	id [String]	UI部品の部品ID

戻り値	なし
例外	引数が不正な場合、RCF18850のエラーとなります。
説明	指定されたUI部品の要素上でWebブラウザのデフォルトの右クリックの動作を抑止します。

■preventDefaultCtxmByIframeIdメソッド

メソッド	preventDefaultCtxmByIframeId(iframeId)	
引数	iframeId [String]	iframeのID
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定されたiframe上でのWebブラウザの右クリックの動作を抑止します。	

■resetDefaultCtxmByElementメソッド

メソッド	resetDefaultCtxmByElement(element)	
引数	element [String]	対象要素
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定された要素上でのWebブラウザの右クリックの動作を元に戻します。	

■resetDefaultCtxmByIdメソッド

メソッド	resetDefaultCtxmById(id)	
引数	id [String]	部品ID
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定された要素上でのWebブラウザの右クリックの動作を元に戻します。	

■resetDefaultCtxmByIframeIdメソッド

メソッド	resetDefaultCtxmByIframeId(iframeId)	
引数	iframeId [String]	IframeのID
戻り値	なし	
例外	引数が不正な場合、RCF18850のエラーとなります。	
説明	指定されたiframe上でのWebブラウザの右クリックの動作を元に戻します。	

注意

FragmentContainer上のUI部品に対して、以下のJavaScript APIを設定している場合、FragmentContainerをアンロードする際には、それぞれに対応したJavaScript APIを使用してリセットしてください。

設定しているJavaScript API	リセットに使用するJavaScript API
onContextMenuById	resetContextMenuById

設定しているJavaScript API	リセットに使用するJavaScript API
onContextMenuByElement	resetContextMenuByElement
preventDefaultCtxmByIframeId	resetDefaultCtxmByIframeId

ContextMenuの利用例1

コンテクストに応じて別メニューに切り替える例を以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData1 = {
         menuData: {
                ...
  };
  var modelData2 = {
         menuData: {
                ...
  };
  var eventMap = {  // メニュー項目が選択された際に呼び出されるイベントリスナ
        ctxm1:{  ctxm1_item01: func01 },
        ctxm2:{  ctxm2_item01: func02 }
  };
  RCF.addInitializedListener(function(eventObject) {

      rcf.event.EventRegistrar.registerEvents(eventMap, "eventMap");

      // ページ全体でWebブラウザのデフォルトの右クリックの動作を抑止。
         var ctxm_mgr = rcf.widget.menu.ContextMenuManager.getInstance();
         ctxm_mgr.preventDefaultCtxmByElement(document);

      // 右クリックで呼び出すContextMenuを切替え表示。
         ctxm_mgr.onContextMenuByElement(document, mkContextMenu);

      // マウス左クリックで非表示にするためのイベントリスナを呼び出す
         ctxm_mgr.onClickHideMenu( hideContextMenu);
  });
  function mkContextMenu(eventObject) {
  var focus_id = fm1.getFocusedComponentId();
        if (focus_id == "t1") {          // FocusがTextInputのt1にある
            ctxm1.show(eventObject);        // 表示
        }
        else if (focus_id == "t2") {   //FocusがTextInputのt2にある
            ctxm2.show(eventObject);        // 表示
        }
        else {
            var ctxm_mgr = rcf.widget.menu.ContextMenuManager.getInstance();
            ctxm_mgr.hideAll();
        }
  }
  // マウス左クリックで起動されるイベントリスナ
  function hideContextMenu(eventObject) {
    var ctxm_mgr = rcf.widget.menu.ContextMenuManager.getInstance();
    ctxm_mgr.hideAll();
  }
  // メニュー項目の選択された際に呼び出されるイベントリスナ
  function func01(eventObject) {
      alert("func01が呼び出されました");
  }
  function func02(eventObject) {
      alert("func02が呼び出されました");
  }
```

```
</script>
<body>
    <div rcf:id="model1" rcf:type="MenuModel" rcf:object="modelData1"></div>
    <div rcf:id="model2" rcf:type="MenuModel" rcf:object="modelData2"></div>
    <div rcf:type="ContextMenu" rcf:id="ctxm1" rcf:data="{model1.menuData}"></div>
    <div rcf:type="ContextMenu" rcf:id="ctxm2" rcf:data="{model2.menuData}"></div>
    <div rcf:type="FocusManager" rcf:id="fm1" rcf:targets="t1;t2"></div>
    <div rcf:type="TextInput" rcf:id="t1"></div>
<div rcf:type="TextInput" rcf:id="t2"></div>

</body>
```

ContextMenuの利用例2

コンテクストに応じてメニューの内容を動的に切り替える例を以下に示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData1 = {
      menuData: {
          name:     "root"
          children: [
           { // 1つ目のメニュー項目
              name:    "item01",
              label:     "項目1",
              children:[]
           }
      ...
  };

RCF.addInitializedListener(function(eventObject) {

    // 対象要素のViewContainer上でデフォルトの右クリックの動作を抑止する。
    var ctxm_mgr = rcf.widget.tree.ContextMenuManager.getInstance();
    ctxm_mgr.preventDefaultCtxmById("vc01");

    // 右クリックで呼び出す関数を定義
    ctxm_mgr.onContextMenuById("vc01", mkContextMenu);
});
function mkContextMenu(eventObject) {

    // 例:TextInputとTextAreaでメニューの内容を切り替える
    // フォーカスがTextInputにある場合はメニュー項目をTextInput用にする
    // フォーカスがTextAreaにある場合はメニュー項目をTextArea用にする
     var focus_id = fm1.getFocusedComponentId();
     var compo = RCF.getComponent(focus_id);

     if (compo.type == "TextInput" ) {
         func_addChild("sub-TextInput");        // TextInput用のメニュー項目を追加
     }
     else if (compo.type == "TextArea" ) {
         func_addChild("sub-TextArea");         // TextArea用のメニュー項目を追加
     }
     ctxm1.show(eventObject);       // 表示
}
function func_addChild(targetName) {
    // 追加するメニュー項目
    var _newMenuNode = {
      name:             targetName,
      label:             targetName,
      eventType:      "menuSelect",
      children: []
    };
```

```
    // 追加メニュー項目のパスの部分データを取得
    var _provider = ctxm1.getNodeDataProvider( "/item01" );
    if ( _provider != null ) {
        // 子メニュー項目の配列を取得
        var _children = _provider.getDataProvider("children");
        // 子メニュー項目の配列からすべて削除
        var len = _children.getLength();
        for(var i=0; i<len; i++ ) {
            _children.removeItemAt(0);
        }
        // 子メニュー項目の配列に1つのメニュー項目のデータを追加
        _children.addItem(_newMenuNode);
    }
}
</script>
<body>
    <div rcf:id="model1" rcf:type="MenuModel" rcf:object="modelData1"></div>
    <div rcf:type="ContextMenu" rcf:id="ctxm1" rcf:data="{model1.menuData}" ></div>
    <div rcf:id="vc01" rcf:type="ViewContainer">
    <div rcf:type="FocusManager" rcf:id="fm1" rcf:targets="t1;ta1"></div>
      <div rcf:type="TextInput" rcf:id="t1"></div>
      <div rcf:type="TextArea" rcf:id="ta1"></div>
    </div>
</body>
```

補足事項

- **ContextMenuで表示する項目のサイズ**
  ContextMenuで表示する項目のサイズを設定する際の単位は、“px”を推奨しています。そのほかの単位、およびパーセント値(%)による指定をした場合、表示が崩れる可能性があります。
- **zIndex**
  ContextMenuを最前面に表示するには、zIndexの仕組みを利用します。zIndexの値は、大きいものから前面に表示されます。Windowなどほかの部品でzIndexを指定している場合は、ContextMenuのzIndexをほかの部品のzIndexより大きい値に設定してください。
- **ContextMenuが表示されない場所**
  Select部品が単一選択であるリストボックス上、およびComboBox部品の選択リスト上では、ContextMenuは表示できません。
- **style属性**
  本部品は、 タグのstyle属性(position、top、leftなど)が利用できない部品の1つです。詳細は、“1.3.5 UI部品の タグおよびタグで利用できる属性”を参照してください。

2.8 画面部品共通

ここでは、画面部品に共通する情報について説明します。

2.8.1 画面部品共通プロパティ

画面部品で共通のプロパティを、以下に示します。

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
id	String	部品のIDを指定します。(注) 指定する文字列は、以下の条件に従う必要があります。 ・ 最初の文字は半角英文字 ・ 2文字目以降の文字は、半角英文字、半角数字、アンダー	可	部品名_連番	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		バー、ハイフン、コロン、ピリオドのどれか 先頭および末尾の空白は、無視されます。上記以外の文字が指定された場合、エラーとなります。また、空文字列を指定することはできません。 ただし、rcf:idで指定した値をJavaScriptの変数としたオブジェクトとして操作したい場合(バインディング式含む)は、idには以下の文字列を指定してください。 ・ 最初の文字は半角英文字 ・ 2文字目以降の文字は、半角英文字、半角数字、アンダーバーのどれか 設定した値を変更することはできません。					
type	String	部品タイプを指定します。 設定した値を変更することはできません。	不可	なし	値	不可	不可
styleClass	String	スタイルのクラス名を指定します。 機能部品、機能付加部品には指定できません。 setPropertyにより変更することはできません。	可	""	値	不可	不可

注) HTMLのid属性とは異なります。HTMLのid属性を指定する場合は、このrcf:id属性とは別に、id属性を指定してください。

2.8.2 画面部品共通イベントリスナ

画面部品で共通のイベントリスナを、以下に示します。

名前	説明	イベントオブジェクト
onPropertyChange	プロパティが変更されたときに呼ばれます。	PropertyChangeEvent
onShow	部品が可視化されたときに呼ばれます。	ActionEvent
onHide	部品が不可視化されたときに呼ばれます。	
onClick	マウスでクリックされたときに呼ばれます。	
onDblClick	マウスでダブルクリックされたときに呼ばれます。	
onMouseDown	マウスで押されたときに呼ばれます。	
onMouseUp	マウスが離されたときに呼ばれます。	
onMouseOver	マウスが部品の上に重ねられたときに呼ばれます。	
onMouseOut	マウスが部品の上から外れたときに呼ばれます。	
onMouseMove	マウスが部品の上で動いたときに呼ばれます。	

名前	説明	イベントオブジェクト
onKeyPress	部品にフォーカスがある状態でキーが押されて離されたときに呼ばれます。	
onKeyDown	部品にフォーカスがある状態でキーが押されたときに呼ばれます。	
onKeyUp	部品にフォーカスがある状態でキーが離されたときに呼ばれます。	

マウスでのダブルクリックによるイベントについては、“5.2.1 マウスのダブルクリックによるイベントでの注意事項”を参照してください。

注意

propertychangeイベント、valuechangeイベントに関する注意事項

- IE 6 IE 7 IE 8
  Internet Explorerの場合、TextInputなどの文字入力部品でIMEを使用した文字入力を行うと、変換途中の状態もすべてpropertychange、valuechangeイベントにより通知されます。
  なお、変換途中では、IME/ブラウザの仕様により、いったん文字が消されて書き直されるようなイベントが発生する場合(2回発生)があります。

- Firefox2 Firefox3
  Firefoxの場合は、変換確定時にだけイベントが発生します。

propertychangeイベントが発生しない場合

部品のプロパティ値を、データプロバイダを用いて部分更新した場合は、propertychangeイベントは発生しません。

また、部品に対して以下の操作を行った場合も、プロパティ値が部分更新されるため、propertychangeイベントは発生しません。

部品	プロパティ	操作
Calendar	selectedDates	日付の選択状態の変更
CheckBoxGroup	selectedValues	グループ内のCheckBoxのチェック状態の変更
CheckList	selectedIndexes	項目の選択状態の変更(複数選択時)
Select	selectedIndexes	項目の選択状態の変更
	selectedValues	項目の選択状態の変更
SelectList	selectedIndexes	項目の選択状態の変更(複数選択時)
TableEdit	data	セルの編集
	selectedRows	行の選択状態の変更
TableView	data	ソートの実行
	selectedRows	行の選択状態の変更
DataGrid	data	セルの編集、ソートの実行
	selectedRows	行の選択状態の変更

ポイント

イベントリスナには、画面部品共通イベントリスナと、画面部品共通イベントリスナ名に部品名のイベントが発生した箇所を付加したイベントリスナが定義されているものがあります。
例:Panel部品のonMouseDownとonMouseDownBody
これらのイベントリスナは、常に内部の部品のイベントから通知が開始され、順次外側の部品に通知されていきます。

イベントの伝播

マウス関連のイベント(mousedown、mouseup、mouseoverなど)およびキー関連のイベント(keypress、keydown、keyup)は、発生したノードから親要素に伝播する(バブルアップする)仕組みになっています。

このため、コンテナ部品やテーブル部品、カレンダ部品などについては、内部のHTML要素上および部品でマウスイベントおよびキーイベントが発生した場合、イベントの伝播により、内部で発生したイベントもその部品のイベントとして発生し、イベントリスナが呼ばれます。

例えば、ViewStackなどのコンテナ部品はフォーカスを持たないため、単体ではkeydownなどのイベントは発生しません。しかし、コンテナ内に配置した部品でkeydownイベントが発生した場合、イベントの伝播によってコンテナ部品のonKeyDownイベントリスナも実行されます。

2.8.3 画面部品共通JavaScript API

画面部品で共通のJavaScript APIを、以下に示します。

- getPropertyメソッド
- setPropertyメソッド
- getDataProviderメソッド
- getStyleメソッド
- setStyleメソッド
- showメソッド
- hideメソッド
- isVisibleメソッド

getPropertyメソッド

メソッド	getProperty(name)	
引数	name [String]	プロパティ名
戻り値	[任意]	プロパティ値
例外	引数が省略された場合、または空文字列が指定された場合、エラーになります。(RCF11006)	
説明	プロパティ値を取得します。 スタイルプロパティを取得する場合は、getStyleメソッドを利用してください。	

setPropertyメソッド

メソッド	setProperty(name,value)	
引数	name [String]	プロパティ名
	value [任意]	プロパティ値
戻り値	なし	
例外	以下の場合にエラーとなります。 ・引数が省略された場合(RCF11006) ・nameに空文字列が指定された場合(RCF11006) ・nameに存在していないプロパティ名を指定した場合(RCF11008) ・変更不可のプロパティを指定した場合(RCF11002)	

説明	プロパティ値を設定します。 スタイルプロパティを設定する場合は、setStyleメソッドを利用してください。

getDataProviderメソッド

メソッド	getDataProvider(name)	
引数	name [String]	プロパティ名
戻り値	[任意] (データプロバイダの種類による)	データプロバイダ
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合、または空文字列が指定された場合(RCF11006) ・ nameに存在していないプロパティ名を指定した場合(RCF11008)	
説明	部品のプロパティを操作するデータプロバイダを取得します。	

getStyleメソッド

メソッド	getStyle(name)	
引数	name [String]	スタイルプロパティ名
戻り値	[任意]	プロパティ値
例外	なし	
説明	スタイルの値を取得します。 ブラウザによって取得できる値が異なる場合があります。 RCF.addInitializedListenerに設定した関数内で実行した場合、正しい値が取得できないことがあります。	

setStyleメソッド

メソッド	setStyle(name,value)	
引数	name [String]	スタイルプロパティ名
	value [任意]	プロパティ値
戻り値	なし	
例外	なし	
説明	スタイルを設定します。 指定できないスタイルプロパティ名および指定できない値が設定された場合は、表示に反映されません。 RCF.addInitializedListenerに設定した関数内で使用した場合、正しく動作しません。	

showメソッド

メソッド	show()
引数	なし

戻り値	なし
例外	なし
説明	部品を可視化します。 さらにこの部品にshowイベントが発行され、onShowイベントリスナが呼ばれます。

hideメソッド

メソッド	hide()
引数	なし
戻り値	なし
例外	なし
説明	部品を不可視化します。 さらにこの部品にhideイベントが発行され、onHideイベントリスナが呼ばれます。

isVisibleメソッド

メソッド	isVisible()	
引数	なし	
戻り値	[Boolean]	部品の可視状態 true:可視化 false:不可視化
例外	なし	
説明	部品の可視状態を取得します。	

2.8.4 文字種指定

画面部品で指定できる文字種とその値について、説明します。

識別子	文字種	値
DIGIT	数字	0～9
LETTER	英字	a～zおよびA～Z
LOWERCASE	英小文字	a～z
UPPERCASE	英大文字	A～Z
SPECIAL	特殊文字	Unicodeの以下の文字 ・ ASCIIの0x20～0x2fまでの範囲 ・ ASCIIの0x3a～0x40までの範囲 ・ ASCIIの0x5b～0x60までの範囲 ・ ASCIIの0x7b～0x7eまでの範囲
ALL	すべて	IMEによる入力はできません。

2.9 スタイルプロパティ

画面部品で共通のスタイルプロパティを、以下に示します。
スタイルプロパティの指定方法については、“1.3.2 スタイルの設定”を参照してください。

なお、省略した場合はブラウザの標準の値となります。

注意

スタイルプロパティに不正な値が設定された場合、無視されます。

ポイント

部品全体に対するスタイルプロパティ名について

部品全体に対するスタイルプロパティに付けられる名前の例を、以下に示します。

- color:色指定
- backgroundColor:背景色指定
- fontSize:フォントサイズ指定

部品の特定のパーツに対するスタイルプロパティ名について

画面部品の特定のパーツに付けられるスタイルプロパティの名前には、その部品を表すプレフィックスが以下のように付加されます。

[プレフィックス][最初が大文字のプロパティ名]

タイトル部分のプレフィックスが“title”の場合の例を、以下に示します。

- titleColor:タイトル部分の色指定
- titleBackgroundColor:タイトル部分の背景色指定
- titleFontSize:タイトル部分のフォントサイズ指定

サイズ

名前	説明
width	部品の幅を指定します。 CSSのwidthプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
height	部品の高さを指定します。 CSSのheightプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。

カラー

名前	説明
color	文字の色を指定します。 CSSのcolorプロパティの値を指定できます。
backgroundColor	背景色を指定します。 CSSのbackground-colorプロパティの値を指定できます。

フォント

名前	説明
fontFamily	文字の表示フォントを指定します。 CSSのfont-familyプロパティの値を指定できます。

名前	説明
fontSize	文字サイズを指定します。 CSSのfont-sizeプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
fontStyle	文字のイタリック表示を指定します。 CSSのfont-styleプロパティの値を指定できます。
fontWeight	文字のウェイトを指定します。 CSSのfont-weightプロパティの値を指定できます。
lineHeight	行間を指定します。 CSSのline-heightプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。

テキスト

名前	説明
textIndent	字下げ幅を指定します。 CSSのtext-indentプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
textAlign	テキストの行揃えを指定します。 CSSのtext-alignプロパティの値を指定できます。
textDecoration	文字装飾を指定します。 CSSのtext-decorationプロパティの値を指定できます。
letterSpacing	文字の間隔を指定します。 CSSのletter-spacingプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
wordSpacing	単語間の間隔を指定します。 CSSのword-spacingプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
whiteSpace	スペースの扱いを指定します。 CSSのwhite-spaceプロパティの値を指定できます。

ボーダー

名前	説明
borderWidth	枠線の太さを指定します。 CSSのborder-widthプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
borderStyle	枠線の種類を指定します。 CSSのborder-styleプロパティの値を指定できます。
borderColor	枠線の色を指定します。 CSSのborder-colorプロパティの値を指定できます。

パディング

名前	説明
padding	パディングを指定します。 CSSのpaddingプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。

マージン

名前	説明
margin	マージンを指定します。 CSSのmarginプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。

ポジション

名前	説明
position	要素の配置方法を指定します。 CSSのpositionプロパティの値を指定できます。 Ajaxフレームワークの動作定義により、コンテナ部品内にUI部品およびHTML要素を配置したときの基準位置が変わります。詳細は“1.3.7 コンテナ部品内のレイアウト”および“ユーザーズガイド”を参照してください。
top	要素の上からの位置を指定します。 CSSのtopプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
right	要素の右からの位置を指定します。 CSSのrightプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
bottom	要素の下からの位置を指定します。 CSSのbottomプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。
left	要素の左からの位置を指定します。 CSSのleftプロパティの値を指定できます。 数値で指定する場合、単位を省略することはできません。

オーバーフロー

名前	説明
overflow	領域からあふれた場合の動作を指定します。 CSSのoverflowプロパティの値を指定できます。

セル

名前	説明
textAlign	テキストの行揃えを指定します。 CSSのtext-alignプロパティの値を指定できます。
verticalAlign	セル内の上下の表示位置を指定します。 CSSのvertical-alignプロパティの値を指定できます。
whiteSpace	スペースの扱いを指定します。 CSSのwhite-spaceプロパティの値を指定できます。

第3章 機能部品

本章では、機能部品について説明します。

3.1 モデル定義部品

ここでは、モデル定義部品の設定内容および設定方法について説明します。

3.1.1 Model

Modelは、データモデルを定義する機能部品であり、ユーザアプリケーションのデータと画面部品のプロパティをバインディングするために使用します。

- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API
- スキーマ定義

記述形式

```
<div rcf:type="Model" rcf:object="modelData" ... ></div>
```

または

```
<span rcf:type="Model" rcf:object="modelData" ... ></span>
```

注意

- 子要素を指定することはできません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
object	Object	データとするオブジェクトを表す値です。 objectプロパティには、Array以外の任意のObjectを指定できます。	不可	null	値	不可	可
schema	Object	データのスキーマを表す値です。 データの各プロパティに対する検証内容が定義されています。 詳細は、“スキーマ定義”を参照してください。	可	null	値	不可	不可

部品共通のプロパティも指定できます。詳細は、“2.8.1 画面部品共通プロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

■getPropertyメソッド

メソッド	getProperty(name)	
引数	name [String]	プロパティ名
戻り値	[任意]	プロパティ値
例外	引数が省略された場合、または空文字列が指定された場合、エラーになります。 (RCF11006)	
説明	objectプロパティに設定したオブジェクトのプロパティのコピーを取得します。 プロパティ名には、“a.b.c”のようなプロパティパスも指定することができます。 このメソッドで、Modelのobjectプロパティの値やschemaプロパティの値を取得することはできません。	

■setPropertyメソッド

メソッド	setProperty(name,value)	
引数	name [String]	プロパティ名
	value [任意]	プロパティ値
戻り値	なし	
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合(RCF11006) ・ nameプロパティに空文字列が指定された場合(RCF11006) ・ nameプロパティに存在していないプロパティ名を指定した場合(RCF11008)	
説明	objectプロパティに設定したオブジェクトのプロパティ値を設定します。 プロパティ名には、“a.b.c”のようなプロパティパスも指定することができます。なお、プロパティパスには“[]”は記述できません。配列やオブジェクトを部分更新する場合には、データプロバイダを利用してください。データプロバイダを利用すると、配列の要素の追加や削除、オブジェクトのプロパティの追加が可能になります。 このメソッドで、Modelのobjectプロパティの値やschemaプロパティの値を設定することはできません。	

■getDataProviderメソッド

メソッド	getDataProvider(name)	
引数	name [String]	プロパティ名
戻り値	[任意] (データプロバイダの種類による)	データプロバイダ
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合、または空文字列が指定された場合(RCF11006) ・ nameに存在していないプロパティ名を指定した場合(RCF11008)	

説明	objectプロパティに設定したオブジェクトのプロパティを操作するデータプロバイダを取得します。 プロパティ名には、“a.b.c”のようなプロパティパスも指定することができます。

■validateメソッド

メソッド	validate(continue)	
引数	continue [Boolean]	あるプロパティの検証でエラーが発生した場合、それ以降のプロパティの検証を続行するかどうかを指定します。 true:続行します。 false:続行しません。 省略した場合は、trueになります。
戻り値	[Object]	検証結果 検証が成功した場合はnullになります。
例外	以下の場合に例外が発生します。 ・ schemaプロパティが設定されていない場合(RCF16101) ・ スキーマ定義の記述に誤りがある場合(RCF16190) ・ スキーマ定義で指定された対象となるプロパティが存在しない場合(RCF16104)	
説明	モデルデータの検証を実行します。実行するには、objectプロパティおよびschemaプロパティが指定されている必要があります。 検証を実行した場合、検証したモデルのプロパティが画面部品にバインディングしていて、かつ画面部品にValidationHelperが付加されているとき、ValidationHelperに検証結果イベント(validationerrorまたはvalidationsuccess)が自動的に送出されます。	

validateメソッドの戻り値

検証に失敗した場合、戻り値には、検証エラーに関する情報が含まれます。

```
{
  "プロパティパス": [ {検証エラー情報}, {検証エラー情報}, ... ],
  "プロパティパス": ...
     ...
  "validateAll": 検証エラー情報
}
```

名前	データ型	説明
プロパティパス	Array	プロパティパスは、検証でエラーとなったプロパティへのパスであり、配列には、検証エラーとなった項目の情報(検証エラー情報)が含まれます。
validateAll	Object	validateAllの検証に失敗した場合、検証エラー情報が設定されます。

・ 検証エラー情報
検証エラー情報は、以下のプロパティを持つオブジェクトである。

検証エラー情報(errorResult)

```
{
number: エラーコード
message: メッセージ
error: 例外オブジェクト
}
```

プロパティ名	データ型	説明
number	Number	エラーコード(注)
message	String	メッセージ

プロパティ名	データ型	説明
error	Object	エラーを表すオブジェクト ユーザがスキーマで定義したvalidate、およびvalidateAllでエラーが発生した場合、そのエラーオブジェクトが設定されます。 それ以外は、nullが設定されます。

注) エラーコードを以下に示します。

表3.1 エラーコード

コード	内容
100	typeの検証に失敗しました。
110	notNullの検証に失敗しました。
200	Booleanの制約であるvalueの検証に失敗しました。
300	Stringの制約であるlengthの検証に失敗しました。
301	Stringの制約であるminLengthの検証に失敗しました。
302	Stringの制約であるmaxLengthの検証に失敗しました。
303	Stringの制約であるpatternの検証に失敗しました。
400	Numberの制約であるfiniteの検証に失敗しました。
401	Numberの制約であるintegerの検証に失敗しました。
402	Numberの制約であるmaxInclusiveの検証に失敗しました。
403	Numberの制約であるmaxExclusiveの検証に失敗しました。
404	Numberの制約であるminInclusiveの検証に失敗しました。
405	Numberの制約であるminExclusiveの検証に失敗しました。
406	Numberの制約であるintegerDigitsの検証に失敗しました。
407	Numberの制約であるfractionDigitsの検証に失敗しました。
408	Numberの制約であるpatternの検証に失敗しました。
600	Arrayの制約であるlengthの検証に失敗しました。
601	Arrayの制約であるminLengthの検証に失敗しました。
602	Arrayの制約であるmaxLengthの検証に失敗しました。
603	Arrayの制約であるelementTypeの検証に失敗しました。
604	Arrayの制約であるelementNotNullの検証に失敗しました。
900	validateによる検証に失敗しました。
910	validateAllによる検証に失敗しました。

上記の表は、検証エラー情報のnumberプロパティに設定されるエラーコードです。

■validatePropertyメソッド

メソッド	validateProperty(path)	
引数	path [String]	検証を行うプロパティパス
戻り値	[Array]	検証エラー結果 検証が成功した場合はnullになります。
例外	以下の場合に例外が発生します。 ・ 引数pathが不正な場合(RCF16102)	

	• schemaプロパティが設定されていない場合(RCF16101) • 引数pathで指定されたオブジェクトが存在しない場合(RCF16104) • 引数pathで指定されたオブジェクトに対するスキーマが定義されていない場合(RCF16103) • スキーマ定義の記述に誤りがある場合(RCF16190)
説明	プロパティの検証を実行します。 検証を実行した場合、検証したモデルのプロパティが画面部品にバインディングしていて、かつ画面部品にValidationHelperが付加されているとき、ValidationHelperに検証結果イベント(validationerrorまたはvalidationsuccess)が自動的に送出されます。

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

スキーマ定義

ここでは、モデルオブジェクトに対する検証内容であるスキーマの記述方法について説明します。

■スキーマ全体の記述方法

スキーマは、JavaScriptのオブジェクトとして記述します。

```
{
  "プロパティパス":{ 検証内容 }
  "プロパティパス":{ 検証内容 }
  ...
  validateAll: function(object) {
    // 全体の関連性検証関数
  }
}
```

プロパティ名	データ型	説明	省略
プロパティパス	Object	プロパティパスには、検証したいプロパティ名とその検証内容を指定します。 検証内容については、“■プロパティの検証内容”を参照してください。	可
validateAll	Function	オブジェクト全体の関連性チェックを行う関数を定義します。引数には、objectプロパティで指定したオブジェクトが渡されます。	可

プロパティパス

プロパティパスには、Modelのobjectプロパティで指定したオブジェクト(連想配列)の検証したいプロパティのパス名を指定します。
例えば、パス名は以下のようになります。

```
[例]
var o = {
  a: {
     b: "xxxx",
     c: {
       d: "xxxx"
     }
  },
  f: "xxxx"
};

[パス名]
 a ⇒ "a"
 b ⇒ "a.b"
 d ⇒ "a.c.d"
```

以下に、スキーマの記述例を示します。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
    obj1: {
      string1: "文字列"
    }
  };

  var schema = {
    "obj1.string1":{
      type: "string",
      notNull: true
    },
    validateAll: function(object) {
    ...
    }
  };
//]]>
</script>

  ...

<div rcf:id="model1" rcf:type="Model" rcf:object="modelData" rcf:schema="schema"></div>
```

上記の例では、モデルのデータとして定義されているmodelDataのobj1.string1に対して、スキーマを記述しています。

プロパティパスは、“"obj1.string1"”のように""(ダブルクォーテーション)で囲む必要があります。

また、スキーマ定義では、“validateAll”という名前は予約されており、モデルのデータのオブジェクトに“validateAll”という名前があったとしても、それに対するスキーマを定義することはできません。

validateAll

validateAllには、オブジェクト全体の関連性チェックを行う関数を指定できます。この関数の引数には、objectプロパティで指定されたオブジェクトが渡されます。

validateAllでは、検証に成功した場合はtrue、検証に失敗した場合はfalseまたはエラーを送出します。

以下に例を示します。

```
{

  validateAll: function(object) {
      if (object.a > object.b) {
          return true;
      }
      throw "The value of a is invalid.";
  }
}
```

上記の例では、オブジェクトにあるaとbの比較によるチェックを行っています。aの方が大きい場合は検証が成功したことを示すtrueを、それ以外の場合は検証が失敗したことを表すエラーを返します。

エラー情報は、検証エラー情報のerrorプロパティから取得できます。検証エラー情報については、“JavaScript API”を参照してください。

なお、validateAllの検証結果は、どの画面部品のValidationHelperに対しても、検証結果イベントは送出されません。

■プロパティの検証内容

各プロパティの検証内容の記述方法を以下に示します。

```
{
  type: "データ型",
  notNull: Nullを許可しない場合はtrueを設定
```

```
  constraint: 各データ型に依存した制約条件
  validate: function(value, object) {
      // 独自の検証内容
  }
}
```

名前	データ型	説明	省略
type	String	プロパティのデータ型を指定します。データ型としては、以下が指定できます。 ・ "boolean":Boolean ・ "string" :String ・ "number" :Number ・ "date" :Date ・ "array" :Array ・ "object" :Object(Date、Arrayを含まない) プロパティの値がnullの場合、typeの検証は常に成功します。 また、プロパティがundefinedの場合、検証は常に失敗します。	不可
notNull	Boolean	nullを許可するかしないかを指定します。 ・ true: nullを許可しない プロパティ値がundefinedの場合も検証に失敗します。 ・ false: nullを許可する 常に検証が成功します。	可
constraint	Object	各データ型に依存した制約条件を指定します。 ここで記述できる内容は、typeで指定したデータ型により異なります。 記述内容の詳細については、“Booleanの制約条件”ほかを参照してください。	可
validate	Function	プロパティに対する独自の検証関数を指定します。	可

validate

validateは、プロパティに対する独自の検証関数を指定します。この関数では、第1引数にプロパティの値、第2引数にobjectプロパティに指定されたオブジェクトが渡されます。

validateでは、検証に成功した場合はtrue、検証に失敗した場合はfalseまたはエラーを送出します。

例を以下に示します。

```
{
  validate: function(value, object) {
      if (value > object.a) {
          return true;
      }
      throw "invalid value. ";
  }
}
```

上記の例では、オブジェクトにあるaとプロパティの値の比較によるチェックを行っています。プロパティの値の方が大きい場合は検証が成功したことを示すtrueを、それ以外の場合は検証が失敗したことを表すエラーオブジェクトを返します。

エラーオブジェクトは、検証エラー情報のerrorプロパティから取得できます。

Booleanの制約条件

制約条件を以下に示します。

```
{
  value: 値の状態(Boolean)
}
```

プロパティ名	データ型	説明	省略
value	Boolean	値の状態を指定します。 そのプロパティの値がtrueであるべきならば、trueを指定します。	可

Stringの制約条件

制約条件を以下に示します。“5.1.2 サロゲートペア”も参考にしてください。

```
{
  length: 文字数
  minLength: 最小文字数
  maxLength: 最大文字数
  pattern: 正規表現形式
}
```

プロパティ名	データ型	説明	省略
length	Number	文字数を指定します。	可
minLength	Number	最小文字数を指定します。 文字数が指定した値以上であれば、検証に成功します。	可
maxLength	Number	最大文字数を指定します。 文字数が指定した値以下であれば、検証に成功します。	可
pattern	RegExp	正規表現を指定します。	可

Numberの制約条件

制約条件を以下に示します。

```
{
  finite: 有限数であるかどうか
  integer: 整数であるかどうか
  maxInclusive: 値の範囲の最大値(指定値を含む)
  maxExclusive: 値の範囲の最大値(指定値を含まない)
  minInclusive: 値の範囲の最小値(指定値を含む)
  minExclusive: 値の範囲の最小値(指定値を含まない)
  integerDigits: 整数部の最大桁数
  fractionDigits: 小数部の最大桁数
  pattern: 正規表現形式
}
```

プロパティ名	データ型	説明	省略
finite	Boolean	有限数であるかどうかを指定します。 ・ true:有限数の場合 NaNや無限大は検証に失敗します。 ・ false:有限数でない場合	可
integer	Boolean	整数かどうかを指定します。有限数でない場合は、検証に失敗します。 ・ true:整数の場合 ・ false:整数でない場合	可

プロパティ名	データ型	説明	省略
maxInclusive	Number	値の範囲の最大値を指定します。(指定値を含む) 有限数でない場合は、検証に失敗します。	可
maxExclusive	Number	値の範囲の最大値を指定します。(指定値を含まない) 有限数でない場合は、検証に失敗します。	可
minInclusive	Number	値の範囲の最小値を指定します。(指定値を含む) 有限数でない場合は、検証に失敗します。	可
minExclusive	Number	値の範囲の最小値を指定します。(指定値を含まない) 有限数でない場合は、検証に失敗します。	可
integerDigits	Number	10進数での整数部の最大桁数を指定します。 有限数でない場合、または指数表記の場合、常に検証に失敗します。	可
fractionDigits	Number	10進数での小数部の最大桁数を指定します。 有限数でない場合、または指数表記の場合、常に検証に失敗します。	可
pattern	RegExp	正規表現形式を指定します。 numberオブジェクトを10進数で文字列化し、指定された正規表現形式と一致するかを検証します。 有限数でない場合、または指数表記の場合、常に検証に失敗します。	可

Dateの制約条件

Dateの制約条件はありません。

Arrayの制約条件

制約条件を以下に示します。

```
{
  length: 配列の要素の数
  minLength: 配列の最小数
  maxLength: 配列の最大数
  itemType: 配列の要素のデータ型
  itemNotNull: Nullを禁止するかどうか
}
```

プロパティ名	データ型	説明	省略
length	Number	配列の要素の数を指定します。	可
minLength	Number	配列の要素の最小数を指定します。 要素数が指定した値以上であれば、検証に成功します。	可
maxLength	Number	配列の要素の最大数を指定します。 要素数が指定した値以下であれば、検証に成功します。	可
itemType	String	配列の要素のデータ型を指定します。データ型としては、以下が指定できます。 ・ "boolean" :Boolean ・ "string" :String ・ "number" :Number ・ "date" :Date	可

プロパティ名	データ型	説明	省略
		• "array" :Array • "object" :Object(Date、Arrayを含まない) どれか1つでもデータ型が異なっていれば、検証に失敗します。 要素がnullの場合、typeの検証は常に成功します。	
itemNotNull	Boolean	配列の要素にNullを許可するかどうかを指定します。 • true: Nullを許可しない • false: Nullを許可する 常に検証が成功します。	可

■スキーマ記述例

以下にスキーマの記述例を示します。

このスキーマでは、モデルのデータであるobject1.str1、object1.num1に対する制約を記述しています。

```
<script type="text/javascript">
//<![CDATA[
  var modelData = {
         object1: {
             str1: 'abc111',
             num1: 10.5
         }
  };

  var schema = {
        "object1.str1": {
             type: 'string',
             notNull: true,
             constraint: {
                 minLength: 3,
                 maxLength: 7}
         },
        "object1.num1": {
             type: 'number',
             notNull: true,
             minExclude: 0,
             maxExclude: 20,
             fractionDigits: 2
         }
  };
//]]>
</script>

  ...


<div rcf:id="model1" rcf:type="Model" rcf:object="modelData"  rcf:schema="schema"></div>
```

3.1.2 TreeModel

TreeModelは、TreeViewで使用するデータモデルを定義する機能部品であり、ユーザアプリケーションのデータと画面部品のプロパティをバインディングするために使用します。

- 記述形式
- プロパティ
- イベントリスナ

- JavaScript API
- スキーマ定義

記述形式

```
<div rcf:id="model1" rcf:type="TreeModel" rcf:object="modelData"></div>
```

または

```
<span rcf:id="model1" rcf:type="TreeModel" rcf:object="modelData"></span>
```

注意

- 子要素を指定することはできません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

Modelのプロパティと同様です。詳細は、“3.1.1 Model”の“プロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

■getNodeDataProviderメソッド

メソッド	getNodeDataProvider(rootName,nodePath)	
引数	rootName [String]	ツリーデータのルートを示すプロパティ名
	nodePath [String]	ノードデータを取得するノードのパス名 例:/travel/Asia
戻り値	[Object]	ObjectDataProvider
例外	以下の場合にエラーとなります。 ・引数が省略された場合 ・rootNameが存在しない場合 ・nodePathに空文字列が指定された場合 ・nodePathに存在しないノードのパスが指定された場合	
説明	指定したノードのデータへアクセスするためのデータプロバイダを取得します。ノードのデータは、objectプロパティ中の指定ノードとその子ノードを構成する部分データとなります。	

ModelのJavaScript APIはすべて利用できます。詳細は、“3.1.1 Model”の“JavaScript API”を参照してください。
部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

スキーマ定義

Modelのスキーマ定義と同様です。詳細は、“3.1.1 Model”の“スキーマ定義”を参照してください。

3.1.3 MenuModel

MenuModelは、ContextMenuで使用するデータモデルを定義する機能部品であり、ユーザアプリケーションのデータと画面部品のプロパティをバインディングするために使用します。

- 記述形式
- プロパティ
- イベントリスナ
- Javascript API
- スキーマ定義

記述形式

```
<div rcf:id="model1" rcf:type="MenuModel" rcf:object="modelData"></div>
```

または

```
<span rcf:id="model1" rcf:type="MenuModel" rcf:object="modelData"></span>
```

注意

- 子要素を指定することはできません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

Modelのプロパティと同様です。詳細は、“3.1.1 Model”の“プロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

Javascript API

■getNodeDataProviderメソッド

メソッド	getNodeDataProvider(rootName, nodePath)	
引数	rootName [String]	メニューデータのルートを示すプロパティ名
	nodePath [String]	メニュー項目のデータを取得するメニュー項目のパス 例) /ctxm1/item01
戻り値	[Object]	ObjectDataProvider
例外	以下の場合にエラーとなります。 ・ 引数が省略された場合 ・ nodePathに空文字列が指定された場合 ・ nodePathに存在していないメニュー項目のパスが指定された場合	
説明	ContextMenuの定義データのメニュー項目のデータへアクセスを行うデータプロバイダを取得します。取得したプロバイダを通して、メニュー項目の更新が可能となります。	

ModelのJavaScript APIはすべて利用できます。詳細は、“3.1.1 Model”の“JavaScript API”を参照してください。
部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

スキーマ定義

Modelのスキーマ定義と同様です。詳細は、“3.1.1 Model”の“スキーマ定義”を参照してください。

第4章 機能付加部品

本章では、機能付加部品について説明します。

ポイント

機能付加部品は、1つの画面部品に対して複数設定することができます。

注意

- 1つの画面部品に対して、同じ種類の複数の機能付加部品を設定することはできません。同じ種類の複数の機能付加部品を設定した場合の動作は不定です。
- 機能付加部品が対象としていない部品に付加された場合、デバッグログが出力され、機能付加部品は無視されます。

4.1 入力支援機能付加部品

ここでは、入力支援機能の設定内容および設定方法について説明します。

4.1.1 AutoCompleter

AutoCompleterは、選択可能な文字列リストを表示し、テキストフィールドの入力を支援する機能です。

- 表示例
- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API
- 補足事項

AutoCompleterは、TextInputとSelectListをつなぐ機能付加部品です。

機能付加対象は、TextInputです。

表示例

以下にAutoCompleterの表示例を示します。

図4.1 表示例

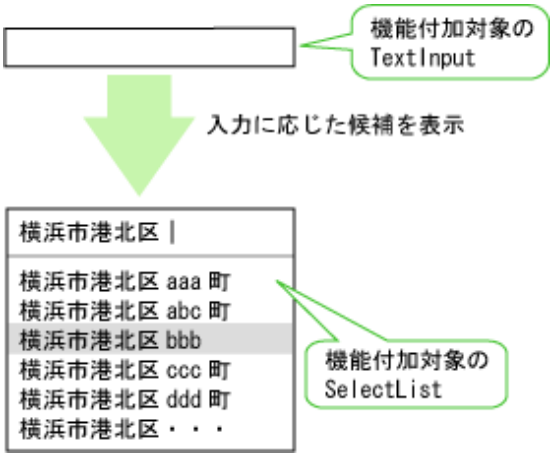
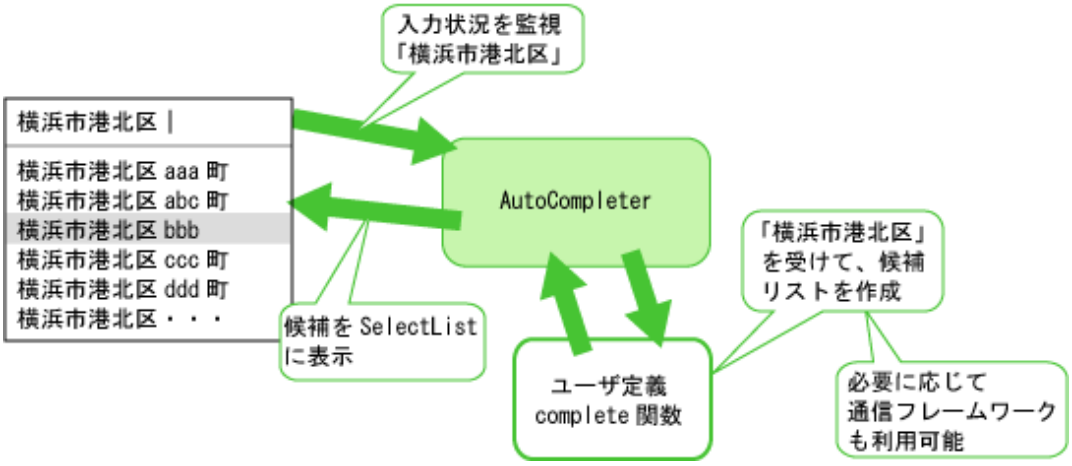


図4.2 動作イメージ



1. 候補リストが表示されている状態では、↓カーソルキーにより選択できます。選択候補リストの一番上が確定候補(マウスオーバーと同じ)になります。
2. 確定候補は、↑↓カーソルキーまたはマウスオーバーで変更できます。
3. 候補リストが表示されている状態でも、選択候補がある状態でも、カーソルはTextInputの中にあり、←→カーソル移動やキー入力を受け付けます。
4. 確定候補は、マウス左クリックまたはEnterキーで確定します。(fixイベント発行)
5. 確定した場合、SelectListのラベルがTextInputに設定されます。ラベルについては、SelectListのlabelProviderプロパティを参照してください。
6. 確定した場合または候補がない場合は、SelectListは消えます。
7. 候補リストに指定されたSelectListのプロパティやスタイルプロパティは、以下のように変更されます。

名前		変更後の値
プロパティ	tabIndex	-1
スタイルプロパティ	position	absolute
	top, left	TextInputの入力部分の左下の座標
	width	(adjustプロパティがtrueの場合だけ)TextInputの入力部分のwidth
	visibility	hidden
	zIndex	TextInputのzIndex+1

記述形式

```
<div rcf:type="AutoCompleter" rcf:target="xxx" rcf:list="xxx"></div>
```

または

```
<span rcf:type="AutoCompleter" rcf:target="xxx" rcf:list="xxx"></span>
```

注意

- 子要素は記述できません。詳細は、"5.1.4 子要素を持たない部品に子要素を記述した場合の動作"を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
target	String	機能付加対象のTextInputのrcf:idを指定します。	不可	-	値	不可	不可
list	String	候補を表示するSelectListのrcf:idを指定します。	不可	-	値	不可	不可
adjust	Boolean	SelectListの幅を変更するかどうかを指定します。 ・ true:候補を表示するSelectListの幅を、機能付加対象のTextInputの幅に合わせる ・ false:SelectListの幅は変更しない	可	true	値	不可	不可
completer	complete関数を持つオブジェクト	キー入力操作ごとに候補の配列を作成する関数です。 パラメタは以下の2つです。 ・ value: TextInputの現在の文字列 ・ callback: 候補リストに表示するデータの配列を渡す関数 配列のメンバは、String型、Object型、またはArray型	可	デフォルトcompleter (注)	値	不可	不可

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

注) デフォルトcompleter
開発初期段階でJavaScriptを記述しなくても動きを確認できるよう、以下の配列をコールバック関数に返します。

```
[
  value + "0", //現在のtextプラス0から9の数字を付けたメンバ数10の配列
  ...
  value + "9"
]
```

completerの例(ローカル)

```
<script type="text/javascript">
//<![CDATA[
  var user_obj = {
  /**
   * param value TextInputの文字列
   * param callback 候補リストを渡す関数
   */
    complete: function(value, callback) {
                //
                //候補一覧の配列を作成しコールバック関数に渡します。
                //
                var ret = ['候補1', '候補2', '候補3'];
                callback(ret);  //自発的にコールバック関数の呼出しを行う。
    }
  };
//]]>
</script>

<div rcf:id="input0" rcf:type="TextInput"></div>
```

```
<div rcf:id="output0" rcf:type="SelectList"></div>
<div rcf:type="AutoCompleter" rcf:target="input0" rcf:list="output0" rcf:completer="user_obj">
</div>
```

completerの例(非同期通信時)クライアント側

```
<script type="text/javascript">
//<![CDATA[
  var user_obj = {
    complete: function(value, callback) {
      var uiSet2Bean = {
        keyword:value
      };

      var reqParam = {
        beanId:"UiSet2Bean",
        verb:"search"
      };

      var option = {
        url:"acf/apc",
        callback:function(res){
          callback(res.menu);
        }
      };
      UjiRequest.send(uiSet2Bean,reqParam,option);
    }
  };
//]]>
</script>

<div rcf:id="autoCompleterInput" rcf:type="TextInput" rcf:value="parts-"></div>
<div rcf:id="autoCompleterList" rcf:type="SelectList"></div>
<div rcf:type="AutoCompleter" rcf:target="autoCompleterInput"
     rcf:list="autoCompleterList" rcf:completer="user_obj"></div>
```

completerの例(非同期通信時)サーバ側ビジネスクラス

```
  public Object search(DispatchContext context, uiSet2.UiSet2Bean dataBean) {
    /* keywordで開始される文字をAutoCompleterのメニューとして返します */

    String keyword = dataBean.getKeyword();
    ArrayList menu = new ArrayList();

    for (int i=0;i<MENULIST.length;i++){
      if(MENULIST[i].startsWith(keyword)){
        menu.add(MENULIST[i]);
      }
    }
    dataBean.setMenu(menu);
    return dataBean;
  }
```

非同期通信によって入力値の候補をサーバアプリケーションから取得する場合は、上記のほかにもApcoordinator連携に必要なコマンドマップやデータBeanがあります。

非同期通信によって入力値の候補を取得するサンプル(uiSet2)については、“ユーザーズガイド”の“サンプルアプリケーション”を参照してください。

イベントリスナ

名前	説明	イベントオブジェクト
onFix	入力値が確定した場合に呼ばれます。	ValueChangeEvent

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

固有のJavaScript APIはありません。

補足事項

IMEの候補リストの表示タイミング
IMEが有効になっている場合、候補リストが表示されるタイミングは、ブラウザによって異なります。

- IE 6 IE 7 IE 8
  IMEの変換中にも、候補リストが表示されます。(ただし、候補の選択ができるのは、変換を完了してからです。)
- Firefox2 Firefox3
  IMEの変換が完了した時点で、リストが表示されます。

候補リスト非表示のタイミング
表示された候補リストは、以下のどれかのタイミングで非表示となります。

- 候補を確定した場合
- ユーザの入力により、確定候補がなくなった場合(complete関数が空の配列を返した場合など)
- 機能を付加したTextInput部品がフォーカスを失った場合(TextInput外部の領域をクリックしたり、Internet Explorerでブラウザのスクロールバーを操作した場合など)

adjust=falseとした場合のリストのサイズ
adjustプロパティをfalseにする場合には、SelectListにサイズを指定してください。
SelectListのサイズを指定しないでadjustプロパティをfalseにした場合、候補リストが正しく表示されません。

マウスによる候補選択時に発生するTextInputのイベント
マウスによる候補の選択を行った場合、Internet ExplorerではTextInputからフォーカスがいったん外れます。これは、Internet ExplorerとFirefoxのブラウザの動作の違いによるものです。そのため、Internet ExplorerとFirefoxでは、TextInputのイベント発生が異なります。

- IE 6 IE 7 IE 8
  マウスによる候補選択時にいったんフォーカスが外れるため、そこでchangeイベントやblurイベントが発生します。また、そのあと、TextInputにフォーカスを戻しているため、focusイベントが発生します。
- Firefox2 Firefox3
  マウスによる候補選択時にフォーカスが外れないため、イベントは発生しません。

候補リストに指定されたSelectListのプロパティについて
候補リストに指定されたSelectListの以下のプロパティは、設定しても意味がありません。

- options (complete関数の戻り値で上書きされます。)
- multiple (falseに上書きされます。)
- selectedIndex、selectedIndexes (optionsが頻繁に入れ替わるため、この値を直接扱う意味がありません。)

4.1.2 Limiter IE 6 IE 7 IE 8

Limiterは、入力文字を制限する機能付加部品です。よく使われる可能性のある検証ルールをあらかじめ実装したものとして、NumeralOnlyLimiter、EnableCharTypeLimiterがあります。
本部品は、ユーザが独自の検証ルールを作成する場合に使用します。

- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API

なお、Limiterは、対象となる部品へのキー入力および文字列のペーストに対する制限を行う機能部品です。これら以外の方法による部品への値設定(初期値設定、APIによる更新、モデルとのバインディング)については、Limiterでは制限しません。

機能付加対象は以下のとおりです。

- TextInput
- ComboBox
- DateInput
- NumberInput

注意

Limiterを利用できるブラウザ IE 6 IE 7 IE 8

Limiterは、Internet Explorerでだけ利用可能な部品です。ほかのブラウザでは、動作しません。また、Limiterは、1つの部品につき1つだけ設定できます。
Limiterを追加すると、入力フィールドに以下の制約が追加されます。

- IMEを使った入力はできません。(IMEは常に無効です。)
- 上書き入力はできません。常に挿入による入力になります。

記述形式

```
<div rcf:type="Limiter" rcf:target="xxx"></div>
```

または

```
<span rcf:type="Limiter" rcf:target="xxx"></span>
```

注意

- 子要素は記述できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
target	String	機能付加対象のrcf:idを指定します。	不可	―	値	不可	不可
limiter	doLimit関数およびdoLimitOnPaste関数を持つオブジェクト	キー入力制限を行う関数 (注)	可	キー入力を制限しないで、ペーストを禁止する関数	値	不可	不可

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

注) limiterプロパティ
limiterプロパティは、キーが入力されたときのキー入力の制限を行うdoLimit関数、およびペーストされたときの制限を行うdoLimitOnPaste

関数を持つオブジェクトを指定します。オブジェクトには、doLimit、doLimitOnPaste関数のどちらか、または両方を持つことができます。例えば、doLimitだけを持つオブジェクトを指定した場合、ペーストされたときの制限はデフォルト動作になります。

- doLimit関数
  doLimit関数のシグネチャは以下のとおりです。
  boolean doLimit(limitObj)

  引数のlimitObjには、以下の連想配列が渡されます。

```
{
  charCode: キーコード
  cursor: カーソル位置
  value: 現在の文字列
}
```

名前	データ型	説明
charCode	Number	入力されようとしているキーのキーコード
cursor	Number	カーソル位置
value	String	現在の文字列 ただし、文字列の一部分が選択状態の場合は、選択状態部分を除いた文字列が渡されます。

  doLimit関数は、引数により、入力されようとしているキーを許可するかどうかを判断し、許可する場合はtrue、許可しない場合はfalseを返すように実装します。

注意

doLimit関数の中では、入力部品からフォーカスが外れるような処理や副作用を及ぼすような処理は行わないでください。それらの処理を行った場合、正常に入力できなくなります。

- doLimitOnPaste関数
  doLimitOnPaste関数のシグネチャは以下のとおりです。
  String doLimitOnPaste (obj)

  引数objには、以下の連想配列が渡されます。

```
{
  pasteString: ペーストされようとしている文字列
  cursor: カーソル位置
  value: 現在の文字列
}
```

名前	データ型	説明
pasteString	String	ペーストされようとしている文字列
cursor	Number	カーソル位置
value	String	現在の文字列 ただし、文字列の一部分が選択状態の場合は、選択状態部分を除いた文字列が渡されます。

  doLimitOnPasteでは、引数により、ペーストされようとしている文字列を許可するかどうかを判断し、許可する文字列を返すように実装します。許可しない場合は、""(空文字列)を返します。

注意

doLimitOnPaste関数の中では、入力部品からフォーカスが外れるような処理や副作用を及ぼすような処理は行わないでください。それらの処理を行った場合、正常に入力できなくなります。

doLimitおよびdoLimitOnPasteの例を以下に示します。

入力可能な文字列を、英字(+アンダースコア)で始まる英数字(+アンダースコア)に限定する例

```
<script type="text/javascript">
//<![CDATA[
    var limit1 = {
        doLimit: function(limitObj) {
            // 戻り値
            var acceptable = false;

            // 英字およびアンダースコア（すべての位置）
            if ((limitObj.charCode >= 65 && limitObj.charCode <= 90) ||
                (limitObj.charCode >= 97 && limitObj.charCode <= 122) ||
                limitObj.charCode == 95) {
                acceptable = true;
            }
            // 数字（2文字目以降）
            else if (limitObj.charCode >= 48 && limitObj.charCode <= 57) {
                if (limitObj.cursor > 0) {
                    acceptable = true;
                }
            }

            return acceptable;
        },

        doLimitOnPaste: function(obj) {
            // 返却文字列
            var result = "";

            // 文字の追加位置が先頭か否かを判定するフラグ
            var firstChar = (obj.cursor == 0) ? true : false;

            // ペーストする文字列を先頭から順にチェック
            for (var i = 0; i < obj.pasteString.length; i++) {
                var code = obj.pasteString.charCodeAt(i);
                var acceptable = false;

                // 英字およびアンダースコア（すべての位置）
                if ((code >= 65 && code <= 90) ||
                    (code >= 97 && code <= 122) ||
                    code == 95) {
                    acceptable = true;
                }
                // 数字（2文字目以降）
                else if (code >= 48 && code <= 57) {
                    if (!firstChar) {
                        acceptable = true;
                    }
                }

                // 入力可能な文字ならば、返却する文字列に追加
                if (acceptable) {
                    result += obj.pasteString.charAt(i);
                    firstChar = false;
                }
            }

            return result;
        }
    };
//]]>
</script>
```

```
  ...

<div rcf:id="textInput1" rcf:type="TextInput"></div>
<div rcf:id="limiter" rcf:type="Limiter" rcf:target="textInput1" rcf:limiter="limit1"></div>
```

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

4.1.3 NumeralOnlyLimiter IE6 IE7 IE8

NumeralOnlyLimiterはLimiterのひとつであり、入力文字を数値に関する文字だけに制限する機能付加部品です。

- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API

NumeralOnlyLimiterを使用すると、デフォルトでは数字だけが入力できます。
プロパティの設定により、小数点、3桁区切り文字、負号を入力可能とすることができます。小数点と負号は、入力位置および個数が制約されます。

小数点や負号が入力可能な場合、文字列のペーストでは、入力位置および個数に制約があります。ペーストした結果は、ペースト文字列の先頭から1文字ずつ入力した場合と同じ結果となります。

例1: 入力フィールドが“-123”(負号入力可能)で、先頭に“999”をペーストしようとした場合、負号の前には文字は入力不可なので、何もペーストされません。

例2: 入力フィールドが空で、ペースト文字列が“12.34.56”の場合、“12.3456”がペーストされます。(12のうしろの小数点は有効、34のうしろの小数点は2個目となるため無効)

機能付加対象は以下のとおりです。

- TextInput
- ComboBox
- DateInput
- NumberInput

NumeralOnlyLimiterを利用できるブラウザ IE6 IE7 IE8

NumeralOnlyLimiterは、Internet Explorerでだけ利用可能な部品です。ほかのブラウザでは動作しません。また、Limiterは、1つの部品につき1つだけ設定できます。
NumeralOnlyLimiterを追加すると、入力フィールドに以下の制約が追加されます。

- IMEを使った入力はできません。(IMEは常に無効です。)
- 上書き入力はできません。常に挿入による入力になります。

記述形式

```
<div rcf:type="NumeralOnlyLimiter" rcf:target="xxx"></div>
```

または

```
<span rcf:type="NumeralOnlyLimiter" rcf:target="xxx"></span>
```

注意

- 子要素は記述できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
target	String	機能付加対象のrcf:idを指定します。	不可	－	値	不可	不可
decimalSeparator	String	小数点記号を指定します。(1文字の半角特殊文字または英字) 指定された場合、指定された値を小数点記号と解釈し、入力を許可します。 小数点記号は入力文字列内に1つしか入力できません。 2文字以上は指定できません。また、groupSeparatorおよびminusSignと同じ文字は指定できません。 デフォルトでは、何も入力を許可しません。	可	""	値	不可	不可
groupSeparator	String	1000の区切り文字を指定します。(1文字の半角特殊文字または英字) 指定した場合、指定された文字を1000の区切り文字と解釈し、入力を許可します。 区切り文字の入力はどこにでもでき、3桁ごとのチェックは行われません。 2文字以上は指定できません。また、decimalSeparatorおよびminusSignと同じ文字は指定できません。 デフォルトでは、何も入力を許可しません。	可	""	値	不可	不可
minusSign	String	マイナス記号を指定します。(1文字の半角特殊文字または英字) 指定した場合、指定された文字をマイナス記号と解釈し、入力を許可します。 マイナス記号は、入力文字列の先頭にだけ入力できます。 2文字以上は指定できません。また、decimalSeparatorおよびgroupSeparatorと同じ文字は指定できません。 デフォルトでは、何も入力を許可しません。	可	""	値	不可	不可
allowPaste	Boolean	ペーストの有効/無効を指定します。 ・ true:数字だけがペーストされます。ただし、decimalSeparator、	可	false	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		groupSeparator、minusSignを設定した場合は、設定したそれぞれの値もペーストされます。 ・ false:ペーストされません。					

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

4.1.4 EnableCharTypeLimiter IE6 IE7 IE8

EnableCharTypeLimiterはLimiterのひとつであり、有効文字種を指定することで、入力文字種を制限する機能付加部品です。

- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API

機能付加対象は以下のとおりです。

- TextInput
- ComboBox
- DateInput
- NumberInput

EnableCharTypeLimiterを利用できるブラウザ IE6 IE7 IE8

EnableCharTypeLimiterは、Internet Explorerでだけ利用可能な部品です。ほかのブラウザでは、動作しません。また、Limiterは、1つの部品につき1つだけ設定できます。
EnableCharTypeLimiterを追加すると、入力フィールドに以下の制約が追加されます。

- IMEを使った入力はできません。(IMEは常に無効です。)
- 上書き入力はできません。常に挿入による入力になります。

記述形式

```
<div rcf:type="EnableCharTypeLimiter" rcf:target="xxx"></div>
```

または

```
<span rcf:type="EnableCharTypeLimiter" rcf:target="xxx"></span>
```

注意

- 子要素は記述できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
target	String	機能付加対象のrcf:idを指定します。	不可	-	値	不可	不可
types	String	有効文字種を指定します。 (注)	可	ALL	値	不可	不可
allowPaste	Boolean	ペーストの有効/無効を指定します。 ・ true:有効文字種だけがペーストされます。 ・ false:ペーストされません。	可	false	値	不可	不可

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

注) typesプロパティ
typesプロパティは、“;”を区切り文字とし、有効文字種を指定します。

以下に例を示します。

```
<div rcf:type="EnableCharTypeLimiter" rcf:target="target1" rcf:types="DIGIT; LETTER"></div>
```

- 指定できる有効文字種については、“2.8.4 文字種指定”を参照してください。
  なお、ALLが指定された場合でも、IMEによる入力はできません。
- 文字種の指定が含まれていない場合、および文字種として正しくない値が指定された場合は、エラーとなります。
- 文字種の指定では、大文字と小文字を区別します。ただし、“all”では区別されません。
- 複数個の文字種を指定した場合、入力文字が指定された文字種のどれかに該当すれば、有効となります。
  上記の例("DIGIT;LETTER")では、数字と英字(大小)が有効になります。
- 範囲の重複("LETTER;UPPERCASE")および重複指定("DIGIT;DIGIT")は、特にエラーにはなりません。

イベントリスナ

固有のイベントリスナはありません。

JavaScript API

固有のJavaScript APIはありません。

4.1.5 ValidationHelper

ValidationHelperは、検証を行うトリガとなるイベントを指定し、そのイベントの発生時にプロパティの値の検証を実行する機能付加部品です。

- 記述形式
- プロパティ
- イベントリスナ

- JavaScript API
- 補足事項

検証対象となるプロパティの条件は以下のとおりです。

- プロパティがモデルとバインディングしている。
- バインディングしているデータに対するスキーマが、モデルに定義されている。

指定したイベントが発生すると、その画面部品で上記の条件を満たすプロパティはすべて検証されます。
検証結果は、イベントとしてユーザに通知されます。

利用方法の概要は、“ユーザーズガイド”の“入力データの検証”を参照してください。

機能付加対象は以下のとおりです。

- フォーム部品(Text、Buttonを除く)
- RadioButtonGroup
- CheckBoxGroup
- TreeView
- ContextMenu

記述形式

```
<div rcf:type="ValidationHelper" rcf:target="xxx" rcf:events="xxx1; xxx2; ... "></div>
```

または

```
<span rcf:type="ValidationHelper" rcf:target="xxx" rcf:events="xxx1; xxx2; ..."></span>
```

注意

- 子要素は記述できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
target	String	機能付加対象のrcf:idを指定します。	不可	－	値	不可	不可
events	Array	検証を行うイベント名のリストを指定します。 (注)	不可	指定なし	値	不可	不可

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

注) eventsプロパティ
eventsプロパティには、検証を行うイベントを指定します。例えば、TextInputに対してchangeイベントのタイミングで検証を行いたい場合は、以下のように指定します。

```
<div rcf:type="ValidationHelper" rcf:target="target1" rcf:events="change"></div>
```

複数記述する場合は、“;”を区切り文字として、“xxxxx; yyyy; zzzz”のように記述します。

機能付加対象とその部品で指定できる検証のトリガとなるイベントは、以下のとおりです。

機能付加対象	イベント
TextInput	valuechange、change、focus、blur
CheckBox	checkstatuschange、focus、blur
RadioButton	checkstatuschange、focus、blur
TextArea	valuechange、change、focus、blur
Select	change、focus、blur
ComboBox	valuechange、change、focus、blur
DateInput	valuechange、datechange、change、focus、blur
NumberInput	valuechange、numberchange、change、focus、blur
MaskedTextInput	focus、blur
MaskedDateInput	datechange、focus、blur
SelectList	change、focus、blur
CheckList	change、focus、blur
CheckBoxGroup	selected、deselected
RadioButtonGroup	selected、deselected

複数のイベントを指定した場合、ある動作で複数回検証が実行される場合があります。

例えば、TextInputで、changeとblurを指定した場合、フォーカスを失ったときに値が変更されていれば、changeイベントとblurイベントの両方が発生するので、複数回検証が実行されます。

上記以外の部品およびイベントを指定した場合、動作は不定になります。

イベントリスナ

名前	説明	イベントオブジェクト
onValidationError	プロパティの検証でエラーがあった場合に呼ばれます。	ValidationEvent
onValidationSuccess	プロパティの検証が成功した場合に呼ばれます。	

検証に関するイベント

検証が実行された場合、validationerrorまたはvalidationsuccessイベントが発生します。
画面部品で複数のプロパティがバインディングしている場合、プロパティごとにイベントが発生します。

また、以下の場合もイベントリスナが呼ばれます。

- ModelのJavaScript APIから、対象がバインディングしているプロパティの検証を実行された場合
- 同じModelのプロパティにバインディングしている別の部品に付加されているValidationHelperにより検証が実行された場合

JavaScript API

固有のJavaScript APIはありません。

補足事項

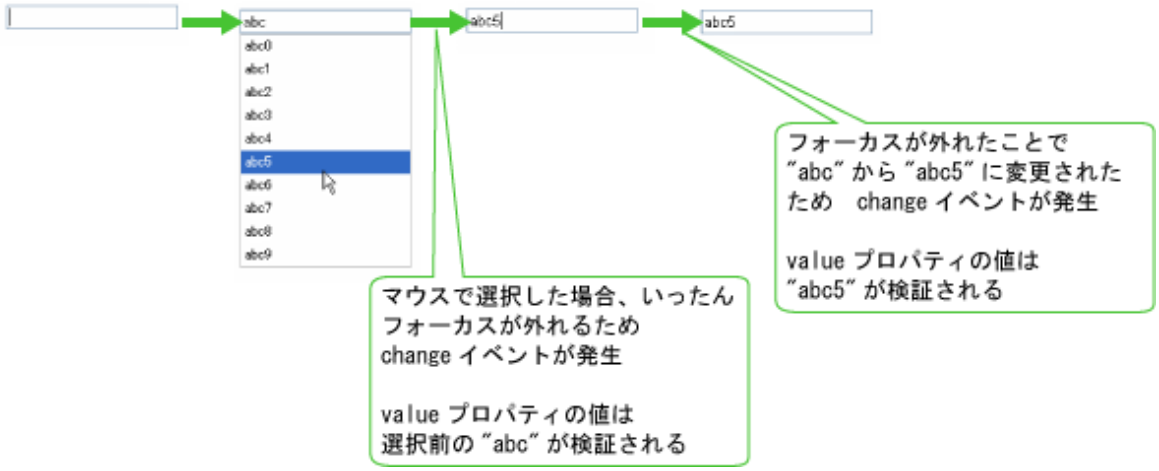
TextInputとAutoCompleterとValidationHelperを組み合わせた場合

TextInputに対して機能付加部品としてAutoCompleterを指定した場合、マウスによる候補選択時に発生するTextInputのイベントがInternet ExplorerとFirefoxで異なります。

詳細は、AutoCompleterの“補足事項”の“マウスによる候補選択時に発生するTextInputのイベント”を参照してください。

このため、TextInputに対してAutoCompleterとValidationHelperを指定し、change、blur、focusのどれかにより検証を行うように設定した場合、Internet Explorerではマウスによる候補選択時に発生するchange、blur、focusイベントにより検証が実行されます。

以下、changeイベントで検証を行うようにした場合の動作例を説明します。



マウスによる選択を行った場合、Internet Explorerではいったんフォーカスが外れるため、最初にフォーカスが当たったときからvalueの値が変更していれば、changeイベントが発生します。このchangeイベントでは、選択される前の値(上記例では“abc”)が検証されます。そして、マウスによる選択を行った値がTextInputに設定されます。

そのあと、TextInputからフォーカスを外すと、選択前の値と選択後の値が異なっていた場合には再度changeイベントが発生します。これにより再度検証が行われます。

4.2 フォーカス制御機能付加部品

ここでは、フォーカス制御機能の設定内容および設定方法について説明します。

4.2.1 FocusManager

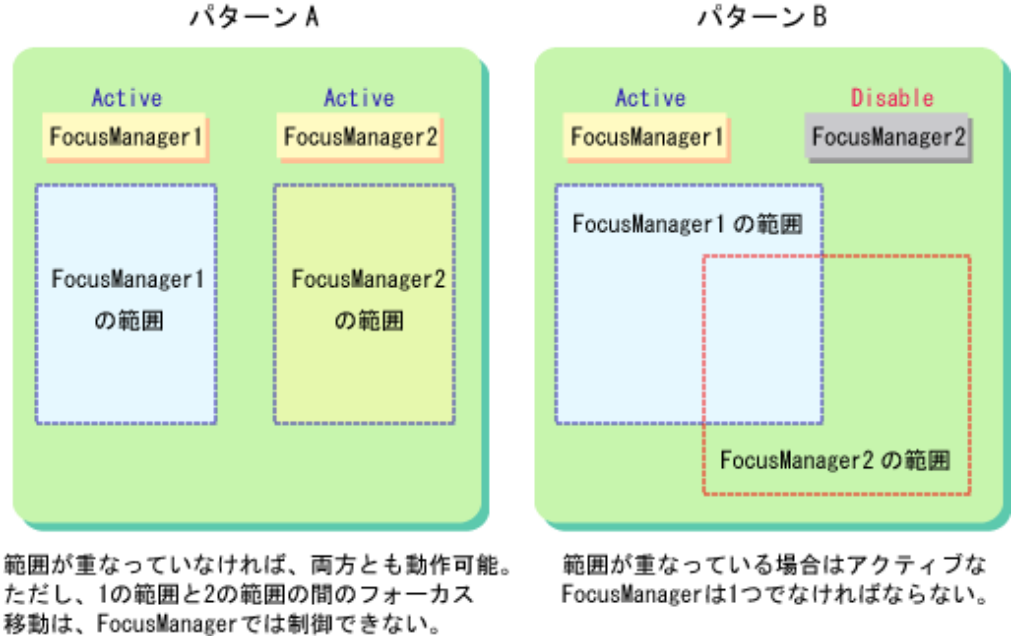
FocusManagerは、フォーカスを制御する部品であり、任意のキーでのフォーカスの移動およびフォーカスの移動順番を指定することができます。

- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API
- 補足事項

FocusManagerは、ページ内に複数記述することができます。この場合の注意事項は以下のとおりです。

- 複数のFocusManagerを記述し、それぞれに別々の部品を登録した場合、両方を動作させることができます。(パターンA)
  ただし、各FocusManagerが管理する部品間のフォーカス移動を、FocusManagerで制御することはできません。
- 複数のFocusManagerを記述し、ある部品を複数のFocusManagerに登録した場合(制御範囲に重なりがある場合)、1つのFocusManagerだけが有効になるようにユーザロジックで制御する必要があります。(パターンB)

図4.3 複数のFocusManagerの動作



パターンBの場合、FocusManagerの切替えは、JavaScriptから行います。

以下に例を示します。

```
<script type="text/javascript">
//<![CDATA[
  function changeFocusManager () {
    // FocusManagerの切替え
    manager1.setProperty("enabled", false);
    manager2.setProperty("enabled", true);
  }

//]]>
</script>

<div rcf:type="FocusManager" rcf:id="manager1" rcf:targets="..."></div>
<div rcf:type="FocusManager" rcf:id="manager2" rcf:targets="..." rcf:enabled="false"></div>
```

上記の例では、FocusManagerはmanager1およびmanager2の2つが定義されており、manager2は無効(enabledプロパティがfalse)になっています。
JavaScriptのchangeFocusManager関数では、manager1を無効にし、manager2を有効にすることにより、FocusManagerをmanager2に切り替えています。

注意

- FocusManagerのフォーカスの移動順番は、FocusManagerのtargetsプロパティで指定された順番で制御されます。画面部品に指定されているtabIndexプロパティは参照しません。
- フォーカスの移動先の部品が以下のどちらかの状態のとき、その部品にはフォーカスは移動せず、次の部品に移動します。
  - その部品が無効化されている場合
  - その部品またはその部品の親要素が表示されていない場合

記述形式

```
<div rcf:type="FocusManager" rcf:targets="xxx1; xxx2; ..." ... ></div>
```

または

```
<span rcf:type="FocusManager" rcf:targets="xxx1; xxx2; ..." ... ></span>
```

注意

子要素を指定することはできません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
targets	Array	フォーカスを制御する部品IDのリストを指定します。 (注1)	不可	-	値	可	不可
rotate	Boolean	フォーカス移動の循環(rotate)の有効/無効を指定します。 ・ true:フォーカスがリストの最後の部品にあるときに、フォーカスを次に移動しようとすると、リストの最初の部品にフォーカスが移動します。 ・ false:フォーカスは最後の部品から動きません。	可	false	値	不可	不可
tabEnabled	Boolean	ブラウザのデフォルトのTabキーでのフォーカス移動の有効/無効を指定します。 ・ true: Tabキーでのフォーカス移動可 ・ false: Tabキーでのフォーカス移動不可	可	true	値	不可	不可
nextKey	String	フォーカスを次に移すキーリスト (注2)	可	13 (Enterキー)	値	不可	不可
previousKey	String	フォーカスを前に移すキーリスト (注2)	可	13+SHIFT (Enterキー+Shiftキー)	値	不可	不可

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

注1) targetsプロパティ
targetsプロパティでは、フォーカスを制御する部品のIDのリストを指定します。指定できる部品は以下のとおりです。

- フォーム部品(Textは除く)
- TabPanel
- TableView

- TableEdit
- DataGrid
- Calendar
- CalendarButton
- TreeView
- ScrapingView(コンテンツ内は除く)
- RadioButtonGroup
- CheckBoxGroup

targetsプロパティは、RadioButtonGroupおよびCheckBoxGroupのグループごとか、またはグループに含まれる画面部品のどちらかにだけ指定できます。

また、このリストはフォーカスの移動順番も示しており、リストで指定された順番でフォーカスが移動します。

以下の例では、textInput1、textarea1、select1の順番でフォーカスが移動します。

```
<div rcf:id="textInput1" rcf:type="TextInput"></div>
<div rcf:id="select1" rcf:type="Select"></div>
<div rcf:id="textarea1" rcf:type="TextArea"></div>
<div rcf:type="FocusManager" rcf:targets="textInput1:textarea1:select1" ... ></div>
```

また、targetsプロパティの値は、動的に変更することができます。変更するには、setPropertyメソッドを利用します。

注2) キーの指定
nextKeyプロパティおよびpreviousKeyプロパティでは、フォーカスを移動させるキーを指定します。
キー指定は、“+”を区切り文字とし、以下を指定できます。

- 数字または別名:キーコード
- ALT:Altキー
- CTRL:Ctrlキー
- SHIFT:Shiftキー

キーコードは必須であり、最初に指定する必要があります。

正しい指定例	内容
13	Enterキー
13+SHIFT	Enterキー + Shiftキー
13+CTRL+SHIFT	Enterキー + Ctrlキー + Shiftキー

誤った指定例	内容
SHIFT	キーコードが指定されていなので、設定不可
CTRL+13	キーコードが最初に指定されていないので、設定不可

キーコードには、数字の直接指定のほか、以下の別名を指定することができます。
別名による指定は、その別名が表すキーの実際のキーコードを指定した場合と同様に動作します。
例えば、“ENTER+SHIFT”は、“13+SHIFT”と同じ値として扱います。

別名	対象となるキー	実際のキーコード
BACKSPACE	Backspaceキー	8
TAB	Tabキー	9
ENTER	Enterキー	13
ESC	Escキー	27

別名	対象となるキー	実際のキーコード
SPACE	スペースキー	32
PAGEUP	PageUpキー	33
PAGEDOWN	PageDownキー	34
END	Endキー	35
HOME	Homeキー	36
LEFT	←(左カーソルキー)	37
UP	↑(上カーソルキー)	38
RIGHT	→(右カーソルキー)	39
DOWN	↓(下カーソルキー)	40
INSERT	Insertキー	45
DELETE	Deleteキー	46
F1	F1キー	112
F2	F2キー	113
F3	F3キー	114
F4	F4キー	115
F5	F5キー	116
F6	F6キー	117
F7	F7キー	118
F8	F8キー	119
F9	F9キー	120
F10	F10キー	121
F11	F11キー	122
F12	F12キー	123

Tabキーによるフォーカス移動の制御

ブラウザは、デフォルトでTabキーおよびTabキー+Shiftキーでフォーカス移動が可能です。

FocusManagerを使用した場合でも、デフォルトではブラウザのTabキーによるフォーカス移動は有効です。
ブラウザのTabキーによるフォーカス移動を禁止し、FocusManagerを利用してTabキーによるフォーカス移動をしたい場合は、以下のように指定します。

- nextKeyプロパティおよびpreviousKeyプロパティに、それぞれ、Tabキー、Tabキー+Shiftキーを指定する。

イベントリスナ

固有のイベントリスナはありません。部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

独自のキー操作を設定している場合は、“5.2.3 keypressイベントに関する注意事項 ”および“5.2.5 部品に対するキー入力に関する注意事項”を参照してください。

JavaScript API

■nextメソッド

メソッド	next()
引数	なし
戻り値	なし

例外	なし
説明	フォーカスを次に移動します。

■previousメソッド

メソッド	previous()
引数	なし
戻り値	なし
例外	なし
説明	フォーカスを前に移動します。

■getFocusedComponentIdメソッド

メソッド	getFocusedComponentId()	
引数	なし	
戻り値	[String]	フォーカスを保持している部品ID
例外	なし	
説明	フォーカスを保持している部品IDを取得します。 フォーカスを保持している部品が、FocusManagerでフォーカスを制御していない場合は、空文字を返します。	

部品共通のJavaScript APIもあります。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

補足事項

フォーカス移動キー設定時には、以下の注意事項があります。

- 画面部品で有効なキーやグローバルイベント制御を行っているキーが重複してフォーカス移動キーに指定された場合、画面部品での処理が行われる前にフォーカスが移動するなど、予期しない動作をする場合があります。フォーカス移動キーには、それらと干渉しないキーを設定してください。

  例)
  “13(Enter)”および“13+SHIFT(Enter+Shift)”がフォーカス移動キーに指定された場合、TextAreaで改行が入力できなくなります。改行を入力できるようにするには、どちらかを別のキーに割り当てる必要があります。

- ブラウザのショートカットキーなどと一致するキーがフォーカス移動キーに指定された場合、フォーカス移動処理が優先され、元々の処理がキャンセルされる場合があります。フォーカス移動キーには、有効にしたいショートカットなどと干渉しないキーを設定してください。

  例)
  “66+CTRL(Ctrl + B)”をフォーカス移動キーに指定した場合、機能付加対象の部品でCtrl + Bを入力しても、以下のブラウザの標準動作が行われなくなります。

  - 
    お気に入りの整理
  - 
    ブックマークの表示

- 数字や文字をフォーカス移動キーに割り当てる場合は、CTRLまたはALTと組み合わせて指定してください。数字や文字のキーを単独でフォーカス移動キーに割り当てた場合、IMEが有効になっているとキーコードを判別できないため、フォーカスの移動が正常に行われません。

4.3 グループ化機能付加部品

ここでは、グループ化機能の設定内容および設定方法について説明します。

4.3.1 CheckBoxGroup

CheckBoxGroupは、複数のCheckBoxからなるチェックボックスグループを形成する機能付加部品です。

- 記述形式
- プロパティ
- イベントリスナ
- JavaScript API

チェックボックスグループを形成することにより、以下が可能になります。

- グループ内で選択されている複数のチェックボックスの値の取得
- グループ内のチェックボックスの選択状態が変更されたことによるイベント発生
- FocusManager制御対象
  - CheckBoxGroup自体をFocusManagerのフォーカス移動対象として指定可能
  - 構成するCheckBoxを直接FocusManagerのフォーカス移動対象とする設定は無効

記述形式

```
<div rcf:type="CheckBoxGroup" rcf:targets="xxx1; xxx2; ... "></div>
```

または

```
<span rcf:type="CheckBoxGroup" rcf:targets="xxx1; xxx2; ... "></span>
```

注意

- 子要素は記述できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
targets	Array	グループを形成するCheckBoxのidのリストです。 グループ内のカーソルキーによるフォーカスの制御は、本プロパティの指定順に従います。 本プロパティが以下の値の場合は、エラーとなります。 ・指定されたidを持つ部品が存在しない ・指定されたidを持つ部品がCheckBoxではない ・指定されたCheckBoxのvalueが空文字列である ・グループ内に同じvalueを持つCheckBoxが存在する	不可	―	値	不可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
selectedValues	String型のArray	選択されているCheckBoxのvalueのリストです。すべてのCheckBoxが非選択の場合は、長さ0の配列となります。 グループ内のどのvalueとも一致しない値が指定された場合は、エラーとなります。 初期値として本プロパティを利用する場合の注意 本プロパティが長さ0の配列の場合、関係するCheckBoxのcheckedプロパティ指定が有効となります。 長さ0の配列以外の場合、CheckBoxのcheckedプロパティ指定は無効となり、本プロパティで指定された値を持つCheckBoxが選択状態となります。	可	[]	値、バインド	可	可

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

チェックボックスグループの形成

チェックボックスグループを形成するには、グループに含めるCheckBoxのidのリストをtargetsプロパティに指定します。
以下に例を示します。

```
...
<span rcf:id="checkBox1" rcf:type="CheckBox" rcf:value="val1" rcf:label="チェックボックス1"></span>
<span rcf:id="checkBox2" rcf:type="CheckBox" rcf:value="val2" rcf:label="チェックボックス2"></span>
 ...
<span rcf:id="group1" rcf:type="CheckBoxGroup" rcf:targets="checkBox1; checkBox2" ... ></span>
 ...
```

フォーカス制御

CheckBoxGroupをFocusManagerに登録することにより、グループ単位でフォーカスを制御します。
チェックボックスグループ内のフォーカスの移動はカーソルキーで行うことができ、フォーカスの移動は、targetsプロパティに指定された順番になります。

イベントリスナ

名前	説明	イベントオブジェクト
onSelected	グループ内のCheckBoxが選択されたときに呼ばれます。	CheckStatusChangeEvent
onDeselected	グループ内のCheckBoxの選択が解除されたときに呼ばれます。	

部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

4.3.2 RadioButtonGroup

RadioButtonGroupは、複数のRadioButtonからなる相互排他的なグループを形成する機能付加部品です。

- 記述形式
- プロパティ
- イベントリスナ

- JavaScript API

ラジオボタングループを形成することにより、以下が可能になります。

- RadioButtonGroupに属するRadioButtonは、HTMLの同名のラジオボタン群と同様の動きをする
  - グループのRadioButton群で1つのフォーカスを構成
  - グループ内の選択状態を、カーソルキーで変更可能
- FocusManager制御対象
  - RadioButtonGroup自体をFocusManagerのフォーカス移動対象として指定可能
  - 構成するRadioButtonを直接FocusManagerのフォーカス移動対象とする設定は無効

記述形式

```
<div rcf:type="RadioButtonGroup" rcf:targets="xxx1; xxx2; ... "></div>
```

または

```
<span rcf:type="RadioButtonGroup" rcf:targets="xxx1; xxx2; ... "></span>
```

注意

- 子要素は記述できません。詳細は、“5.1.4 子要素を持たない部品に子要素を記述した場合の動作”を参照してください。
- 本部品は画面に表示されないため、 タグおよびタグのどちらで記述しても違いはありません。

プロパティ

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
targets	Array	グループを形成するRadioButtonのidのリストです。 グループ内のカーソルキーによるフォーカスの制御は、本プロパティの指定順に従います。 本プロパティが以下の値の場合は、エラーとなります。 ・指定されたidを持つ部品が存在しない ・指定されたidを持つ部品がRadioButtonではない ・指定されたRadioButtonのvalueが空文字列である ・グループ内に同じvalueを持つRadioButtonが存在する	不可	-	値	不可	不可
selectedValue	String	選択されているRadioButtonのvalueです。 すべてのRadioButtonが非選択の場合は空文字列になります。 グループ内のどのvalueとも一致しない値が指定された場合は、エラーとなります。 初期値として本プロパティを利用する場合の注意	可	""	値、バインド	可	不可

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
		本プロパティが空文字列の場合、checkedプロパティがtrueのRadioButtonのなかで、targetsの指定順が最後のものを選択状態にします。 空文字列以外の場合、RadioButtonのcheckedプロパティ指定は無効とし、本プロパティで指定された値を持つRadioButtonが選択状態となります。					

そのほかにも、画面部品および機能付加部品で共通のプロパティを指定することができます。詳細は、“2.8.1 画面部品共通プロパティ”、および“4.4.1 機能付加部品共通プロパティ”を参照してください。

ラジオボタングループの形成

ラジオボタングループを形成するには、targetsプロパティにグループに含めるRadioButtonのidのリストを指定します。

以下に例を示します。

```
...
<span rcf:id="radioButton1" rcf:type="RadioButton" rcf:value="val1" rcf:label="ラジオボタン1"></span>
<span rcf:id="radioButton2" rcf:type="RadioButton" rcf:value="val2" rcf:label="ラジオボタン2"></span>
 ...
<span rcf:id="group1" rcf:type="RadioButtonGroup" rcf:targets="radioButton1; radioButton2" ... ></span>
 ...
```

フォーカス制御

RadioButtonGroupをFocusManagerに登録することにより、グループ単位でフォーカスを制御します。

ラジオボタングループ内のフォーカスの移動はカーソルキーで行うことができ、フォーカスの移動順番は、targetsプロパティに指定された順番になります。

イベントリスナ

名前	説明	イベントオブジェクト
onSelected	グループ内のRadioButtonが選択されたときに呼ばれます。	CheckStatusChangeEvent
onDeselected	グループ内のRadioButtonの選択が解除されたときに呼ばれます。 (注)	

注) イベント発生順はdeselected→selected

部品共通のイベントリスナは、“2.8.2 画面部品共通イベントリスナ”を参照してください。

JavaScript API

固有のJavaScript APIはありません。部品共通のJavaScript APIは、“2.8.3 画面部品共通JavaScript API”を参照してください。

4.4 機能付加部品共通

ここでは、機能付加部品に共通する情報について説明します。

4.4.1 機能付加部品共通プロパティ

ここでは、機能付加部品で共通のプロパティについて説明します。

表の項目の意味は、“Text”の“プロパティ”を参照してください。

名前	データ型	説明	省略	省略値	属性指定	更新	部分更新
enabled	Boolean	機能の有効/無効を指定します。 ・ true:有効 ・ false:無効	可	true	値、バインド	可	不可

第5章 注意事項

本章では、UI部品を使用する際の注意事項を説明します。

5.1 画面部品の注意事項

画面部品に関する注意事項を示します。

5.1.1 画面部品全般に関する注意事項

画面部品は、HTML、JavaScript、およびCSSにより実現されています。

そのため、ブラウザによる機能範囲の違いや動作の違いにより、部品の表示や動作が異なる部分があります。

5.1.2 サロゲートペア

画面部品では、サロゲートペアの文字を2文字として扱います。

例えば、TextInputの入力できる文字列の長さを制限するmaxLengthプロパティやModelの検証機能でのstringに対する制約条件であるlength、maxLength、minLengthなどの文字列の長さを判定する部分では、サロゲートペアの文字は2文字と解釈して処理が行われます。

5.1.3 画面部品表示時のエラー

画面部品の表示時に発生する代表的なエラーは、以下のとおりです。
エラーが発生すると、画面部品の表示が中断されます。

- rcf:idで指定したIDが重複している場合(RCF11000)
- Booleanを指定する属性で、“true”または“false”以外の文字列を指定した場合(RCF11001)
- Numberを指定する属性で、数値と解釈できない文字列を指定した場合(RCF11001)
- Objectを指定する属性で、Objectと解釈できない文字列を指定した場合(RCF11001)
- 更新不可のプロパティ(id、typeなど)を更新しようとした場合(RCF11002)
- その部品の必須プロパティが指定されていない場合(RCF11003)
- バインディング式で指定したコンポーネントが見つからなかった場合(RCF11004)
- 異なるデータ型のプロパティ同士をバインディングした場合(RCF11007)

そのほか、画面部品固有のエラーやメッセージの詳細については、“ユーザーズガイド”の“エラーメッセージ”を参照してください。

5.1.4 子要素を持たない部品に子要素を記述した場合の動作

子要素を持たない部品に、子要素を記述しないでください。子要素を記述した場合、以下の動作になりますので、注意してください。

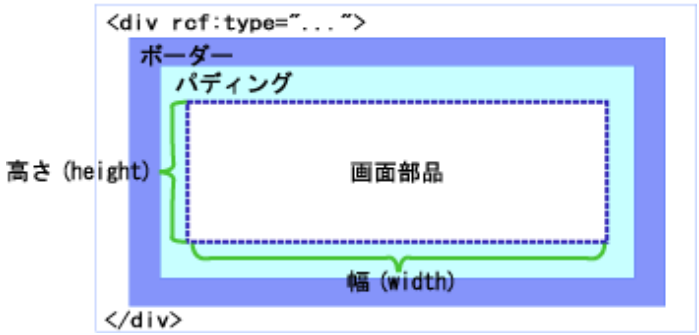
- 子要素に タグまたはタグによりUI部品を記述した場合
  子要素に記述された部品は無視されます。
- 子要素に上記以外の一般的なHTMLタグを記述した場合
  動作は不定です。

5.1.5 画面部品の幅と高さ

画面部品は、 タグまたはタグで記述します。
タグまたはタグに挟まれた部分が、画面部品の幅と高さになります。
画面部品の幅と高さは、ボーダーおよびパディングを含みません。

タグで記述した場合の高さと幅について、下図に示します。

図5.1 タグでの幅と高さ



ただし、以下の部品では、ボーダーは幅と高さに含まれます。

- Button
- Calendar

5.1.6 画面部品のレイアウト

画面部品は、Windowsの標準のレイアウトを前提として設計されています。
Windowsの[画面のプロパティ]-[デザインの詳細]などでレイアウトを変更している場合、画面部品のレイアウトが崩れる場合があります。
また、ブラウザの文字サイズを動的に変更した場合も、画面部品のレイアウトが崩れる場合があります。

5.1.7 テキスト入力部品でのCtrlキー+zによるやり直し

テキスト入力部品でCtrlキー+zが押された場合、その部品にフォーカスが当たったときの状態に戻ります。

テキスト入力部品とは、以下のものです。

- TextInput
- TextArea
- DateInput
- NumberInput
- MaskedTextInput
- MaskedDateInput
- ComboBox

5.1.8 ツールチップの表示

ツールチップの表示には、Internet ExplorerとFirefoxで、以下のような違いがあります。

- enabledプロパティにfalseを設定し、部品を非活性にした場合
  - IE 6 IE 7 IE 8
    ツールチップが表示されます。
  - Firefox2 Firefox3
    ツールチップは表示されません。
- 画像にaltを指定した場合(ComboBoxなどの画像を含む部品)
  - IE 6 IE 7 IE 8
    altがツールチップとして表示されます。

- Firefox2 Firefox3
  altはツールチップとして表示されません。

5.1.9 Dateオブジェクトとタイムゾーン

UTCとローカル時間の設定

UI部品の中には、プロパティ値やAPIのインタフェースとして、DateオブジェクトまたはDateオブジェクトを含んだ配列やオブジェクトを持っているものがあります。これらのDateオブジェクトの年月日・時分秒を設定・参照する際には、ローカル時間とUTCのどちらかを選択します。デフォルトはローカル時間です。

UTCとローカル時間の指定には、以下の2種類の方法があります。

- RCF_configで設定する
  RCF_configオブジェクトのutcプロパティに、trueまたはfalseを指定します。
  - true: UTCで動作する
  - false: ローカル時間で動作する

  省略時はfalseになります。
  RCF_configオブジェクトについては、“ユーザーズガイド”を参照してください。

```
例:
    RCF_config = {
    utc: false,
    他の動作設定項目
     ...
    };
```

- 部品のプロパティで設定する
  タグまたはタグで記述する各部品のutcプロパティ値に、trueまたはfalseを指定します。

```
例:
 <div rcf:type="DateInput" rcf:utc="true"></div>
```

両方で指定した場合は、個々の部品でのプロパティ設定が優先されます。

サーバとクライアント(ブラウザ)のロケールが同一の場合は、ローカル時間(デフォルト)が扱いやすい場合が多くなります。しかし、サーバとクライアントでロケールが異なる可能性がある場合は、UTCを使用することで、ロケールの違いに依存せずに動作させることができます。
用途や環境に応じて決定してください。

個々のDateプロパティでUTCとローカル時間のどちらを使用しているかは判断できないため、特別の理由がない限り、システム全体でどちらを使用するかを統一してください。このため、RCF_configでの設定を推奨します。

どちらの設定方法も、動作中に変更することはできません。

使用するメソッド

UTCとローカル時間とでは、それぞれ年月日・時分秒を設定・取得するメソッドが異なります。指定したタイムゾーンに合ったメソッドを使う必要があります。誤ったメソッドを使用した場合は、正しい値が設定・取得されません。

- UTCの場合
  - getUTCFullYear() :年の取得
  - getUTCMonth() :月の取得(0～11)
  - getUTCDate() :日の取得
  - getUTCDay() :曜日の取得(0～6)
  - getUTCHours() :時の取得(0～23)
  - getUTCMinutes() :分の取得

- getUTCSeconds() :秒の取得
- getUTCMilliseconds() :ミリ秒の取得
- setUTCFullYear() :年の設定
- setUTCMonth() :月の設定(0～11)
- setUTCDate() :日の設定
- setUTCHours() :時の設定(0～23)
- setUTCMinutes() :分の設定
- setUTCSeconds() :秒の設定
- setUTCMilliseconds() :ミリ秒の設定

・ ローカル時間の場合

- getFullYear() :年の取得
- getMonth() :月の取得(0～11)
- getDate() :日の取得
- getDay() :曜日の取得(0～6)
- getHours() :時の取得(0～23)
- getMinutes() :分の取得
- getSeconds() :秒の取得
- getMilliseconds() :ミリ秒の取得
- setFullYear() :年の設定
- setMonth() :月の設定(0～11)
- setDate() :日の設定
- setHours() :時の設定(0～23)
- setMinutes() :分の設定
- setSeconds() :秒の設定
- setMilliseconds() :ミリ秒の設定

5.1.10 Window、PopupCalendar部品を自動的に表示させる場合

Window部品やPopupCalendar部品を画面の初期化完了直後に自動的に表示したい場合は、以下のようにタイムアウトを設定して、部品のshowメソッドを呼ぶイベントリスナを作成し、RCF.addInitializedListener()で追加してください。

```
// Window部品を、画面の初期化完了時に自動的に表示する場合の例
RCF.addInitializedListener(
  function() {
    // wid1はWindow部品のインスタンス
    setTimeout( function(){wid1.show();}, 100);
  });
```

タイムアウトを設定しないで直接showメソッドを呼ぶと、部品の描画が崩れる可能性があります。

ポイント

Window部品は、showWindowプロパティをtrueにすることにより、自動的に表示させることもできます。

5.1.11 ページあたりの画面部品の個数について

ページあたりの画面部品の個数については、“1.3.6 ページあたりの画面部品の個数”を参照してください。

5.1.12 ページの拡大/縮小機能に関する注意点 IE 7 IE 8 Firefox3

ページの拡大/縮小機能は使用できません。ページを拡大/縮小した場合、表示が乱れる場合があります。

5.2 全部品共通の注意事項

全部品共通の注意事項を示します。

5.2.1 マウスのダブルクリックによるイベントでの注意事項

マウスでダブルクリックした場合、ブラウザの動作の違いにより、Internet ExplorerとFirefoxではイベントの発生する順番が異なります。

- IE 6 IE 7 IE 8
  mousedown → mouseup → click → mouseup → dblclick
- Firefox2 Firefox3
  mousedown → mouseup → click → **mousedown** → mouseup → **click** → dblclick

Internet Explorerでは、2回目のmousedownイベントとclickイベントが発生しません。

5.2.2 データプロバイダに関する注意事項

各部品のプロパティで、部分更新が可能と明記されているプロパティについては、データプロバイダにより、オブジェクトのプロパティや配列の要素などデータの一部分を変更することができます。

データプロバイダを利用する場合の注意事項について、以下に示します。

- データプロバイダは、部品のどのプロパティでもgetDataProviderにより取得することができます。
- 部品のプロパティで“更新”が“不可”の場合、データプロバイダのsetDataメソッドで更新することはできません。エラー(RCF11002)になります。
  “更新”が“可能”の場合、setDataによりプロパティの値が設定できます。
- 部品のプロパティで“部分更新”が“可”の場合、データプロバイダのsetData以外の更新メソッド(注)でも部分更新が行えます。
  “部分更新”が“不可”の場合は、それらのメソッドを呼び出すとエラー(RCF11010)になります。
- プリミティブ型(String、Number、Boolean)の場合、部分更新という概念はありません。そのため、各部品の“更新”および“部分更新”欄は不可になっています。
  “更新”が“可”となっているプロパティは、データプロバイダのsetDataメソッドで更新することが可能です。

注) 以下のメソッドです。

- ObjectDataProvider
  - setProperty
  - addProperty
- ArrayDataProvider
  - addItem
  - insertItemAt
  - removeItemAt
  - replaceItemAt
  - sortItems

- DateDataProvider
  - setData以外のsetから始まるメソッド

データプロバイダの種類やメソッドについては、“付録B データプロバイダ”を参照してください。

5.2.3 keypressイベントに関する注意事項 IE6 IE7 IE8

Internet Explorerでは、英数字、記号、ESC、スペース、Enterなどのキー以外が押された場合、keypressイベントは発生しません。

また、以下の部品では部品固有のキー操作を可能としているため、Internet Explorerでkeypressイベントが発生しないキーがあります。

画面部品

画面部品名	keypressが発生しないキー
SelectList	スペース
CheckList	スペース
TabPanel	nextKeyプロパティ、previousKeyプロパティに指定したキー
Calendar	スペース
PopupCalendar	スペース、Enter、ESC
CalendarButton	スペース、Enter
TableView	スペース
TableEdit	スペース、Enter、ESC(注)
DataGrid	スペース、Enter、ESC(注)

注) 編集のキャンセルが実行された場合。それ以外はイベントが発生します。

機能付加部品

機能付加部品の場合、機能付加対象の部品によって、一部のキーについてはkeypressイベントが発生しません。

例えば、FocusManagerの場合、フォーカス制御の対象となっている画面部品で、一部のキーについてkeypressイベントが発生しません。

画面部品名	keypressが発生しないキー
FocusManager	nextKeyプロパティ、previousKeyプロパティに指定したキー
AutoCompleter	Enter

5.2.4 フォーカス移動におけるInternet Explorerの動作に関する注意事項 IE6 IE7 IE8

Internet Explorerでは、画面をフレーム分割したり、<iframe>タグを使用したりしている場合で、かつそれらのフレームにスクロールバーが表示されている場合、マウスでスクロールバーをクリックした際にフォーカス移動上の不具合が発生します。

あるUI部品上にフォーカスが当たっているときに、ほかのフレームのスクロールバーをマウスでクリックすると、部品に対してblurイベントが発生するにもかかわらず、依然としてその部品に対してキー入力が可能な状態となります。このような状態でキー入力を行うと、入力したデータが失われたり誤ったデータが設定されたりする危険性があります。このような状態で、キー入力を行わないように注意してください。

スクロールバー以外の部分をクリックすることで、フォーカスとキー入力の可・不可の関係は、正しい状態に戻ります。

また、画面設計を行う際にも、フレームにスクロールバーが表示されないよう考慮してください。

5.2.5 部品に対するキー入力に関する注意事項

FocusManager部品におけるキー操作でのフォーカス移動や、Calendar部品における空白キーでの日付選択など、いくつかの部品ではキー操作による独自の処理が定義されています。

これらの部品に対してキー操作を行う際にIMEがONになっていると、正しく処理が行われない場合があります。そのような場合はIMEをOFFにして、直接入力モードでキー操作を行うようにしてください。

また、Microsoft IME以外の日本語入力システムを使用している場合は、システムによっては入力システムを停止させる必要があります。

5.2.6 組込みオブジェクトの利用に関する注意事項

UI部品のプロパティおよびモデルデータやAPIの引数として、数値、論理値、文字列を使用する場合、組込みオブジェクト(Number、Boolean、Stringオブジェクト)は使用できません。基本データ型(Number、Boolean、String型)を使用してください。

以下の例では、“設定1”のボタンを押した場合には正常にプロパティの設定が行われますが、“設定2”のボタンを押した場合には、Number型のオブジェクトを設定しようとしたため、エラーとなります。

```
<script type="text/javascript">
//<![CDATA[
  function set1() {
    // 基本型で設定(正常終了)
    list.setProperty('selectedIndex', 1);
  }

  function set2() {
    // Number型オブジェクトで設定(エラー)
    list.setProperty('selectedIndex', new Number(1));
  }
//]]>
</script>

 :
<div rcf:id="list" rcf:type="SelectList" rcf:options="foo;bar"></div>
<div rcf:type="Button" rcf:onClick="set1()">設定1</div>
<div rcf:type="Button" rcf:onClick="set2()">設定2</div>
 :
```

5.2.7 Number型のデータに関する注意事項

Internet Explorer 6、Internet Explorer 7、およびFirefoxでは、Number型を、IEEE 754で規定される64ビット倍精度浮動小数点数で扱う仕様になっています。そのため、Ajaxフレームワークで扱うNumber型のデータについては、浮動小数点演算特有の誤差が生じる場合があります。

付録A イベントオブジェクト

ここでは、画面部品で発生する各イベントで取得できるイベントオブジェクトについて説明します。

各イベントのtypeプロパティには、イベント名が指定されています。イベント名は、イベントリスナ名の先頭の“on”を削除し、残りをすべて小文字で表現したものになっています。

A.1　共通イベント

ここでは、部品で共通するイベントのイベントオブジェクトについて説明します。

A.1.1　ActionEvent

ActionEventは、画面部品が操作されたときに発生するイベントです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  browserEvent: ブラウザのイベント
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
browserEvent	Object	ブラウザのイベント JavaScriptが提供するイベントオブジェクトをラップしたオブジェクトが格納されています。詳細は、“A.1.3 BrowserEvent”を参照してください。

A.1.2　PropertyChangeEvent

PropertyChangeEventは、モデルの値が変更されたときに発生するイベントです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  propertyName: '変更された値のプロパティ名',
  oldValue: '変更前のプロパティの値',
  newValue: '変更後のプロパティの値'
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
propertyName	String	値が変更されたプロパティ名
oldValue	(任意の型)	変更前のプロパティの値
newValue	(任意の型)	変更後のプロパティの値

A.1.3 BrowserEvent

BrowserEventは、ブラウザ上で発生したイベントをラップしているオブジェクトです。

記述形式

```
{
  target: イベントを送出したノード,
  type: イベント名,
  nativeEvent: ブラウザのイベント
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したノード
type	String	イベント名
nativeEvent	Object	ブラウザが送出したイベントオブジェクト オブジェクトの内容はブラウザごとに異なります。 nativeEventは参照だけにしてください。nativeEventに対して、イベントのバブルアップの中止や、イベントのキャンセルなどを行った場合、部品の動作が不定になります。

A.2 カスタムイベント

ここでは、部品ごとのカスタムイベントのイベントオブジェクトについて説明します。

A.2.1 ValueChangeEvent

ValueChangeEventは、値が変更されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  oldValue: 変更前の値,
  newValue: 変更後の値
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
oldValue	(任意の型)	変更前のプロパティの値
newValue	(任意の型)	変更後のプロパティの値

使用イベントリスナ

- onValueChange (TextInput、TextArea、ComboBox、DateInput、NumberInput、MaskedTextInput、MaskedDateInput)
- onDateChange (DateInput、MaskedDateInput)
- onNumberChange (NumberInput)

- onYearChange (MaskedDateInput)
- onMonthChange (MaskedDateInput)
- onDayChange (MaskedDateInput)
- onHourChange (MaskedDateInput)
- onMinuteChange (MaskedDateInput)
- onSecondChange (MaskedDateInput)
- onFix (AutoCompleter)
- onDataChange (ScrapingView)

A.2.2 CheckStatusChangeEvent

CheckStatusChangeEventは、選択状態が変更されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  checked: 選択状態
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
checked	Boolean	選択状態 ・ true:選択されています。 ・ false:選択されていません。

使用イベントリスナ

- onCheckStatusChange (CheckBox、RadioButton)
- onSelected (CheckBoxGroup、RadioButtonGroup)
- onDeselected (CheckBoxGroup、RadioButtonGroup)

A.2.3 OptionStateChangeEvent

OptionStateChangeEventは、選択リストの項目の選択状態が変更されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  index: インデックス,
  value: 項目の値
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
index	Number	インデックス
value	String	項目の値

使用イベントリスナ

- onOptionSelected (Select、SelectList、CheckList)
- onOptionDeselected (Select、SelectList、CheckList)
- onChange (CheckList)

A.2.4 DateParseEvent

DateParseEventは、文字列からDateオブジェクトへの変換が実行された場合に使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  error: エラーオブジェクト,
  value: 入力テキスト,
  oldDate: 実行前のdateプロパティの値,
  newDate: 実行後のdateプロパティの値
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(DateInput)
type	String	イベント名
error	Object	エラーオブジェクト 変換に成功した場合はnull
value	String	入力テキスト
oldDate	Date	実行前のdateプロパティの値
newDate	Date	実行後のdateプロパティの値

使用イベントリスナ

- onDateParseSuccess (DateInput)
- onDateParseError (DateInput)

A.2.5 NumberParseEvent

NumberParseEventは、文字列からnumberへの変換が実行された場合に使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  error: エラーオブジェクト,
  value: 入力テキスト,
  oldNumber: 実行前のnumberプロパティの値,
  newNumber: 実行後のnumberプロパティの値
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(NumberInput)
type	String	イベント名
error	Object	エラーオブジェクト
value	String	入力テキスト
oldNumber	Number	実行前のnumberプロパティの値
newNumber	Number	実行後のnumberプロパティの値

使用イベントリスナ

- onNumberParseSuccess (NumberInput)
- onNumberParseError (NumberInput)

A.2.6 WrongKeyPressEvent

WrongKeyPressEventは、フォーマットパターンと異なるキーが入力された場合に使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  charCode: 文字コード,
  cursor: カーソル位置
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
charCode	Number	入力された文字の文字コード
cursor	Number	カーソル位置

使用イベントリスナ

- onWrongKeyPress (MaskedTextInput、MaskedDateInput)

A.2.7 InvalidValueErrorEvent

InvalidValueErrorEventは、フォーマットパターンと異なった文字列が指定されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  error: エラーオブジェクト,
  value: valueプロパティの値
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(MaskedTextInput)
type	String	イベント名
error	Object	エラーオブジェクト
value	String	valueプロパティの値

使用イベントリスナ

- onInvalidValueError (MaskedTextInput)

A.2.8 CalendarSelectionChangeEvent

CalendarSelectionChangeEventは、カレンダの選択状態が変更されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  date: 変化した日付(Date型)
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
date	Date	変化した日付

使用イベントリスナ

- onDateSelected (Calendar)
- onDateDeselected (Calendar)
- onDateFocused (Calendar)
- onDateFixed (PopupCalendar)
- onDateDblClicked (Calendar)

A.2.9 DisplayMonthChangeEvent

DisplayMonthChangeEventは、カレンダの表示月が変更されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  year: 表示年,
  month: 表示月
  firstDateOfMonth: 表示月の開始日(Date型),
  lastDateOfMonth:  表示月の最終日(Date型),
  firstDateOfCalendar: 前月終盤も含むカレンダ開始日(Date型),
  lastDateOfCalendar:  次月初旬も含むカレンダ最終日(Date型)
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
year	Number	表示年
month	Number	表示月(0-11)
firstDateOfMonth	Date	表示月の開始日
lastDateOfMonth	Date	表示月の最終日
firstDateOfCalendar	Date	前月終盤も含むカレンダ開始日
lastDateOfCalendar	Date	次月初旬も含むカレンダ最終日

使用イベントリスナ

- onDisplayMonthChanged (Calendar)
- onBeforeDisplayMonthChanged (Calendar)

A.2.10 ItemEvent

ItemEventは、テーブル部品の行や列、およびセルが操作されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  row: 行インデックス番号,
  column: 列名,
  browserEvent: ブラウザから送出されたイベント
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名

名前	データ型	説明
row	Number	行インデックス番号 列ヘッダで発生したイベント(onXXXXColumnイベント)の場合、またはダミー列の場合は、-1が設定されます。
column	String	列名 onXXXXRowイベントの場合、またはダミー列の場合は、空文字列が設定されます。
browserEvent	Object	ブラウザから送出されたネイティブイベント 以下のイベントリスナでは、nullが設定されます。 ・ onFocusRow ・ onBlurRow ・ onSelectRow ・ onDeselectRow ・ onFocusCell ・ onBlurCell ・ onSelectCell ・ onDeselectCell 詳細は、“A.1.3 BrowserEvent”を参照してください。

使用イベントリスナ

- onClickColumn (TableView、TableEdit、DataGrid)
- onDblClickColumn (TableView、TableEdit、DataGrid)
- onMouseDownColumn (TableView、TableEdit、DataGrid)
- onMouseUpColumn (TableView、TableEdit、DataGrid)
- onMouseOverColumn (TableView、TableEdit、DataGrid)
- onMouseOutColumn (TableView、TableEdit、DataGrid)
- onMouseMoveColumn (TableView、TableEdit、DataGrid)
- onClickRow (TableView、TableEdit、DataGrid)
- onDblClickRow (TableView、TableEdit、DataGrid)
- onMouseDownRow (TableView、TableEdit、DataGrid)
- onMouseUpRow (TableView、TableEdit、DataGrid)
- onMouseOverRow (TableView、TableEdit、DataGrid)
- onMouseOutRow (TableView、TableEdit、DataGrid)
- onMouseMoveRow (TableView、TableEdit、DataGrid)
- onFocusRow (TableView、DataGrid)
- onBlurRow (TableView、DataGrid)
- onSelectRow (TableView、TableEdit、DataGrid)
- onDeselectRow (TableView、TableEdit、DataGrid)
- onClickCell (TableView、TableEdit、DataGrid)
- onDblClickCell (TableView、TableEdit、DataGrid)

- onMouseDownCell (TableView、TableEdit、DataGrid)
- onMouseUpCell (TableView、TableEdit、DataGrid)
- onMouseOverCell (TableView、TableEdit、DataGrid)
- onMouseOutCell (TableView、TableEdit、DataGrid)
- onMouseMoveCell (TableView、TableEdit、DataGrid)
- onFocusCell (TableEdit)
- onBlurCell (TableEdit)
- onSelectCell (TableEdit)
- onDeselectCell (TableEdit)
- onOpenDetail (DataGrid)
- onCloseDetail (DataGrid)
- onClickLink (DataGrid)
- onClickImage (DataGrid)

A.2.11 DataChangeEvent

DataChangeEventは、テーブル部品のdataプロパティの変更によって発生するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  operation: '操作名',
  propertyName: 'プロパティ名',
  row: 行インデックス番号,
  column: 列名,
  newValue: 変更後の値,
  oldValue: 変更前の値
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
operation	String	操作 operationに設定される値は、以下のとおりです。 ・ set: 配列のインスタンスを変更 ・ remove: 配列要素を削除 ・ add: 配列要素を追加 ・ replace: 配列要素を置換 ・ insert: 配列要素を挿入 ・ update: 配列要素のあるプロパティの値を変更/ある要素にプロパティを追加 ・ sort: ソートの実行

名前	データ型	説明
propertyName	String	operationが‘update’の場合は、変更されたプロパティ名 operationが‘update’以外の場合は、空文字列
row	Number	変更された行インデックス番号 operationが‘sort'または‘set'の場合は、-1
column	String	operationが‘update’の場合は、変更されたカラム名 operationが‘update’以外の場合は、空文字列
newValue	Object	変更後の値 operationの値によって、以下のように設定されます。 ・‘set’: null ・‘remove’: null ・‘replace’: 置換後の新しい配列要素 ・‘update’: 変更後のプロパティの値 ・‘add’: 追加された配列要素 ・‘insert’: 追加された配列要素 ・‘sort’: null
oldValue	Object	変更前の値 operationの値によって、以下のように設定されます。 ・‘remove’: 削除された配列要素 ・‘replace’: 置換対象となった配列要素 ・‘update’: 変更前のプロパティの値 ・その他: null

使用イベントリスナ

- onDataChange (TableView、TableEdit、DataGrid)

A.2.12 ValidationEvent

ValidationEventは、検証が実施されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  component: 検証を行った部品,
  propertyName: 検証を行ったプロパティ名,
  value: 検証を行ったプロパティの値,
  results: 検証結果
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名
component	Object	検証を行った部品

名前	データ型	説明
propertyName	String	検証を行った部品のプロパティ名
value	Object	検証を行ったプロパティの値
results	Array	検証エラー情報の配列 validationsuccessの場合はnull

使用イベントリスナ

- onValidationError (ValidationHelper)
- onValidationSuccess (ValidationHelper)

検証エラー情報

ValidationEventのresultsには、検証エラー情報が含まれています。resultsプロパティは、以下のオブジェクトの配列が取得できます。

```
{
 number: エラーコード,
 message: エラーメッセージ,
 error: エラーオブジェクト
}
```

名前	データ型	説明
number	Number	エラーコード
message	String	エラーメッセージ
error	Object	エラーオブジェクト nullの場合もあり

エラーコードの詳細は、“3.1.1 Model”の“JavaScript API”を参照してください。

A.2.13 FragmentStateChangeEvent

FragmentStateChangeEventは、FragmentContainerで状態が変化したときに使用するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'statechange',
  state: 状態文字列
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(FragmentContainer)
type	String	イベント名 “statechange”固定
state	String	状態文字列 FragmentContainer正常系状態の種類に応じた文字列

使用イベントリスナ

- onStateChange (FragmentContainer)

A.2.14 FragmentErrorEvent

FragmentErrorEventは、FragmentContainerでエラーが起きたときに発生するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'fragmenterror',
  state: エラー発生時のFragmentContainerの状態を表す文字列,
  message: 'メッセージ',
  cause: 内部の例外オブジェクト
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(FragmentContainer)
type	String	イベント名 “fragmenterror”固定
state	String	エラーが発生したときのFragmentContainerの状態を表す文字列
message	String	以下の種類があります。 ・ サーバ接続前(接続できないなど):状況を表すメッセージ ・ サーバ接続後:HTTPレスポンスコードとレスポンス本体 ・ タイムアウト時:タイムアウトを示すメッセージ ・ フラグメントHTML内のJavaScript実行時のエラーのメッセージ ・ フラグメントHTMLのパース時のエラーメッセージ
cause	Object	内部の例外オブジェクト 内部の例外が存在しない場合はnull

使用イベントリスナ

- onFragmentError (FragmentContainer)

A.2.15 SelectedIndexChangeEvent

SelectedIndexChangeEventは、選択されたインデックスが変化したときに発生するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'selectedindexchange',
  index: インデックス
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト
type	String	イベント名 “selectedindexchange”固定

名前	データ型	説明
index	Number	選択されたインデックス

使用イベントリスナ

- onSelectedIndexChange (ViewStack、TabPanel)

A.2.16 TreeDataChangeEvent

TreeDataChangeEventは、TreeViewのdataプロパティの変更によって発生するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  nodePath: ノードのパス名,
  propertyName: 変更されたプロパティ名
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(TreeView)
type	String	イベント名
nodePath	String	ノードのパス名 イベントが発生しているノードまでのパスが設定されます。ノードとノードの区切り記号は、/です。 例:/root/子ノード名
propertyName	String	値が変更されたプロパティ名 全体が変更された場合は、nullが設定されます。

使用イベントリスナ

- onDataChange (TreeView)

A.2.17 TreeNodeEvent

TreeNodeEventは、TreeViewで、ノードが操作(展開、折りたたみ、選択)されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  nodePath: ノードのパス名,
  browserEvent: ブラウザから送出されたイベント
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(TreeView)
type	String	イベント名

名前	データ型	説明
nodePath	String	ノードのパス名 展開、折りたたみ、選択されているノードまでのパスが設定されます。ノードとノードの区切り記号は、/です。 例: /root/子ノード名
browserEvent	Object	ブラウザから送出されたネイティブイベント 詳細は、“A.1.3 BrowserEvent”を参照してください。 以下のイベントリスナでは、nullが設定されます。 ・ onTreeNodeExpand ・ onTreeNodeCollapse ・ onTreeNodeSelect

使用イベントリスナ

- onTreeNodeExpand (TreeView)
- onTreeNodeCollapse (TreeView)
- onTreeNodeSelect (TreeView)

A.2.18 SelectFilterEvent

SelectFilterEventは、DataGridのフィルタ機能により表示状態が変更することで発生するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  column: 絞り込みを行った列の列名,
  keyword: 絞り込みキーワード
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(DataGrid)
type	String	イベント名
column	String	絞り込みを行った列名
keyword	String	絞り込みを行ったときの選択値、または手入力された文字列

使用イベントリスナ

- onSelectFilter (DataGrid)

A.2.19 SelectedPulldownChangeEvent

SelectedPulldownChangeEventは、DataGridの選択されているプルダウンの項目が変更されたときに呼ばれるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
```

```
  type: 'selectedpulldownchange',
  row: プルダウンを変更した行のインデックス番号,
  column: プルダウンを変更した列の列名,
  index: 変更後の選択項目のインデックス番号,
  item: 変更後の選択項目の項目内容
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(DataGrid)
type	String	イベント名 “selectedpulldownchange”固定
row	Number	プルダウンを変更した行のインデックス番号
column	String	プルダウンを変更した列の列名
index	Number	変更後の選択項目のインデックス番号
item	String	変更後の選択項目の項目内容

使用イベントリスナ

- onSelectedPulldownChange (DataGrid)

A.2.20 CheckItemChangeEvent

CheckItemChangeEventは、DataGridのチェックボックスの値が変更されたときに呼ばれるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  row: 行インデックス番号
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(DataGrid)
type	String	イベント名 以下のどれかが設定されます。 ・ setcheckbox: チェックボックスがONになった ・ resetcheckbox: チェックボックスがOFFになった ・ setallcheckbox: すべての行のチェックボックスがONになった ・ resetallcheckbox: すべての行のチェックボックスがOFFになった
row	Number	イベントリスナによって以下のどちらかが設定されます。 ・ onSetCheckBox、onResetCheckBoxイベントの場合 チェックした行のインデックス番号 ・ onSetAllCheckBox、onResetAllCheckBoxイベントの場合 -1

データオブジェクトの先頭からのインデックス番号をrowプロパティに設定します。

ソートした場合は、ソートにおいて振り直されたインデックス番号がrowプロパティに設定されます。

使用イベントリスナ

- onSetCheckBox (DataGrid)
- onResetCheckBox (DataGrid)
- onSetAllCheckBox (DataGrid)
- onResetAllCheckBox (DataGrid)
- onSetAllDisplayCheckBox (DataGrid)
- onResetAllDisplayCheckBox (DataGrid)

A.2.21 DetailItemEvent

DetailItemEventは、DataGrid部品のツリーで展開された詳細データの行が操作されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  detailRow: クリックした詳細データの行のインデックス番号,
  detailColumn: クリックした詳細データの列の列名,
  row: 行インデックス番号
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(DataGrid)
type	String	イベント名
detailRow	Number	クリックした詳細データの行のインデックス番号
detailColumn	String	クリックした詳細データの列の列名
row	Number	行インデックス番号

使用イベントリスナ

- onClickDetailLink (DataGrid)
- onClickDetailCell (DataGrid)

A.2.22 ContextMenuEvent

ContextMenuEventは、ContextMenuでメニュー項目が操作(表示、非表示、選択)されたときに使用されるイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  nodePath: メニュー項目のパス名,
  browserEvent: ブラウザから送出されたイベント
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(ContextMenu)
type	String	イベント名
nodePath	String	メニュー項目のパス名 表示、非表示、選択されているメニュー項目までのパスが設定されます。メニュー項目とメニュー項目の区切り記号は、/です。 例: /root/子階層のメニュー項目名
browserEvent	Object	ブラウザから送出されたネイティブイベント 詳細は、“A.1.3 BrowserEvent”を参照してください。 以下のイベントリスナでは、nullが設定されます。 ・ onContextMenuShow ・ onContextMenuHide ・ onContextMenuSelect ・ 指定イベントタイプのイベントリスナ

使用イベントリスナ

- onContextMenuShow (ContextMenu)
- onContextMenuHide (ContextMenu)
- onContextMenuSelect (ContextMenu)
- 指定イベントタイプのイベントリスナ (ContextMenu) (注)

注) メニュー項目を選択した場合に送出されるイベントタイプ名は、“ContextMenuのID”＋“_”＋eventType の形式となります。

このイベントタイプ名に対応するイベントリスナをイベントマップに定義し、rcf.event.EventRegistrar.registerEventsで登録することにより、個々のメニュー項目に対応したユーザ独自のイベントリスナを呼び出すことができます。

A.2.23 MenuDataChangeEvent

MenuChangeEventは、ContextMenuのdataプロパティの変更によって発生するイベントオブジェクトです。

記述形式

```
{
  target: イベントを送出したオブジェクト,
  type: 'イベント名',
  nodePath: メニュー項目のパス名,
  propertyName: 変更されたプロパティ名
}
```

モデルプロパティ

名前	データ型	説明
target	Object	イベントを送出したオブジェクト(ContextMenu)
type	String	イベント名
nodePath	String	メニュー項目のパス名 イベントが発生しているメニュー項目までのパスが設定されます。メニュー項目とメニュー項目の区切り記号は、/です。 例: /root/子階層のメニュー項目名

名前	データ型	説明
propertyName	String	値が変更されたプロパティ名 全体が変更された場合は、nullが設定されます。

使用イベントリスナ

- onDataChange (ContextMenu)

付録B データプロバイダ

ここでは、データプロバイダについて説明します。

B.1　データプロバイダの種類

データプロバイダは、部品のgetDataProviderメソッドにより取得できます。
データプロバイダは、getDataProviderの引数に指定したプロパティのデータ型によって、以下の5つの種類があります。

データ型	データプロバイダ
Boolean、Number、String	PrimitiveDataProvider
Array	ArrayDataProvider
Date	DateDataProvider
Object(Array、Date、null以外)	ObjectDataProvider
Object(null)	NullDataProvider

データプロバイダごとにAPIは異なります。詳細は、“B.2 データプロバイダのJavaScript API”を参照してください。

B.2　データプロバイダのJavaScript API

以下のデータプロバイダのAPIについて説明します。

- PrimitiveDataProvider
- ArrayDataProvider
- DateDataProvider
- ObjectDataProvider
- NullDataProvider

PrimitiveDataProvider

プロパティ	データ型	説明
providerType	String	データプロバイダのタイプ “PrimitiveDataProvider”が設定されています。

■getDataメソッド

メソッド	getData()
説明	値を取得します。 取得した値は参照用です。更新しないでください。

■setDataメソッド

メソッド	setData(value)	
引数	value [Boolean、Number、またはString]	値
説明	値を設定します。 各部品で更新不可となっているプロパティに対して、値を設定することはできません。エラー(RCF11002)になります。	

■getDataTypeメソッド

メソッド	getDataType()
戻り値	Boolean、Number、Stringのどれか
説明	データ型を取得します。

ArrayDataProvider

プロパティ	データ型	説明
providerType	String	データプロバイダのタイプ “ArrayDataProvider”が設定されています。

■getDataメソッド

メソッド	getData()
説明	値を取得します。 取得した値は参照用です。更新しないでください。

■setDataメソッド

メソッド	setData(value)	
引数	value [Array]	値
説明	値を設定します。 各部品で更新不可となっているプロパティに対して、値を設定することはできません。エラー(RCF11002)になります。	

■getDataTypeメソッド

メソッド	getDataType()
戻り値	Array
説明	データ型を取得します。

■getDataProviderメソッド

メソッド	getDataProvider(path)	
引数	path [String]	パス
説明	データプロバイダを取得します。	

■addItemメソッド

メソッド	addItem(item)	
引数	item [任意]	配列要素
説明	配列の最後に要素を追加します。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

■getItemAtメソッド

メソッド	getItemAt(index)

引数	index [Number]	インデックス
説明	配列の要素を取得します。 取得した値は参照用です。更新しないでください。	

■getLengthメソッド

メソッド	getLength()
説明	配列の長さを取得します。

■insertItemAtメソッド

メソッド	insertItemAt(index, item)	
引数	index [Number]	インデックス
	item [任意]	要素
説明	指定した位置に要素を挿入します。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

■removeItemAtメソッド

メソッド	removeItemAt(index)	
引数	index [Number]	インデックス
説明	指定した位置の要素を削除します。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

■replaceItemAtメソッド

メソッド	replaceItemAt(index, item)	
引数	index [Number]	インデックス
	item [任意]	要素
説明	指定した位置の要素を置き換えます。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

■sortItemsメソッド

メソッド	sortItems(comparator)	
引数	comparator [function]	比較関数(省略可)
説明	配列をソートします。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

DateDataProvider

プロパティ	データ型	説明
providerType	String	データプロバイダのタイプ “DateDataProvider”が設定されています。

■getDataメソッド

メソッド	getData()
説明	値を取得します。 取得した値は参照用です。更新しないでください。

■setDataメソッド

メソッド	setData(value)	
引数	value [Date]	値
説明	値を設定します。 各部品で更新不可となっているプロパティに対して、値を設定することはできません。エラー(RCF11002)になります。	

■getDataTypeメソッド

メソッド	getDataType()
戻り値	Date
説明	データ型を取得します。

■getTimeメソッド

メソッド	getTime()
説明	1970年1月1日午前0時からのミリ秒を取得します。

■setTimeメソッド

メソッド	setTime(time)	
引数	time [number]	ミリ秒
説明	1970年1月1日午前0時からのミリ秒を設定します。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

ObjectDataProvider

プロパティ	データ型	説明
providerType	String	データプロバイダのタイプ “ObjectDataProvider”が設定されています。

■getDataメソッド

メソッド	getData()
説明	値を取得します。 取得した値は参照用です。更新しないでください。

■setDataメソッド

メソッド	setData(value)	
引数	value [Object]	値
説明	値を設定します。 各部品で更新不可となっているプロパティに対して、値を設定することはできません。エラー(RCF11002)になります。	

■getDataTypeメソッド

メソッド	getDataType()
戻り値	Object
説明	データ型を取得します。

■getDataProviderメソッド

メソッド	getDataProvider(path)	
引数	path [String]	パス
説明	データプロバイダを取得します。	

■getPropertyメソッド

メソッド	getProperty(name)	
引数	name [String]	プロパティ名
説明	プロパティ値を取得します。 取得した値は参照用です。更新しないでください。	

■setPropertyメソッド

メソッド	setProperty(name, value)	
引数	name [String]	プロパティ名
	value [任意]	値
説明	プロパティ値を設定します。 存在していないプロパティに対して設定することはできません。 各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。	

■addPropertyメソッド

メソッド	addProperty(name, value)	
引数	name [String]	プロパティ名
	value [任意]	値
説明	プロパティ値を追加します。 すでに存在しているプロパティに対して設定する場合は、setPropertyを利用してください。	

	各部品で更新不可および部分更新不可となっているプロパティに対して、値を更新することはできません。エラー(RCF11010)になります。

■getPropertyNameメソッド

メソッド	getPropertyName()
説明	プロパティ名のリスト(配列)を取得します。

NullDataProvider

プロパティ	データ型	説明
providerType	String	データプロバイダのタイプ “NullDataProvider”が設定されています。

■getDataメソッド

メソッド	getData()
説明	値を取得します。 取得した値は参照用です。更新しないでください。

■setDataメソッド

メソッド	setData(value)	
引数	value [Objectまたはは Date]	値
説明	値を設定します。 各部品で更新不可となっているプロパティに対して、値を設定することはできません。エラー(RCF11002)になります。	

■getDataTypeメソッド

メソッド	getDataType()
戻り値	Object
説明	データ型を取得します。

付録C 機能別リファレンス

機能別に使用する部品および参照先を以下に示します。

機能分類	機能概要	参照先	備考
画面の表示と入力	テキストを表示する	2.1.1 Text	
	ボタンを表示する	2.1.5 Button	
	リストボックスを表示する	2.1.7 Select	
	リストボックスを表示する(カスタマイズ表示)	2.1.13 SelectList	
	チェックボックスを表示する	2.1.3 CheckBox	
	チェックボックスをグループ化する	4.3.1 CheckBoxGroup	
	ラジオボタンを表示する	2.1.4 RadioButton	
	ラジオボタンをグループ化する	4.3.2 RadioButtonGroup	
	チェックボタンをリスト表示する	2.1.14 CheckList	
	文字列を入力する(単一行)	2.1.2 TextInput	
	文字列を入力する(複数行)	2.1.6 TextArea	
	文字列を入力する(穴埋め)	2.1.11 MaskedTextInput	IE 6 IE 7 IE 8 Internet Explorerだけ使用可
	コンボボックスから入力する	2.1.8 ComboBox	
	日付と時間を入力する	2.1.9 DateInput	
	日付と時間を入力する(穴埋め)	2.1.12 MaskedDateInput	IE 6 IE 7 IE 8 Internet Explorerだけ使用可
	数字を入力する	2.1.10 NumberInput	
	カレンダを表示する	2.4.1 Calendar	
	ポップアップカレンダを表示する	2.4.2 PopupCalendar	
	ポップアップカレンダ表示用ボタンを表示する	2.4.3 CalendarButton	
	テーブルを表示する	2.3.1 TableView	
	テーブルを表示・編集する	2.3.2 TableEdit	
	高度なテーブルを編集する	2.3.3 DataGrid	IE 6 IE 7 IE 8 Internet Explorerだけ使用可
	テーブルの列を定義する(TableView、TableEdit)	2.3.4 ViewColumn	

機能分類	機能概要	参照先	備考
	テーブルの列を定義する(DataGrid)	2.3.5 ViewColumnGrid	 Internet Explorerだけ使用可
	テーブルの列をグループ化する	2.3.6 ViewColumnGroup	
	テーブルの列にチェックボックスを配置する(DataGrid)	2.3.7 ViewColumnCheck	 Internet Explorerだけ使用可
	テーブルの列にツリー展開用ボタンを配置する(DataGrid)	2.3.8 ViewColumnTree	 Internet Explorerだけ使用可
	テーブルの列にプルダウンを配置する(DataGrid)	2.3.9 ViewColumnSelect	 Internet Explorerだけ使用可
	テーブルの列にイメージを配置する(DataGrid)	2.3.10 ViewColumnImage	 Internet Explorerだけ使用可
	ツリー形式で項目リストを表示する	2.5.1 TreeView	
	スクレイピングしたコンテンツを表示する	2.6.1 ScrapingView	
	マウスの右クリックによりコンテキストメニューを表示する	2.7.1 ContextMenu	
画面入力の支援	フォーカスの順序を指定する	4.2.1 FocusManager	
	入力用文字列リストを表示する	4.1.1 AutoCompleter	
	入力文字を制限する(キー入力、ペースト入力)	4.1.2 Limiter	 Internet

機能分類	機能概要	参照先	備考
			Explorerだけ使用可
	入力文字を操作/制限する(数値入力)	4.1.3 NumeralOnlyLimiter	 Internet Explorerだけ使用可
	入力文字を制限する(文字種)	4.1.4 EnableCharTypeLimiter	 Internet Explorerだけ使用可
	入力データを検証する	4.1.5 ValidationHelper	
画面全体の定義	画面情報を指定する	2.2.1 ViewContainer	
	画面情報を指定する(タイトルとボディあり)	2.2.2 Panel	
	画面情報を指定する(ブラウザ内で移動可能)	2.2.6 Window	
	画面情報を別ファイルで指定する	2.2.5 FragmentContainer	
	切替え対象の画面情報をグループ化する	2.2.3 ViewStack	
	切替え対象の画面情報をグループ化する(タブあり)	2.2.4 TabPanel	
データの操作	データモデルを定義する	3.1.1 Model	
	ツリー部品のデータモデルを定義する	3.1.2 TreeModel	
	メニュー部品のデータモデルを定義する	3.1.3 MenuModel	

付録D 付属機能リファレンス

各部品の付属機能について以下に示します。

○:使用できる

部品名	モデルバインディング	フォーマッタ	入力制限	フォーカス制御	自動脱出	複数選択	データ変換	レンダラ	備考
Text	○	○							
Button	○			○					
Select	○			○		○			
SelectList	○	○		○		○		○	
CheckBox	○			○					
CheckBoxGroup	○								
RadioButton	○			○					
RadioButtonGroup	○								
CheckList	○	○		○		○			
TextInput	○	○	○	○	○				
TextArea	○	○		○					
MaskedTextInput	○	○	○	○	○				 InternetExplorerだけ使用可
ComboBox	○	○	○	○	○				
DateInput	○	○	○	○	○		○		
MaskedDateInput	○	○	○	○	○				 InternetExplorerだけ使用可
NumberInput	○	○	○	○	○		○		
Calendar	○	○		○		○		○	
PopupCalendar	○	○		○				○	
CalendarButton	○			○					
TableView	○			○		○			
TableEdit	○			○		○			
DataGrid	○			○		○		○	 InternetExplorerだけ使用可

部品名	モデルバインディング	フォーマッタ	入力制限	フォーカス制御	自動脱出	複数選択	データ変換	レンダラ	備考
ViewColumn				○				○	
ViewColumnGrid				○				○	IE 6 IE 7 IE 8 InternetExplorerだけ使用可
ViewColumnGroup									
ViewColumnCheck				○				○	IE 6 IE 7 IE 8 InternetExplorerだけ使用可
ViewColumnTree				○				○	IE 6 IE 7 IE 8 InternetExplorerだけ使用可
ViewColumnSelect				○				○	IE 6 IE 7 IE 8 InternetExplorerだけ使用可
ViewColumnImage				○				○	IE 6 IE 7 IE 8 InternetExplorerだけ使用可
TreeView	○			○					
ScrapingView	○			○					
ContextMenu	○								
FocusManager									
AutoCompleter									
Limiter			○						IE 6 IE 7 IE 8 InternetExplorerだけ使用可
NumeralOnlyLimiter			○						IE 6 IE 7

部品名	モデルバインディング	フォーマッタ	入力制限	フォーカス制御	自動脱出	複数選択	データ変換	レンダラ	備考
									IE 8 InternetExplorerだけ使用可
EnableCharTypeLimiter			○						IE 6 IE 7 IE 8 InternetExplorerだけ使用可
ValidationHelper	○								
Model	○								
TreeModel	○								
MenuModel	○								

部品名	画面情報	画面情報の切替え	モデルバインディング	レンダラ
ViewContainer	基本			
Panel	タイトル部・ボディ部			
Window	内部ウィンドウ			
FragmentContainer	別ファイル			
ViewStack		画面の同じ位置で表示を切替え	○	
TabPanel		タブで切替え	○	○

付録E テーブル部品機能比較

テーブル部品(TableView、TableEdit、DataGrid)の機能比較を以下に示します。

○:使用できる

機能	TableView	TableEdit	DataGrid
フォーカス	行単位	セル単位	行またはセル単位
選択	行単位	セルまたは行単位	行またはセル単位
セルの編集		○	○
行のソート	○		○
フィルタ用コンボボックスの配置(行ヘッダ)			○
チェックボックスの配置(セル)			○
プルダウンの配置(セル)			○
イメージの配置(セル)			○
アンカーリンク(セル)			○
表示の切替え(ページ遷移)			○
順次表示(1行ずつ表示)			○
列結合			○
列の固定表示			○
セルの表示カスタマイズ(renderer)	○	○	○
選択セルの表示カスタマイズ(selectedRenderer)			○
選択行の表示カスタマイズ(selectedRenderer)			○
ヘッダやデータの複数行表示	○	○	○
ツリー展開(詳細データの表示)			○

用語集

Ajax(Asynchronous JavaScript + XML)

JavaScriptのXMLHttpRequestオブジェクトを利用して非同期にサーバと通信し、ページ全体を読み込まずにページの必要な部分だけを書き換えるWebアプリケーションの開発技術です。

Ajaxフレームワーク環境定義ファイル

Ajaxフレームワークの通信フレームワークの動作環境を設定するXML形式のファイルです。

Ajaxページエディタ

Ajaxフレームワークアプリケーションで利用する画面をGUIベースで編集するツールです。
EclipseおよびInterstage Studio Java EEワークベンチのプラグインとして動作します。

Apcoordinator

Java(TM) 2 Platform, Enterprise Edition(J2EE)に従ったアプリケーションの構築を支援するアプリケーションフレームワーク製品です。

Apcoordinator連携機能

Webブラウザ上のJavaScriptアプリケーションから、Apcoordinatorを利用する機能です。

Eclipse

オープンソースとして公開されている統合開発環境(IDE)です。
Eclipseのプラグイン機能を使用することで、開発環境を拡張することが可能です。

HTTPリクエスト

WebブラウザからWebサーバに対して処理や動作を要求する送信です。HTTPプロトコルに従って送信されます。

HTTPレスポンス

Webサーバからブラウザに送られるリクエストの処理結果です。

Interstage

富士通の提供するアプリケーションサーバ製品のファミリー名です。

Interstage Application Server

富士通の提供するアプリケーションサーバ製品です。

Interstage Studio

Ajaxフレームワークアプリケーション(Ajaxフレームワークを利用したWebアプリケーション)を開発するときに利用する、コンポーネント指向のJava統合開発環境です。アプリケーションを効率的に開発するための各種デザインツール、開発支援ツール、コンポーネント部品が組み込まれています。

Interstage Studio Java EEワークベンチ

Interstage Studioが提供するワークベンチの1つで、Java EEに準拠したアプリケーションの開発に用います。

Interstage Studioワークベンチ

Interstage Studioが提供するワークベンチの1つで、通常のアプリケーションの開発に用います。

JSON(JavaScript Object Notation)

JavaScriptから派生した、データ交換を行うためのデータ記述形式の1つで、テキストベースのデータフォーマットです。

JSP(JavaServer Pages)

Javaによる動的Webページです。JSPタグやJSPスクリプトレット記述により、サーバでの動作を記述します。

J2EE(Java 2 Enterprise Edition)

Sun Microsystems社のプログラミング言語“Java 2”の機能セットの1つで、企業の業務システムや電子商取引などで使われるサーバに必要な機能をまとめたものです。標準機能セットのJava 2 Standard Edition(J2SE)に、サーバ用のAPIや諸機能を付加したものといえます。

REST(Representational State Transfer)

HTTPプロトコルのGETやPOSTメソッドによるリクエストに対して、XMLで応答を返すようなシステムです。
Webサービスのうち、REST形式で動作するサービスをRESTサービスと呼びます。

RIA(Rich Internet Application)

Javaアプレット、Ajaxなどを用いたユーザインタフェースによって、単純なHTMLで記述されたページに比べて操作性や表現力に優れたWebアプリケーションのことをいいます。

SSL(Secure Sockets Layer)

Netscape Communications社が提唱した、インターネット上で情報をのぞき見されないようにするためのネットワークセキュリティ方式です。Webサーバへの接続時に、まず相手の認証や使用する暗号などを確認し、相互に認証したあとで送受信を行うという2階層になっています。

UI部品

クライアントで使用するAjaxフレームワークの部品群の総称です。UI部品には、以下の3種類があります。

- 画面部品
- 機能部品
- 機能付加部品

Webアプリケーション

HTMLファイル、イメージファイル、サーブレット、JSPファイルなどのWebリソースと、Webアプリケーション環境定義ファイルから構築します。機能を、1つのWebアプリケーションのパッケージ単位として開発できます。

Webサービス

HTTPなどのインターネット関連技術を応用して、Web上で公開されているサービスです。

WYSIWYG(What You See Is What You Get)

出力イメージ(文書や画面)を確認しながら編集できるという概念を表します。

XHTML(Extensible HyperText Markup Language)

HTMLをXMLの文法で定義し直したマークアップ言語です。

XML(Extensible Markup Language)

文書、データの意味や構造を記述するための汎用的なマークアップ言語です。

XSL(Extensible Stylesheet Language)

XMLのスタイルシートを記述する言語です。

XSLT(XML Stylesheet Language Transform)

XMLの構造を別の形式に変形するための変換ルールを記述するものです。

XSLTプロセッサ

XSLTの定義に従って、XML形式のデータをHTML形式やほかのドキュメント形式へと変換するものです。
スクレイピングツールで利用しています。

アクションイベント

画面部品が操作されたときに発生するイベントです。

アクセスログ機能

ブラウザ上のJavaScriptアプリケーションからWebサービスへのアクセスログを収集し、出力する機能です。

アダプタ

様々なWebサービスとのアクセスを制御する部品の総称です。マッシュアップフレームワークでは、RESTサービスやHTMLサービスのためのアダプタを提供しています。

アプリケーションサーバ

アプリケーションの実行環境と運用の管理機能を提供するソフトウェアです。

イベント

画面上の操作に対して発生する通知のことです。ユーザが画面部品に対して操作したときに発生するイベントと、画面そのものに対して操作したときに発生するイベント(グローバルイベント)の2種類があります。

イベントオブジェクト

イベントが発生してリスナが呼ばれた場合、リスナ関数の引数に渡されるものです。主なイベントオブジェクトに、アクションイベントとプロパティチェンジイベントがあります。

イベントハンドリング機能

画面に対するイベントリスナを管理し、イベントの発生時に適切なユーザロジックを呼び出す機能です。クライアントフレームワークの機能の1つです。

イベントリスナ

イベントの受信を行うためのメソッドが定義されたインタフェースです。

イベントログ機能

通信フレームワークのクライアント側のJavaScriptから、サーバ側のログを出力する機能です。

画面部品

入力フィールド、チェックボックス、テーブルなど、画面に表示される機能を提供する部品です。

簡易通信方式

サーブレット連携機能において、アプリケーションを容易に構築するために利便性を重視した通信の方式です。アプリケーションは、JavaScriptアプリケーションとビジネスロジック間の通信にサーブレット連携機能のAPIを使用します。JavaScriptのオブジェクトとビジネスロジックのJavaBeanは、サーブレット連携機能の中で相互変換されます。

機能付加部品

入力フィールドへの入力時に文字列を補完するオートコンプリーション機能、フォーカス制御機能など、画面部品に対して機能を付加する部品です。

機能部品

データモデルを定義するなど、画面内部に機能をもつ部品です。

クライアント通信API

ブラウザ上のJavaScriptアプリケーションから、Webサービスを呼び出す機能です。

クライアントフレームワーク

Webブラウザ上のJavaScriptで動作するAjaxフレームワークの機能です。クライアントフレームワークには、以下の4つの機能があります。

- UI部品(画面部品、機能部品、機能付加部品)
- モデルバインディング機能
- イベントハンドリング機能
- グローバルイベント制御機能

クロスドメイン制約

JavaScriptのXMLHttpRequestオブジェクトを利用してサーバと通信する場合、セキュリティを確保するため、JavaScriptを読み込んだページがあるサーバ以外にはアクセスできません。この制約をクロスドメイン制約と呼びます。

グローバルイベント

ファンクションキーを押した場合など、画面そのものに対して操作したときに発生するイベントのことです。

グローバルイベント制御機能

画面そのものに対して操作したときに発生するイベントを制御する機能です。F1～F12までのファンクションキーに対して、keydown、keypress、およびkeyupのイベントを検出することが可能です。クライアントフレームワークの機能の1つです。

サービス管理機能

ブラウザ上のJavaScriptアプリケーションからアクセスするWebサービスを管理する機能です。

サービス認証機能

Webサービスにアクセスする際に、認証情報を付加する機能です。

サーブレット

Webサーバ上で動作するJavaアプリケーションです。ダイナミックな処理結果を返却するアプリケーションの開発に適しています。

サーブレット連携機能

ブラウザ上のJavaScriptアプリケーションから、サーブレットを利用する機能です。

サロゲートペア

従来のUnicode(UTF-16)では未使用だった0xD800～0xDBFF(1024通り)を上位サロゲート、0xDC00～0xDFFF(1024通り)を下位サロゲートと規定し、上位サロゲート＋下位サロゲートの32ビットで1文字を表現する方法です。

自動脱出機能

最大文字数を超えて文字列が入力されたときに、次の画面部品に自動的にフォーカスが移動する機能です。
自動脱出機能を利用するには、対象部品に最大入力文字数、およびフォーカスの移動先を指定しておく必要があります。

スクレイピング機能

各サービスから取得したデータのうち、実際に必要なデータだけを切り出す機能です。

スクレイピングツール

Ajaxフレームワークアプリケーションで利用するWebアプリケーションをGUIベースでスクレイピングし、XSLファイルを作成するツールです。
EclipseおよびInterstage Studio Java EEワークベンチのプラグインとして動作します。

セキュリティ機能

リクエストの正当性を確認する機能です。

セキュリティロール

Interstage管理コンソールを使用するユーザに割り当てられた権限です。
セキュリティロールによって、ユーザをグループ分けすることができます。

セション

サーバとクライアントとの接続を表します。クライアントごとに1つのセションが作成されます。

遅延読込み

画面の一部分を別に用意しておき、ユーザが画面を操作している間に、裏で画面情報を読み込むことをいいます。

通信フレームワーク

Webブラウザで動作するJavaScriptアプリケーションとJ2EEサーバ間の通信を支援するAjaxフレームワークの機能です。通信フレームワークには、以下の機能があります。

- Apcoordinator連携機能
- サーブレット連携機能
- データ型変換機能
- イベントログ機能

データBean

クライアントとビジネスクラスの間のデータ受渡しを行うJavaBeans形式のクラスです。

データBean共有

AjaxフレームワークアプリケーションとApcoordinatorアプリケーションとで、データBeanを共有するための機能です。データBeanを共有することにより、Apcoordinatorアプリケーションで利用するデータBeanの情報を、Ajaxフレームワークアプリケーションで更新することができます。

データ型変換機能

クライアントのJavaScriptとサーバのJava間での双方向オブジェクト変換機能です。通信フレームワークでは、メソッド実行前後に、データ型変換機能を利用してオブジェクト変換を行います。

データ形式変換機能

アダプタから受け取ったXML情報をクライアントのJavaScriptで使えるように、JSON形式に変換する機能です。

汎用通信方式

サーブレット連携機能において、アプリケーションの自由度を重視した通信の方式です。アプリケーションは、JavaScriptのオブジェクトとビジネスロジックのJavaBeanとの変換にサーブレット連携機能のAPIを使用します。JavaScriptアプリケーションとビジネスロジック間の通信は、WebブラウザとWebサーバの機能を使用して実装します。

ビジネスクラス

ビジネスロジックを記述するJavaのクラスをビジネスクラスと呼びます。

ビジネスメソッド

ビジネスクラス内に記述されたメソッドで、クライアントから送信されたデータを処理します。

ビジネスロジック

アプリケーションが実現する業務処理そのものをビジネスロジックと呼びます。

非同期通信

送信側と受信側とでデータ送出のタイミングを合わせる必要がなく、任意のタイミングでデータを送信する通信方式です。

プロパティチェンジイベント

モデルの値が変更されたときに発生するイベントです。

マッシュアップ

複数の異なるWebサービスやアプリケーションを組み合わせて、1つの新しいサービスを形作ることです。

マッシュアップ定義ファイル

マッシュアップフレームワークの動作環境を設定するXML形式のファイルです。

マッシュアップフレームワーク

WebブラウザとWebサービスの通信を中継するAjaxフレームワークの機能です。複数のWebサービスを組み合わせたWebアプリケーションの作成が可能となります。
マッシュアップフレームワークには、以下の機能があります。

- クライアント通信API
- マッシュアッププロキシ
- アダプタ

マッシュアッププロキシ

クライアント通信APIからの呼出しに応じて、様々なWebサービスから情報を取得する機能です。
マッシュアッププロキシでは、Webサービスを呼び出すために以下の機能を提供します。

- サービス管理機能
- サービス認証機能
- データ形式変換機能
- セキュリティ機能
- アクセスログ機能

モデルオブジェクト

ユーザアプリケーションが利用するデータを定義したものです。モデルオブジェクトはモデル定義部品で定義します。

モデルバインディング機能

画面部品から入力されたデータをモデルオブジェクトに反映したり、モデルオブジェクト内のデータが更新されたときに画面部品に値を反映したりする機能です。クライアントフレームワークの機能の1つです。

ユーザ認証

ユーザを、ユーザID/パスワードや証明書により判定し確認する処理です。

ユーザロジック

サーバとの通信量を最適化するためにJavaScriptで実装したアプリケーションです。

ログ出力レベル

ログのレベルは、ログの詳細度を表します。レベルには、高い順から、TRACE、INFO、WARN、ERRORの4種類があります。特定のレベル以下のログだけを出力するように設定して、不要なログを抑止することができます。

ワークベンチ

EclipseやInterstage Studioが提供する、アプリケーションを開発するための統合開発環境です。プロジェクトの定義から実行まで、すべての開発作業を行うことができます。

索 引