

# WinActorからExcelのマクロを非同期で実行するサンプルシナリオ

WinActorのエミュレーションを使用してExcelのマクロを非同期で実行するサンプルシナリオになります。

シナリオからExcelのマクロを実行し、エラーが発生した際にシナリオが止まることなく継続して処理することができます。

## サンプルシナリオ概要

目次
はじめに
本サンプルシナリオの概要
使用するWinActorモジュール
動作確認環境
本サンプルシナリオの使い方
Step 1. Office(Excel)を確認
Step 2. サンプルシナリオの読み込み
Step 3. 動作に必要な変数を設定
Step 4. エラー判定までの待機時間を設定
Step 5. サンプルシナリオの実行
Step 6. サンプルシナリオの終了後の確認
トラブルシューティング
サンプルシナリオ解説
1.マクロの非同期実行
2.正常終了判定
変更履歴

## はじめに

WinActorのエミュレーションを使用してExcelのマクロを非同期で実行するサンプルシナリオ(以降、本サンプルシナリオ)について説明します。

本サンプルシナリオはマクロの非同期実行と特定のエラー・正常終了判定を行います。

本ドキュメントでは「Excelマクロを非同期で実行(エラー判定あり).ums6」について説明しており、マクロの非同期実行のみを行うシナリオ「Excelマクロを非同期で実行(エラー判定なし).ums6」も同梱してます。

## 本サンプルシナリオの概要

本サンプルシナリオは、以下の処理を行います。

実行したいマクロが登録されているExcelファイルを開きます。

エミュレーションを利用して指定されたマクロを実行します。

指定時間待機します。

待機完了後、エラーダイアログが表示されているか判定を行います。

設定されている場合は正常終了ダイアログが表示されているか判定を行います。

判定結果によって処理を分岐させ、それぞれに指定した処理を実行します。

## 使用するWinActorモジュール

本サンプルシナリオでは、以下のWinActorモジュールを使用します。

ノード/アクション/ウィンドウ状態待機

ノード/アクション/指定時間待機

04\_自動記録アクション/エミュレーション.ums6

14\_入力欄操作/エミュレーションで文字送信.ums6

18\_Excel関連/01\_ファイル操作/Excel開く(前面化).ums6

## 動作確認環境

本サンプルシナリオは以下の環境で動作確認しています。

Windows 10

WinActor 6.3.0、7.1.0

Microsoft Excel 2019

## 本サンプルシナリオの使い方

本サンプルシナリオの実施方法は以下の通りです。

### Step 1. Office(Excel)を確認

本サンプルシナリオは、WinActorが動作するPCとOfficeに含まれるExcelを用いて、マクロ実行するシナリオです。  
手元のPCで動作が確認できるように、上記のアプリがインストールされているか、確認してください。

### Step 2. サンプルシナリオの読み込み

ダウンロードしたサンプルシナリオのZIPファイルを展開し、展開したフォルダに含まれている本サンプルシナリオをWinActorで読み込みます。

### Step 3. 動作に必要な変数を設定

本サンプルシナリオを実行するために必要な値を、変数一覧で設定します。

事前に設定が必要な変数は下記に記載しております。

#### ファイル名

実行するマクロが登録されているファイル名を、絶対パスまたは相対パスで指定してください。

例) 正常パターンマクロ.xlsm

#### マクロ名

実行するマクロ名を指定してください。

例) Sample1

#### 正常終了判定

正常終了判定をする場合は、1を指定してください。

※正常終了判定をする場合はサンプルシナリオ解説の[2.正常終了判定](#)に記載の手順で正常終了時のダイアログを設定する必要があります。

例) 1

変数一覧						
グループ名	変数名	現在値	初期化しない	初期値	コメント	
変更可能変数	ファイル名		<input type="checkbox"/>		【必須】実行するマクロが登録されているファイル名を、絶対パスまたは相対パスで指定してください。	
	マクロ名		<input type="checkbox"/>		【必須】実行するマクロ名を指定してください。	
	正常終了判定		<input type="checkbox"/>		【任意】正常終了確認をする場合は、1を指定してください。	
変更不可変数	コンパイルエラー判...		<input type="checkbox"/>			
	VBエラー判定結果		<input type="checkbox"/>			
	正常終了判定結果		<input type="checkbox"/>	true		

変数一覧画面

### Step 4. エラー判定までの待機時間を設定

マクロ実行開始から、エラー判定処理開始までの待機時間を実行するマクロの処理時間に合わせて「指定時間待機ノード」に直接設定してください。

初期値は10秒です。

プロパティ - 指定時間待機

名前	指定時間待機
コメント	

☒ 指定した時間待つ

待機時間  ミリ秒

☐ 指定した時刻まで待つ

時刻

☐ 指定時刻チェック

時刻

チェック結果

時刻の入力例

開始時刻	13:00
開始日時	2012/04/01 13:00
開始・終了時刻	09:00-18:00
開始・終了日時	2012/04/01 13:10-2012/04/03 00:00

日付形式

タイムゾーン

OK キャンセル

### 待機時間設定

## Step 5. サンプルシナリオの実行

WinActorで本サンプルシナリオを実行します。

同梱されている正常パターンマクロ.xlsmを指定した場合、10秒後にExcelからダイアログが表示され、マクロが正常終了します。

エラーパターンマクロ.xlsmを指定した場合、マクロでエラーが発生します。

## Step 6. サンプルシナリオの終了後の確認

実行結果として、判定結果に従ってメッセージが表示されていることを確認してください。

WinActor

マクロ実行は正常に終了しました。

OK

### 正常終了メッセージ

WinActor

マクロ実行中にエラーが発生しました。

OK

## トラブルシューティング

本サンプルシナリオでは、変数指定が不十分であった場合、エラーメッセージが出力されます。

詳細については、[エラーメッセージ一覧 \(https://winactor.biz/samplescenario/errmsg\\_10953.html\)](https://winactor.biz/samplescenario/errmsg_10953.html)を参照してください。

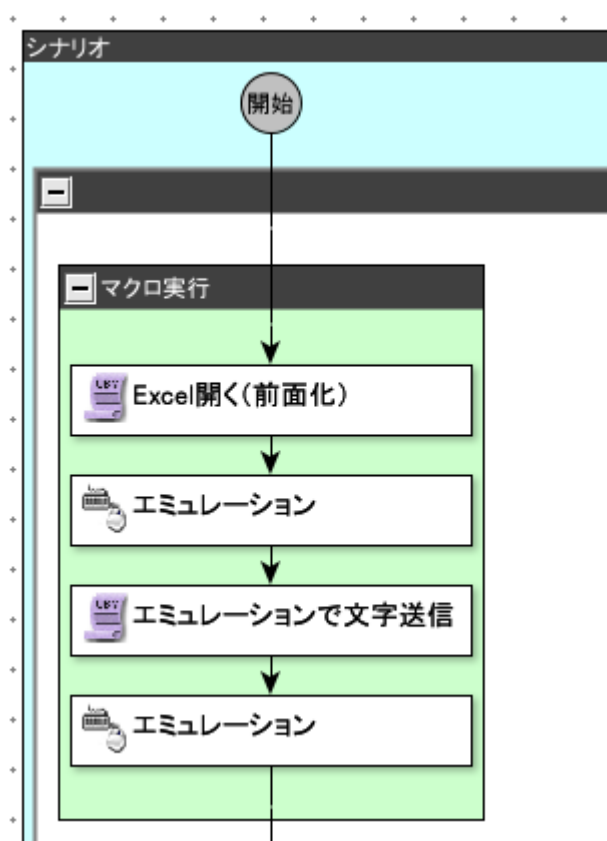
## サンプルシナリオ解説

本サンプルシナリオで使用している実装テクニックについて解説します。

### 1. マクロの非同期実行

本サンプルシナリオのマクロ実行は、「エミュレーション」ノードを使用したキー操作で実現しています。

以下の操作でマクロ実行を開始し、処理の完了を待たずにシナリオを進めることが可能です。



マクロ実行シナリオ

### 2. 正常終了判定

本サンプルシナリオの正常終了判定は、「ウィンドウ状態待機」ノードの状態取得機能活用することで実現しています。

以下のように、ウィンドウ識別名に正常終了時に出力されるダイアログを予め設定してください。

ダイアログタイトルやOKボタンのレイアウトなどが変更されると正常に判定できない場合があるので、都度設定しなおしてください。

正常終了判定を行わない場合はマクロ名が誤っていた場合でもエラー判定となりません。

プロパティ - ウィンドウ状態待機

名前

正常終了確認


コメント

取得結果

正常終了判定結果

ウィンドウ識別名

MicrosoftExcel



画面の変化

画面が表示されるまで

状態取得のみ

タイムアウト

10,000

ミリ秒

OK

キャンセル

正常終了確認用ノード

変更履歴

版数
1.0版
日付
2020/10/15
修正内容
初版

版数
1.1版
日付
2024/1/15
修正内容
<div><div>・注意事項を削除</div><div>・ファイル名を「SS2009_10953_ExcelOperationRunMacroAsynchronously_1.0.1.zip」に変更</div></div>