X-point

JavaScript
プログラミングガイド

2026/05/01 版

ATLED

はじめに

◆本書の目的

本書は、X-point ウェブフォーム上で使用可能な API とそれらを使用した実装例を元に、X-point での JavaScript 利用方法について説明しています。本書の内容をよくお読み頂いた上で、JavaScript の実装を行なってください。

◆対象とする読者

本書は JavaScript の知識をお持ちの「X-point」のフォーム管理者を対象としています。フォーム管理者とは、eFormMaker でのフォームの作成や、フォーム上で使用されるコンポーネントなどの作成・維持管理を行う際に必要な本システムの管理権限を持つユーザを指します。

◆対応バージョン（2026/05/01 時点）

eFormMaker	X-point
eFormMaker v3.13	X-point v3.13

◆免責事項

実装した JavaScript によって発生したトラブルにつきましては弊社では責任を負いかねます。
ご利用はお客様の責任で行っていただきますようお願いいたします。

◆製品名について

本書中、「X-point サーバ」は「X-point」と表記しています。
また、各製品の名称は対応バージョンを省略してある箇所もありますのでご了承ください。

◆商標について

X-point、エクスポイントの名称およびロゴは株式会社エイトレッドの商標または登録商標です。
Microsoft、MS-DOS、Windows は米国 Microsoft Corporation の米国およびその他の国における登録商標です。Macintosh、MacOS は Apple Computer, Inc.の米国およびその他の国における登録商標です。Adobe、Acrobat、Adobe Acrobat は Adobe Systems, Inc.の商標または登録商標です。ORACLE、Java、JavaScript は、Oracle Corporation 及びその子会社、関連会社の米国及びその他の国における登録商標です。デスクネッツ、desknet's は株式会社ネオジャパンの登録商標です。サイボウズ、Cybozu はサイボウズ株式会社の登録商標です。

その他、記載された会社名およびロゴ、製品名などは該当する会社の商標または登録商標です。本書では、©、®、(TM) の表示を省略しています。ご了承ください。

◆製作著作

目次／索引

1. eFormMakerでのJavaScript実装方法

この章では「eFormMaker」上で JavaScript が実装可能な箇所について説明します。

1.1. フィールドのイベントハンドラに実装

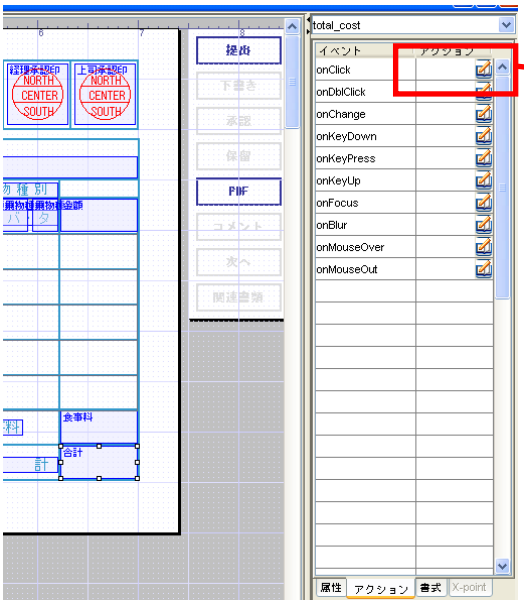
eFormMaker で定義したフィールドのイベントハンドラに JavaScript を実装することが出来ます。
実装可能なイベントハンドラは以下の通りです。

▼ フィールドタイプ別イベントハンドラー覧

	ラベル	文字	数値	整数	テキストエリア	パスワード	ボタン	ラジオボタン	チェックボックス	コンボボックス	リストボックス	西暦	月	日	曜日	日時	印影	バーコード	イメージ
onClick クリックした時	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×	○
onDblClick ダブルクリックした時	○	○	○	○	○	○	×	○	○	×	×	○	○	○	×	○	×	×	○
onChange 値が変更され、フォーカスを失う時	×	○	○	○	○	○	×	○	○	○	○	○	○	○	○	○	×	×	×
onKeyDown いずれかのキーが押された時	×	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×	×
onKeyPress いずれかのキーが押されて放された時	×	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×	×
onKeyUp いずれかのキーが放された時	×	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	×	×	×
onFocus フォーカスが当たった時	×	○	○	○	○	○	×	○	○	×	×	○	○	○	×	○	×	×	×
onBlur フォーカスが外れた時	×	○	○	○	○	○	×	○	○	×	×	○	○	○	×	○	×	×	×
onMouseOver マウスカーソルが重なった時	○	○	○	○	○	○	×	○	○	×	×	○	○	○	×	○	×	×	○
onMouseOut マウスカーソルが重なった後に外れた時	○	○	○	○	○	○	×	○	○	×	×	○	○	○	×	○	×	×	○
onCalcValidator フィールドに設定された計算式が実行される直前	×	×	○	○	×	×	×	×	×	×	×	○	○	○	×	○	×	×	×

JavaScript の実装はフィールドのプロパティ画面より行います。フィールドを選択状態にし、プロパティ画面のアクションタブからイベントハンドラを選択すると [JavaScript 入力画面]が表示されますので、ここに JavaScript を記述します。

▼ プロパティ画面



▼ JavaScript 入力画面

フィールドに指定するイベントハンドラは X-point 上においては、書類編集が可能な状態であるか否かにより動作する、しないに違いがあります。

イベント種類	フォームの状態（isEditMode の結果）	
	編集可（true）	編集不可（false）
onClick	○	×
onDblClick	○	×
onChange	○	×
onKeyDown	○	×
onKeyPress	○	×
onKeyUp	○	×
onFocus	○	○
onBlur	○	○
onMouseOver	○	○
onMouseOut	○	○

※ ○　・・・　使用可、　　×　・・・　使用不可

マウスやフォーカスの移動により発生するイベントハンドラは入力フォームに変化をつけるようなことができるように編集不可の場合にも JavaScript が実行されます。
その為、項目の値や状態を変更する事ができてしまうため、実行できることが不都合である場合があります。その様な場合には isEditMode () を利用して対処します。

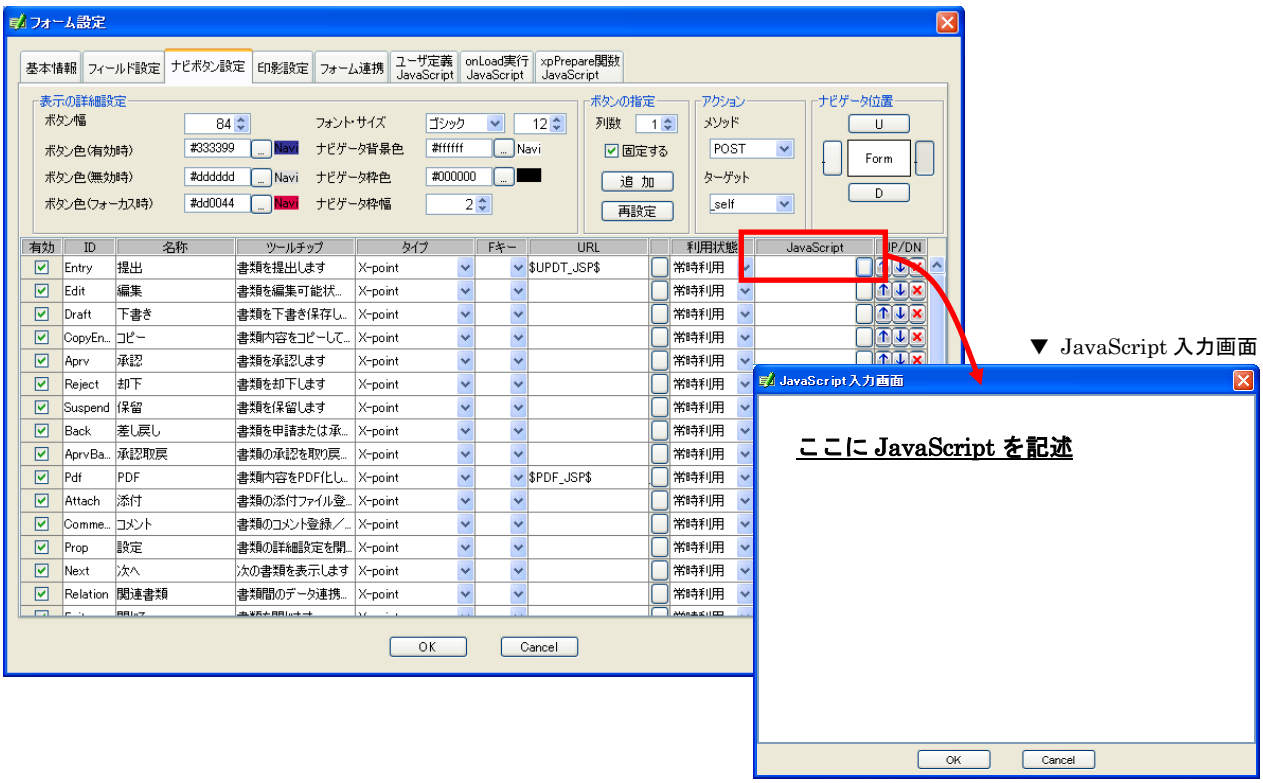
【編集状態を考慮した記述方法】

```
if (isEditMode()) {

  a = a + 1;   // 編集可能なとき
} else {

  a = a + 2;   // 編集不可のとき
}
```

1.2. ナビボタンに実装

既存のナビボタン、又は新たに追加したナビボタンに対して JavaScript を実装することが出来ます。
実装は[フォーム設定]-[ナビボタン設定]タブより行います。

▼ ナビボタン設定タブ

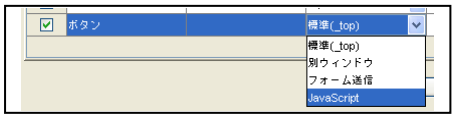


▼ JavaScript 入力画面

! 注意事項

既存のナビボタンに対して実装した処理は、ナビボタン毎に決められた処理（提出ボタンであれば提出処理）の前に実行されます。
また、新たに追加したナビボタンに対して実装する場合は、タイプに「JavaScript」を選択しておく必要があります。

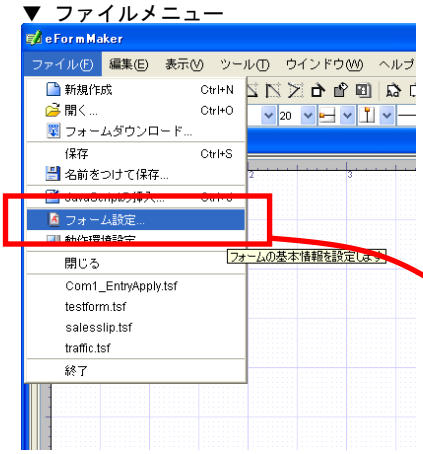
▼ ナビボタンのタイプを選択



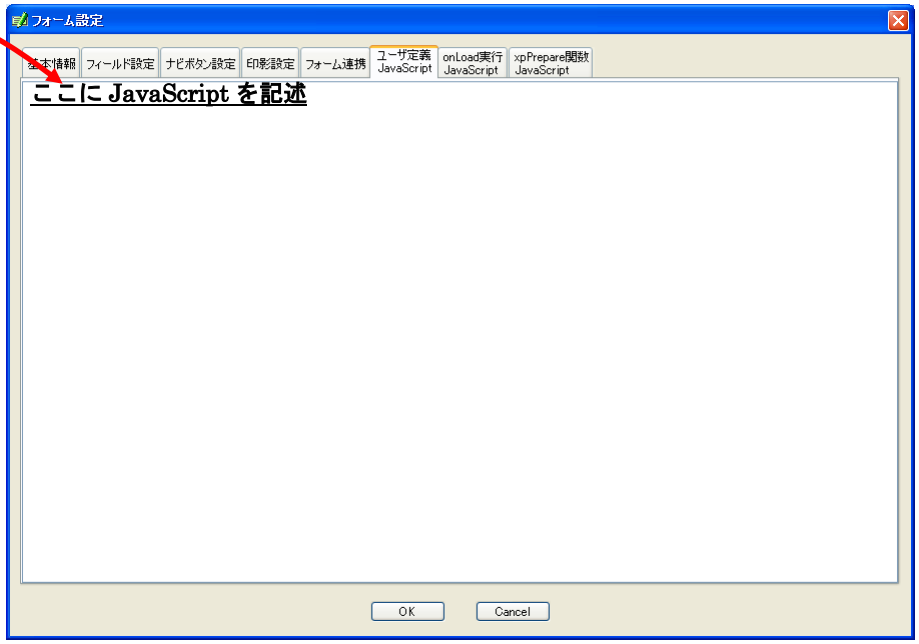
1.3. 共通処理を実装

フォーム内で共通で使用したい function や変数を記述する場合は、[フォーム設定]-[ユーザ定義 JavaScript]タブに記述します。
ここで記述された function や変数をフィールドのイベントハンドラから呼び出すことが出来ます。

▼ ファイルメニュー

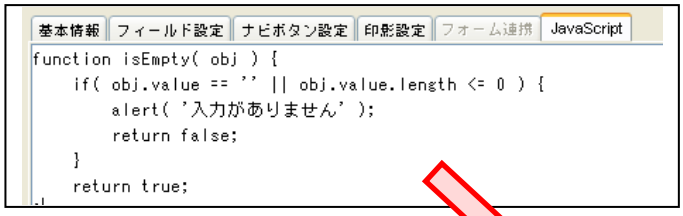


▼ JavaScript タブ

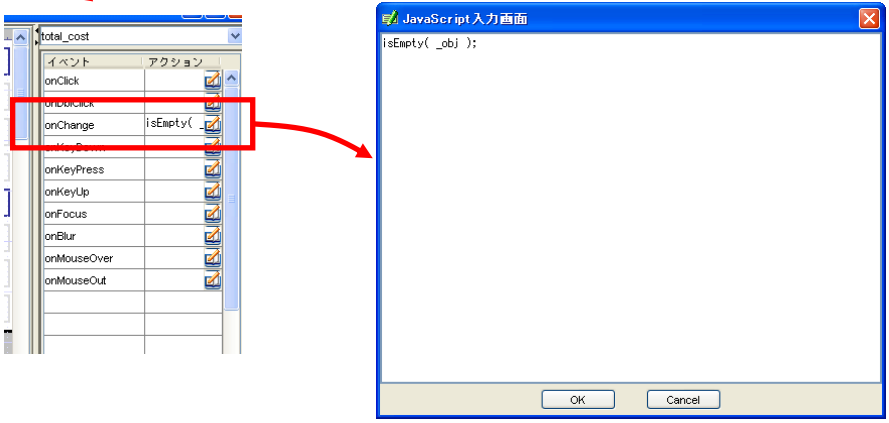


例えば、入力チェックを行う function をこの JavaScript タブで記述しておき、フィールドイベント内の入力チェックで利用することが可能です。

▼ JavaScript タブで記述しておき・・・



▼ フィールドイベント内で利用する



! 注意事項

フォーム内に記述されるに JavaScript の関数名は、一意である必要があります。
「複数枚フォーム」のような複数のフォームにまたがって書かれる JavaScript の関数名も同様に一意である必要があります。
複数枚フォームで利用されるフォーム内に同一名称の関数名がある場合、あとのページの JavaScript 関数が有効になります。

1.4. 動作確認方法

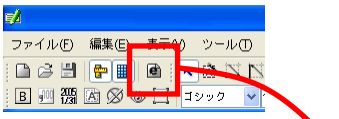
JavaScript の実装内容によって動作確認の方法が異なります。動作確認が可能な箇所は大きく分けて以下の通りです。

1） eFormMaker プレビューで確認

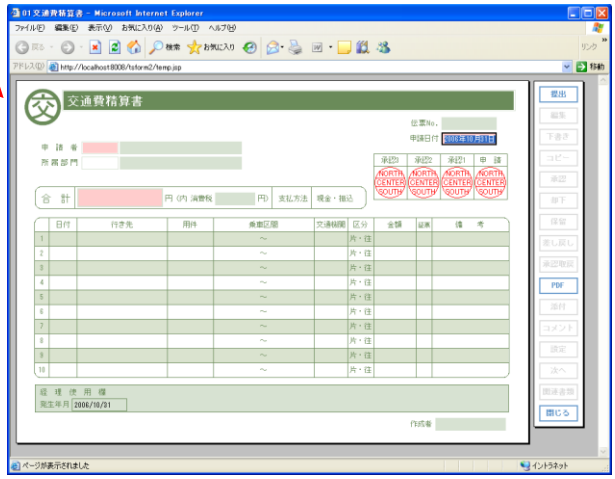
フィールドのイベントハンドラに対する実装であれば、eFormMaker プレビューから動作を確認します。eFormMaker のプレビューアイコンからプレビュー画面を表示して動作確認を行ってください。

！注意事項

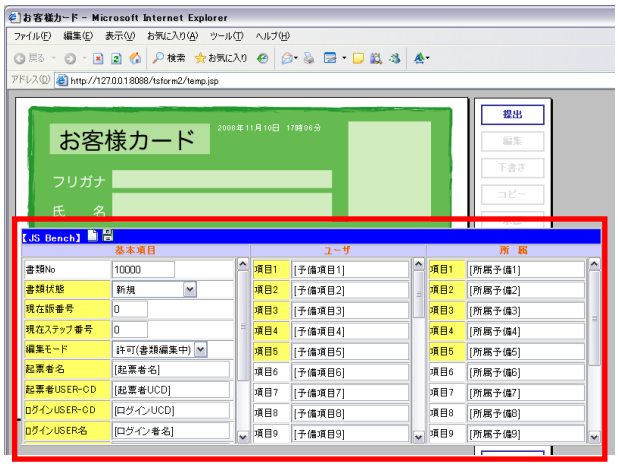
eFormMaker プレビューで一次確認を行うことはできますが、実際に運用する前に必ず、**X-point 上で実際にワークフローを回して動作の確認**を行うようにしてください。



▼ プレビュー画面を表示し、実際に値を入力して確認



eFormMaker プレビューで X-point の API やログイン情報を利用した動作の確認を行う場合は、eFormMaker の動作環境の設定で“JS Bench”を有効にします。



“JS Bench”機能を有効にするとプレビューの際、画面上に W/F の状態を仮設定するためのパネルがフォーム上に表示されるようになります。パネル上で指定した項目は、ワークフロー関連の API からの戻り値として返されるようになります。

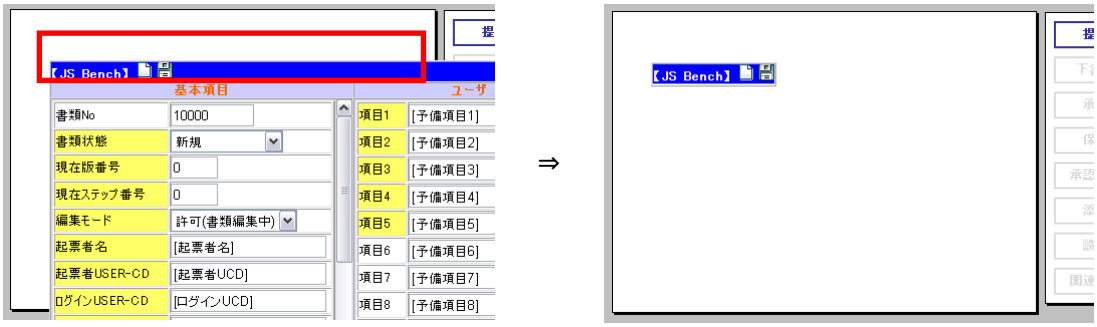
パネル画面は青のタイトルバー部でマウス左ボタンを押した状態で移動させることができます。

【その他の操作】

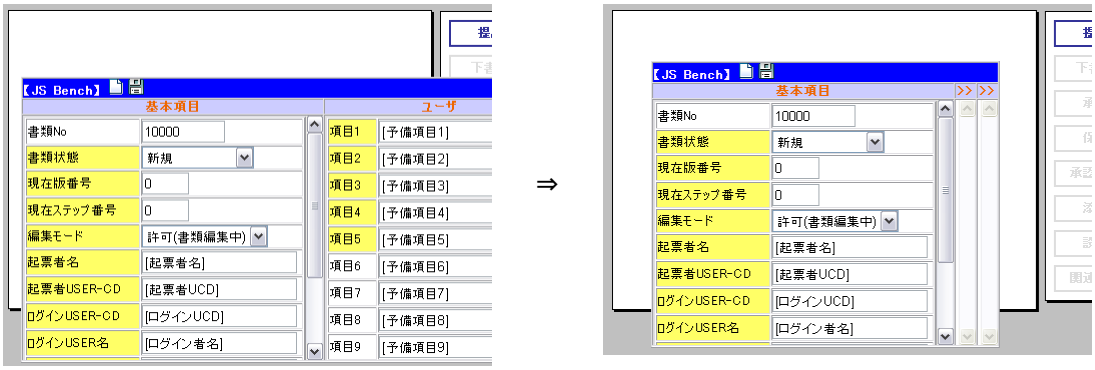
“JS Bench”で指定するパラメータのうち背景が黄色で表示されている項目は、保存アイコン（  ）をクリックするとブラウザ Cookie に保存することができます。
黄色い項目以外には“JS Bench”の表示座標、表示状態が保存されます。

各項目の名称欄上でマウスカーソルを止めると、API 名が表示されます。

青のタイトルバー（【JS Bench】と表示）上でダブルクリックするとパネルを最小化することができます。



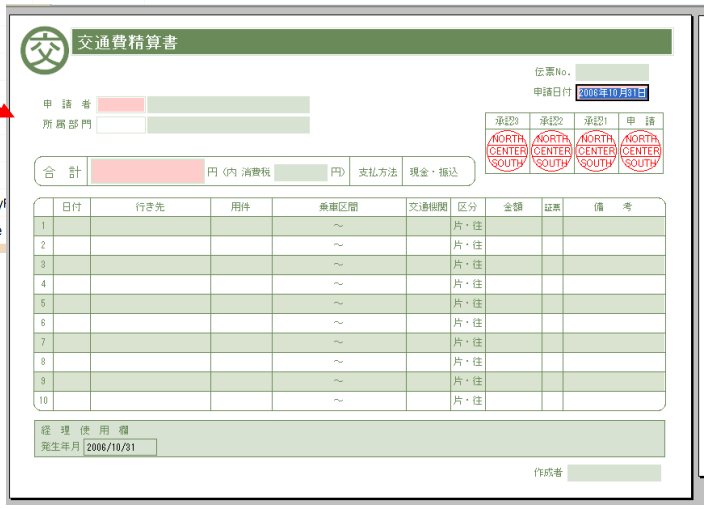
“ユーザ”、“所　属”の文字上でマウスをダブルクリックすると項目を折りたたむことができます。



2） X-point プレビューで確認

後述する複数枚フォーム向けの API やログイン情報を取得する API を利用した実装であれば、X-point プレビューから動作を確認します。管理者サイトのフォーム管理画面よりフォームを登録または修正し、一覧からプレビュー画面を表示して動作確認を行ってください。

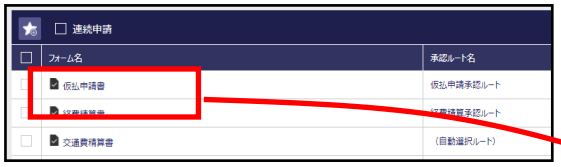
▼ プレビュー画面を表示し、実際に値を入力して確認



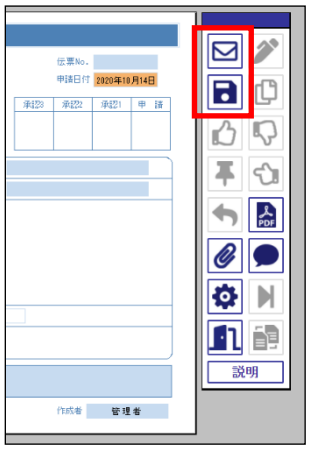
3） 実際にワークフローを回して確認

ナビボタンに対する実装であれば、実際にワークフローを回して各ナビボタンの動作を確認します。
ワークフローを回す際の設定手順は、X-point 管理者機能マニュアル「8.1.3 フォームの公開手順」を参照してください。

▼ フォームライブラリより新規提出状態で開く

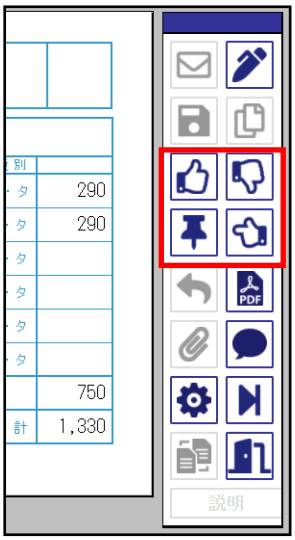


▼ 提出・下書きボタン等の動作確認



▼ 「承認」画面より承認状態で開く

▼ 承認・差し戻しボタン等の動作確認



! 注意事項

本番稼動後のメンテナンスを考慮して、本番環境用ドメインとは別に動作確認用ドメインを作成しておくことを推奨いたします。
動作確認用ドメインで実装に問題がないことを確認したうえで、本番環境用ドメインに反映するといった運用を行ってください。

2. X-point JavaScript API

この章では X-point で用意されている JavaScript API について説明します。
eFormMaker での JavaScript 実装の際には、これらの API を利用して頂くことになります。

2.1. フィールドに対するアクセス

フォームに定義したフィールドに対してアクセスする場合は以下の function を用います。

▼ フィールドアクセス function 一覧

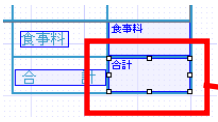
戻り値	function 名	eFormMaker プレビュー	X-point プレビュー	ワークフロー
該当オブジェクト	getElementById_XP(フィールド ID) フィールドオブジェクトを取得する	○	○	○
該当オブジェクト	getElementById_XP(フィールド ID,ページ番号) フィールドオブジェクトを取得（複数枚フォーム用）	○ (※)	○	○
該当オブジェクトの value	getFieldValue_XP(フィールド ID) 該当フィールドの値（value）を返す。指定されたフィールド ID が存在しない場合は警告メッセージを表示する。	○	○	○
該当オブジェクトの value	getFieldValue_XP(フィールド ID, ページ番号) 該当フィールドの値（value）を返す。（複数枚フォーム用） 指定されたフィールド ID が存在しない場合は警告メッセージを表示する。	○ (※)	○	○
（戻り値なし）	setFieldValue_XP(フィールド ID, 値) 該当フィールドに値をセットする。指定されたフィールド ID が存在しない場合は警告メッセージを表示する。	○	○	○
（戻り値なし）	setFieldValue_XP(フィールド ID, 値, ページ番号) 該当フィールドに値をセットする。指定されたフィールド ID が存在しない場合は警告メッセージを表示する。	○ (※)	○	○
該当グループの行数	getGroupTableRows(グループ ID) 指定した表定義グループの繰り返し行数を返す。	○	○	○
該当グループの行数	getGroupTableRows(グループ ID, ページ番号) 指定した表定義グループの繰り返し行数を返す。（複数枚フォーム用）	○ (※)	○	○

！注意事項

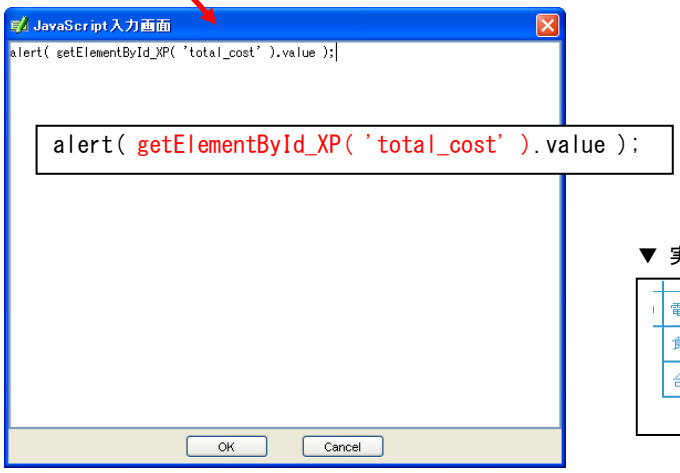
※ eFormMaker でプレビューを行った場合はページ番号が無視された動作となります。
※ eFormMaker ではページ数、現在ページ番号は常に 1 を返します。
※ これらの関数を複数枚フォーム／単一フォームの両方で使用する場合、“getFormPageNo” API を使用してページを取得することで記述を統一化できます。
例）var fld = getElementById_XP('textfield1', getFormPageNo('form_code'));
※ 数値フィールドと整数フィールドに対して getFieldValue_XP()で値の取得を行った際に、フィールド値が空白だった場合は初期値として「0」が戻り値となります。
※ ” getElementById_XP” の” I” はアルファベット大文字の「I：アイ」を入力します。小文字の「l：エル」や数字の「1」と間違えやすいのでご注意ください。

例えば、フォーム上に定義された合計金額フィールドに入力されている値を取得したい場合は、以下のような実装を行います。

▼ onChange に JavaScript を記述

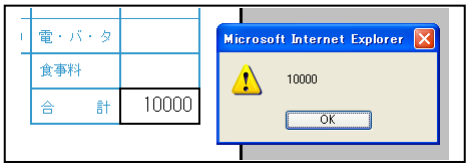


タグ名：　　合計
フィールド ID：total_cost




```
alert( getElementById_XP( 'total_cost' ).value );
```

▼ 実行結果



また、表定義内フィールドに対してアクセスする場合は、以下のようにフィールド ID の末尾に行番号をつけてアクセスします。

▼表定義となっている金額フィールド 2 行目の値をアラート表示する場合

タグ名：金額
フィールド ID：money
行番号：2 行目

```
alert( getElementById_XP( 'money_1' ).value );
```

※フィールド ID の末尾につける行数は、**「1 行目が 0、2 行目が 1、…」**と 0 から始まりますのでご注意下さい。

2.2. ナビボタンに対するアクセス

フォームのナビボタンに対して有効／無効を切り替えたり、ラベル文字を変更するには以下の function を用います。

▼ フィールドアクセス function 一覧

戻り値	function 名	eFormMaker プレビュー	X-point プレビュー	ワークフロー
（戻り値なし）	setNavigatorDisabled(ナビボタンのタイプ, true or false) 指定したタイプのナビボタンを有効・無効を切り替えます。 有効/無効の指定 ：（無効=true 有効=false） ※フォームを開いた時点で有効であるボタンのみ、切り替えが可能です。	△（注※）	○	○
（戻り値なし）	setNavigatorEnabled(ナビボタンのタイプ, true or false) 指定したタイプのナビボタンを有効・無効を切り替えます。 有効/無効の指定 ：（無効=false 有効=true） ※フォームを開いた時点で有効であるボタンのみ、切り替えが可能です。	△（注※）	○	○
（戻り値なし）	setNavigatorLabel(ナビボタンのタイプ, ラベル文字列) 指定したタイプのナビボタンに指定されているラベル（ボタンの見出し）を変更します。 ※ナビボタン名の変更はフォームの再読み込みが行われると元に戻ります。再読み込みは、提出、編集、下書き、コピー、承認、却下、保留など様々な操作で発生します。常に変更したい場合は、フォーム表示のイベントで変更しておきます。	△（注※）	○	○

ナビボタンのタイプは次の表に記載された定数を使用します。

定　義　パ　ラ　メ　ー　タ			
提出	NavigatorTypes.Entry	設定	NavigatorTypes.Prop
PDF	NavigatorTypes.Pdf	コメント	NavigatorTypes.Comment
添付	NavigatorTypes.Attach	下書き	NavigatorTypes.Draft
閉じる	NavigatorTypes.Exit	コピー	NavigatorTypes.Copy
編集	NavigatorTypes.Edit	次へ	NavigatorTypes.Next
承認	NavigatorTypes.Aprv	次ページ	NavigatorTypes.NextPage
却下	NavigatorTypes.Reject	前ページ	NavigatorTypes.PrevPage
保留	NavigatorTypes.Suspend	承認取戻	NavigatorTypes.AprvBack
差し戻し	NavigatorTypes.Back	関連書類	NavigatorTypes.Relation

【記述例】

新規提出時または下書き時に PDF ボタンを無効化するには以下のような記述となります。

```
// 新規提出時または下書き時か
if ( getStatus() == '' || getStatus() == 0 ) {

    // ナビボタン PDF を無効化する
    setNavigatorDisabled( NavigatorTypes.Pdf , true );
}
```

！注意事項

※　eFormMaker プレビューで動作させる場合には、第一引数に「ナビボタンのタイプ」ではなく、「ナビボタンの ID」を指定する必要があります。「ナビボタンの ID」とは、eFormMaker ＞［ファイル］メニュー ＞ フォーム設定 ＞［ナビボタン設定］タブにて指定された ID です。

※　追加したナビボタンに対しても、eFormMaker ＞［ファイル］メニュー ＞ フォーム設定 ＞［ナビボタン設定］タブ から ID を設定することで、上記【記述例】のように、追加分のナビボタンに対しての操作を行うことができます。

2.3. より高度なフィールドへのアクセス

フォームに定義したフィールドを取得する際に“XWEB.getXPObject()”を利用するとより高度な操作を行う事ができます。
“XWEB.getXPObject()”を利用すると、ラジオボタン、チェックボックス、コンボボックス、リストボックスに対して、JavaScriptで無効化(disabled)と背景色の変更(backgroundcolor)を行う場合といった操作が可能になります。

2.3.1. フィールド・オブジェクトの取得

戻り値	function 名	eFormMaker プレビュー	X-point プレビュー	ワークフロー
X-point フィールドオブジェクト	XWEB.getXPObject(フィールド ID [, ページ番号]) ※複数枚フォームで使用する場合、ページ番号は省略できません。 ※複数枚フォーム、単一フォームの両方で使用する場合、 “getFormPageNo”API を使用してページを取得します。	○ (※)	○	○

フィールドの取得

```
var fld=XWEB.getXPObject('textfield1');

// 複数枚フォームの２ページ目で使われている textfield1 を取得
var fld=XWEB.getXPObject('textfield1', 2);

// 複数枚フォームで'form_code'が使われているページの textfield1 を取得
var fld=XWEB.getXPObject('textfield1', getFormPageNo('form_code'));
```

！注意事項

※　eFormMaker でプレビューを行った場合はページ番号が無視（常に 1 が指定）された動作となります。

2.3.2. フィールドタイプの判定オブジェクトの取得

取得したフィールドの種別が何であるかを判定する場合は XWEB.FieldType パラメータ定義を利用します。
独自に作成した JavaScript 関数等で仮引数に設定されているフィールドタイプを判断するような場合に便利です。

```
var fld=XWEB.getXPObject(id);
switch(fld.type){
   case XWEB.FieldType.Text:
      break;
   case XWEB.FieldType.Number:
      break;
}
```

XWEB.FieldType パラメータには次のタイプが登録されています。

定　義　パ　ラ　メ　ー　タ	
XWEB.FieldType.Label	XWEB.FieldType.Stamp
XWEB.FieldType.Text	XWEB.FieldType.Barcode
XWEB.FieldType.Number	XWEB.FieldType.Hidden
XWEB.FieldType.Integer	XWEB.FieldType.DateTime
XWEB.FieldType.TextArea	XWEB.FieldType.Table
XWEB.FieldType.Password	XWEB.FieldType.Hline
XWEB.FieldType.Button	XWEB.FieldType.Vline
XWEB.FieldType.RadioButton	XWEB.FieldType.Line
XWEB.FieldType.CheckBox	XWEB.FieldType.Rect
XWEB.FieldType.ComboBox	XWEB.FieldType.Fill
XWEB.FieldType.List	XWEB.FieldType.Image
XWEB.FieldType.Year	XWEB.FieldType.ImageUpload
XWEB.FieldType.Month	
XWEB.FieldType.Day	
XWEB.FieldType.WeekDay	

2.3.3. フィールド情報の取得と操作

プロパティの属性値を取得します。変更はできません。

プロパティ名	内　　容	備　　考
type	フィールドのタイプ	XWEB.FieldType
id	フィールドの ID 名 表定義内では ID 名に"_行番号"となります。	

フォームの状態に従ってフィールドの情報を取得します。

メソッド名	内　　容	引　　数
getMaxrow	表の定義行数を返す tbl.getMaxrow()	なし
getXPField	表を構成するフィールドを行番号と ID 名の組み合わせで取得する。 tbl.getXPField(rownum,'textfield1')	
getXPItemNames	表を構成するフィールド名を Array 配列で返します。 var items=tbl.getXPItemNames();	
getRoundType	小数点以下の処理方法を返す fld.getRoundType()	なし
getRoundPosition	小数点以下のまるめ桁数を返す fld.getRoundPosition()	なし
getValue	フィールドの値を取得する。 リストボックスの場合は先頭の選択値 fld.getValue() 数値､整数フィールドは Number 型 Object､それ以外は String 型 Object が基本の戻り値になります。 数値フィールドが空欄の場合０が戻されます。	表以外はなし。 表の場合は(行番号,ID 名)
getTextValue	数値フィールド、整数フィールドの設定値を文字列のまま取得する それ以外のフィールドは getValue と同じ値を戻します。	
isVisible	フィールドが可視状態であるか否かを調べる。 fld.isVisible()	なし
isDisabled	フィールドが無効状態であるか調べる fld.isDisabled()	なし
isEnabled	フィールドが有効状態であるか調べる fld.isEnabled()	なし
isTextVertical	ラベルが縦書き指定であるか調べる fld.isTextVertical()	なし
isComma	カンマ区切り指定の有無を返す fld.isComma()	なし
isZeroFill	フィールドが前ゼロ詰めか否かを返す fld.isZeroFill()	なし
isScroll	表定義がスクロール定義か否かを返す fld.isScroll()	なし
isRoundField	数値フィールドに桁数まるめ指定の有無を返す fld.isRoundField()	なし
setFontSize	フィールドの表示フォント・サイズを変更する fld.setFontSize(15)	整数
setFontName	フィールドの表示フォント名を変更する fld.setFontName('MS UI Gothic')	フォント名(文字列)
setFontWeight	文字の太さを設定する fld.setFontWeight('bold') ※ ブラウザ種類により指定通りにならない場合があります。	指定は以下の 13 種類から。 normal,bold,lighter, bolder,100,200,300,400, 500,600,700,800,900
setFontStyle	文字スタイルを変更する fld. setFontStyle('italic') ※ ブラウザ種類により指定通りにならない場合があります。	標準　　：normal イタリック：italic 斜体　　：oblique
setTextDecoration	文字に下線・上線・打ち消し線の付加、点滅等をさせる。 fld.setTextDecoration('underline') ※ ブラウザ種類により指定通りにならない場合があります。	標準　　：none 下線　　：underline 上線　　：overline 打消し線：line-through
setForegroundColor	フィールドの文字色を変更する fld.setForegroundColor('#FF00FF') ※ ボタン、ラジオボタン、チェックボックスには指定できません。	色コード(#000000)文字列
setBackgroundColor	フィールドの背景色を変更する fld.setBackgroundColor('#FF00FF') ※縦書きラベルは背景色が未指定の場合動作しません。	色コード(#000000)文字列

メソッド名	内　　容	引　数
setRectColor	フィールドの枠表示色を変更する fld.setRectColor('#FF00FF')	色コード(#000000)文字列
setColor	図形の色を変更する fld.setColor('#FF00FF')	色コード(#000000)文字列
setFocus	フィールドにフォーカスを設定する fld.setFocus()	なし
setValue	フィールドに値を設定する fld.setValue('**** TEXT *****') ※ラジオボタンで空文字列('')を設定すると、 ボタンが全て選択されていない状態になります。	文字列
setVisible	フィールドの可視・不可視を設定する fld.setVisible(true)	True/false true ： 可視 false ： 不可視
setDisabled	入力フィールドを無効にする fld.setDisabled(true)	True/False true ： 対象フィールドが 無効になる false ： 対象フィールドが 有効になる
setTextVisible	フィールドに指定されている文字列の可視状態を変更する。 fld.setTextVisible(true)	True/false true ： 可視 false ： 不可視

setImeMode	入力フィールドの IME 状態を変更する。
	例： fld.setImeMode(XWEB.IMEType.auto);
	fld.setImeMode(XWEB.IMEType.active);
	fld.setImeMode(XWEB.IMEType.inactive);
	fld.setImeMode(XWEB.IMEType.disabled);
	※ IE のみで動作します。
		XWEB.IMEType
		.auto ： 自動
		.active ： ON にする
		.inactive ： OFF にする
		.disabled ： 無効にする

setReadOnly	入力フィールドを編集不可にする。 fld.setReadOnly(true)	True/False true ： 編集不可 false ： 編集可
setEnabled	入力フィールドを有効にする fld.setEnabled(true)	True/False true ： 有効 false ： 無効
setTooltipText	フィールドにツールチップを設定する fld.setTooltipText('**** Tooltip *****')	文字列
addRoundType	まるめ桁数を指定します。 fld.addRoundType(pos, type) ※　フォームデザイン時にまるめ指定がされている場合に限り 変更指定が可能です。 ※　設定の解除はできません。 ※　数値フィールドにのみ指定できます。	小数点以下桁数を数値で指定します。 まるめタイプは１～３で指定します。 １：四捨五入 ２：切り捨て ３：切り上げ
getTopText	印影の上段に指定されている文字列を取得します。 例： alert('上段='+fld.getTopText()); ※印影の場合にのみ利用できます。	なし
getCenterText	印影の中上段に指定されている文字列を取得します。 例： alert('中段='+fld.getCenterText()); ※印影の場合にのみ利用できます。	なし
getBottomText	印影の下段に指定されている文字列を取得します。 例： alert('下段='+fld.getBottomText()); ※印影の場合にのみ利用できます。	なし
calculate	フィールドに設定されている計算式（合計・四則演算）を実行します。 例： fld.calculate(); ※計算式が設定できる数値・整数・西暦・月・日・日時フィールドのみ利用できます。	なし

execMasterSeek	マスタ参照コンポーネントが設定されているフィールドのマスタ検索を実行します。
	例：表以外のフィールド
	fld.execMasterSeek();
	例：表の１行目のフィールド
	fld.execMasterSeek(0);
	※ マスタ検索の結果を書類に反映するためには、マスタレコードの主キーが一意であることが必要です。
		表以外はなし。
		表の場合は(行番号)

注意事項

※ setValue()で変更できる値を除き、殆どの設定は入力フォーム（HTML）上でのみ動作します。PDF には反映されません。
※ 背景色、線色など一部の項目については PDF への反映が可能な物もありますが、ブラウザ操作の制約上全てを反映することはできません。期待する動作が可能であるかはフォーム作成時にプレビュー等を行いよく確認して下さい。
※ フィールドの種別により設定できないものもあります。
指定できないメソッドを利用すると JavaScript エラーが発生しますので注意して下さい。

【例】ラジオボタン、チェックボックス、コンボボックス、リストボックスに対して、JavaScript で無効化(disabled)と背景色の変更(backgroundcolor)を行う。

【フィールドを無効化する場合】
```
XWEB.getXPObject( フィールド ID ).setDisabled(true);
```

【フィールドの無効を解除する場合】
```
XWEB.getXPObject( フィールド ID ).setDisabled(false);
```

【フィールドの背景色を変更する場合】
```
XWEB.getXPObject( フィールド ID ).setBackgroundColor("#FF0000");
```

※色の指定は RGB の 16 進数で設定します。
※ラジオボタン、チェックボックスについては背景色を設定することができません。
※複数枚フォームに対して使用する場合は、次のようにページ番号を引数に含めます

```
XWEB.getXPObject( フィールド ID, ページ番号 ).setDisabled(true);
XWEB.getXPObject( フィールド ID, ページ番号 ).setDisabled(false);
XWEB.getXPObject( フィールド ID, ページ番号 ).setBackgroundColor("#FF0000");
```

2.4. ドキュメント情報/ログインユーザ情報の取得

書類 No などのドキュメントに関する情報や、ログインユーザの情報を JavaScript で取得することが出来ます。
取得可能な情報は以下の通りです。

▼ ドキュメント/ログインユーザ情報取得 function 一覧

戻り値	function 名	eFormMaker プレビュー	X-point プレビュー	ワークフロー
書類 No	getDocumentId()	△	×	○
書類状況（ステータス） ※各状況の戻り値は以下の様になります。 空白…新規提出時 0…下書き 1…承認中 2…保留中 3…却下 4…申請者差し戻し 5…承認者差し戻し 6…承認完了	getStatus()	△	×	○
現在版番号	getCurrentVersionNo()	△	×	○
現在ステップ番号 ※１	getCurrentStepNo()	△	×	○
編集モードかどうか（true/false）	isEditMode()	△	×	○
起票者名	getApplyUserName()	△	×	○
起票者のユーザコード	getApplyUserCode()	△	×	○
ログインユーザのユーザコード	getLoginUserCode()	△	○	○
ログインユーザの氏名	getLoginUserName()	△	○	○
ログインユーザのカナ	getLoginUserKana()	△	○	○
ログインユーザの E-Mail	getLoginUserEmail()	△	○	○
ログインユーザのユーザ予備 1～30	getLoginUserSpareItem1～30()	△	○	○
ログインユーザのメイン所属グループコード	getLoginUserGroupCode()	△（※３）	○	○
ログインユーザのすべての所属グループコード　※２	getLoginUserGroupCodeAll()	△（※４）	○	○
ログインユーザのメイン所属グループ名	getLoginUserGroupName()	△（※３）	○	○
ログインユーザのすべての所属グループ名 ※２	getLoginUserGroupNameAll()	△（※３）	○	○
ログインユーザのメイン所属グループ予備 1～15	getLoginUserGroupSpareItem1～15()	△	○	○
ログインユーザのメイン所属グループの役職コード	getLoginUserPartCode()	△	○	○
ログインユーザのすべての所属グループの役職コード　※２	getLoginUserPartCodeAll()	△（※４）	○	○
ログインユーザのメイン所属グループの役職名	getLoginUserPartName()	△	○	○
ログインユーザのすべての所属グループの役職名　※２	getLoginUserPartNameAll()	△（※３）	○	○
ログインユーザのログイン ID ※５	getCurrentLoginName()	×	○	○
サーバシステム時刻の Date オブジェクト	XWEB.Date.get()	○	○	○
ログイン中のドメインコード	getLoginDomainCd()	×	○	○
ページ数	getMaxPageNo() フォームを構成しているページ数を返す。（複数枚フォーム用） 複数枚フォームではない場合、必ず 1 を返します。	△	○	○
現在ページ番号	getCurrentPageNo() 現在のページ番号を返す。（複数枚フォーム用） 複数枚フォームではない場合、必ず 1 を返します。	△	○	○
表示フォームのコード	getFormCd() 表示フォームの登録フォームコード、複数枚フォームの場合は複数枚フォームの登録コードを返します。	○	○	○
表示フォームの利用フォームコード	getFormCdList() 表示中のフォームに利用されているフォームコードをページ番号順に配列形式(Array)で返します。	△	○	○

戻り値	function 名	eFormMaker プレビュー	X-point プレビュー	ワーク フロー
指定フォームの利用ページ番号	getFormPageNo(フォームコード) 指定されたフォームコードが入力フォームの何ページ目で利用されているかを返します。(１～)	△ (常に１)	○	○
関連元フォームの書類 ID	getParentDocumentId() 関連元フォームが存在する場合に、書類 ID が取得できます。 元フォームが無い場合は undefined を返します。	△	○	○
なし	moveToPage(ページ番号) 複数枚フォームを表示する際、指定番号のページに表示を切り替えます。	△	△	○
書類に添付されているファイル総数	getDocumentAttachedFileCount()	×	△	○
指定番号の添付ファイル名	getDocumentAttachedFileName(番号) 添付ファイルを０から順に指定します。	×	△	○
指定番号の添付ファイルサイズ（バイト）	getDocumentAttachedFileSize(番号) 添付ファイルを０から順に指定します。	×	△	○
指定番号の添付ファイルタイプ	getDocumentAttachedFileContentType(番号) 添付ファイルを０から順に指定します。 MIME タイプ(例 : "text/plain"、"image/jpeg")を返します。	×	△	○
指定番号の添付ファイルコメント ※５	getDocumentAttachedFileRemark(番号) 添付ファイルを０から順に指定します。	×	△	○
書類のコメント総数	getDocumentCommentCount()	×	△	○
指定番号のコメントに付く重要マーク有無 ※５	getDocumentCommentAttentionFlag(番号) コメントを０から順に指定します。 false:通常、true:重要	×	△	○
指定番号のコメントに付く差し戻しマーク ※５	getDocumentCommentBackFlag(番号) コメントを０から順に指定します。 false:通常、true:差し戻しコメント	×	△	○
指定番号のコメント内容 ※５	getDocumentCommentContent(番号) コメントを０から順に指定します。	×	△	○
指定番号のコメント記入者 ※５	getDocumentCommentWritername(番号) コメントを０から順に指定します。	×	△	○
指定番号のコメント記入日 ※５	getDocumentCommentWriteDate(番号) コメントを０から順に指定します。	×	△	○
関連書類の作成総数	getDocumentRelationDocCount()	×	△	○

！注意事項

ドキュメント/ログインユーザ情報は X-point サーバで管理されているため、eFormMaker プレビューから動作確認をすることは出来ません。

※１ getCurrentStepNo()の戻り値は数値型で返されますが、X-point にて“フォーム互換設定”画面で“文字列”に指定する事が出来ます。互換設定で“文字列”に指定した場合は、JavaScript の動作がアップロード後に変化する場合がありますのでご注意願います。
なお、X-point V1.8.0.xx 以前は文字列型で返されていましたので、V1.8.0.xx 以前の動作を維持したい場合は、“文字列”を指定して下さい。

※２ getLoginUserGroupCodeAll()、getLoginUserGroupNameAll()、getLoginUserPartCodeAll()、getLoginUserPartNameAll()は、Array 配列で返却されます。上記４つの関数では必ず同じ件数が取得され、配列番号が同じものが対応した情報となります(コードに対する名称、など)。
例えば、groupCodes = getLoginUserGroupCodeAll(); のように使用したとき、groupCodes[0]でデフォルトで選択されているグループのコードを取得できます。
さらに、partNames = getLoginUserPartNameAll(); のようにして役職名称の配列を取得したとき、partNames[0]が groupCodes[0]に対応した役職名称となります。
所属グループ数を取得するには、上記の例であれば count = groupCodes.length;で取得することができます。

※３ eFormMaker のプレビュー時にメソッドに応じた固定文字列が返されます。

※４ eFormMaker のプレビュー時には空の配列データ(new Array())が返されます。

※５ 新規書類を開いた直後とそれ以外で、同じ番号を指定しても取得される内容が異なります。
新規書類を開いた直後のタイミングで番号を０から順にコメント情報を取得した場合は、一番新しいコメント情報から取得されます。
それ以外のタイミングの場合は、一番古いコメント情報から取得されます。

2.5. アクションの前後処理

フォームのロードや提出ボタン押下時の前処理など、何らかのアクションの前後に JavaScript で処理を実装することが出来ます。以下の名称の function を[フォーム設定]-[OnLoad 実行 JavaScript](※)、[xpPrepare 関数 JavaScript]タブに関数の処理内容を記述すれば、それぞれのタブに応じたアクション前後で処理が実行されます。

▼ アクション前後処理タブ一覧

戻り値	タブ	詳細タブ	対応アクション	eFormMaker プレビュー	X-point プレビュー	ワークフロー
	OnLoad 実行 JavaScript	-	フォームロード	△	○	○
true/false	xpPrepare 関数 JavaScript	Entry(提出)	提出ボタン押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Edit(標準編集)	通常フォームにて、編集ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	WfEdit(W/F 編集)	ワークフローフォームにて、編集ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Draft(下書き)	下書きボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	CopyEntry(コピー)	コピーボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Aprv(承認)	承認ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Reject(却下)	却下ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Suspend(保留)	保留ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Back(差し戻し)	差し戻しボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	AprvBack(承認取戻)	承認取戻ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Attach(添付)	添付ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Pdf(PDF 出力)	PDF ボタンを押下	△	○	○
true/false	xpPrepare 関数 JavaScript	Comment(コメント)	コメントボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Prop(設定)	設定ボタンを押下	△	×	○
true/false	xpPrepare 関数 JavaScript	Next(次へ)	次へボタンを押下	×	×	○
true/false	xpPrepare 関数 JavaScript	Exit(閉じる)	閉じるボタンを押下	×	×	○
true/false	xpPrepare 関数 JavaScript	Relation(関連書類)	関連書類ボタンを押下	×	×	○
true/false	xpPrepare 関数 JavaScript	NextPage(次ページ)	次ページへボタンを押下	×	×	○
true/false	xpPrepare 関数 JavaScript	PrevPage(前ページ)	前ページへボタンを押下	×	×	○

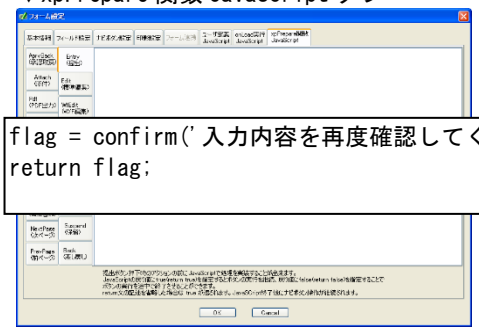
※ 戻り値に false を返すとアクションが途中で終了します
※ eFormMaker でプレビュー実行時に動作確認を行うためには、ナビボタンの ID 名に“xpPrepare”以降の名称が指定されている必要があります。(ワークフローフォームの編集ボタンは WfEdit ではなく Edit を指定します)
※ eFormMaker のナビボタン設定で各ボタンに指定できる JavaScript と X-point 側で作成可能な JavaScript("xpPrepare"で始まる function)の実行順序は変更される場合がありますので、ナビボタンに JavaScript を記述する場合は、どちらか一方のみにしてください。
※ 複数枚フォームで initXpForm や xpPerpare****を利用した場合、構成するフォームに複数の initXpForm、や xpPerpare**** が存在すると、それらのうち一つのみが実行されます。
X-point v1.8.4.01 以降で eFormMaker v1.8.4.01 以降を利用しフォーム設定の「onLoad 実行 JavaScript」、「xpPrePare 関数 JavaScript」に定義する場合は、フォームの 1 ページ目から順に「onLoad 実行 JavaScript」、「xpPrePare 関数 JavaScript」に指定されている JavaScript が実行されます。
※ 単一フォーム、複数枚フォームで共通の JavaScript を記述する場合は“getFormPageNo()”関数を利用して、フォームが何ページ目で利用されているかを常に意識してプログラムを作成して下さい。

!注意事項

ナビボタンのアクションに対応したタブ関数の実装結果は、「1.2. ナビボタンに実装」の方法で実装した場合の結果と同じになります。両方に実装した場合はナビボタンのアクションに対応した function が先に実行されます。

例えば、提出ボタン押下時に確認ダイアログを表示させ入力内容に問題がないか確認させたい場合、[フォーム設定]-[xpPrepare 関数 JavaScript]タブに以下のようなコードを記述します。

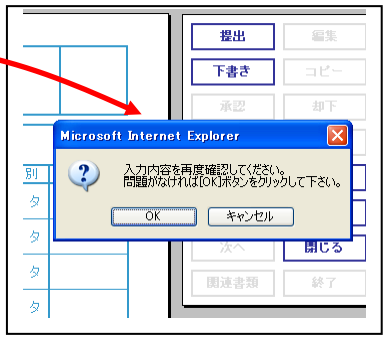
▼xpPrepare 関数 JavaScript タブ



```
flag = confirm('入力内容を再度確認してください。¥n 問題がなければ[OK]ボタンをクリックしてください。');
return flag;
```

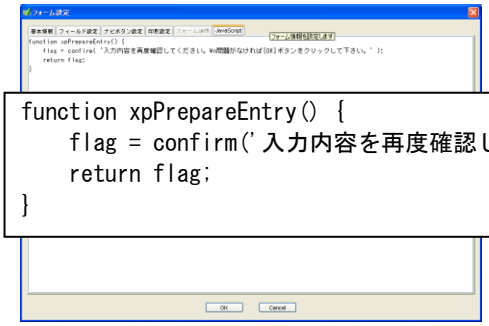
▼ 提出ボタンを押下すると・・・

▼ 確認ダイアログが表示される



X-point v1.8.4.00 以前に作成されたフォームでは、[フォーム設定]-[ユーザ定義 JavaScript]タブに以下のように記述されています。

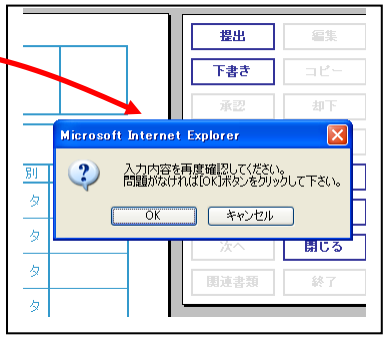
▼ JavaScript タブ



```
function xpPrepareEntry() {
    flag = confirm('入力内容を再度確認してください。¥n 問題がなければ[OK]ボタンをクリックしてください。');
    return flag;
}
```

▼ 提出ボタンを押下すると・・・

▼ 確認ダイアログが表示される



2.6. ナビボタンの処理

既存のナビボタンではなく、独自の処理を行うナビボタンを作成したい場合（独自処理の後に提出処理を行うなど）は、ナビボタンに記述する JavaScript 内で以下の function を利用します。
例えばコピーボタンの機能を加えたオリジナルのコピーボタンを両方用意する場合は、以下の function 一覧にある doCopy()をオリジナルのコピーボタンの処理内に記述します。

▼ ナビボタン処理 function 一覧

戻り値	function 名	対応処理	eFormMaker プレビュー	X-point プレビュー	ワークフロー
（戻り値なし）	doEntry()	提出ボタン押下時の処理	△	×	○
（戻り値なし）	doEdit()	編集ボタン押下時の処理	△	×	○
（戻り値なし）	doDraft()	下書きボタン押下時の処理	△	×	○
（戻り値なし）	doCopy()	コピーボタン押下時の処理	△	×	○
（戻り値なし）	doApprove()	承認ボタン時の処理	△	×	○
（戻り値なし）	doReject()	却下ボタン時の処理	△	×	○
（戻り値なし）	doSuspend()	保留ボタン時の処理	△	×	○
（戻り値なし）	doBack()	差し戻しボタン時の処理	△	×	○
（戻り値なし）	doApproveBack()	承認取戻ボタン時の処理	△	×	○

例えば、提出ボタンを新たに作成し項目が未入力であれば“編集モード”に、記入済であれば“提出”をするボタンを作る場合に利用します。

```
if( …… ) {
    alert( '〇〇欄が記入されていません。' );
    doEdit();
}else{
    doEntry();
}
```

また、これらの doXXX()関数は、ナビボタンの実際の実行可否を判定しません。実行可否の状態は以下の変数から取得することができます。doXXX()関数利用時は、これらの変数の値を確認するようにしてください。

▼ ナビボタン実行可否判定変数一覧

ボタン種別	変数名	値	対応する function
提出ボタン	xpoint.grant.**entry**	実行可能：TRUE、実行不可：FALSE	doEntry()
編集ボタン	xpoint.grant.**edit**	〃	doEdit()
下書きボタン	xpoint.grant.**draft**	〃	doDraft()
コピーボタン	xpoint.grant.**copy**	〃	doCopy()
承認ボタン	xpoint.grant.**approve**	〃	doApprove()
却下ボタン	xpoint.grant.**reject**	〃	doRecject()
保留ボタン	xpoint.grant.**suspend**	〃	doSuspend()
差し戻しボタン	xpoint.grant.**back**	〃	doBack()
承認取戻ボタン	xpoint.grant.**aprvBack**	〃	doApproveBack()

▼ ボタン実行可否判定例

```
if( xpoint.grant.edit ) {
    doEdit();
}else{
    alert('編集権限がありません');
}
```

！注意事項

※ ナビボタン処理 function は内部で Submit 処理や別ウィンドウ表示処理が実行されるため、処理の最後に実行されるように実装してください。
※ eFormMaker ではダミー関数が定義されているのみです。実行しても何も行なわれません。

2.7. イベントハンドラ内での自身参照

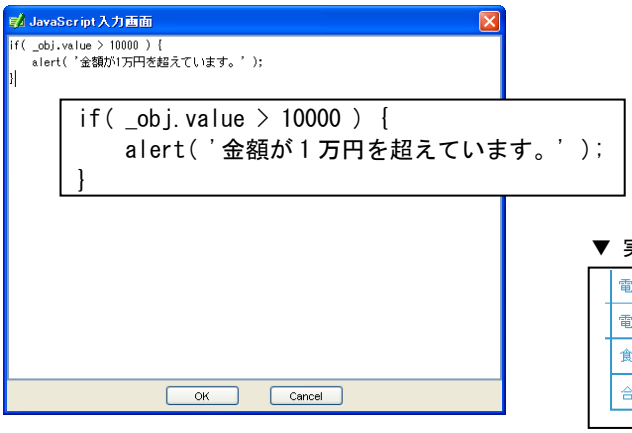
イベントハンドラ内で入力された金額の値をチェックするといった実装をする際に、自身のフィールド（this）を参照する必要があります。参照可能な変数は以下の通りです。

▼ 参照可能変数一覧

変数名	概要
_obj	イベントハンドラが実行されたフィールドのオブジェクト。
_no	表定義の行番号を保持する変数。表定義内のフィールドの場合のみ参照可能。 行数-1 の値が格納されている。(1 行目には 0 がセットされている)
event	イベントハンドラが受け取ったイベントへの参照オブジェクト。 ※マスタ参照、ミラー定義、setValue()関数から onChange イベントハンドラを作動させた場合、event 変数は undefined となります。

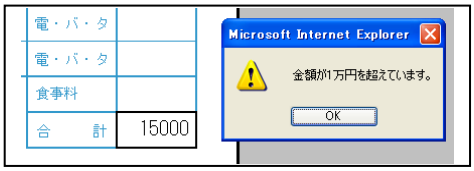
例えば、入力された金額が 1 万円以上かどうかをチェックする JavaScript を実装した場合、以下のように”_obj”変数を参照します。

▼ onChange イベントの JavaScript 入力画面




```
if( _obj.value > 10000 ) {
    alert( '金額が 1 万円を超えています。' );
}
```

▼ 実行結果



！注意事項

_obj の実体はフィールドのタイプによって異なるため、「2.1. フィールドに対するアクセス」の getFieldValue_XP()等の function をご利用頂くことを推奨いたします。

2.8. ユーティリティ API

どの実装箇所でも利用して頂けるユーティリティ API が用意されています。

戻り値	function 名	概要	eFormMaker プレビュー	X-point プレビュー	ワークフロー
文字列	trim(文字列)	前後の空白文字（スペース、復帰文字、タブ、改行文字）を削除する			
文字列	ltrim(文字列)	前方の空白文字（スペース、復帰文字、タブ、改行文字）を削除する			
文字列	rtrim(文字列)	後方の空白文字（スペース、復帰文字、タブ、改行文字）を削除する			
true/false	isWeekDay(日付値)	平日（日曜以外）かどうかを返す。	○	○	○
true/false	isHoliday(日付値)	祝日かどうかを返す ※「国民の祝日に関する法律」に定められた日を祝日と判定します	○	○	○
true/false	isProperDate(日付値)	存在する日付かどうかを返す	○	○	○
文字列	getDaysnote(日付値)	ドメイン管理「カレンダ」に登録されたカレンダで指定日の備考に登録された文字列を取得します	－ 国民の祝日に定められた祝日名のみを返す	○	○
true/false	isBusinessDay(日付値)	ドメイン管理「カレンダ」に登録された内容に基づき営業日(true)、休業日(false)を判定します	－	○	○
数値	XpNetwork.dateif(日付値, 日付値)	ドメイン管理「カレンダ」に登録された内容に基づき指定日間の営業日数を返す	－	○	○
日付値 (yyyy/MM/dd)	XpNetwork.add(日付値, 加算する営業日数)	ドメイン管理「カレンダ」に登録された内容に基づき指定日に営業日数加算した日付を返す	－	○	○
日付値 (yyyy/MM/dd)	XpNetwork.sub(日付値, 減算する営業日数)	ドメイン管理「カレンダ」に登録された内容に基づき指定日に営業日数減算した日付を返す	－	○	○

※ 日付値として指定可能な日付書式は以下の通り

- yyyy 年 MM 月 dd 日
- yyyy/MM/dd
- yyyyMMdd
- yyyy/MM（日は 01 で補完される）
- MM/dd（年は現在年で補完される）
- MMdd（年は現在年で補完される）

※ ドメイン管理「カレンダ」で登録された内容に基づく日付情報取得や計算は、ログイン中ユーザが利用するカレンダに何が割り当てられているかにより結果が異なります。

※ 「XpNetwork.dateif」「XpNetwork.add」「XpNetwork.sub」は先頭部の名称「XpNetwork」が「xpNetwork」「XpNetworkdays」「xpNetworkdays」である場合も同様に動作します。

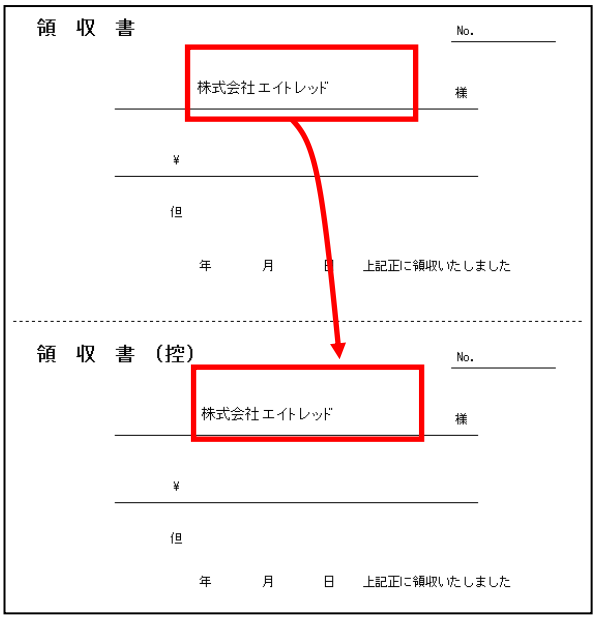
3. サンプル集

この章では、「2. X-point JavaScript API」でご紹介した API を使用したサンプルコードを掲載しています。JavaScript 実装のヒントとしてご活用ください。

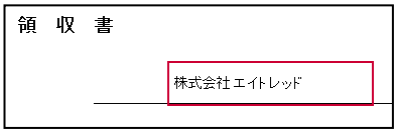
※　掲載のサンプルコードは動作を保障するものではありません
※　実際のコーディングはフィールドの定義内容や eFormMaker、X-point のバージョンによって異なります

3.1. 入力した値を他のフィールドにコピーする

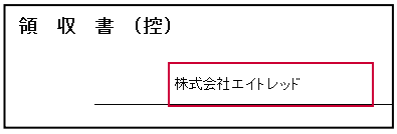
例）領収書に入力した内容を控えにそのままコピーする



▼ フィールド情報



タイプ：文字フィールド
タグ名：会社名
ID：company_name1



タイプ：文字フィールド
タグ名：会社名（控）
ID：company_name2

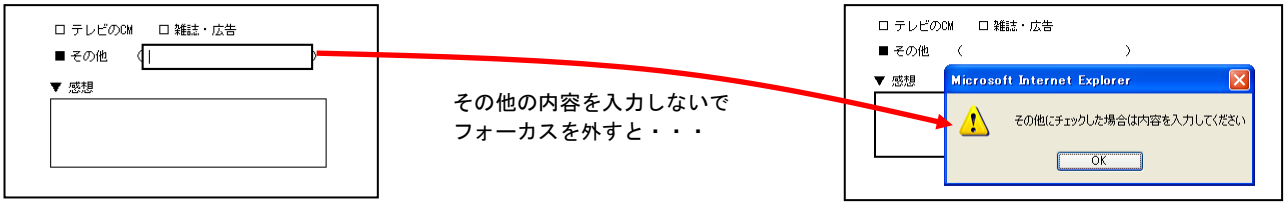
▼ サンプルコード

company_name1 の onChange イベントハンドラに以下のコードを記述

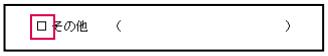
```
setFieldValue_XP('company_name2',_obj.value);　　… company_name2 に company_name1 の入力値をセットする
```

3.2. 条件付きの入力チェック

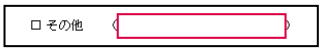
例）アンケート項目で”その他”にチェックした場合は、必ず”その他”の内容を入力させる



▼ フィールド情報



タイプ：チェックボックス
タグ名：その他チェック
ID：others_check
チェック値：1



タイプ：文字フィールド
タグ名：その他の内容
ID：others_content

▼ サンプルコード

others_content の onBlur イベントハンドラに以下のコードを記述

```
if( getFieldValue_XP('others_check') == 1 && _obj.value == '' ) {    … others_check がチェックされていて、others_content に
                                                                        入力がないかどうかの判定
    alert( 'その他にチェックした場合は内容を入力してください' );    … 警告メッセージを表示する
}
```

尚、フィールドの未入力をチェックする際、上の例では「_obj.value == ''」を利用していますが、getFieldValue_XP() や getElementById_XP().value によって入力値を調べることもできます。ただし、フィールドタイプによって以下のように未入力の場合の戻り値が異なりますのでご注意ください。

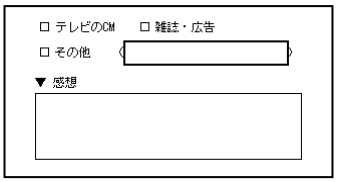
【未入力フィールドの値取得結果】

フィールドタイプ	getFieldValue_XP() の結果	getElementById_XP().value の結果
文字フィールド	'' (空)	'' (空)
数値フィールド	0	'' (空)
整数フィールド	0	'' (空)
テキストエリア	'' (空)	'' (空)
パスワードフィールド	'' (空)	'' (空)
ラジオボタン	'' (空)	'' (空)
チェックボックス	'' (空)	'' (空)
コンボボックス	undefined	'' (空)
リストボックス	undefined	'' (空)
西暦フィールド	'' (空)	'' (空)
月フィールド	'' (空)	'' (空)
日フィールド	'' (空)	'' (空)
曜日フィールド	undefined	'' (空)
日時フィールド	'' (空)	'' (空)

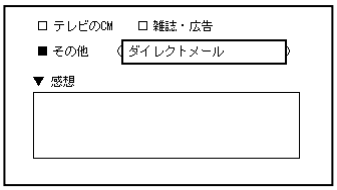
3.3. フィールドの ReadOnly を切り替える

例）アンケート項目で”その他”にチェックした場合にのみ、”その他”の内容を入力可能にする

▼ その他にチェックしていないので入力できない



▼ その他にチェックすると入力可能になる



▼ フィールド情報

※ 「3.2. 条件付きの入力チェック」を参照

▼ サンプルコード

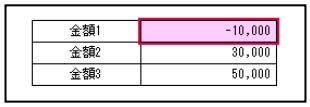
others_check の onChange イベントハンドラに以下のコードを記述

```
var othersContent = XWEB.getXPObject('others_content');      … others_content を取得する

if( getFieldValue_XP('others_check') == 1 ) {                … others_check がチェックされているかの判定
    othersContent.setReadOnly(false);                        … チェックされている場合は ReadOnly を外す
} else {
    othersContent.setReadOnly(true);                         … チェックされていない場合は ReadOnly にする
    othersContent.setValue('');
}
```

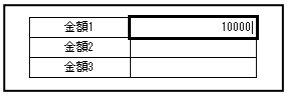
3.4. フィールドの背景色を切り替える

例）入力した金額の正負で背景色を切り替える

▼ 金額が負の場合は背景色をピンクにする



▼ フィールド情報



タイプ：数値フィールド
タグ名：金額 1～3
ID：amount1～3

▼ サンプルコード

amount1～3 の onChange イベントハンドラに以下のコードを記述

```
var fld = XWEB.getXPObject( 'amount1' );                    … フィールドの取得
                                                              ※該当のフィールド ID を指定します（amount1～3）
if( fld.getValue() < 0 ) {                                  … 入力値が負かどうかの判定
    fld.setBackgroundColor('#ffccff');                      … 入力値が負であれば、背景色をピンクに切り替える
} else {
    fld.setBackgroundColor('');
}                                                           … 入力値が正であれば、背景色をクリアする
```

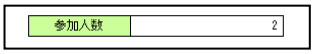
3.5. 入力されているフィールドをカウントする

例）参加者一覧の入力内容から参加人数をカウントする

▼ 参加者の氏名が入力されている数だけ参加人数にセットする

▼ フィールド情報

タイプ：文字フィールド
タグ名：氏名 1～5、住所 1～5
ID：name1～5、address1～5



タイプ：数値フィールド
タグ名：参加人数
ID：numbers

▼ サンプルコード

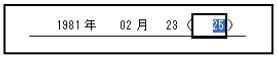
name1～5 の onChange イベントハンドラに以下のコードを記述

```
count = 0;                                                    … 最終的な参加人数の値を初期化する

if( getFieldValue_XP( 'name1' ) != '' ) count++;              … 氏名 1～5 に入力があれば参加人数を+1 する
if( getFieldValue_XP( 'name2' ) != '' ) count++;
if( getFieldValue_XP( 'name3' ) != '' ) count++;
if( getFieldValue_XP( 'name4' ) != '' ) count++;
if( getFieldValue_XP( 'name5' ) != '' ) count++;

setFieldValue_XP( 'numbers',count  );                         … 最終的な参加人数をセットする
```

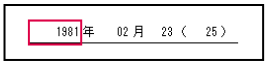
3.6. 生年月日（西暦）から年齢を自動計算

例）生年月日の年月日を入力すると、年齢が自動的に計算される

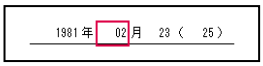
▼年月日それぞれに値を入れると年齢が自動的に計算される



▼ フィールド情報

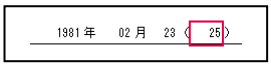


タイプ：西暦フィールド
タグ名：生年月日-年
ID：birthyear



タイプ：月フィールド
タグ名：生年月日-月
ID：birthmonth

タイプ：日付フィールド
タグ名：生年月日-日
ID：birthday



タイプ：整数フィールド
タグ名：年齢
ID：age

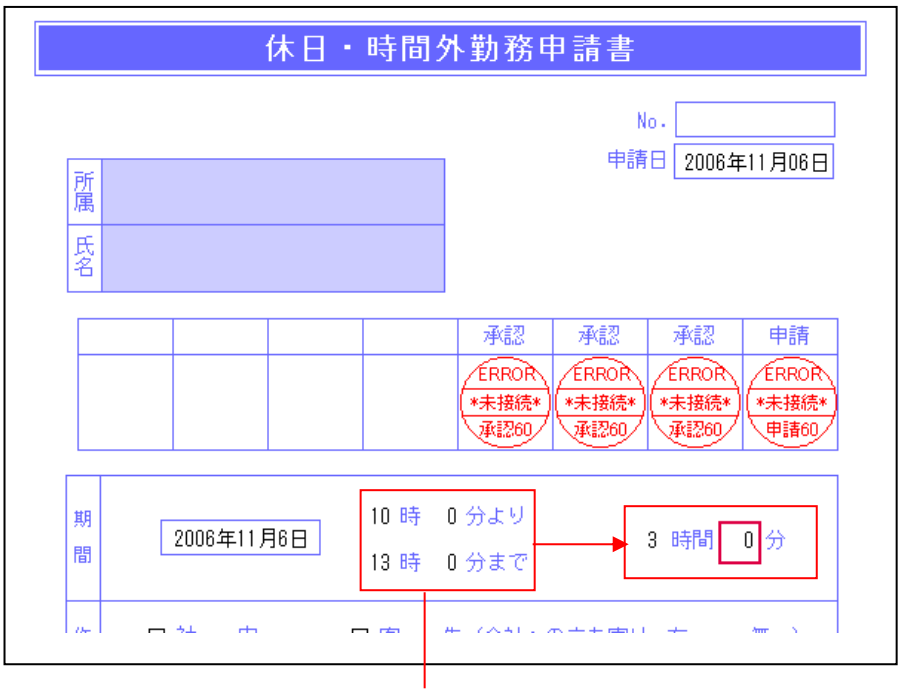
▼ サンプルコード

birthyear、birthmonth、birthday それぞれの onChange イベントハンドラに以下のコードを記述

```
var birthyear  = getFieldValue_XP( 'birthyear'  );   … 生年月日-年を取得する
var birthmonth = getFieldValue_XP( 'birthmonth' );   … 生年月日-月を取得する
var birthday   = getFieldValue_XP( 'birthday'   );   … 生年月日-日を取得する

if( birthyear != '' && birthmonth != '' && birthday != '' ) {   … 年月日が入力されているかの判定
  age      = 0;                                      … 最終的な年齢を初期化
  now      = new Date();                             … 現在日付を取得
  nowyear  = now.getFullYear();                      … 現在年を取得
  nowmonth = now.getMonth() + 1;                     … 現在月を取得（-1 された値が格納されているので+1 して補正）
  nowday   = now.getDate();                          … 現在日を取得

  if( nowmonth < Number( birthmonth ) ) {            … 生年月日-月が現在月以降かの判定
    age = nowyear - Number( birthyear ) - 1;         … 年の差にさらに-1 した値が年齢になる
  } else if( nowmonth > Number( birthmonth ) ) {     … 生年月日-月が現在月以前かの判定
    age = nowyear - Number( birthyear );             … 年の差の値が年齢になる
  } else if( nowmonth == Number( birthmonth ) ) {    … 生年月日-月が現在月かの判定
    if( nowday < Number( birthday ) ) {              … 生年月日-日が現在日以降かの判定
      age = nowyear - Number( birthyear ) - 1;       … 年の差にさらに-1 した値が年齢になる
    } else {
      age = nowyear - Number( birthyear );           … 年の差の値が年齢になる
    }
  }
  setFieldValue_XP( 'age', age );                    … 最終的な年齢をセットする
}
```

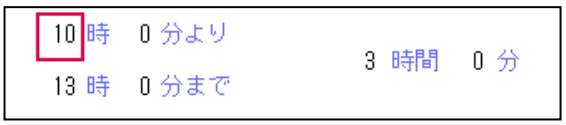
3.7. 開始時間、終了時間から経過時間を自動計算

例）時間外勤務申請書の開始時刻・終了時刻から期間を自動計算する

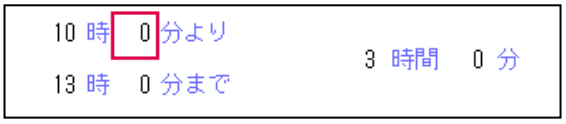


開始時刻、終了時刻が入力されたら自動的に計算

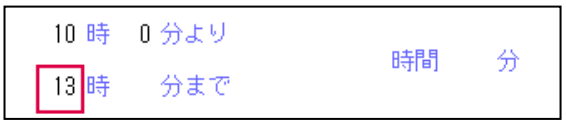
▼ フィールド情報



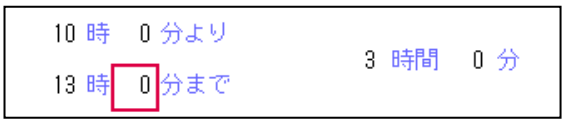
タイプ：整数フィールド
タグ名：期間自時
前ゼロ詰め：false
ID：kikan_from_hh



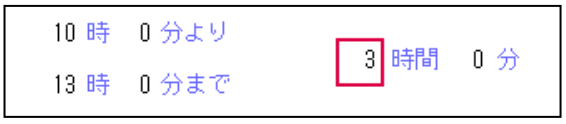
タイプ：整数フィールド
タグ名：期間自分
前ゼロ詰め：false
ID：kikan_from_mm



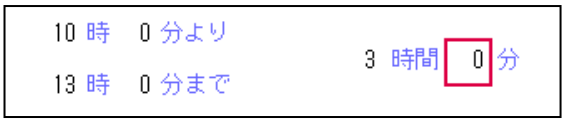
タイプ：整数フィールド
タグ名：期間至時
前ゼロ詰め：false
ID：kikan_to_hh



タイプ：整数フィールド
タグ名：期間至分
前ゼロ詰め：false
ID：kikan_to_mm



タイプ：整数フィールド
タグ名：期間時
前ゼロ詰め：false
ID：kikan_hh



タイプ：整数フィールド
タグ名：期間分
前ゼロ詰め：false
ID：kikan_mm

▼ サンプルコード

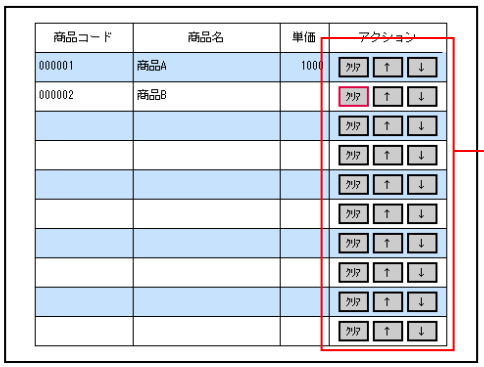
kikan_from_hh、kikan_from_mm、kikan_to_hh、kikan_to_mm の onChange イベントハンドラに以下のコードを記述

```
var kikan_from_hh = getElementById_XP( 'kikan_from_hh' );           … 期間自時を取得する
var kikan_from_mm = getElementById_XP( 'kikan_from_mm' );           … 期間自分を取得する
var kikan_to_hh   = getElementById_XP( 'kikan_to_hh'   );           … 期間至時を取得する
var kikan_to_mm   = getElementById_XP( 'kikan_to_mm'   );           … 期間至分を取得する
var kikan_hh      = getElementById_XP( 'kikan_hh'      );           … 期間時を取得する
var kikan_mm      = getElementById_XP( 'kikan_mm'      );           … 期間分を取得する

kikan_hh.value = '';                                                … 期間時を初期化する
kikan_mm.value = '';                                                … 期間分を初期化する

if( kikan_from_hh.value != null && kikan_from_mm.value != null &&   … 期間自時、期間自分、期間至時、期間至分が全て入力さ
        kikan_to_hh.value != null && kikan_to_mm.value != null ) {      れているか判定
    from_time = ( Number( kikan_from_hh.value ) * 60 ) +            … 開始時刻が分単位になるよう計算する
                Number( kikan_from_mm.value );
    to_time   = ( Number( kikan_to_hh.value   ) * 60 ) +            … 終了時刻が分単位になるよう計算する
                Number( kikan_to_mm.value );

    if( to_time > from_time ) {                                     … 開始時刻<終了時刻の関係かどうか判定
        kikan_time = to_time - from_time;                           … 期間を計算

        kikan_hh.value = Math.floor( kikan_time / 60 );             … 期間から時間を計算して、期間時に値をセット
        kikan_mm.value = kikan_time - ( kikan_hh.value * 60 );      … 期間から分を計算して、期間分に値をセット
    }
}
```

※ このサンプルコードは終了時刻が 0 時以降のケースを考慮しておりません

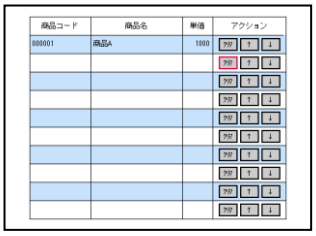
3.8. 表定義テーブルのある行をクリア・並び替えする

例）表定義に入力値をクリア、並び替えするボタンを用意する

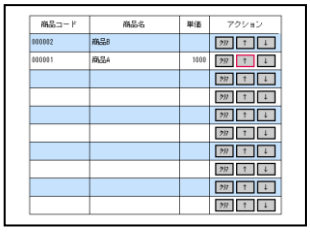


【クリア】ボタン…対象行の値をクリアする
【↑】ボタン…対象行とその上の行の値を置き換える
【↓】ボタン…対象行とその下の行の値を置き換える

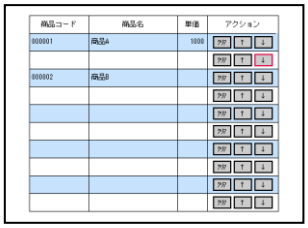
▼ 対象行のクリア



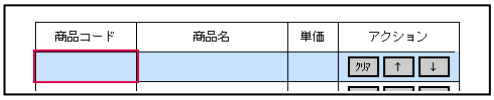
▼ 上へ移動



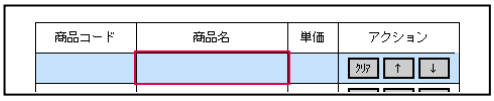
▼ 下へ移動



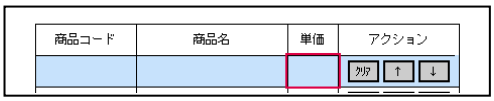
▼ フィールド情報



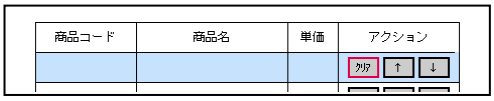
タイプ：文字フィールド
タグ名：商品コード
ID：item_code



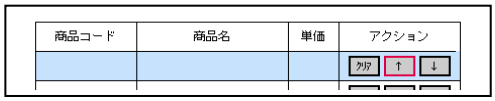
タイプ：文字フィールド
タグ名：商品名
ID：item_name



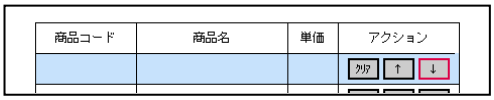
タイプ：数値フィールド
タグ名：単価
ID：unit



タイプ：ボタン
タグ名：クリアボタン
ID：button_clear



タイプ：ボタン
タグ名：上へボタン
ID：button_up



タイプ：ボタン
タグ名：下へボタン
ID：button_down

▼ サンプルコード

[フォーム設定]-[ユーザ定義 JavaScript]欄に以下のコードを記述

```
function clearRecord( _obj, _no ) {                                          … クリア処理を行う function
    var item_code = getElementById_XP( 'item_code_' + _no );                 … 対象行の商品コードを取得する
    var item_name = getElementById_XP( 'item_name_' + _no );                 … 対象行の商品名を取得する
    var unit      = getElementById_XP( 'unit_'      + _no );                 … 対象行の単価を取得する

    item_code.value = '';                                                    … 商品コードをクリアする
    item_name.value = '';                                                    … 商品名をクリアする
    unit.value      = '';                                                    … 商品単価をクリアする
}

function upRecord( _obj, _no ) {                                             … 上への並び替え処理を行う function
    if( _no > 0 ) {                                                          … 対象行が 1 行目でないか判定
        var item_code = getElementById_XP( 'item_code_' + _no );             … 対象行の商品コードを取得する
        var item_name = getElementById_XP( 'item_name_' + _no );             … 対象行の商品名を取得する
        var unit      = getElementById_XP( 'unit_'      + _no );             … 対象行の単価を取得する

        var target_item_code = getElementById_XP( 'item_code_' + ( _no - 1 ) );  … 対象行の上の行の商品コードを取得する
        var target_item_name = getElementById_XP( 'item_name_' + ( _no - 1 ) );  … 対象行の上の行の商品名を取得する
        var target_unit      = getElementById_XP( 'unit_'      + ( _no - 1 ) );  … 対象行の上の行の単価を取得する

        replaceValue( item_code, target_item_code );                         … 商品コードを置き換える
        replaceValue( item_name, target_item_name );                         … 商品名を置き換える
        replaceValue( unit     , target_unit      );                         … 単価を置き換える
    }
}

function downRecord( _obj, _no ) {                                           … 下への並び替え処理を行う function
    if( _no < 9 ) {                                                          … 対象行が最終行でないか判定
        var item_code = getElementById_XP( 'item_code_' + _no );             … 対象行の商品コードを取得する
        var item_name = getElementById_XP( 'item_name_' + _no );             … 対象行の商品名を取得する
        var unit      = getElementById_XP( 'unit_'      + _no );             … 対象行の単価を取得する

        var target_item_code = getElementById_XP( 'item_code_' + ( _no + 1 ) );  … 対象行の下の行の商品コードを取得する
        var target_item_name = getElementById_XP( 'item_name_' + ( _no + 1 ) );  … 対象行の下の行の商品名を取得する
        var target_unit      = getElementById_XP( 'unit_'      + ( _no + 1 ) );  … 対象行の下の行の単価を取得する

        replaceValue( item_code, target_item_code );                         … 商品コードを置き換える
        replaceValue( item_name, target_item_name );                         … 商品名を置き換える
        replaceValue( unit     , target_unit      );                         … 単価を置き換える
    }
}

function replaceValue( object1, object2 ) {                                  … 置換処理を行う function
    dummy = object2.value;                                                       2 つの object 間で value を置き換える
    object2.value = object1.value;
    object1.value = dummy;
}
```

button_clear の onClick イベントハンドラに以下のコードを記述

```
clearRecord( _obj, _no )                                                     … クリア処理を行う function を実行する
```

button_up の onClick イベントハンドラに以下のコードを記述

```
upRecord( _obj, _no )                                                        … 上への並び替え処理を行う function を実行する
```

button_down の onClick イベントハンドラに以下のコードを記述

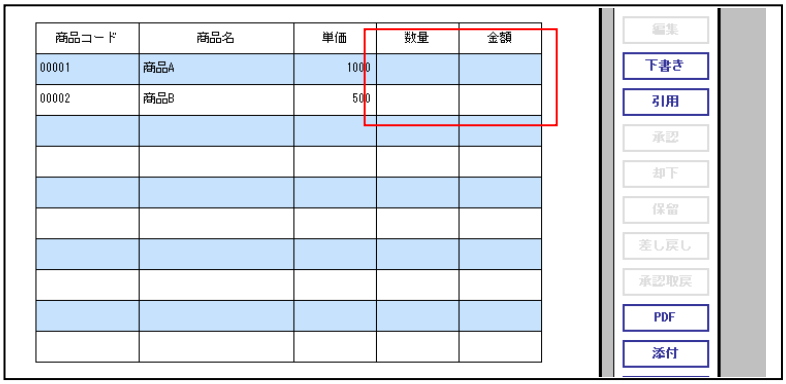
```
downRecord( _obj, _no )                                                      … 下への並び替え処理を行う function を実行する
```

3.9. コピー時にあるフィールドに処理を加えるボタンを作成する

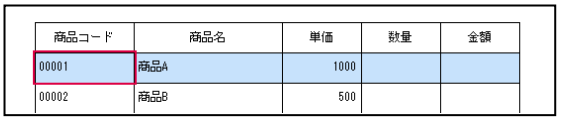
例）数量、金額をクリアしてからコピー処理を実行するボタンを用意する

数量、金額をクリアしてから
コピー処理を実行

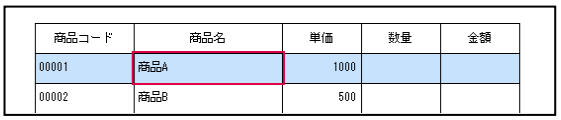
▼ 数量、金額がクリアされる



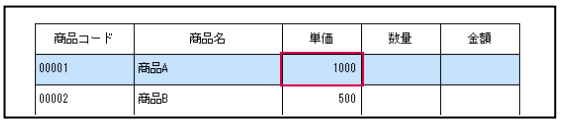
▼ フィールド情報



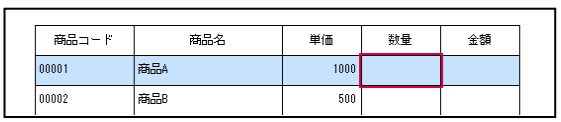
タイプ：文字フィールド
タグ名：商品コード
ID：item_code



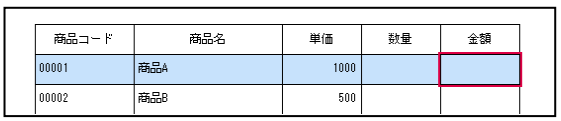
タイプ：文字フィールド
タグ名：商品名
ID：item_name



タイプ：数値フィールド
タグ名：単価
ID：unit



タイプ：整数フィールド
タグ名：数量
ID：quantity

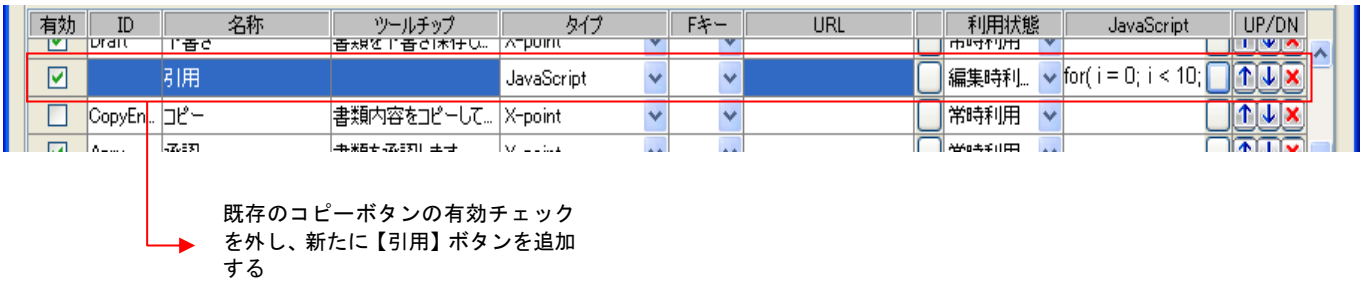


タイプ：数値フィールド
タグ名：金額
ID：amount

▼ サンプルコード

[フォーム設定]-[ナビボタン設定]画面より新規にナビボタンを追加し、追加したナビボタンの JavaScript 欄に以下のコードを記述

▼ ナビボタン設定画面



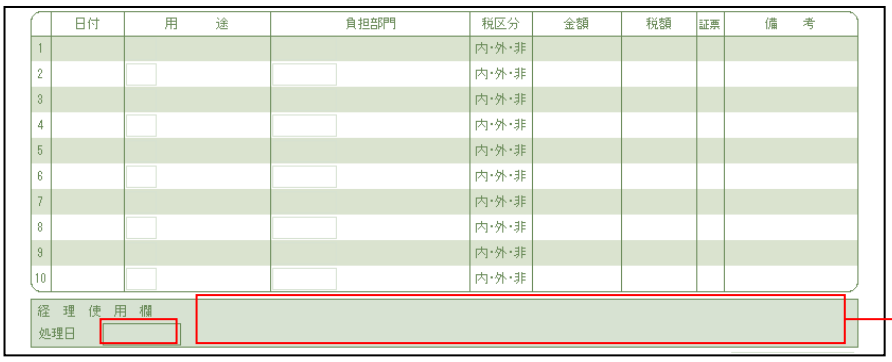
既存のコピーボタンの有効チェックを外し、新たに【引用】ボタンを追加する

```
for( i = 0; i < 10; i++ ) {                          … 行数分ループする
    setFieldValue_XP('quantity_' + i, '' );          … 数量を初期化する
    setFieldValue_XP('amount_' + i, '' );            … 金額を初期化する
}

doCopy();                                            … コピー処理を実行する
```

※ フォーム表示時の表定義内のフィールドの ID は「(フィールドの ID)_(行番号-1)」となります。例えば商品コードの場合は「item_code_0」、「item_code_1」、「item_code_2」・・・となります。

3.10. あるフィールドを特定のステップでのみ編集できるようにする

例）経理使用欄は経理承認のステップ（承認ステップ３）でのみ編集可能とする



経理使用欄の処理日及び備考は経理の承認ステップ以外は読み取り専用

▼ フィールド情報

タイプ：文字フィールド
タグ名：処理日
ID：shori_date

タイプ：テキストエリア
タグ名：経理使用欄
ID：keiri_ran

▼ フィールド情報

[フォーム設定]-[OnLoad 実行 JavaScript]欄に以下のコードを記述します。

```
if( isEditMode() ) {                                          … 現在編集可能な状態かどうかを取得する
  var stepCnt = getCurrentStepNo();                           … 現在の承認ステップを取得する
  var isKeiriStep= (stepCnt == 3);                            … 経理承認のステップかどうか判定

  getElementById_XP('shori_date').readOnly = !isKeiriStep;    … 経理承認のステップでなければ読取専用に設定
  getElementById_XP('keiri_ran' ).readOnly = !isKeiriStep;    … 経理承認のステップでなければ読取専用に設定
}
```

■改訂履歴

改版	改版内容
2021 年 4 月 1 日版	初版
2021 年 9 月 15 日版	軽微な誤字を修正
2021 年 10 月 26 日版	「2.2 ナビボタンに対するアクセス」注意事項の追記
2022 年 1 月 28 日版	eFormMaker のプレビューで getFieldValue_XP の戻り値が undefined になる問題が解決したため注意記載を削除
2022 年 6 月 21 日版	「2.3.2. フィールドタイプの判定オブジェクトの取得」に イメージアップロード・フィールド（XWEB.FieldType.ImageUpload）を追加
2023 年 2 月 08 日版	「2.3.3.フィールド情報の取得と操作」の関数「execMasterSeek」の説明の誤りを修正
2023 年 4 月 01 日版	「2.8 ユーティリティ API」に「getDaysnote」、「isBusinessDay」、「XpNetwork.dateif」、「XpNetwork.add」、「XpNetwork.sub」を追加
2024 年 2 月 21 日版	「2.6 ナビボタンの処理」の関数「doEdit」の説明を修正
2024 年 3 月 18 日版	「2.4. ドキュメント情報/ログインユーザ情報の取得」に「関連書類の作成総数」を追加
2025 年 10 月 14 日版	「2.6. ナビボタンの処理」の実行可否の説明を修正
2025 年 11 月 10 日版	「2.3.3.フィールド情報の取得と操作」の「setTextVisible」の内容に、実装例を追記
2026 年 4 月 8 日版	「2.6. ナビボタンの処理」の実行可否の説明を修正
2026 年 5 月 1 日版	「2.8. ユーティリティ API」の isBusinessDay の eFormMaker プレビューの補足説明を削除