AgileWorks R3

X-WebForm JavaScript プログラミングガイド

R3.3 第 1 版(2026/07/15)

ATLED

目次／索引

1. はじめに 4
1.1. 対象読者 4
1.2. 用語 4
1.3. JavaScript プログラミングを始める前に 4
1.4. 制限事項 4
ドキュメントビューアの編集モードについて 4
2. JavaScript の定義 5
2.1. JavaScript 設定 5
基本設定 5
2.2. ユーザー定義 6
2.3. OnLoad イベントの定義 6
2.4. OnLoad finally イベントの定義 6
2.5. UnLoad イベントの定義 6
2.6. PageChange イベントの定義 7
2.7. PageChangeValidator イベントの定義 7
2.8. フィールドアクションの定義 7
2.9. 外部 JavaScript ファイルのロード 8
2.10. JavaScript の挿入 9
2.11. 関数名の制限 9
3. X-WebForm JS API の利用 10
3.1. フィールド API 10
3.2. フィールドアクション API 18
3.3. サーバー日時の取得 19
4. AgileWorks JS API の利用 21
4.1. ワークフロー情報の取得 (WorkflowInfo) 21
4.2. ワークフローイベントの登録 (WorkflowEvent) 27
4.3. X-WebForm でプレビューする際の注意事項 30
5. 実装サンプル 31
5.1. 条件付き入力チェック 31
5.2. 編集不可状態の切り替え 31
5.3. フィールドの色の切り替え 32

◆ 改版履歴

版数	年月日	改版内容
第 1 版	2026 年 07 月 15 日	第 1 版作成

1. はじめに

X-WebForm はフォームをブラウザに表示するため、JavaScript を利用したプログラムを行うことが可能です。
但し、一般的な JavaScript コーディングとは異なり、X-WebForm と AgileWorks に用意された各種 API を利用しながらコーディングしていくことになります。本書ではフォーム上で JavaScript プログラミングを行う手順について説明します。

1.1. 対象読者

本書では、実際の JavaScript コーディング例を元に各 API の仕様に関する説明を記載しています。
そのため、HTML/CSS/JavaScript でのプログラミングに関する基本的な知識を有する方を対象としています。

1.2. 用語

用語	説明
X-WebForm JS API	X-WebForm が提供する、フィールド操作等を行うための API を指します。本 API は X-WebForm プレビュー、AgileWorks の両環境で利用することができます。
AgileWorks JS API	AgileWorks が提供する、ワークフロー情報の取得や書類操作時のイベント登録等を行うための API を指します。本 API は AgileWorks 環境でのみ利用可能です。

1.3. JavaScript プログラミングを始める前に

JavaScript でのプログラミングは、必須項目の文字色・背景色の変更やフィールドの動的な編集不可制御といった、ユーザビリティを向上させるようなシーンに適しています。フィールドの入力チェックや値変更するといった処理を実装することも可能ですが、より値の整合性が厳しく求められるようなシーンでは、SDK を利用してサーバーサイドで実装を行うことが望ましいです。

また AgileWorks では、ドキュメントビューアからのワークフロー操作だけなく、一括承認・確認機能や SDK、モバイルサイトといった様々な箇所で、ワークフロー操作を行うためのインターフェースを提供しております。そのため、JavaScript の実行に依存してしまうとこれらの機能を利用できないというデメリットが発生します。

JavaScript プログラミングは適切な利用シーンかどうかを考慮したうえでご利用ください。

1.4. 制限事項

ドキュメントビューアの編集モードについて

AgileWorks 上でフォームを利用する際に利用するドキュメントビューアには編集モードという概念があります。

作成したフォームを AgileWorks のワークフローで回付しドキュメントビューアで表示した場合に、初回は非編集モードで表示され、全てのフィールドが編集不可状態となります。【編集】メニューをクリックすると編集モードに切り替わり、各フィールドが入力可能な状態となります。

そのため、ドキュメントビューアが非編集モードの場合に JavaScript でフィールドの値を変更しても、変更した値が保存されないという制限事項があります。

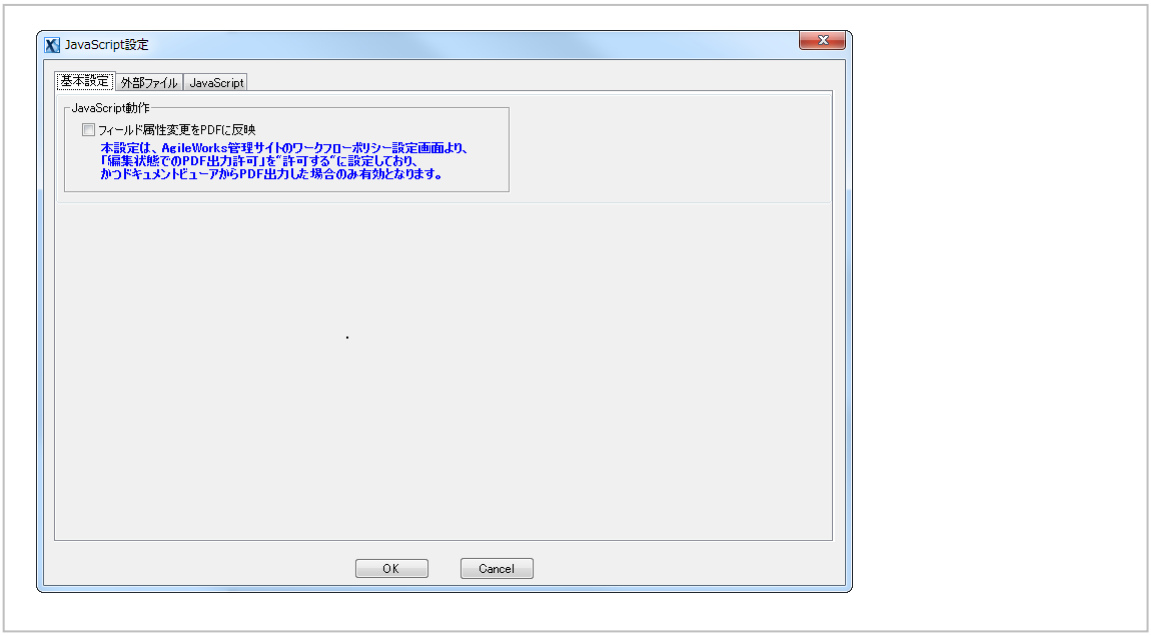
2. JavaScriptの定義

2.1. JavaScript 設定

メニュー[JavaScript 設定]より、フォームで動作させる JavaScript に関する設定を行います。

基本設定

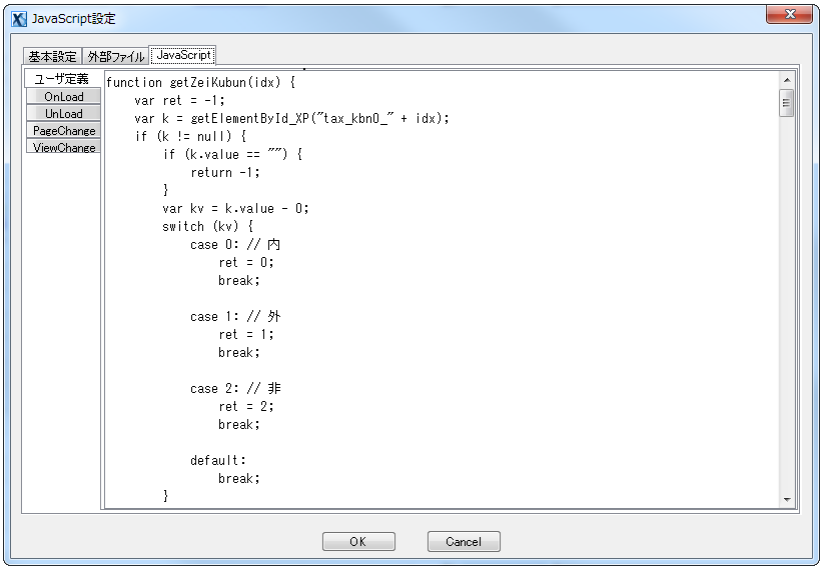
JavaScript 動作に関する基本設定を行います。



項目	説明
フィールド属性変更を PDF に反映	この項目がチェックされていると、HTML フォームに記述した JavaScript で XWEB.getObject() メソッドを介してフィールドの色などを変更した結果が PDF にも反映されます。 以下の項目のみが反映対象となります。 ・ 文字色 ・ 背景色 ・ 枠線色 ・ 可視/不可視 本設定は以下の条件に一致している場合のみ有効となります。PAPI やモバイルサイトなどから PDF 出力を行った場合は有効とならないため、本設定の有効化は推奨いたしません。 ・ AgileWorks 管理サイトのワークフローポリシー設定画面より、「編集状態での PDF 出力許可」を"許可する"に設定している ・ ドキュメントビューアから PDF 出力を行なっている

2.2. ユーザー定義

X-WebForm の[ファイル]-[JavaScript 設定]-[JavaScript]-[ユーザー定義]より、独自の JavaScript を定義することが可能です。ここで定義された JavaScript はフォーム表示時に実行されます。そのため、フィールドやボタンなどで、JavaScirpt を記述する際に関数名のみを書くという簡潔な記述が可能になる為、ここでは主にグローバル変数や複数の箇所から実行される共通関数などを定義します。



2.3. OnLoad イベントの定義

X-WebForm の[ファイル]-[JavaScript 設定]-[JavaScript]-[OnLoad]-[OnLoad]より、フォームが表示されたタイミングで実行される JavaScript を定義することが可能です。

2.4. OnLoad finally イベントの定義

X-WebForm の[ファイル]-[JavaScript 設定]-[JavaScript]-[OnLoad]-[OnLoad finally]より、フォームが表示されたタイミングで実行される JavaScript を定義することが可能です。

OnLoad finally イベントは OnLoad イベントとは異なり、各フィールドのミラー処理や数値の端数処理等の初期設定処理が実行された後に実行されます。そのため、フィールド値を元に何らかの実装を行う場合は OnLoad finally を利用してください。

[処理順序]
1.OnLoad に記述された JavaScript
2.X-WebForm による各フィールドの初期設定処理(※)
3.OnLoad finally に記述された JavaScript

※「各フィールドの初期設定処理」とは、内部的に行われる
・数値フィールドで小数点以下端数処理
・ミラー定義項目の初回動作
・日付セットの曜日設定
等の各設定処理を指します。

2.5. UnLoad イベントの定義

X-WebForm の[ファイル]-[JavaScript 設定]-[JavaScript]-[UnLoad]より、ブラウザに表示されている内容が破棄、またはリロードされたタイミングで実行される JavaScript を定義することが可能です。

2.6. PageChange イベントの定義

X-WebForm の[ファイル]-[JavaScript 設定]-[JavaScript]-[PageChange]-[onChange]より、ページ構成がレイヤーページ方式の場合に、ページを切り替えた際に実行される JavaScript を定義することが可能です。

引数には表示されるページ番号 pgno が渡されます。ページ番号は 0 から開始されます。

```
alert((pgno + 1) + "ページ目が表示されました。");
```

2.7. PageChangeValidator イベントの定義

X-WebForm の[ファイル]-[JavaScript 設定]-[JavaScript]-[PageChange]-[validator]より、ページ構成がレイヤーページ方式の場合に、ページを切り替える直前に実行される JavaScript を定義することが可能です。

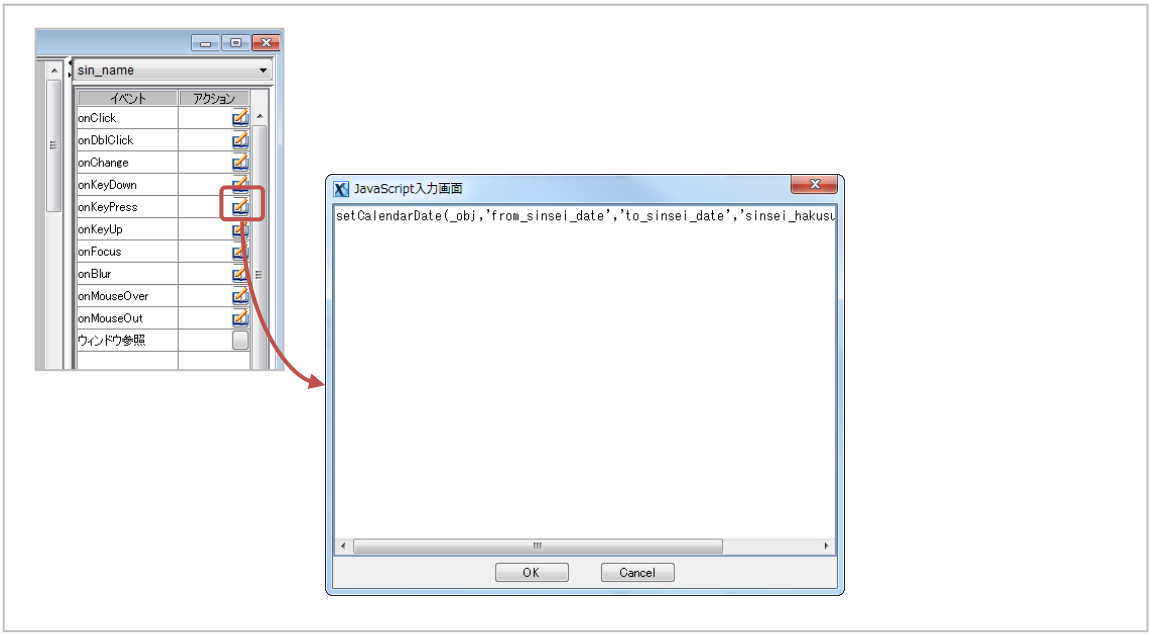
引数には表示されるページ番号 pgno、現在のページ番号 curno が渡されます。ページ番号は 0 から開始されます。フォーム表示直後の場合、curno は undefined になります。

戻り値を false とすることで、ページ切り替えを行わないよう制御することが可能です。

```
if (curno != undefined) {
    var value = XWEB.getObject("textfield1").getValue();
    if (!value) {
        return false;
    }
}
return true;
```

2.8. フィールドアクションの定義

各フィールドのアクション時に実行される JavaScript を定義することが可能です。



表明細内のフィールドの場合は行番号_no が渡されます。行番号は 0 から開始されます。

```
alert((_no + 1) + "行目のフィールドで実行されました");
```

各フィールドタイプで利用可能なアクションは以下の通りです。

アクション	ラベル	文字	数値	整数	テキストエリア	ボタン	ラジオボタン	チェックボックス	コンボ	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 クリック時に動作するアクション	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	-	-	○
onDblClick ダブルクリック時に動作するアクション	○	○	○	○	○	○	○	○	-	-	○	○	○	○	○	-	-	○
onChange(※1) 値変更時に動作するアクション	-	○	○	○	○	-	○	○	○	○	○	○	○	○	○	-	-	-
onKeyDown キー押下時に動作するアクション	-	○	○	○	○	○	○	○	○	○	○	○	○	○	○	-	-	-
onKeyPress キー押下中に動作するアクション	-	○	○	○	○	○	○	○	○	○	○	○	○	○	○	-	-	-
onKeyUp キーを離した時に動作するアクション	-	○	○	○	○	○	○	○	○	○	○	○	○	○	○	-	-	-
onFocus フォーカスが当たった時に動作するアクション	-	○	○	○	○	-	○	○	-	-	○	○	○	○	○	-	-	-
onBlur フォーカスが外れた時に動作するアクション	-	○	○	○	○	-	○	○	-	-	○	○	○	○	○	-	-	-
onMouseOver マウスカーソルが重なった場合に動作するアクション	○	○	○	○	○	-	○	○	-	-	○	○	○	○	○	-	-	○
onMouseOut マウスカーソルが外れた時に動作するアクション	○	○	○	○	○	-	○	○	-	-	○	○	○	○	○	-	-	○
onCalcValidator 計算処理実行前に動作するアクション	-	-	○	○	-	-	-	-	-	-	○	○	○	○	○	-	-	-

※1　フィールド API「setValue」「clearAll」実行後の onChange アクションはラジオボタンフィールドもしくはチェックボックスフィールドの場合のみ動作します。それ以外のフィールドの場合は動作しません。

2.9.　外部 JavaScript ファイルのロード

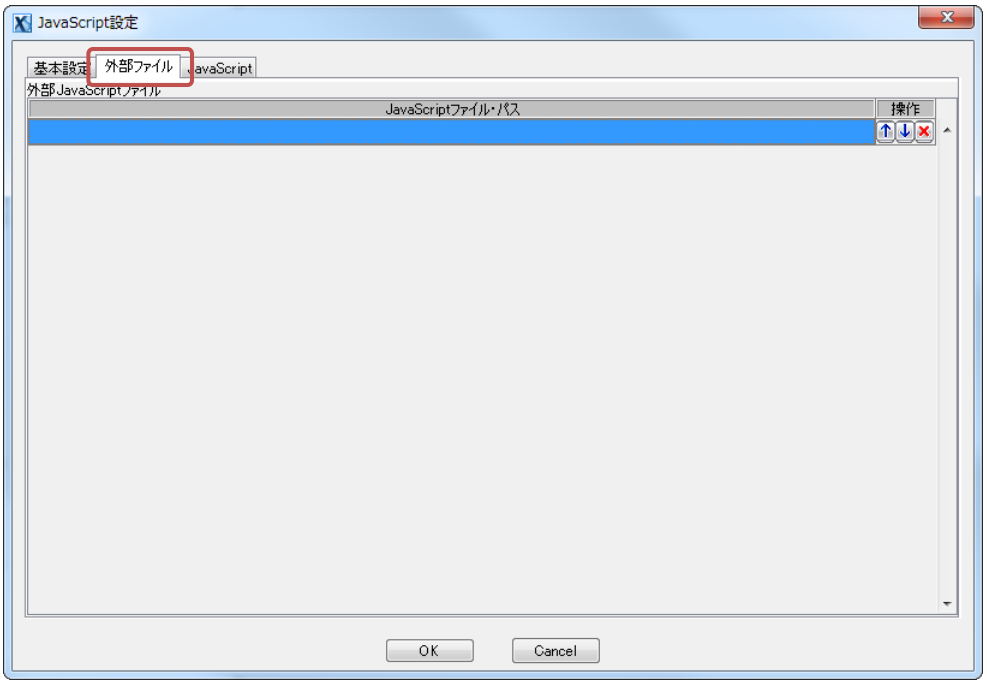
外部 JavaScript ファイルを設定します。

JavaScript を外部ファイルとして配置する場合、以下の注意点があります。

1. JavaScript ファイルは、AgileWorks アプリケーションサーバーから HTTP 通信で参照できる場所に配置し、絶対 URL で指定する
   例えば､AgileWorks アプリケーションサーバー上の Apache DocumentRoot 配下/js というディレクトリを用意して、その中に配置するなどします。
   ○ http://<Server>/js/common.js
   × /usr/local/apache2/js/common.js（サーバーのディレクトリ指定は NG）
   × c:¥Program Files¥Apache Software Foundation¥Apache2.2¥js¥common.js（サーバーのディレクトリ指定は NG）

2. JavaScript ファイルの文字コードは、UTF-8 にする
   AgileWorks 側 HTML ファイルの文字コードが UTF-8 である為、JavaScript 内で全角を含む記述が存在すると文字化け等の原因となります。ファイル文字コードの変更はテキストエディタ等を利用してください。

3. 指定する外部 JavaScript ファイルの数はあまり多くなりすぎないよう注意する
   Load する JavaScript ファイルが増えると、Web サーバーへの通信数が増える為、できるだけ 1 つのファイルにまとめることを推奨します。



2.10. JavaScript の挿入

X-WebForm の[ファイル]-[JavaScript 挿入]より、選択された JavaScript ファイルを読み込み、フォームのユーザー定義 JavaScript として設定します。
読み込み後にユーザー定義 JavaScript を編集した場合も、元のファイルは修正されません。

2.11. 関数名の制限

AgileWorks では、『フィールドの ID』または以下のキーワードを含む関数を内部的に自動生成する為、独自に実装する JavaScript 関数（Function）には該当のキーワードを含めないでください。

なお、以下は一般的に利用の可能性がある代表的な例で、この表に含まれないキーワードでも発生する場合があります。

▼キーワード例

AgileWorks	BigDecimal	I18N	JKL	MathContext
WorkflowEvent	WorkflowInfo	XGEN	XGENFig	XGENHandler
XWEB	XWEVT	expfield	sumfield	exp
sum	onclick	ondblclick	onchange	onkeydown
onkeypress	onkeyup	onfocus	onblur	onmouseover
onmouseout	oncalcvalidator			

3. X-WebForm JS APIの利用

3.1. フィールド API

フォーム上に配置されたフィールドのオブジェクト(本書での型表記は Field)を取得する場合、XWEB.getObejct()を利用します。引数にはフィールドの ID を指定します。

```
var fieldObject = XWEB.getObject("textfield1");
```

取得したフィールドオブジェクトには以下のプロパティ、メソッドが提供されています。
プロパティを変更することはできません。

```
fieldObject.id // プロパティにアクセス
fieldObject.getValue() // メソッドを実行
```

プロパティ

id

型:
String

説明:
フィールドの ID。
表明細内のフィールドの場合は、ID の後ろに"_(行番号)"が付与されます。

type

型:
String

説明:
フィールドのタイプ。
XWEB.FieldType のいずれかが格納されているため、以下の要領でフィールドのタイプを判定することも可能です。

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

XWEB.FieldType の定義一覧は以下の通りです。

項目	説明
XWEB.FieldType.Label	ラベル
XWEB.FieldType.Text	文字フィールド
XWEB.FieldType.Number	数値フィールド
XWEB.FieldType.Integer	整数フィールド
XWEB.FieldType.TextArea	テキストエリア
XWEB.FieldType.Button	ボタン
XWEB.FieldType.RadioButton	ラジオボタン
XWEB.FieldType.CheckBox	チェックボックス
XWEB.FieldType.ComboBox	コンボボックス
XWEB.FieldType.List	リストボックス
XWEB.FieldType.Year	西暦フィールド
XWEB.FieldType.Month	月フィールド
XWEB.FieldType.Day	日フィールド
XWEB.FieldType.WeekDay	曜日フィールド
XWEB.FieldType.Stamp	印影フィールド
XWEB.FieldType.Barcode	バーコードフィールド

XWEB.FieldType.Hidden	不可視フィールド
XWEB.FieldType.Table	表明細グループ
XWEB.FieldType.Hline	水平線
XWEB.FieldType.Vline	垂直線
XWEB.FieldType.Line	直線
XWEB.FieldType.Rect	長方形枠
XWEB.FieldType.Fill	長方形
XWEB.FieldType.Image	画像フィールド

メソッド

addRoundType(pos, type)

引数:

Number	pos	小数点以下桁数を指定します
Number	type	端数処理を指定します。 1: 四捨五入 2: 切り捨て 3: 切り上げ

戻り値:
void

説明:
小数点以下桁数、端数処理を指定します。設定の解除はできません。

backField()

戻り値:
Field

説明:
フォーカス取得可能な、前のフィールドを取得します。

clearAll()

戻り値:
void

説明:
表明細内の全てのフィールドの値をクリアします。
フィールドオブジェクトが表明細グループの場合に限り利用可能です。

```
var table = XWEB.getObject("group1");
table.clearAll();
```

getBackgroundColor()

戻り値:
String

説明:
背景色を、RGB 表記の文字列("rgb(255,0,0)"等)で取得します。

getColor()

戻り値:
String

説明:
線や長方形などの図形の色を、RGB 表記の文字列("rgb(255,0,0)"等)で取得します。

getField(rownum, id)

引数:

Number	rownum	行番号(0 から開始される)
String	id	フィールドの ID

戻り値:
Field

説明:
表明細内のフィールドを取得します。
フィールドオブジェクトが表明細グループの場合に限り利用可能です。

```
var table = XWEB.getObject("group1");
for (var row = 0, maxrow = table.getMaxrow(); row < maxrow; row++) {
    var field = table.getField(row, "textfield1");
}
```

getForegroundColor()

戻り値:
String

説明:
文字色を、RGB 表記の文字列("rgb(255,0,0)"等)で取得します。

getItemNames()

戻り値:
Array<String>

説明:
表を構成するフィールド ID を格納した配列を取得します。
フィールドオブジェクトが表明細グループの場合に限り利用可能です。

```
var table = XWEB.getObject("group1");
var names = table.getItemNames();
for (var i = 0, length = names.length; i < length; i++) {
    var name = names[i];
}
```

getMaxrow()

戻り値:
Number

説明:
表明細の最大行数を取得します。フィールドオブジェクトが表明細グループの場合に限り利用可能です。

```
var table = XWEB.getObject("group1");
var maxrow = table.getMaxrow();
```

getObject(rownum, id)

引数:

Number	rownum	行番号(0 から開始される)
String	id	フィールドの ID

戻り値:
Field

説明:
表明細内のオブジェクトを取得します。(入力フィールドと図形との区別を行いません)
フィールドオブジェクトが表明細グループの場合に限り利用可能です。

```
var table = XWEB.getObject("group1");
for (var row = 0, maxrow = table.getMaxrow(); row < maxrow; row++) {
    var field = table.getObject(row, "rectangle1");
}
```

getOptionLists(flag)

引数:

boolean	flag	配列に格納される情報に選択状態かどうかの情報を含めるかどうか。 true の場合、選択肢の値と選択状態かどうかのプロパティを持つオブジェクトが格納されます。

リストボックスのみ指定可能です。

```
[{"value" : "男性", "selected" : false}, {"value" : "女性",
"selected" : true}]
```

false の場合、選択肢の値自体が格納されます。

```
["男性", "女性"]
```

戻り値:
Array<Object>

説明:
リストボックスの選択肢情報を格納した配列を取得します。

getRecColor()

戻り値:
String

説明:
枠線色を、RGB 表記の文字列("rgb(255,0,0)"等)で取得します。

getRoundPosition()

戻り値:
Number

説明:
小数点以下桁数を数値で取得します。

getRoundType()

戻り値:
Number

説明:
端数処理を取得します

1: 四捨五入
2: 切り捨て
3: 切り上げ

getTextValue()

戻り値:
String

説明:
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します。

getTooltipText()

戻り値:
String

説明:
ツールチップ文字列を取得します。

getValue()

戻り値:
Object

説明:
フィールドの値を取得します。

リストボックスの場合は、選択されている項目のうち先頭の選択値が返ります。
数値・整数フィールドは Number 型オブジェクト、その他のフィールドは String 型オブジェクトが返ります。

getValue(rownum, id)

引数:

Number	rownum	行番号(0 から開始される)
String	id	フィールドの ID

戻り値:
Object

説明:
フィールドが表明細グループの場合に、表明細内のフィールドの値を取得します。
戻り値の仕様は getValue() と同様になります。

isComma()

戻り値:
boolean

説明:
カンマ区切り指定の有無を返します。

isDisabled()

戻り値:
boolean

説明:
無効状態かどうかを返します。

isEnabled()

戻り値:
boolean

説明:
有効状態かどうかを返します。

isHoliday(date)

引数:

String	date	yyyy-MM-dd || yyyy/MM/dd || yyyyMMdd （ゼロ埋め）

戻り値:
boolean

説明:
指定日が休日かどうかを返します。
休日の場合は true を返します。休日＝非営業日

isReadOnly()

戻り値:
boolean

説明:
編集不可かどうかを返します。

isRoundField()

戻り値:
boolean

説明:
端数処理を行うかどうかを返します。
数値・整数フィールドの場合は true を返します。

isScroll()

戻り値:
boolean

説明:
表明細にスクロール設定がされているかどうかを返します。
フィールドオブジェクトが表明細グループの場合に限り利用可能です。

isTextVertical()

戻り値:
boolean

説明:
ラベルが縦書き指定かどうかを返します。

isVisible()

戻り値:
boolean

説明:
可視状態かどうかを返します。

nextField()

戻り値:
Field

説明:
フォーカス取得可能な、次のフィールドを取得します。

optionListAdd(caption, value)

引数:

String	caption	選択肢の表示文言
String	value	選択肢の値

戻り値:
void

説明:
コンボボックス、リストボックスの選択肢を追加します。

optionListClear(addBlank)

引数:

boolean	addBlank	先頭に空白を追加するかどうか。 true を指定すると先頭に空白が追加されます。

戻り値:
void

説明:
コンボボックス、リストボックスの選択肢をクリアします。

optionListRemove(value)

引数:

String	value	選択肢の値

戻り値:
void

説明:
コンボボックス、リストボックスの選択肢のうち、指定された値を持つ候補を削除します。

setBackgroundColor(color)

引数:

String	color	RGB 表記の文字列("rgb(255,0,0)"等)

戻り値:
void

説明:
背景色を変更します。
※ボタンには利用できません。

setColor(color)

引数:

String	color	RGB 表記の文字列("rgb(255,0,0)"等)

戻り値:
void

説明:
線や長方形などの図形の色を変更します。

setDisabled(disabled)

引数:

boolean	disabled	無効とするかどうか

戻り値:
void

説明:
有効・無効状態を変更します。
表明細グループに対して実行した場合は、表明細内のすべてのフィールドの有効・無効状態を変更します。

setEnabled(enabled)

引数:

boolean	enabled	有効とするかどうか

戻り値:
void

説明:
有効・無効状態を変更します。

setFocus()

戻り値:
void

説明:
フィールドにフォーカスを当てます。
フォーカス移動時に発生する onChange/onBlur イベントにて本 API を利用することは出来ません。

setFontName(fontName)

戻り値:
void

引数:

String	fontName	表示フォントの名称("MS UI Gothic"等)

説明:
表示フォントを変更します。

setFontSize(fontSize)

引数:

Number	fontSize	フォントサイズ(整数)

戻り値:
void

説明:
フォントサイズを変更します。
※Internet Explorer の場合、9 以降では動作しません。

setFontWeight(fontWeight)

引数:
String fontWeight 文字の太さを指定する

標準: normal
ボールド: bold

戻り値:
void

説明:
文字の太さを指定します。

setForegroundColor(color)

引数:
String color RGB 表記の文字列("rgb(255,0,0)"等)

戻り値:
void

説明:
文字色を変更します。

setImeMode(imeMode)

引数:
String imeMode IME 状態

自動: auto
ON にする: active
OFF にする: inactive
無効にする: disabled

戻り値:
void

説明:
IME 状態を変更します。
本設定は Internet Explorer、Firefox(Windows)でのみ動作します。

setReadOnly(readOnly)

引数:
boolean readOnly 編集不可とするかどうか

戻り値:
void

説明:
編集不可状態を変更します。
表明細グループに対して実行した場合は、表明細内のすべてのフィールドの編集不可状態を変更します。

setRectColor(color)

引数:
String color RGB 表記の文字列("rgb(255,0,0)"等)

戻り値:
void

説明:
枠線色を変更します。

setTooltipText(text)

引数:
String text 文字列

戻り値:
void

説明:
ツールチップ文字列を変更します。

setValue(value)

引数:

Object	value	フィールド値 数値・整数フィールドには Number 型オブジェクト、その他のフィールドには String 型オブジェクトを指定する。ラジオボタンに対して空文字("")を指定すると、ボタンが全て未選択の状態となる。

戻り値:
void

説明:
フィールドの値を変更します。

setVisible(visible)

引数:

boolean	visible	可視状態かどうか

戻り値:
void

説明:
可視状態を変更します。


注意事項
上記 API でフィールドのスタイルを変更した場合、PDF 出力時には反映されません。
ただし以下の API での変更については、[ファイル]-[JavaScript 設定]-[基本設定]にある「フィールド属性変更を PDF に反映」にチェックを付けることで反映可能です。

- setVisible
- setForegroundColor
- setBackgroundColor
- setRectColor
- setColor

3.2. フィールドアクション API

フィールドのアクションに定義した JavaScript を別の JavaScript から実行したい場合は、フィールドオブジェクトに対して以下のように API を実行します。

```
XWEB.getObject("textfield1").OnChange();
```

対象フィールドが表明細内のフィールドの場合、引数_no が渡されずに実行されるため、アクションに割り当てられた JavaScript 側で引数未指定の場合を考慮する必要があります。

メソッド

OnBlur()

戻り値:
void

説明:
onBlur アクションに定義された JavaScript を実行します。

OnChange()

戻り値:
void

説明:
onChange アクションに定義された JavaScript を実行します。

OnClick()

戻り値:
void

説明:
onClick アクションに定義された JavaScript を実行します。

OnDblClick()

戻り値:
void

説明:
onDblClick アクションに定義された JavaScript を実行します。

OnFocus()

戻り値:
void

説明:
onFocus アクションに定義された JavaScript を実行します。

OnKeyDown()

戻り値:
void

説明:
onKeyDown アクションに定義された JavaScript を実行します。

OnKeyPress()

戻り値:
void

説明:
onKeyPress アクションに定義された JavaScript を実行します。

OnKeyUp()

戻り値:
void

説明:
onKeyUp アクションに定義された JavaScript を実行します。

OnMouseOver()

戻り値:
void

説明:
onMouseOver アクションに定義された JavaScript を実行します。

OnMouseOut()

戻り値:
void

説明:
onMouseOut アクションに定義された JavaScript を実行します。

3.3. サーバー日時の取得

JavaScript を作成する際、Date オブジェクトを利用して時刻を取得するとクライアント PC の時間が取得されます。クライアント PC の時刻設定が正確であれば問題ありませんが、誤った時刻や日付になっていると不都合な場合があります。その様な場合、サーバー日時を取得する関数が用意されています。

```
var d = XWEB.Date.get();
```

注意事項
サーバー日時にはサーバーの応答速度分が遅れとなって反映される為、完全には一致しません。

4. AgileWorks JS APIの利用

この章では、AgileWorks で用意されている JavaScript API について説明します。X-WebForm で作成したフォームを AgileWorks で動作させる際に、これらの API を利用することができます。

4.1. ワークフロー情報の取得 (WorkflowInfo)

WorkflowInfo オブジェクトを基点として、現在操作している書類のワークフロー情報を取得することができます。ワークフロー情報には、書類の基本情報や回付情報、ログイン中のユーザーの情報等が含まれます。

X-WebForm JS API と AgileWorks JS API を組み合わせて実行することも可能です。

```
// ドキュメントビューアが編集モードの場合、スタイルを変更する
if (WorkflowInfo.isEditable()) {
    XWEB.getObject("textfield1").setBackgroundColor("rgb(255,255,204)");
}
```

WorkflowInfo が提供する API は以下の通りです。

メソッド
`getAdminNo` 戻り値: String 説明: 書類管理番号を取得します。
`getApplyCriterionDate` 戻り値: String 説明: 基準日(yyyy-MM-dd 形式)を取得します。
`getApplyDate` 戻り値: String 説明: 申請日(yyyy-MM-dd HH:mm 形式)を取得します。
`getApplyUnitName` 戻り値: String 説明: 申請組織名称を取得します。
`getApplyUserName` 戻り値: String 説明: 申請ユーザー名称を取得します。
`getDoc` 戻り値: Doc

説明:
書類情報を取得します。

getDocId

戻り値:
Number

説明:
書類 ID を取得します。
書類 ID は書類の初回保存時に発番されます。
書類を新規作成したタイミングでは発番されていないため「-1」が取得されます。

getFixDate

戻り値:
String

説明:
承認完了日(yyyy-MM-dd HH:mm 形式)を取得します。

getFormName

戻り値:
String

説明:
フォーム名称を取得します。

getLoginAccount

戻り値:
LoginAccount

説明:
ログインアカウント情報を取得します。

getOperator

戻り値:
User

説明:
操作ユーザー情報を取得します。

getOperatorOrgUnits

戻り値:
Array<OrgUnit>

説明:
操作ユーザーの所属組織情報を取得します。

getOperatorUniversalRoles

戻り値:
Array<Role>

説明:
操作ユーザーのユニバーサルロール情報を取得します。

getOwnerName

戻り値:
String

説明:
書類オーナー名称を取得します。

getParentDoc

戻り値:
Doc

説明:
親書類情報を取得します。存在しない場合は null が返ります。

getPrincipal

戻り値:
User

説明:
代理承認時における本人ユーザー情報を取得します。

getPrincipalOrgUnits

戻り値:
Array<OrgUnit>

説明:
代理承認時における本人ユーザーの所属組織情報を取得します。

getPrincipalUniversalRoles

戻り値:
Array<Role>

説明:
代理承認時における本人ユーザーのユニバーサルロール情報を取得します。

getRuleName

戻り値:
String

説明:
回付ルール名称を取得します。

getStatusLabel

戻り値:
String

説明:
現在のステータス文字列を取得します。

getTaskStepCode

戻り値:
String

説明:
操作ユーザーのステップコードを取得します。

isEditable

戻り値:
String

説明:
書類が編集状態かどうかを取得します。

Doc が提供する API は以下の通りです。

メソッド

getAdminNo

戻り値:
String

説明:

書類管理番号を取得します。

getCurrentSteps

戻り値:
Array<RuleStep>

説明:
現在ステップのリストを取得します。

回付ルールに並列フローが含まれる場合、書類の回付状況によって複数の RuleStep オブジェクトを格納した配列が返ります。並列フローが存在しない場合は、常に要素が 1 つの配列が返ります。

getId

戻り値:
Number

説明:
書類 ID を取得します。

LoginAccount が提供する API は以下の通りです。

メソッド

getCode

戻り値:
String

説明:
コードを取得します。

getName

戻り値:
String

説明:
名称を取得します。

getLocaleName

戻り値:
String

説明:
ロケール名を取得します。

User が提供する API は以下の通りです。

メソッド

getCode

戻り値:
String

説明:
コードを取得します。

getName

戻り値:
String

説明:
名称を取得します。

getKana

戻り値:

String

説明:
カナを取得します。

getLoginId

戻り値:
String

説明:
ログイン ID を取得します。

getMailAddress

戻り値:
String

説明:
メールアドレスを取得します。

getStampName

戻り値:
String

説明:
印影上の表示名を取得します。

getRemarks

戻り値:
String

説明:
備考を取得します。

OrgUnit が提供する API は以下の通りです。

メソッド

getCode

戻り値:
String

説明:
コードを取得します。

getName

戻り値:
String

説明:
名称を取得します。

getSectionRole

戻り値:
Role

説明:
セクションロールを取得します。

Role が提供する API は以下の通りです。

メソッド

getCode

戻り値:

String

説明:
コードを取得します。

getName

戻り値:
String

説明:
名称を取得します。

RuleStep が提供する API は以下の通りです。

メソッド

getCode

戻り値:
String

説明:
コードを取得します。

getName

戻り値:
String

説明:
名称を取得します。

書類のフォーム名称を取得し、フォーム上のあるフィールドにセットする場合は WorkflowInfo.getFormName を利用します。

```
var formName = WorkflowInfo.getFormName();
XWEB.getObject("formname").setValue(formName);
```

現在ステップに応じてフィールドの編集状態を切り替える場合は、WorkflowInfo.isEditable と WorkflowInfo.getDoc().getCurrentSteps を利用します。ドキュメントビューアが非編集モードの場合は全てのフィールドが readOnly のため、WorkflowInfo.isEditable を利用して編集状態かどうかの確認を行っています。

```
if (WorkflowInfo.isEditable()) {
    var steps = WorkflowInfo.getDoc().getCurrentSteps();
    if (steps != null) {
        for (var i = 0, length = steps.length; i < length; i++) {
            if (steps[i].getCode() == "APPROVE_STEP") {
                XWEB.getObject("textfield1").setReadOnly(true);
                break;
            }
        }
    }
}
```

4.2. ワークフローイベントの登録 (WorkflowEvent)

WorkflowEvent オブジェクトから、書類操作メニューに対するイベントハンドラを登録することができます。この API により、承認メニューをクリックした際に確認ダイアログを表示するなど、操作イベントに応じた処理を実装することが可能です。

WorkflowEvent が提供する API は以下の通りです。

メソッド

```
addEvent(caption, stepCode, func)
```

戻り値:
void

引数:

String caption 回付ルールで設定した操作メニュー文言(承認、却下等)。



String stepCode イベントハンドラの実行ステップのコード。
未指定の場合は全てのステップでイベントハンドラが実行される。



Function func イベントハンドラ(必ず boolean 値を return する)

説明:
イベントハンドラを登録します。
イベントハンドラは必ず boolean 値を返す必要があります。true が返された場合のみ、後続の書類操作処理が実行されます。

以下、非対応の操作メニューとステップとなります。

▼非対応の操作メニュー
・編集
・コピーして新規
・関連書類

```
purgeEvent(caption, stepCode)
```

戻り値:
void

引数:

String caption 回付ルールで設定した操作メニュー文言(承認、却下等)。
String stepCode イベントハンドラの実行ステップのコード。

説明:

指定条件に一致した全てのイベントハンドラを破棄します。

```
removeEvent(caption, stepCode, func)
```

戻り値:
void

引数:

String	caption	回付ルールで設定した操作メニュー文言(承認、却下等)。
String	stepCode	イベントハンドラの実行ステップのコード。
Function	func	イベントハンドラ

説明:
指定条件に一致したイベントハンドラを削除します。

setEditable(caption, stepCode, enabled)

戻り値:
void

引数:

String caption 回付ルールで設定した操作メニュー文言(承認、却下等)。



String stepCode イベントハンドラの実行ステップのコード。
未指定の場合は全てのステップでイベントハンドラが実行される。



boolean enabled True:書類の強制的な編集保存を有効化する False:書類の強制的な編集保存を無効化する

説明:
指定したステップの特定操作を行った場合に、書類の編集モードに関わらず操作後に書類を編集保存するかどうかを指示します。
addEvent で登録したイベントハンドラによって書類内容の変更を行なう場合に、書類の編集モードに関わらずに変更内容を保存するために使用します。

以下のステップと操作メニューの組み合わせにおいて動作します。

▼ステップと操作メニュー
・「作成ステップ」における「送信」
・「作成ステップ」における「代理申請」
・「申請ステップ」における「申請」
・「承認ステップ」における「承認」
・「承認ステップ」における「コメント付き承認」

承認メニュー押下時に、あるフィールドに値が入っていない場合はエラーとする場合の例です。この処理をフォームの onLoad イベント時に実行するようにします。

```
WorkflowEvent.addEvent("承認", "APPROVE_STEP", function() {
    var value = XWEB.getObject("textfield1").getValue();
    if (!value) {
        alert("textfield1 に値を入力してください");
        return false;
    }
    return true;
});
```

定義済みの関数を指定する場合は以下のように記述します。

```
function validateApprove() {
    var value = XWEB.getObject("textfield1").getValue();
    if (!value) {
        alert("textfield1 に値を入力してください");
        return false;
    }
    return true;
}

WorkflowEvent.addEvent("承認", "APPROVE_STEP1", validateApprove);
WorkflowEvent.addEvent("承認", "APPROVE_STEP2", validateApprove);
```

PDF メニュー押下時に全てのステップで JavaScript の確認ダイアログを表示し、OK ボタンを押下した際にのみ PDF 出力を行う場合の実装例です。この処理をフォームの onLoad イベント時に実行するようにします。

```
WorkflowEvent.addEvent("PDF", null, function() {
    return confirm("PDF を出力しますか？");
});
```

承認メニュー押下時に、あるフィールドの値を変更する場合の例です。
WorkflowEvent.setEditable と WorkflowEvent.addEvent を onLoad イベント時に実行するようにします。
setEditable を実行しない場合、編集モードではない状態で書類操作を行うと、変更したフィールド値は保存されません。

```
WorkflowEvent.setEditable("承認", "APPROVE_STEP", true);

WorkflowEvent.addEvent("承認", "APPROVE_STEP", function() {
    XWEB.getObject("textfield1").setValue("APPROVED");
    return true;
});
```

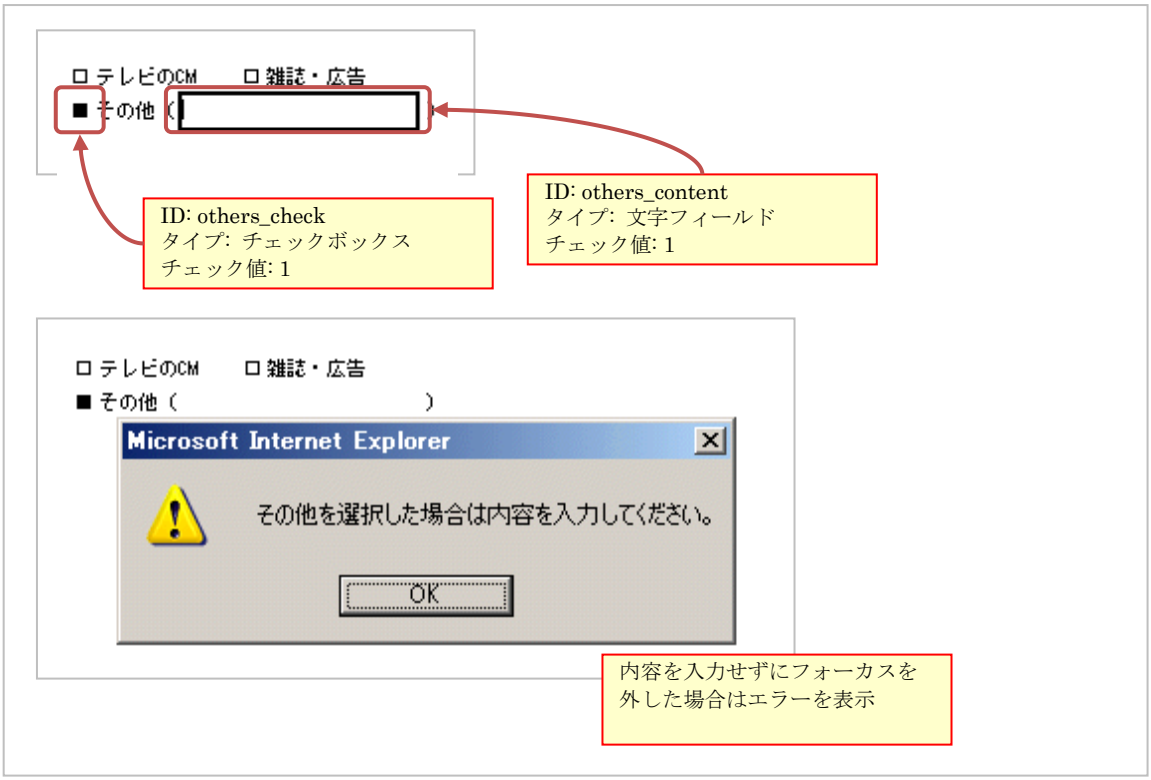
4.3. X-WebForm でプレビューする際の注意事項

X-WebForm JS API を利用した JavaScript 実装は、X-WebForm プレビュー時の操作で動作確認できますが、
AgileWorks JS API を利用した JavaScript 実装は、X-WebForm プレビュー時の操作で動作確認できません。
WorkflowEvent の動作確認や、WorkflowInfo を利用した情報取得等の動作確認は、AgileWorks 管理サイトへアップロードし、ユーザーサイトへ公開設定した後に確認する必要があります。

5. 実装サンプル

5.1. 条件付き入力チェック

以下のようなフィールドを配置した場合に、「その他」が選択された場合は必ず内容させる場合の実装サンプルです。



以下の JavaScript を others_content の onBlur アクションに定義します。

```
// フィールドの取得
var others_check = XWEB.getObject("others_check");
var others_content = XWEB.getObject("others_content");

// 「その他」が選択されており、かつ内容が入力されていないかどうか
if (others_check.getValue() == 1 && others_content.getValue() == "") {
    // メッセージを表示
    alert("その他を選択した場合は内容を入力してください");
}
```

5.2. 編集不可状態の切り替え

「5.1 条件付き入力チェック」の例において、内容の編集不可状態の制御を行う場合の実装サンプルです。

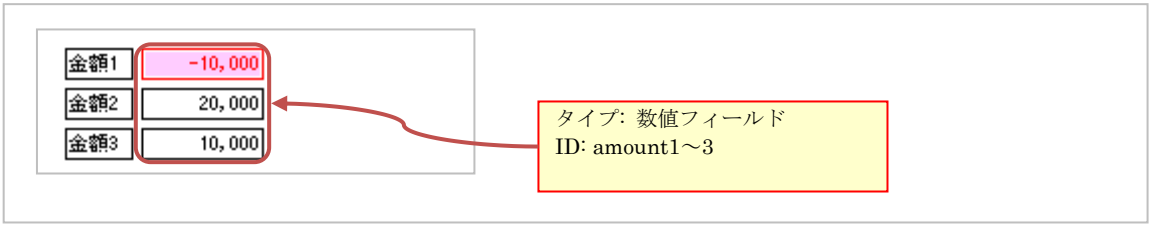
以下の JavaScript を others_check の onChange アクションに定義します。

```
// フィールドの取得
var others_check = XWEB.getObject("others_check");
var others_content = XWEB.getObject("others_content");

if (others_check.getValue() == 1) {
    // その他が選択されているので編集可とする
    others_content.setReadOnly(false);
} else {
    // その他が選択されていないので、編集不可とし値をクリアする
    others_content.setReadOnly(true);
    others_content.setValue("");
}
```

5.3. フィールドの色の切り替え

入力した金額の正負で文字色、背景色、枠線色を切り替える場合の実装サンプルです。



以下の JavaScript を amount1 の onChange アクションに定義します。amount2、amount3 についても 1 行目の ID を変更して同様に定義します。

```
var amount = XWEB.getObject("amount1");

// フィールドの取得
if (amount.getValue() < 0) {
    // 値が負数の場合は色を変更する
    amount.setForegroundColor("rgb(255,0,0)");
    amount.setBackgroundColor("rgb(255,204,255)");
    amount.setRectColor("rgb(255,0,0)");
} else {
    // 値が整数の場合は色を元に戻す
    amount.setForegroundColor("");
    amount.setBackgroundColor("");
    amount.setRectColor("");
}
```