UiPathではじめるRPA入門（１） 〜 概要編

* 2017年11月17日

* [RPA](https://tutorial.co.jp/category/rpa/)

このシリーズでは、RPAソフトウェアの使い方を概説します。選定したソフトウェアは、無料で使えるコミュニティエディションを提供しているUiPathです。UiPathでは、UiPath Academyという無料のEラーニングを提供していますが、コースが長くちょっと試すという分にはかなりの覚悟が必要です。

そこで、一日か二日くらいで大まかな使い方を身に付けることができるように記事をまとめてみようと思います。何らかのプログラミング言語を多少やったことがあるという読者を想定しています。そうでない読者の方でも一通り完了できるレベルではありますが、繰り返し制御、条件分岐、変数、型、エラー処理、クラスなどには困難を感じると思います。

**今回の内容**

* UiPathとは？

では早速、UiPathとは何かの概要を説明します。

**UiPathとは？**

UiPathは、一連のRPAソフトウェアを提供している企業です。今回は、その中のUiPath Studioを用いてロボットの設定を体験します。

**UiPathのRPAソフトウェアの種類**

UiPathが提供するRPAソフトウェアには、幾つかの種類があります。ややこしいと感じるかもしれませんが、一通りロボットの説明をしていきます。

**Attended Robot**

Attended Robot（アテンディットロボット）は、主に従業員のデスクトップ内にインストールし、従業員の操作により動作するロボットです。従業員とともに動くロボットになります。

**Unattended Robot**

Unattended Robot（アンアテンディットロボット）は、サーバー内のバーチャルマシン上で（も）動作し、従業員による操作が不要なロボットです。独立して常時稼働が可能なロボットはこのUnattendedになります。Unattendedを導入することで、よりRPAの効率を高めることができます。

**Orchestrator**

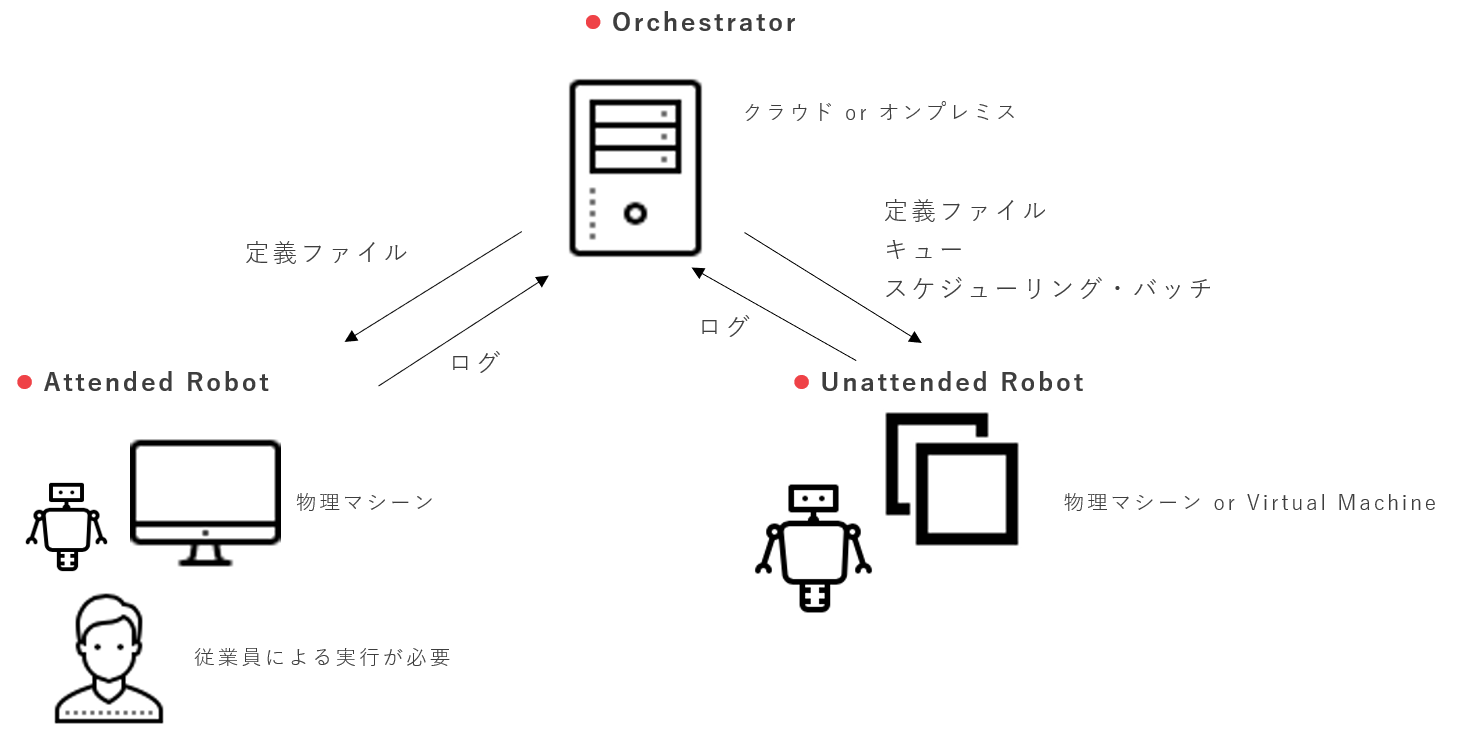
Orchestratorは、Unattended Robot/Attended Robotの定義ファイルのバージョンや稼働状況を管理するサーバーソフトウェアです。NTTデータ社のWinActorで、管理ロボと呼ばれているものもこのOrchestratorに該当します。

Unattended Robotとの連携により効果を発揮し、Unattended Robotへのキューイング、起動、バッチ処理の管理ができます。はじめは導入せずに、ロボットの数が、増えてきたときに導入することも可能です。

**UiPath Studio**

UiPath Studioは、ロボットを設定するためのソフトウェアです。Unattended Robot/Attended Robot共に設定が可能です。動作検証用のロボットが内蔵されています。ロボットを定義するには、このUiPath Studioが必ず必要になります。

**UiPath ロボットの種類の概要**

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_overview/attachment/uipath-robots/)

**RPA構築プロセスと購入ライセンス**

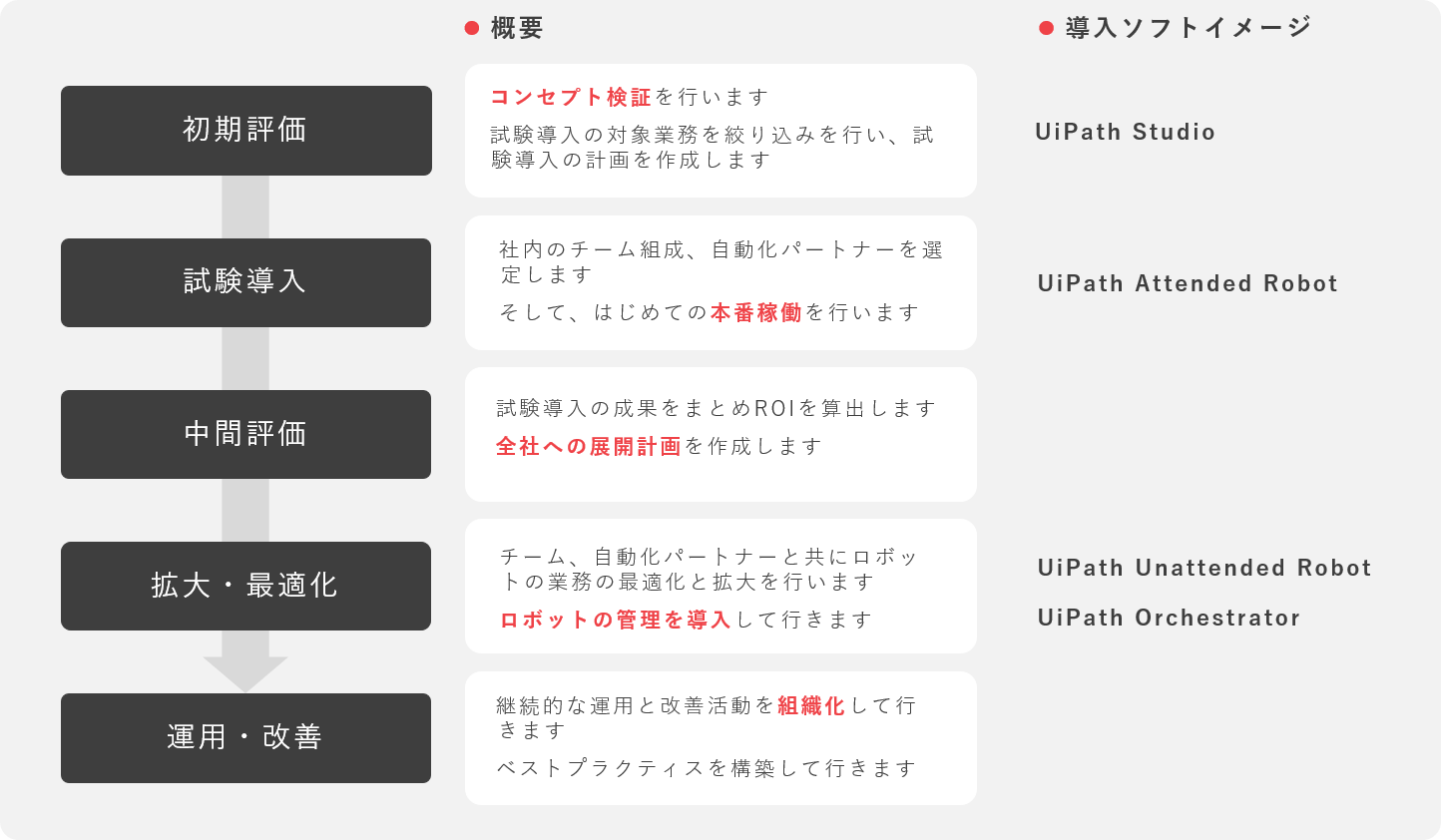
初期評価を行い、部分的な自動化から試験導入を行います。 試験導入後は、段階的に自動化範囲を拡げ運用フェーズへ移行していきます。

初期評価段階では、UiPath Studioの無料の評価版で十分です。60日間のトライアルとなっております。

実際の業務でパイロット導入するときには、Attended RobotかUnattended Robotのライセンスが必要になります。これに加えて、UiPath Studioのエンタープライズライセンスを購入するのもこのタイミングと言えます。Attended Robot１ライセンスにStudio１ライセンスが最小構成となってります。

続いて、中間評価、拡大最適化（全社展開）となると、ロボットの台数の検討や管理の方法の構築に加え、Orchestratorの導入も現実味を帯びてきます。

大規模化を前提とする場合は、初期評価の後は計画的に展開するのが重要です。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_overview/attachment/rpa-process-and-licenses/)

**無料のUiPath Community Edition**

UiPathでは、無料のコミュニティエディションがあります。コミュニティエディションは、小規模事業者や個人の勉強、評価用のもので、UiPath StudioとAttended Robotが使用できます。詳細なライセンス条件は、本家サイトでご確認ください。

このRPA入門ブログでは、UiPath Studioを使っていますが、読者のみなさまは無料のコミュニティエディションで大丈夫です。

[UiPath Community Editionはこちらからダウンロード](https://www.uipath.com/ja/community)

次回より、実際にUiPathを利用して自動化を設定していきます。

**復習**

* UiPathのAttended Robot、Unattended Robot、Orchestratorの役割について説明して見てください

では、また次回

UiPathではじめるRPA入門（２）〜 こんにちは、世界

* 2017年11月21日

* [RPA](https://tutorial.co.jp/category/rpa/)

今回は、UiPathで初めてのプロジェクトを作成してみます。

**今回の内容**

* 新規プロジェクトの作成
* ロボットからのメッセージダイアログの表示（こんにちは、世界）
* アクティビティ、ワークフロー、プロパティの概念
* 変数の簡単な使用方法

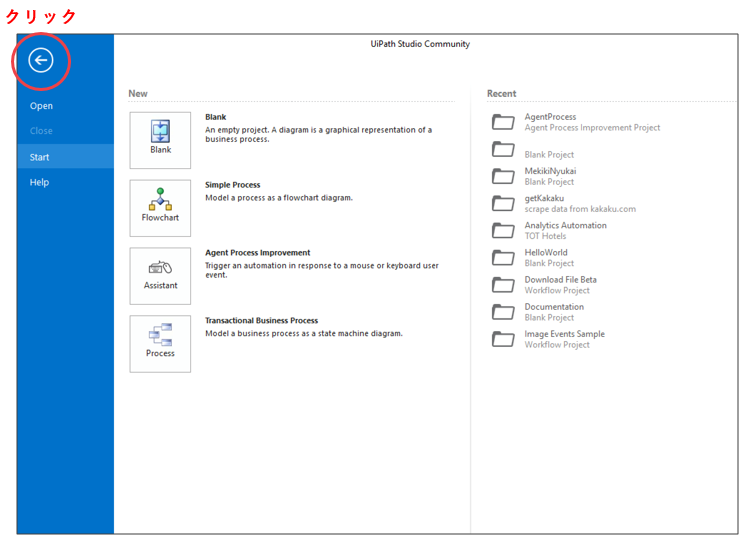
では早速はじめましょう。

[U](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_overview/)[iPathではじめるRPA入門（1）](https://tutorial.co.jp/rpa/2017/11/uipath-overview/)の記事では、RPAの概要と、RPAの代表的ソフトウェアであるUiPathのインストール方法について説明しました。こちらの記事では、[RPA入門 (1)](https://tutorial.co.jp/rpa/2017/11/uipath-overview/)を完了した方を対象に、UiPathのインターフェースの機能説明、及びアクティビティと呼ばれるロボットの基礎的な設定方法について説明していきます。

**インターフェース**

**起動直後の画面**

起動直後は、Start画面が表示されています。とりあえず、「←」ボタンをクリックしてインターフェースの概要をみてみましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface/)

**インターフェース〜全体像**

設定画面の全体は、幾つかのパネルで構成されています。

**アクティビティパネル**

ロボットの動作の最小単位が一覧されています。ここからロボットの動作を選択していきロボットを定義します。

**リボン**

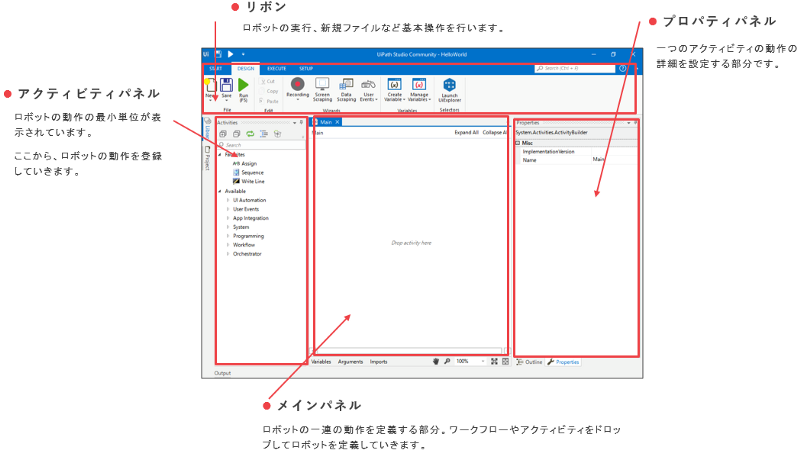
ロボットの実行、新規ファイルの作成、保存などの基本的な操作が一覧されています。

**プロパティパネル**

一つのアクティビティの動作の詳細を設定する部分です。

**メインパネル**

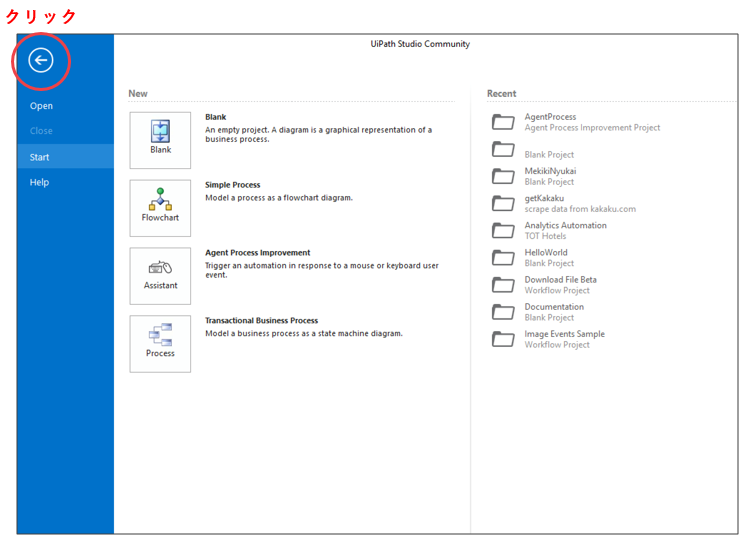
ロボットの一連の動作を定義するところです。ワークフローやアクティビティをドロップしてロボットを定義していきます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_2/)

**インターフェース〜リボン > Start**

**Start**

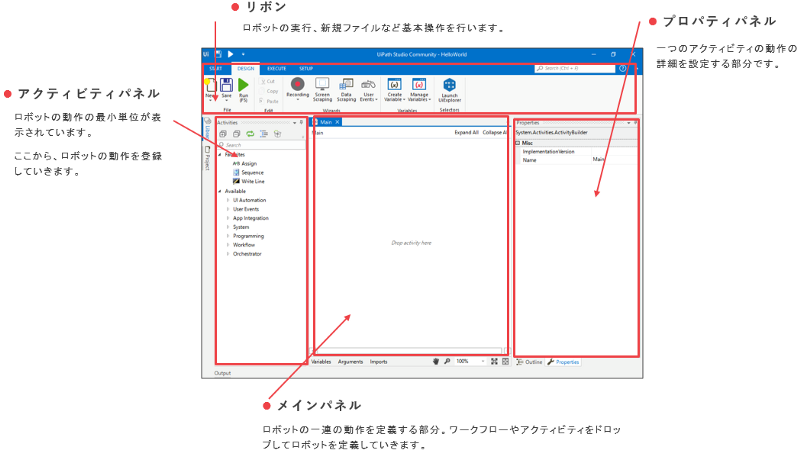
新規プロジェクトを作成するか、最近開いたプロジェクトから作業を開始します。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface/)

**インターフェース〜リボン > Design**

**Design**

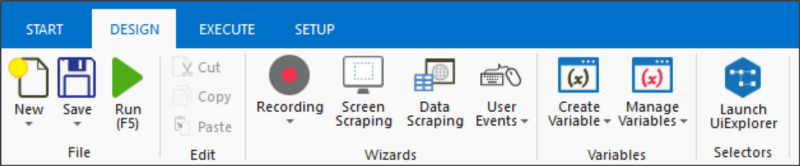
ファイル操作、よく使う機能のウィザード、変数の管理などワークフローを設定する際によく使われる機能にアクセスします。設定中に主に使うメニューになります。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_2/)

**インターフェース〜リボン > Execute**

**Execute**

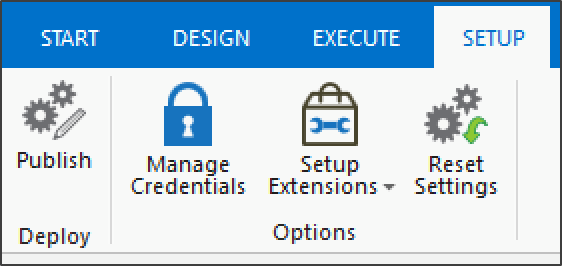
主にデバッグで使うメニューです。プロジェクトの実行からデバッグ実行、ブレークポイントなどデバッグに必要な操作にアクセスします。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_4/)

**インターフェース〜リボン > Setup**

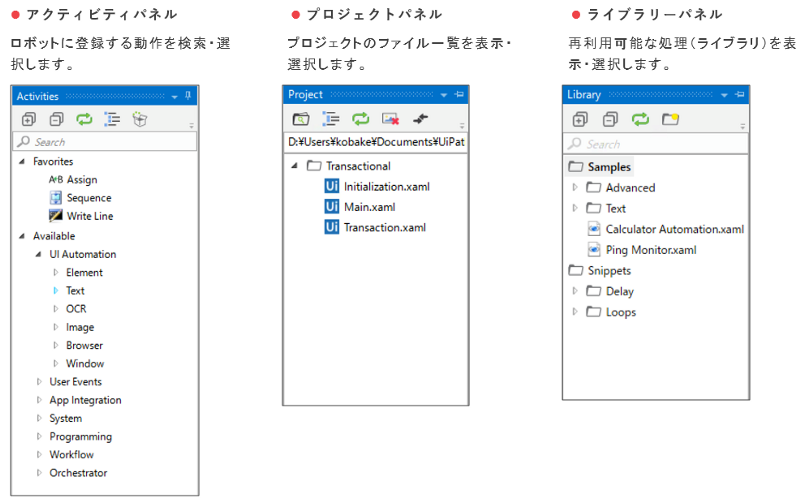
**Setup**

出来上がったロボットのパッケージング、Webサイトのログインなどの認証情報の管理、機能拡張のインストールなどにアクセスします。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_6/)

**インターフェース〜アクティビティパネルほか**

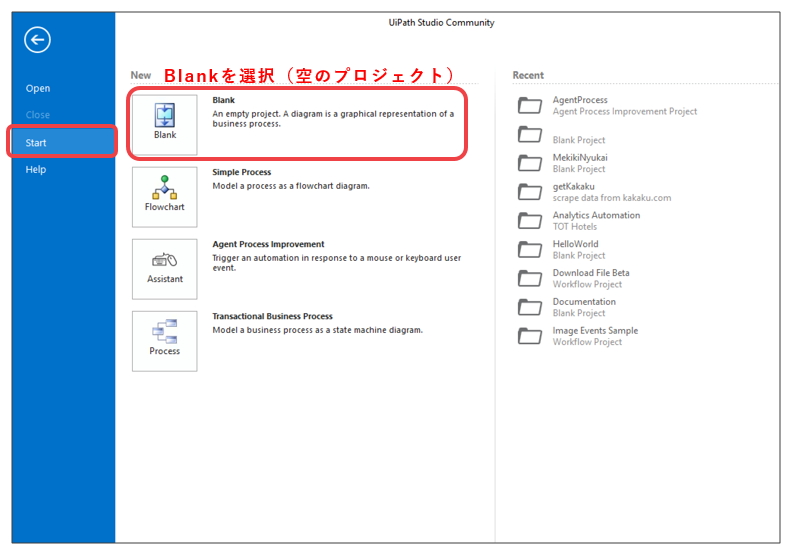
アクティビティパネルの他、プロジェクトのファイル一覧を表示するプロジェクトパネル、再利用可能な処理（ライブラリ）を表示・選択できるライブラリーパネルがあります。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_7/)

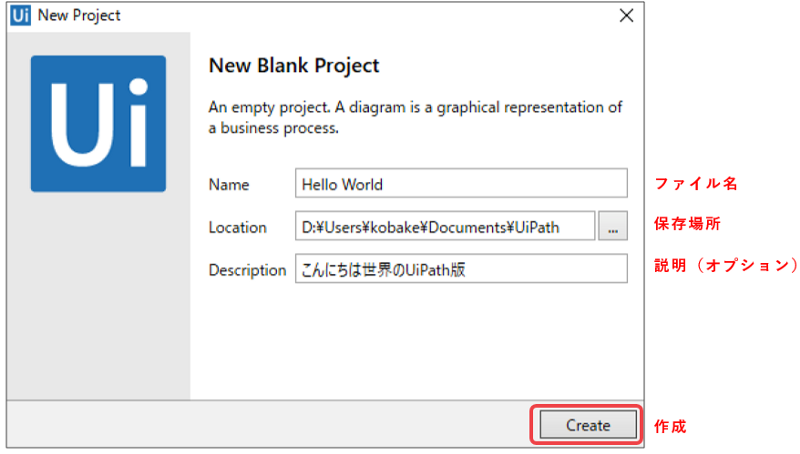
**新規プロジェクトの作成**

さて、早速新規プロジェクトを作成してみましょう。

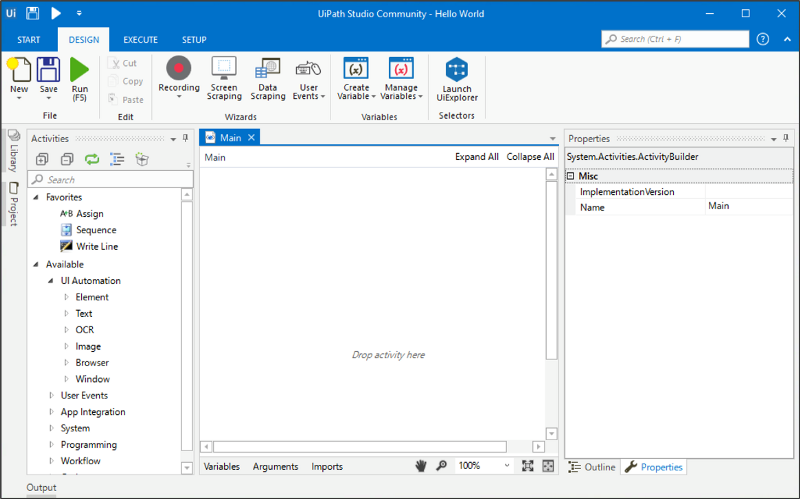
リボン > Start > New から Blank を選択します。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_8/)

ダイアログからファイル名、保存場所、説明を記入し、「Create」します。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_9/)

空のプロジェクトの画面が表示されます。これで新規プロジェクトが作成できました。

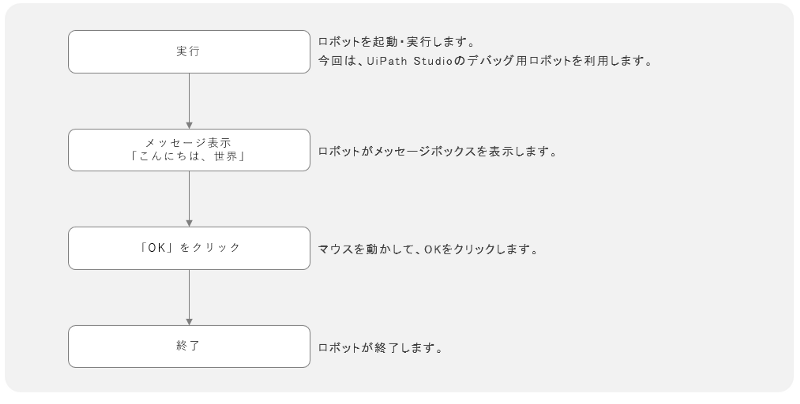
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_10/)

**こんにちは、世界**

**概要**

次に、「こんにちは、世界」とダイアログを表示するだけの簡単なロボットを作成してみましょう。

**作成するロボット**

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_11/)

**手順の解説**

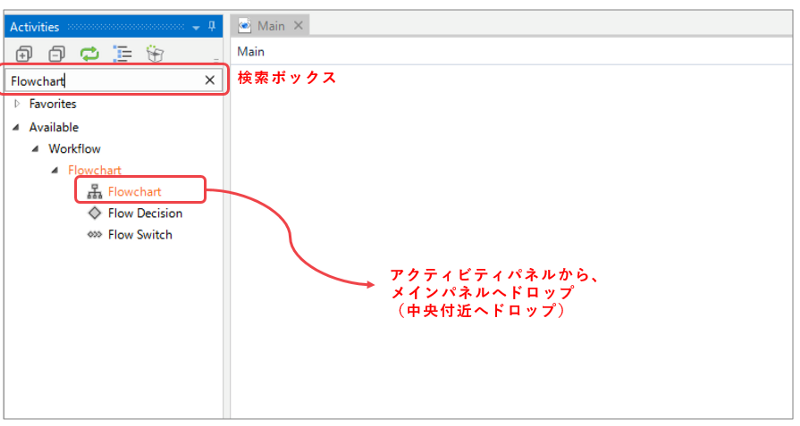
**1. フローチャートの追加**

まずは、プロジェクトにフローチャートを追加します。

アクティビティパネルの検索ボックスに「Flowchart」と入力してください。 検索結果の中に、ハイライトされた「Flowchart」というものがあるのでそれをメインパネルにドロップします。

＊Flowchartの他、Sequenceなども利用できますので試してみてください。

ここでのポイントは、メインパネルの中央付近にドロップすることです。端っこの方にドロップすると反応しない場合があります。反応するかしないかはドラッグ中のポインタのアイコンで判別することができます。

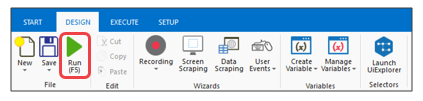
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_12/)

**2. メッセージボックスの追加**

ドロップするとフローチャートがハイライトされた状態で追加されます。

内部の「Start」というアイコンは、フローチャートがここから実行開始されることを示します。

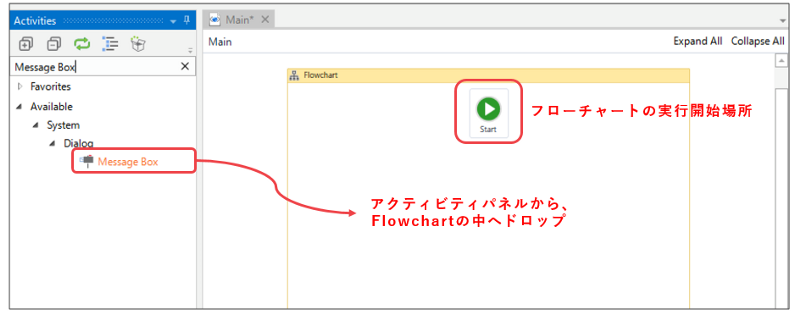
この状態でも、空のロボットとして実行可能です（リボンのRunボタン or F5 で実行してみてください）。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_13/)

実行できましたが、何も起こらないですね。次にアクティビティを追加していきましょう。

アクティビティパネルの検索ボックスに「Message Box」と入力します。

一覧に出て来たハイライトされた Message Box をフローチャートにドロップします。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_14/)

**3. メッセージボックスを設定する**

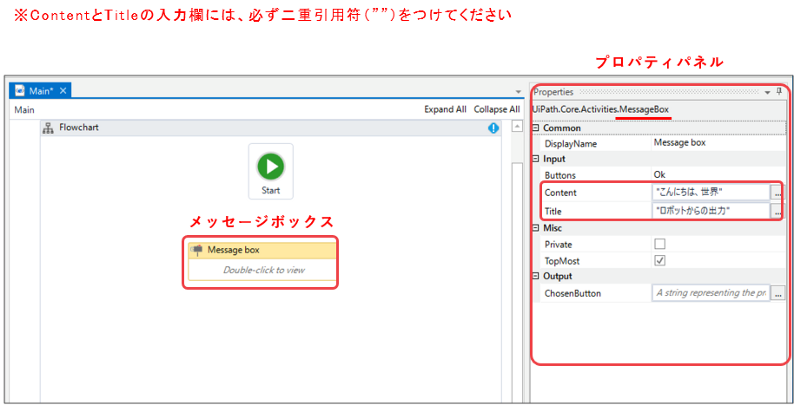
ドロップすると、Message BoxがFlowchartの中に入り、ハイライトされた状態になります。

同時に右側にある、プロパティパネルがMessage Boxになっていることが確認できます。

プロパティパネル内の、Input > Content には、「”こんにちは、世界”」。

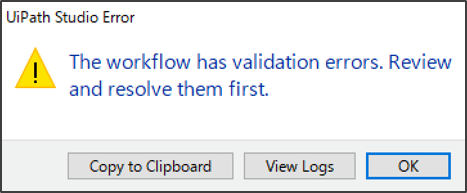
Input > Title には、「”ロボットからの出力”」とでも書いておきましょう。

＊ContentとTitleには、必ず二重引用符（””）をつけてください。これは、コンピューターに入力した値が文字列として認識させるために必要になります。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_15/)

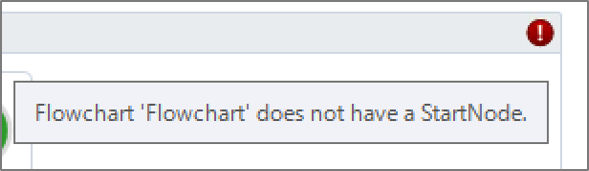
**4. 実行エラーを確認する**

リボン内のRunボタンからロボットを実行すると、エラーが出て実行できないことを確認します。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_16/)

上のようなエラー表示は、設定自体に不足・矛盾などの誤りがある場合に表示され、ロボットが実行できません。OKをクリックして元の画面に戻ります。

Flowchartの右上の赤い（もしくは青い）注意表示にマウスを合わせてみましょう。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_17/)

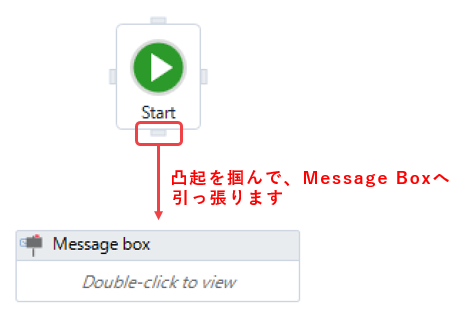
Flowchart ‘Flowchart’ does not have a StartNode.とヒントがでると思います。これは、フローチャートが一番はじめにする処理が定義されていないというエラーです。

この赤い（もしくは青い）注意マークは、残しておくと Run（実行）できません。このマークがでていたらエラーメッセージを確認しましょう。

**5. StartNodeを設定する**

先程のエラーに対応するために、StartNodeを設定しましょう。

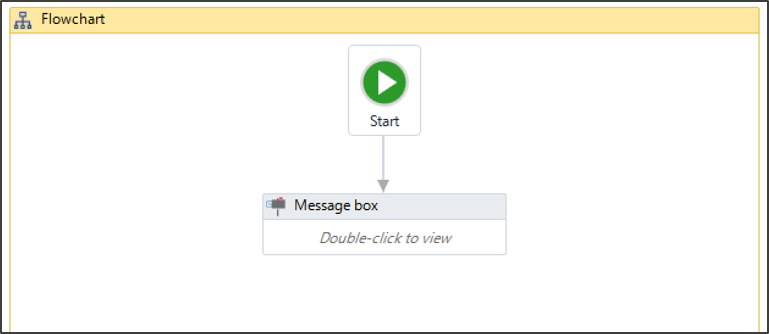
Flowchart内のStartアイコンにマウスオーバーします。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_18/)

マウスオーバーすると、上の絵のようにグレーの四角い凸起がでてきます。下にある一つを引っ張ると線がでてくるので、Message Boxまで引っ張ります。

Message Boxの方も受け入れの凸起がでてくるので、そこへドロップして接続します。

**6. 接続後のフローチャート**

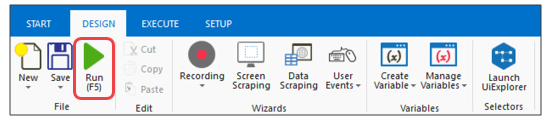
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_19/)

接続すると、上図のようになります。

この時、Flowchartの右上のエラー表示もなくなっていることが確認できますね。

**7. 「こんにちは、世界」ロボットを実行する**

リボン > Design > Runボタンをクリックします

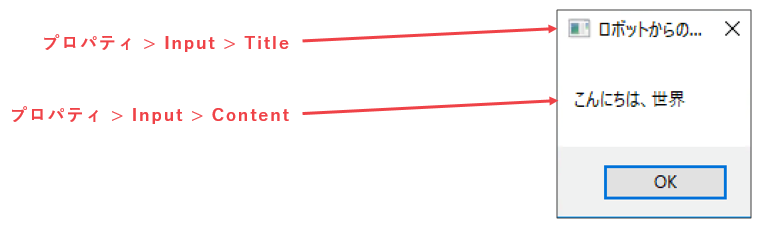
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_20/)

エラー表示もなくなったので、早速実行してみましょう。Runボタンのほか、F5でも実行できます。

メッセージボックスが表示されるので、OKをクリックすればロボットの実行完了です。

メッセージボックスが表示されました。

プロパティパネルでの設定は、メッセージボックスの表示に反映されています。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_21/)

**アクティビティ、ワークフロー、プロパティ**

先程、「こんにちは、世界」と表示するだけのロボットを作成しました。手順について一つずつ復習してみましょう。

**アクティビティ**

ロボットの動作は、アクティビティで定義します。アクティビティは、ロボットに設定可能な定義の全てを含んでいます。

**ワークフロー**

UiPathではワークフローは、アクティビティの一種として登録できます。プロジェクトは必ず上位にワークフローがあります。ワークフローには、Sequence、Flowchart、State Machineなどがありますが、ここでは、Flowchartを最上位に用い、適宜、Sequenceを使って行きます。これらは、互いに入れ子構造にできます。

それぞれの違いを大雑把に説明します。

**Flowchartのイメージ**

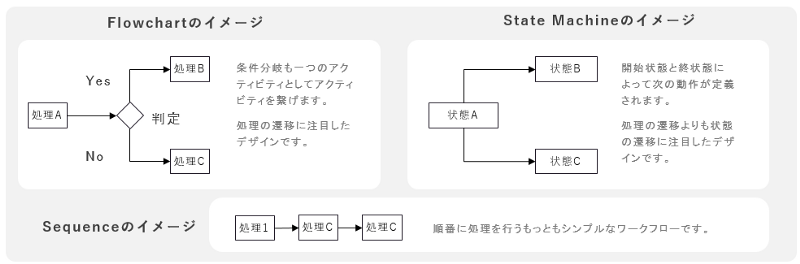
処理の遷移に注目したデザインの方法です。条件分岐、繰り返しなどをつなげてFlowchartを作っていきます。

**State Machineのイメージ**

状態の遷移に注目したデザインになります。開始状態と複数の終状態によって次の動作の定義を変更させます。

**Sequenceのイメージ**

上から順番に処理を行うシンプルなワークフローです。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_22/)

**プロパティ**

プロパティは、アクティビティの具体的な内容が記述されています。

必須プロパティが未定義の場合にはエラーとなります。

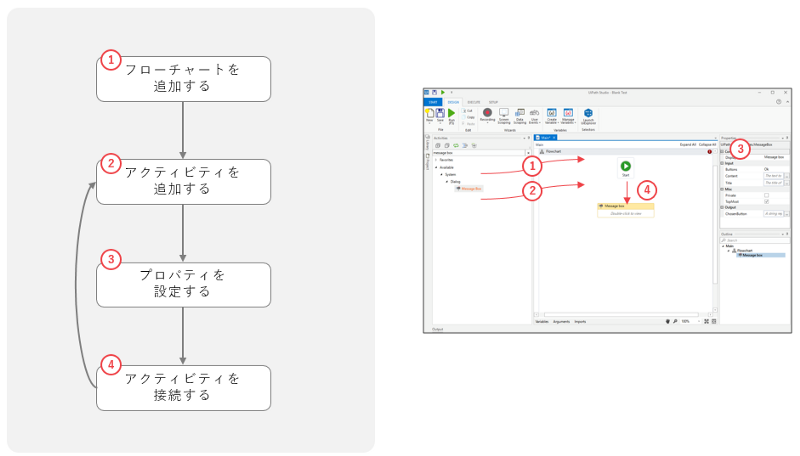
**ロボット定義の流れ**

上記では、「こんにちは、世界」と表示しただけですが、基本的にはこの繰り返しでロボットを定義していきます。

**アクティビティの追加・設定を繰り返すことでロボットを定義していきます**

エラー表示がでていなければ、どの段階でもRunボタンをクリック（もしくは、F5）することでロボットの動作を確認できます。

ステップが大きくなるとロボットを定義するのは困難になってきますが、一つ一つクリアしながら設定していけば難しいことはありません。根気強く設定していってください。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_23/)

**変数の使い方**

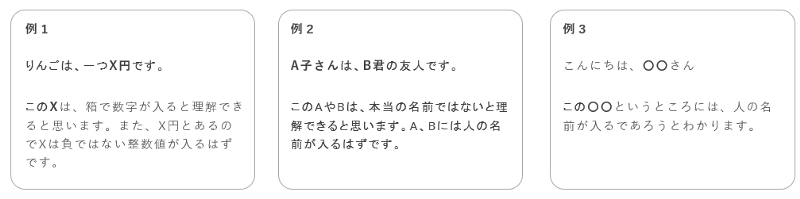
**ロボットの表示をその時々で変えたい時**

先ほど「こんにちは、世界」と表示するだけのロボットに、表示内容を指示するにはどうすればよいでしょうか？表示内容を「こんにちは、佐藤さん」「こんにちは、山田さん」などのようにするだけで、毎回ロボットを作成するのでは大変です。

表示する内容をロボットに入力し、入力された値をロボットが受け取ることができれば同じロボットの汎用性が高まります。この機能を提供するのが変数です。

**変数と型**

変数は、プログラミング経験がある方は特に問題なく扱えると思います。プログラミング経験のない方向けに大雑把に解説します。ここでは変数は一種の箱（入れ物）だと思ってください。箱にも、液体を入れられる箱、冷たいものを入れる箱などいろいろありますが、変数も入れられるものが決まっています。例えば、簡単な例では以下のようになります。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_24/)

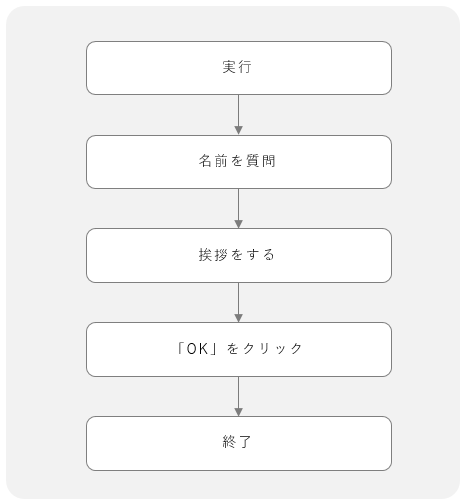
このように、UiPathでも変数には入るものの種類が決まっています。それを、型とよびます。とりあえずは、文字列（String）と整数（Int32）だけ覚えておいてください。この他、だいたいうまい用に変数を扱ってくれるGenericValueという便利な箱もあります。

**ロボットに挨拶させる**

**「こんにちは、世界」を「こんにちは、〇〇」にする**

さて、早速変数を使って、「こんにちは、世界」の世界のところを「こんにちは、〇〇」と変数を使って定義し、入力値によって変更できるようにしましょう。

**作成するロボット**

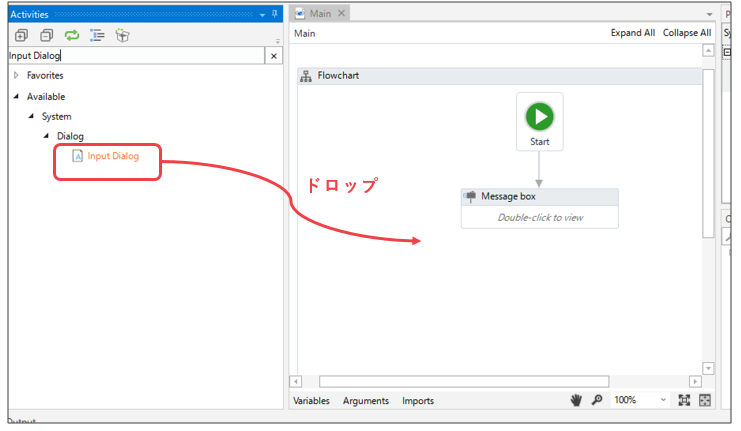
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_25/)

**作成手順**

**1. Input Dialog を追加する**

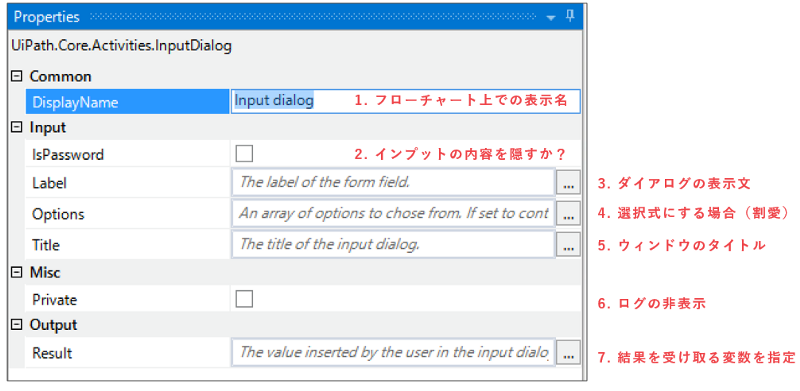
さて、これから〇〇のところに、値を設定したいのですが、ロボットが入力を受け取る簡単な方法が Input Dialog です。これは、ロボットの質問に対して、キーボードから文字列あるいは数値を入力するものになります。

アクティビティパネルの検索ボックスに「Input Dialog」と打ち込んで、Input Dialog をメインパネルへドロップしてください。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_26/)

**2. Input Dialog のプロパティ**

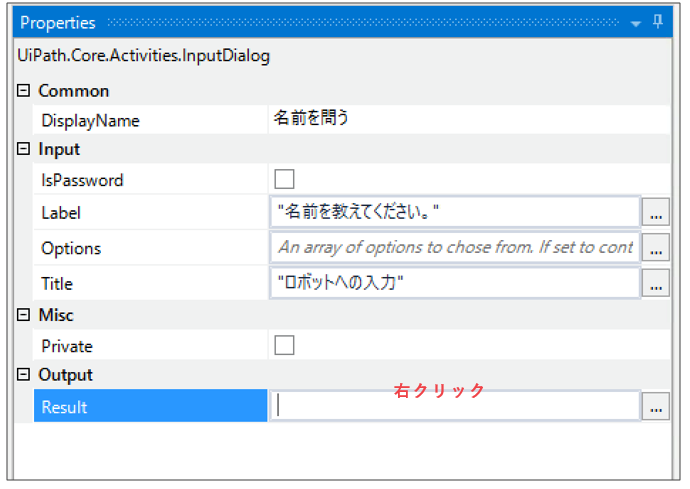
Input Dialog のプロパティは、以下のようになっております。今回は、1. 3. 5. 7. を設定します。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_27/)

**3. Input Dialog のプロパティの設定**

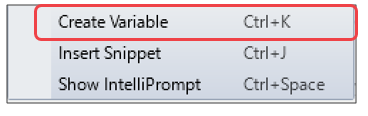
変数以外のところは、以下のように設定してみましょう。

変数のところは、Resultのインプットフィールドのところを右クリックしてください。そうするとこの場で変数を作成することができます。

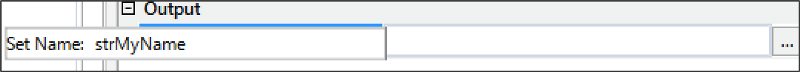
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_28/)

**4. 変数の設定**

右クリックすると、次の表示がでるので、「Create Variable」を選択し、新たに変数を作成します。

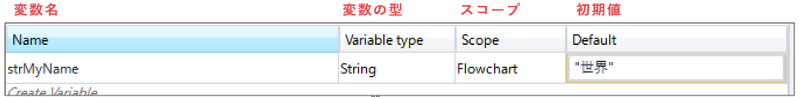
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_29/)

次に、変数名を問われるので、とりあえずstrMyName（引用符不要）と入力してください。名前は基本的にはなんでも良いのですが、社内の規則なども含めてルール化しておくのが望ましいです。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_30/)

入力したら、メインパネル下の「Variables」というタブをクリックして中身を開いてください。その中に先ほど作成した、strMyNameが入っているのが確認できます。

表の見方は、Name, Variable type, Scope, Defaultがそれぞれ、変数名、変数の型、スコープ（変数の有効範囲）、初期値となっております。Defaultのところに、”世界”と入れておきましょう。

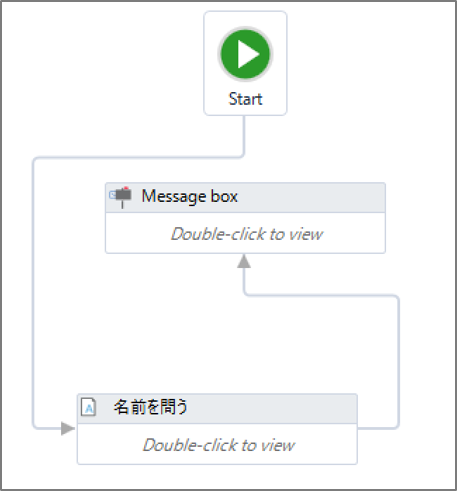
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_31/)

**5. アクティビティを接続する**

さて、Input Dialogの設定ができたので、アクティビティを接続してみましょう。

1. まずは、Message Boxについていた矢印を消します（選択して、deleteキー）
2. 次に、Startから「名前を問う」に矢印を引っ張ります
3. 最後に、「名前を問う」から Message Box に矢印を引っ張ります

最終的に下の図のようになればOKです。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_32/)

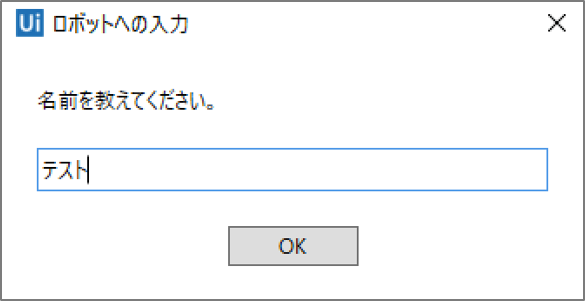
絵的には前後していて綺麗じゃないですね。

各自整形することをおすすめしますが、この状態でも動きます。

**6. ためしに動かしてみる**

この状態で動かしてみましょう。

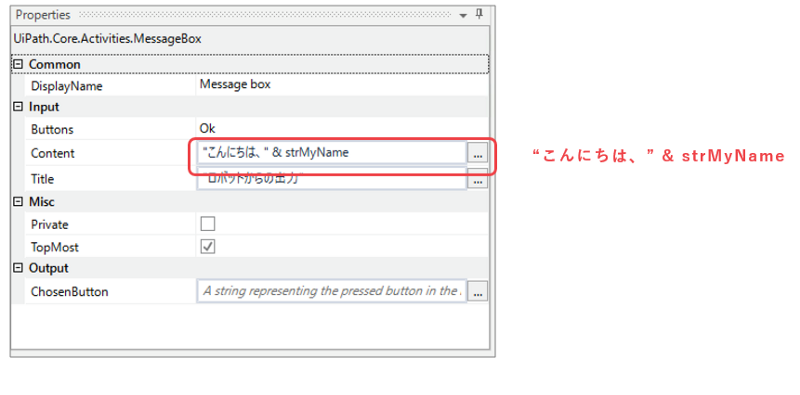
Input Dialog が立ち上がり、名前を聞かれます。ためしなので「テスト」と入力しOKします。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_33/)

相変わらず、結果は、「こんにちは、世界」のままです。 *設定した変数をどこでも受け取っていない* からです。

**7. Message Boxで変数を受け取る**

メッセージボックスのプロパティを見てみると、Contentのところが、 *「”こんにちは、世界”」で固定されています* 。ここに変数をもちいて、 *「”こんにちは、” & strMyName」* としてみてください。変数名があっていれば、strMyNameは、補完候補が表示されます。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_34/)

“”（二重引用符）は、文字列型を作る書き方です。＆は、文字列の結合を意味します。strMyNameは、変数の型でString（文字列）を指定しているので、全て文字列同士となり、＆で結合できます。InputDialogの受け取る変数に、GenericValueを使っている場合は、「&」ではなく、「+」を使ってみてください。

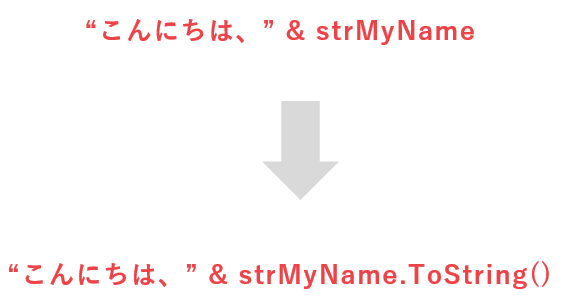
**ここでエラー表示が出る場合**

**型がまちがっている、合わないケースが多い**

先程変数のところで解説しましたが、変数に入れられるものの種類が決まっています。これを型を強制的に String に合わせます。

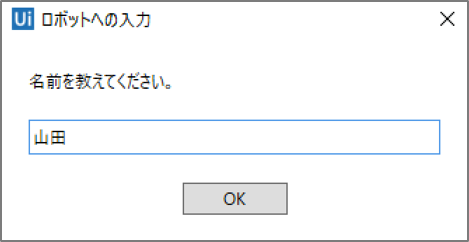
.ToString() という方法を用います。これは、strMyName が ToString()というメソッドを呼び出しているのですが、馴染みのない方は、ToString()で文字列に変換していると考えてください。

なお、GenericValueを用いている場合は、「&」ではなく「+」を使えば大丈夫です。

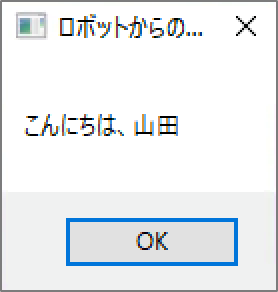
* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_35/)

**8. 修正して、再び動かしてみる**

再びロボットを動かすと、同じように Input Dialog が表示されます。例では「山田」と入れています。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_36/)

OKをすると、狙い通りに「こんにちは、山田」と表示されます。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath_interface/attachment/uipath_interface_37/)

これで、インプットダイアログで設定した任意の名前の挨拶を出力できるようになりました。

**UiPathの情報を調べる**

UiPathは、Web上で簡単に情報を仕入れる事ができます。

**アクティビティを調べる**

<https://www.uipath.com/activities-guide>

上記URLからアクティビティを検索、調べることができます。ただし、英語なので、わからない場合は、がんばってください。日本語に対応しました。

**アクティビティのプロパティを調べる**

<https://www.uipath.com/activities-guide>

同様に上記URLのアクティビティのリンク先から情報を仕入れることができます。  日本語に対応しました。

**エラー表示を調べる**

エラーの多くは、VB.NETの暗黙的型、あるいは、UI要素が見つからないという場合です。VB.NETの場合は、<https://msdn.microsoft.com/ja-jp/library/dd370624.aspx> より情報を検索できます。ただし、ハードルは高めです。

また、Stack Overflowでも幾らかの情報がえられますので参考にしてください。

**復習**

* 新規プロジェクトを作成してみてください
* ロボットからのメッセージダイアログの表示のさせ方を説明してください
* アクティビティ、ワークフロー、プロパティは、それぞれどういう関係にありますか
* 変数は、どのような時に使うか説明してください

それでは、また次回。

UiPathではじめるRPA入門（３）〜 レコーディング編

* 2017年11月21日

* [RPA](https://tutorial.co.jp/category/rpa/)

今回は、前回一つ一つ登録していったアクティビティをレコーディングという方法で一気に登録する方法について学習します。

実際に動かしながらレコーディングの方法に慣れていきましょう。

**レコーディングとは何か？**

**異なるレコーディングの種類**

* Basic レコーディング
* Desktop レコーディング
* Citrix レコーディング
* Web レコーディング

**フロー制御**

* ２種類の条件分岐（Decision フロー と If 文）
* 変数への代入（Assign） と 繰り返し（Do While 文）

早速、始めましょう。

[RPA入門（2）](https://tutorial.co.jp/rpa/2017/11/uipath-interface/)の記事では一つ一つアクティビティ登録する方法を学習しました。本記事ではレコーディングという方法で一気に登録する方法について学習します。実際に動かしながらレコーディングの方法を学習してみて下さい。

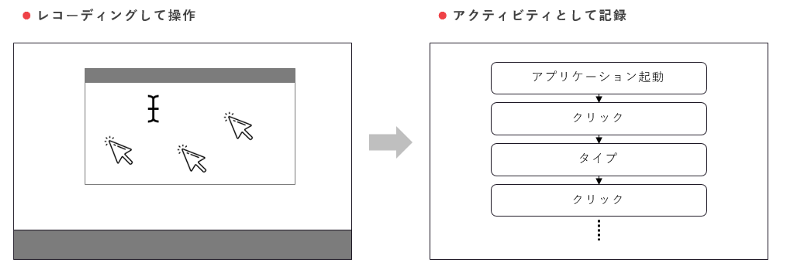
**レコーディングとは？**

UiPathのレコーディング機能とは、あなたがPC画面上で行なった操作を自動的にアクティビティとして記録する手法です。ひとつひとつアクティビティを設定する手間が省けますので、使いこなすことによって、ロボットの設定時間を大幅に短縮することができます。

**アクティビティを自動登録する**

レコーディングについて学びますが、これは、実際にPCを操作して、アクティビティを連続して自動的に登録していくことです。

他のRPAソフトウェアの中には、操作中の画面を動画として記録してアクティビティを作っていくものもありますが、UiPathは、シンプルにアクティビティを自動的に登録してきます。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/1/)

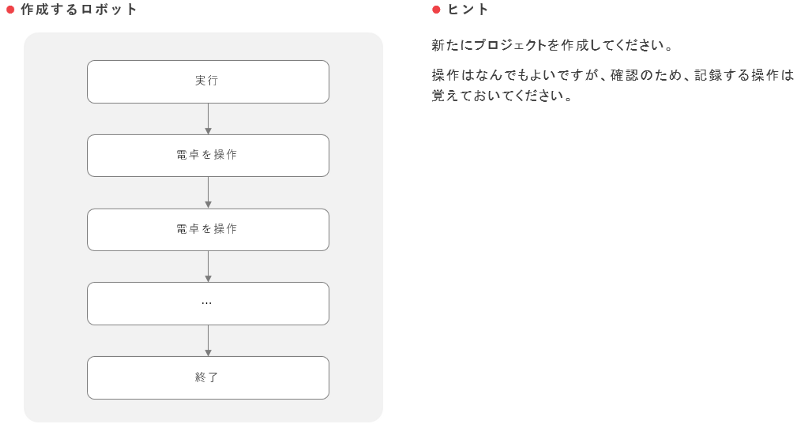
**Basic レコーディングをしてみましょう**

**電卓を操作するロボットを作成します**

電卓を操作起動して、操作させてみましょう。

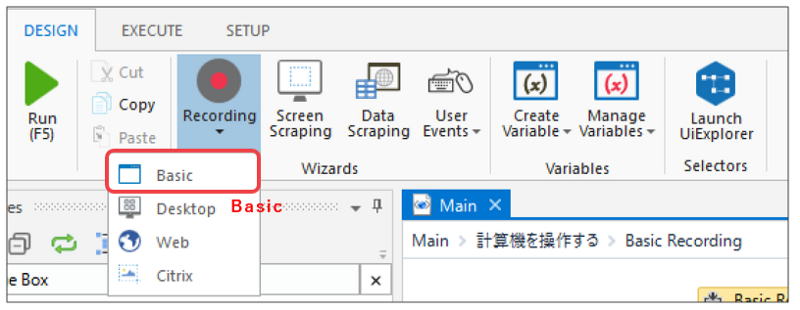
新たにプロジェクトを作成しておいてください。

どんな動作を記録しても良いですが、結果を確認するためどんな動作をさせるかはメモするなど覚えておいてください。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/2/)

**レコーディングの始め方**

リボン > Design ＞ Recording > Basic を選択してください。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/3/)

**レコーディング画面 〜 待機画面**

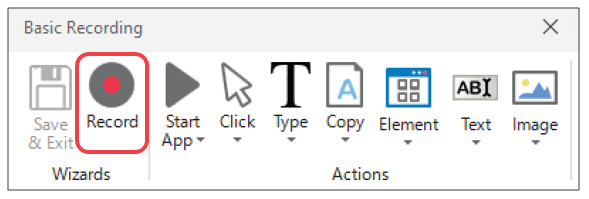
**レコーディング画面**

レコーディングが開始されると、以下のウィンドウが出てきます。

このウィンドウがでている状態では操作は記録されていません。レコーディングの前に準備が必要な場合は、この画面の時に準備しておきましょう。

今回は、予め電卓を起動しておいてください。

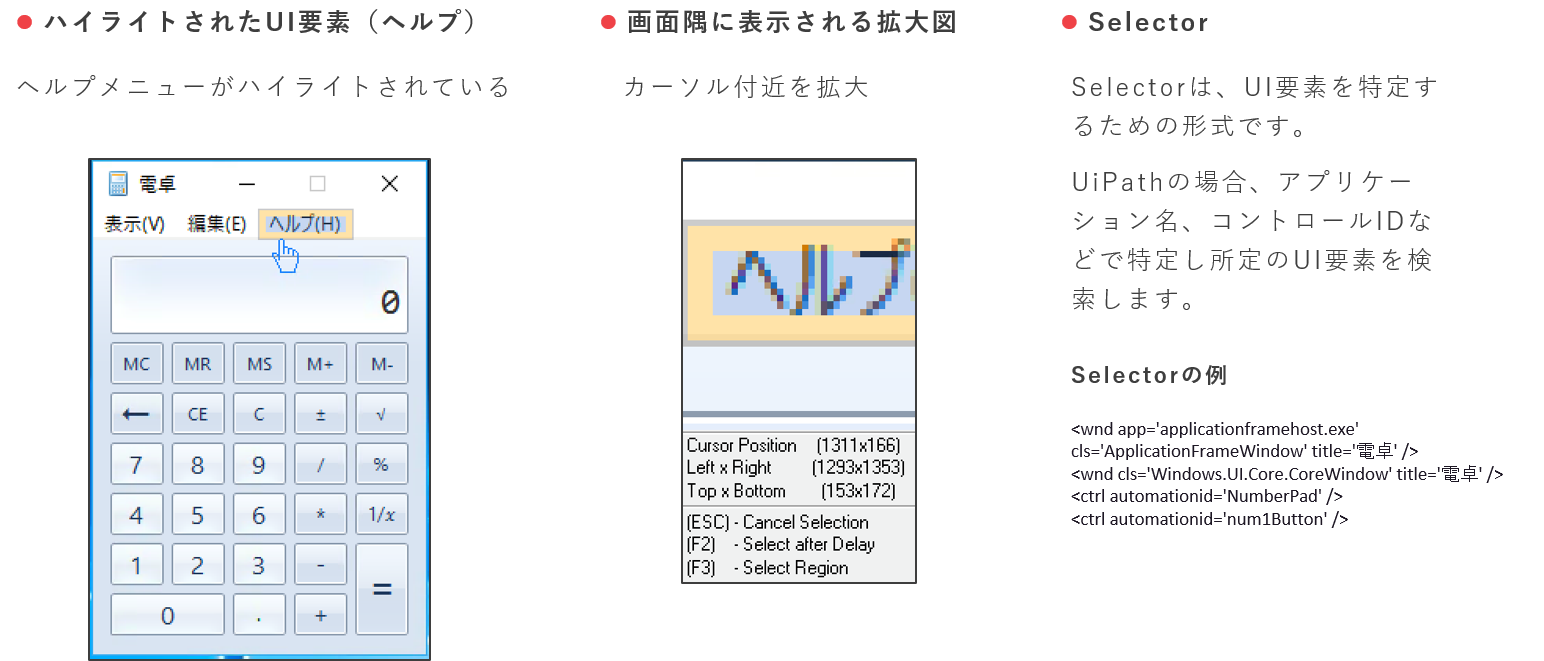
**「Record」ボタンをクリックしてレコーディングを開始します**

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/4/)

**レコーディング中の挙動**

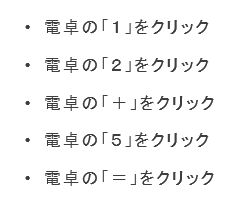
**マウスを移動するとUI要素がハイライトされる**

レコーディング中は、操作対象のUI要素がハイライトされます。ハイライトされた領域がUiPathが認識しているUI要素になります。認識されたUI要素は、Selectorと呼ばれる形式で登録されます。Selectorについての詳細は、ぜひUiPath Academyを受講してみてください。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/5/)

**レコーディングの中身（例）**

テキストでは、以下の操作をレコーディングしたものとしています。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/6/)

実際にレコーディングしてみましょう。

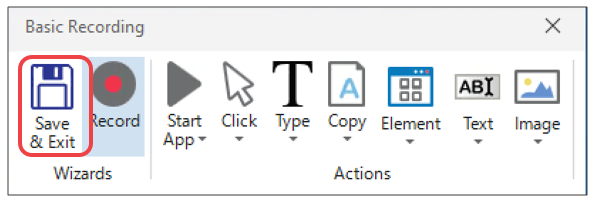
**レコーディングの中断の仕方**

エスケープ（ESC）でレコーディングを中断します。

ESCキーを入力すると、レコーディング画面（待機画面）に戻ります。

**レコーディングを保存する**

レコーディングができていると、待機画面で Save & Exit がクリックできるようになっているのでこれでメイン画面に記録して戻ります。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/7/)

**レコーディング結果**

レコーディングの結果を見てみます

レコーディングの結果は、一つの Sequence 内にまとめられています。

Sequence をダブルクリックして中身を見てみると、先ほどレコーディングしたアクティビティが登録されています。この Sequence を StartNode に設定して実行して見ましょう。（Flowchartを使っていない場合は、StartNodeに設定する必要はありません。）

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/8-2/)

**レコーディング結果の編集**

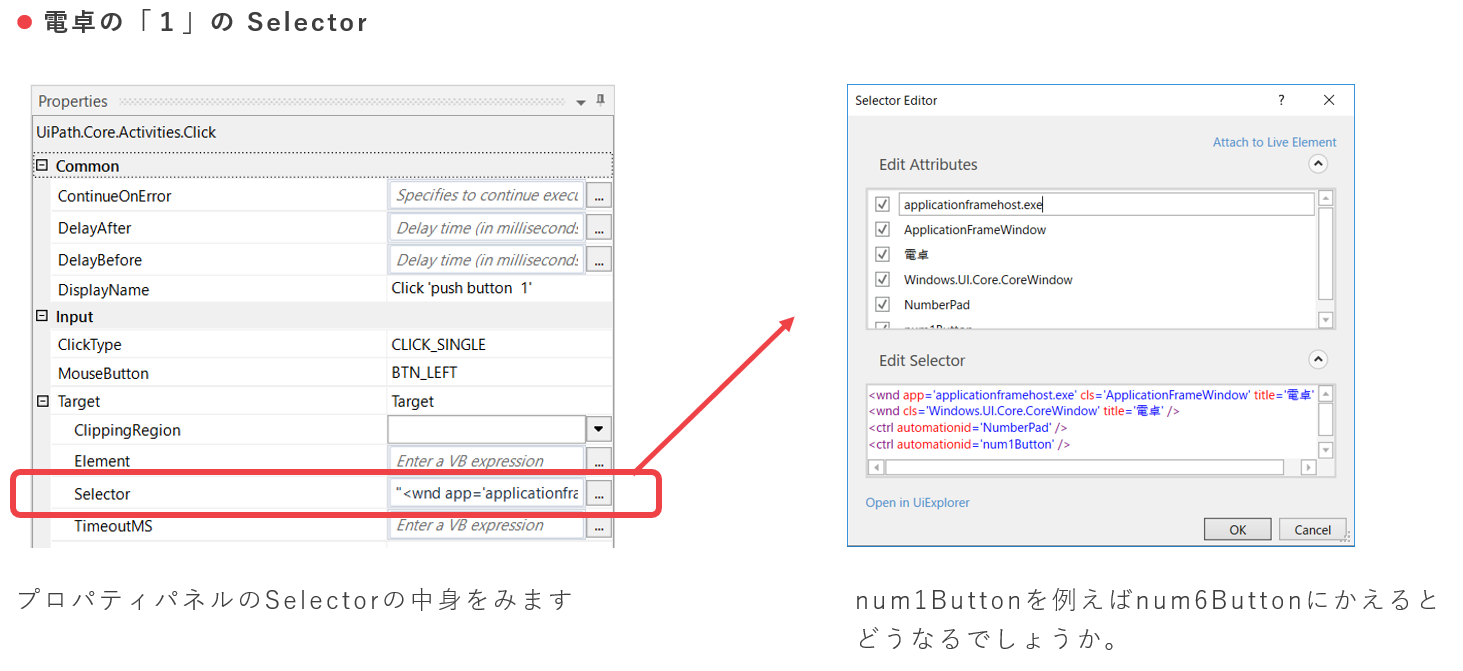
さて、このままでは、レコーディングした全く同じことを繰り返すロボットが作成されただけです。これでは、業務上あまり役に立たないロボットになりそうです。レコーディング中に例えば「１」をクリックしたところを「６」をクリックに変更したい場合は、どうすればよいでしょうか。

先程クリック対象のUI要素は、プロパティの Target > Selector で定義されています。

実は、Selector を変更することによりロボットのクリック対象を変更することができます。直接的に中身を変更することもできます。

電卓の「1」をクリックした時のセレクターを覗いてみましょう。applicationframehost.exe、ApplicationFrameWindow、電卓、num1Buttonなどが並んでいます。順番に、アプリケーションの名前、ウィンドウの名前、タイトル、ボタンの名前かなぁというようなイメージが湧いてくるかと思います。

試しにそっと、num1Buttonをnum6Buttonに変更するとどうなるでしょうか？各自試してみてください。

* [](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/9/)

**レコーディングのテクニック**

**レコーディングできるもの、できないもの**

先ほど、レコーディングを抜ける時には、ESCを押しました。つまりESCの入力自体は、レコーディングできない動作になっているということです。

UiPath のレコーディングには、自動レコーディングできるものとできないものがあります。自動レコーディングできないものについては、手動レコーディングが用意されています。

**自動レコーディングできるもの**

* 左クリック
* 文字入力

**手動レコーディングするもの**

* ショートカットキー
* 修飾キー
* 右クリック
* マウス Hover
* テキストの取得
* 要素、画像の検索
* クリップボードへのコピー
* etc

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/10/)

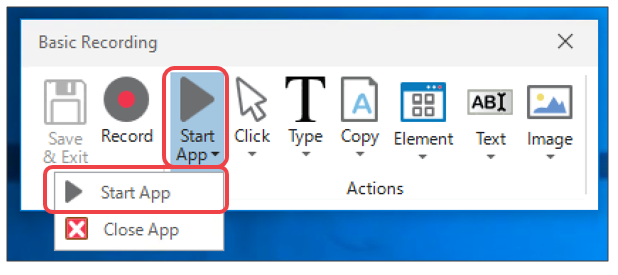
**レコーディングのテクニック 〜 アプリケーションの起動**

先ほど、電卓アプリケーションは事前に起動しておいてからレコーディングしました。

特定のアプリケーションを起動する方法は、実行ファイルをダブルクリックしても良いですが、UiPath では、起動するアプリケーションをウィンドウから選択する方法が用意されています。

レコーディング画面の「Start App」をクリックして、さらに「Start App」を選択します。

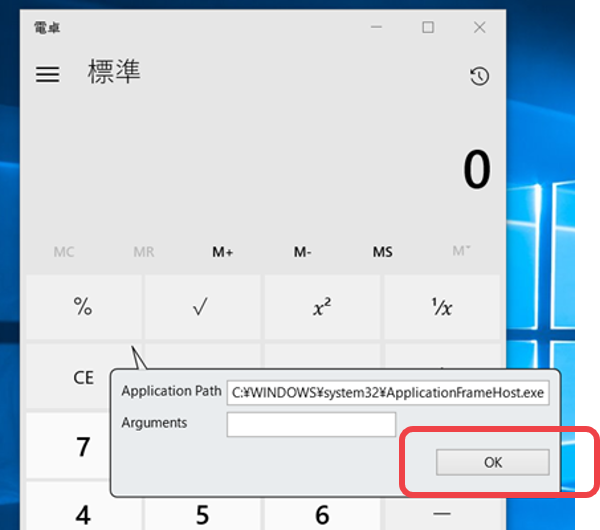
その後、起動済みのアプリケーションのウィンドウを選択してください。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/11/)

選択すると、吹き出しがでてくるので、実行ファイルの場所を確認してください。ここで、win32calc.exeとなっているのがわかりますね。

「Arguments」というのは、起動時に引数を指定する方法ですが、ここでは割愛いたします。コマンドプロンプトからアプリケーションを起動したことがある方にはイメージがわかるかと思います。

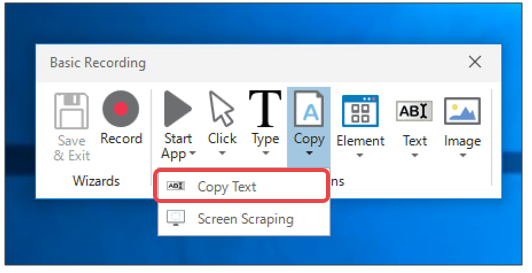
「OK」をクリックしてアプリケーションを起動するだけのロボットを動かしてみましょう。確かに電卓が起動したと思います。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/12/)

**レコーディングのテクニック 〜 テキストの取得**

電卓の計算結果を変数に取得して、これを「Message Box」 で表示してみましょう。

Textの読み込みには、レコーディング仮面の Copy > Copy Text でテキストデータを取得します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/13/)

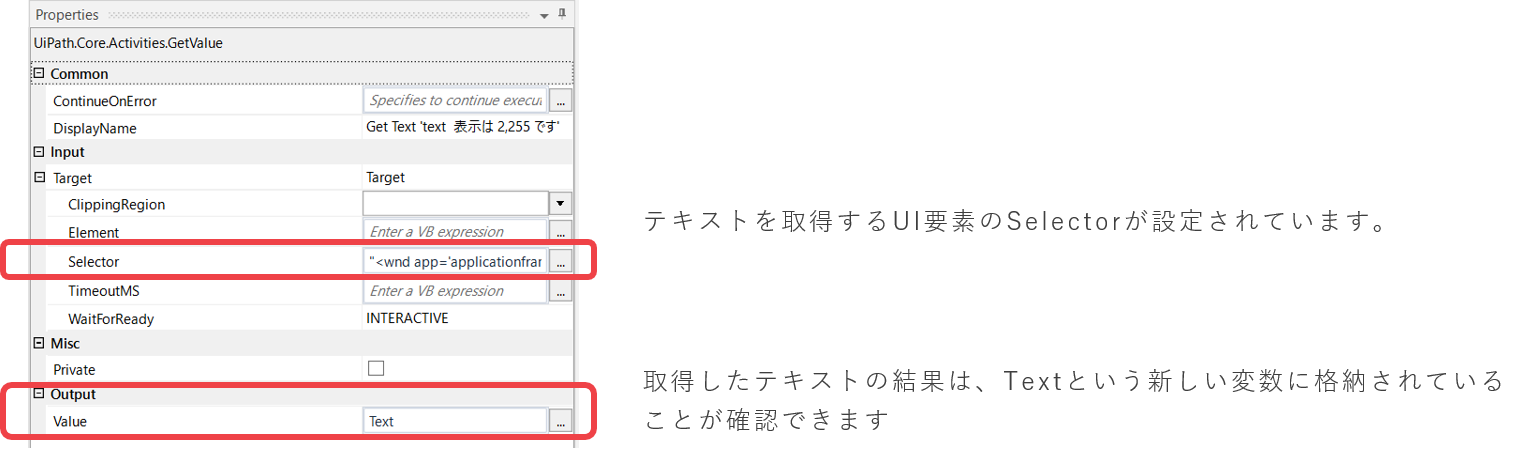
**取得したテキストの確認**

テキストをコピーしたアクティビティは、以下の図のようになります。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/14/)

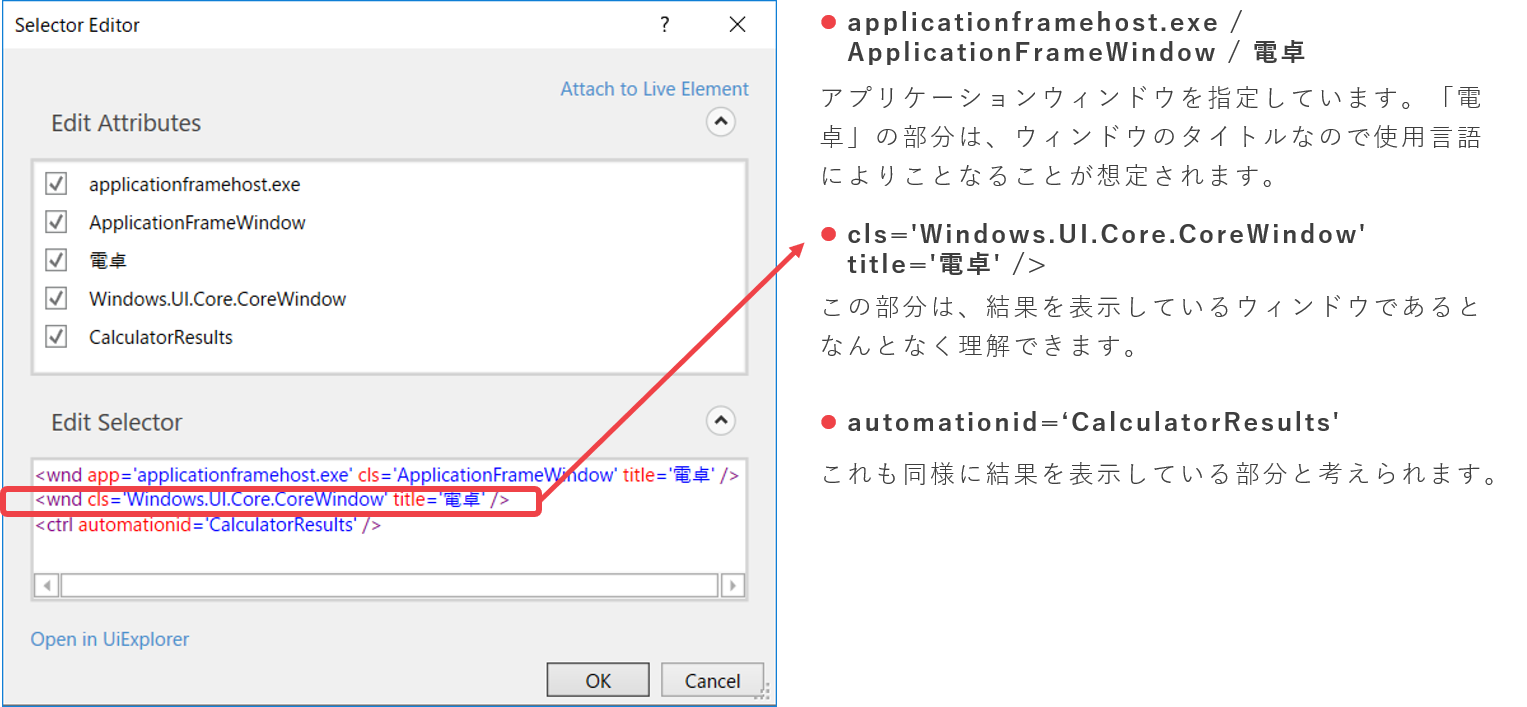
**テキスト取得のアクティビティのプロパティ**

プロパティの Target > Selector と Output > Value を確認してみましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/15/)

**取得したSelector**

今回取得した Selector は以下のようになっています。（2255は、今回の計算結果なので読み替えてください。） ここでは、特に結果を取得したところのSelectorを見てみましょう。ctrlid=’150′ title=’2255′ となっているところがありますね。’2255’は、今回の計算結果が表示されているだけなので、計算結果が変わるとうまくいかない感じがしますね。

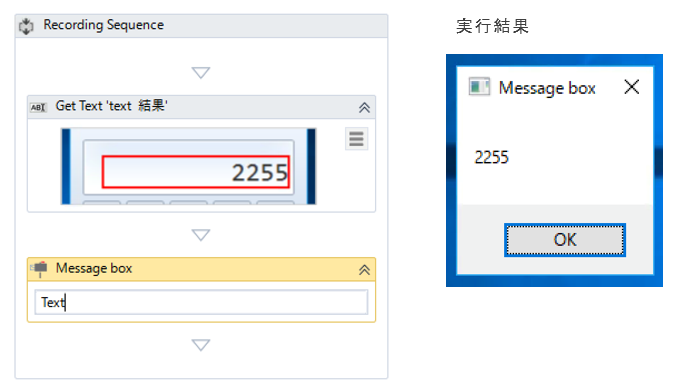
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/16/)

**Message Box の追加**

取得した数字を Message Box に表示させてみます。

Recording Sequenceのなかに、Message Box で取得したテキストを表示させます。

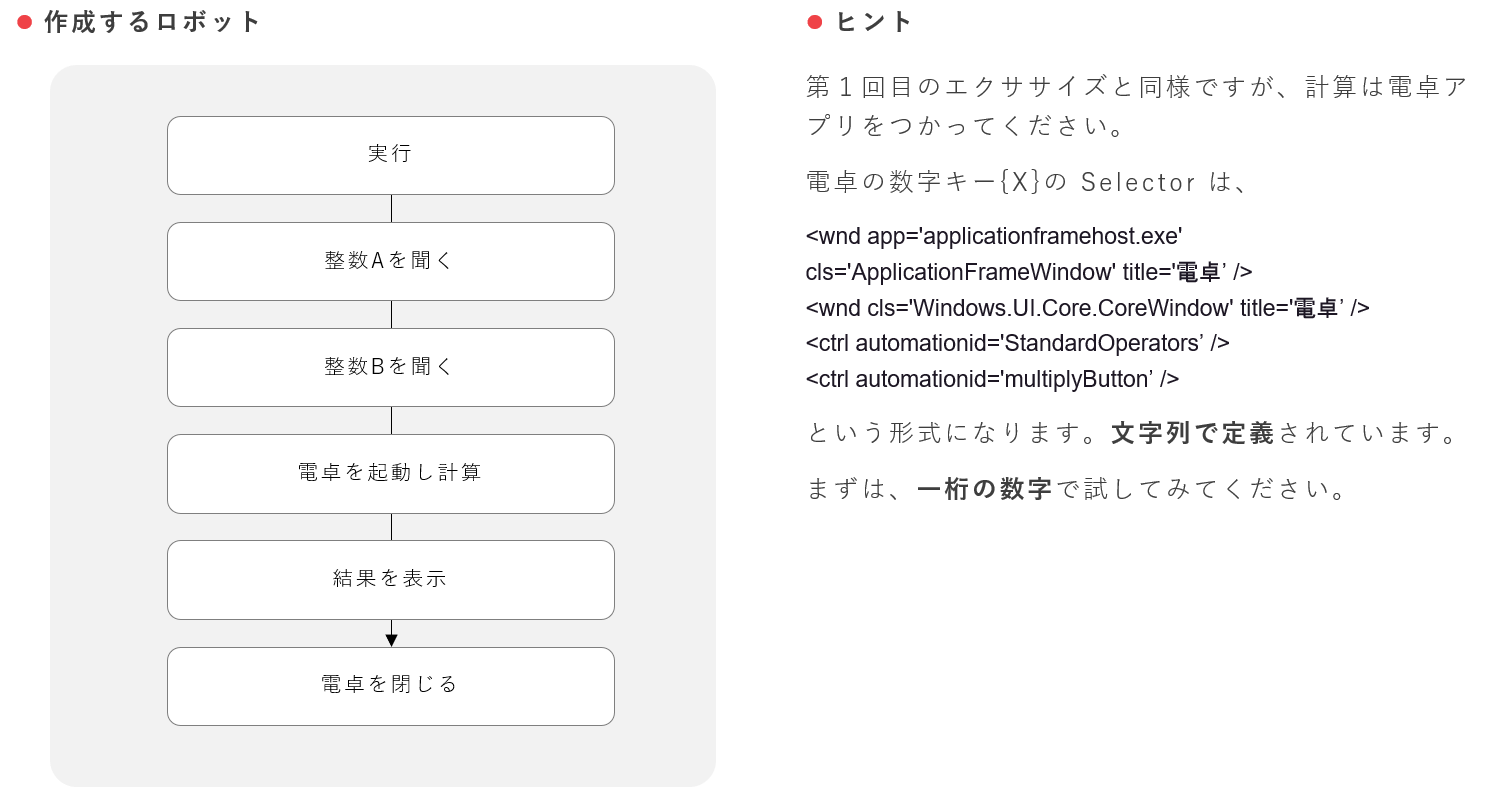
他の値を入力させたりして、実行結果を確認してみてください。結果の取得が上手くいかないときは、先程のSelectorが原因かもしれません。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/17/)

**Basicレコーディング – Exercise**

2つの整数の和を計算するロボットを作ってみましょう。Input Dialog と Message Box を使います。

普通はやらない方法ですが、Selectorを強引に書き換える方法でもクリックする場所を変更できることを試してみましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/18/)

**レコーディングの種類**

既にBasicレコーディングのところで、他のレコーディング方法が見えていたかと思いますが、レコーディングの方法にはいくつかの種類があります。以下、簡単に紹介します。

**Basic**

デスクトップアプリケーションをレコーディングする方法です。複数のアプリケーションをまたいだ操作のレコーディングができます。

**Desktop**

デスクトップアプリケーションをレコーディングする点ではBasicと同じですが、一つのアプリケーションの操作を続ける場合は、Basicよりも高速に動作します。

**Web**

Internet Explorer 上の操作を自動化します。今回は省略します。

**Citrix**

リモートデスクトップなど、Windows上でUI要素が認識されないアプリケーションの自動化に使います。Selectorの概念は、なく操作対象領域の画像マッチで操作を記録していきます。

**Exercise**

Basicと同じように、Desktopレコーディングで電卓の操作をレコーディングしてみましょう。

**Desktop vs Basic レコーディングの結果の比較**

Desktop レコーディングでは、操作対象ウィンドウが固定されます。

Desktopレコーディングでは、すべての操作が、Start Appの中に定義されていることがわかると思います。また、Selectorもシンプルになっています。

一方のBasicレコーディングでは、全体の操作を囲っているものはSequenceのみで、アプリケーションの起動は単独のアクティビティになっています。さらにSelectorは、アプリケーション名から細かく定義されています。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/19/)

**Citrix レコーディング**

先ほどレコーディングした電卓の操作を、Citrixレコーディングでもやってみましょう。

Citrixレコーディングは、画像を認識して操作を登録する方法です。

**画像の認識方法**

Citirixレコーディングで画像を認識させる方法は、テキストと画像の2種類があります。

**テキスト**

画像をOCRなどでテキストに変換して、指定したテキストが見つかったところを操作します。日本語フォントだとうまく機能しない場合もありますが、柔軟な利用ができるので現実的な方法です。

**画像**

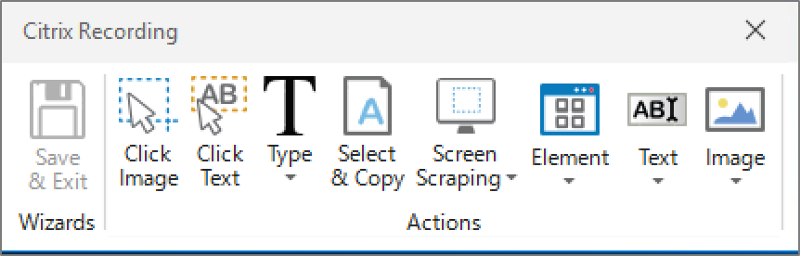
ドラッグアンドドロップで指定した画像を登録し、指定した画像が見つかったところを基準とした場所を座標などで指定し操作します。

**Citrixレコーディングをやってみる**

先ほどレコーディングした電卓の操作を、Citrixレコーディングでもやってみましょう。

Citrixレコーディングは、画像を認識して操作を登録する方法です。

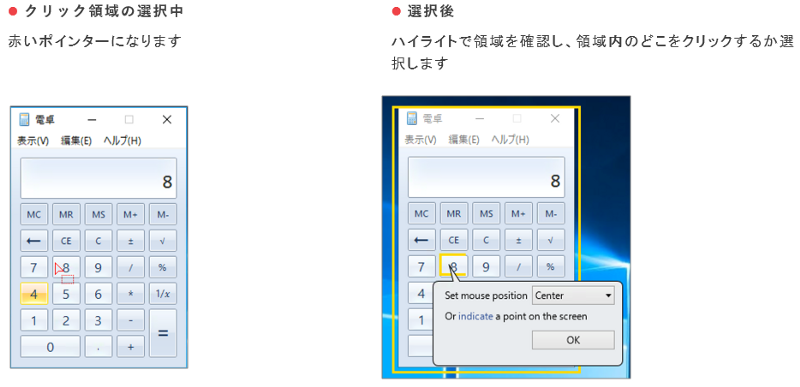
Citrixレコーディングには、自動レコーディングはありません。手動レコーディングで一つずつ登録していきます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/20/)

**Citrix レコーディング 〜 画像クリック**

Click Imageを使うとクリックする領域を選択するアイコンに表示が変わります。

クリックしたい領域をドラッグして指定します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/21/)

**Citrix レコーディング 〜 テキストクリック**

Click Textを使うとOCRを使ってテキストが登場する場所のクリック指定ができます。

クリックしたい領域を選択します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/22/)

**クリックするテキストの選択**

続いて、取得したテキストからクリックするテキストを指定します。

設定したら実際に動かしてみましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/23/)

**フロー制御**

**条件分岐**

条件分岐には、全く同じ動作を定義できる２つの方法があります。それぞれに使いやすいシーンが異なりますが、完全に同じ挙動を記述できます。

**Flow Decision**

フローチャート内での条件分岐を行う方法。より大きな視点から条件を分けたい場合に適しています。

**If**

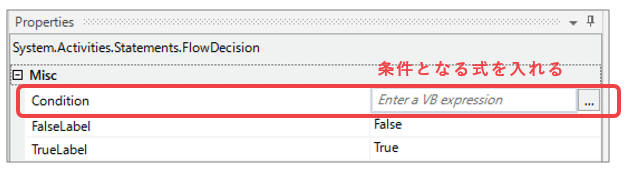
シーケンスに分岐します。小さな分岐を定義する際に便利です。

**Flow Decision**

**概要**

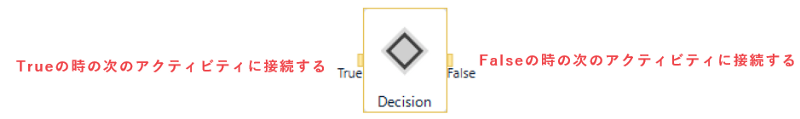
フローチャート内に定義するタイプの条件分岐です。

**プロパティ**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/24/)

**アクティビティ**

評価するCondition（TrueかFalseを返す式）を設定し、Trueの場合、Falseの場合で場合分けします。

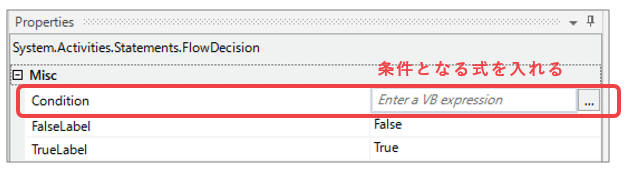
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/25/)

**If**

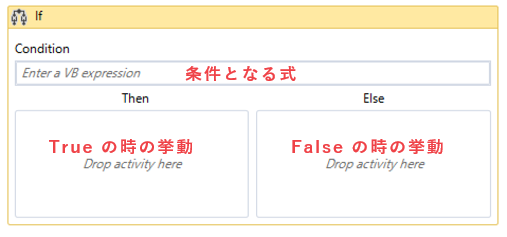
**概要**

シンプルな条件分岐に最適なシーケンスタイプのアクティビティです。

**プロパティ**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/26/)

**アクティビティ**

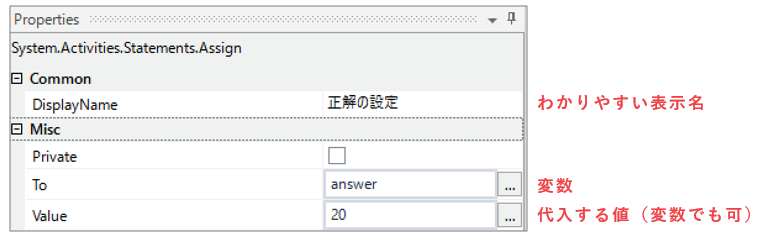
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/27-2/)

**変数への代入**

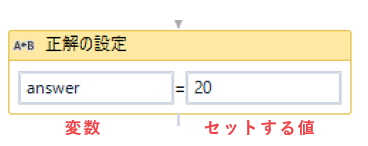
**Assign**

変数に値を代入するアクティビティです。変数の初期値として設定しても同じようなことができますが、アクティビティ内にまとめて整理したい時などに便利です。また、途中で変数の値を変更したいときにも使います。

**プロパティ**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/28-2/)

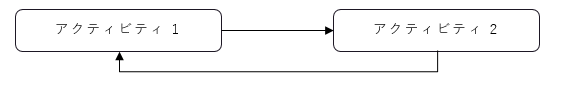
**アクティビティ**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/29/)

**繰り返しの制御**

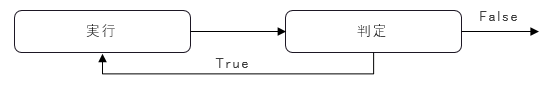
**アクティビティーを戻す**

フローチャートでアクティビティを後ろのアクティビティに戻して接続する方法です。シンプルに矢印で制御できます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/30/)

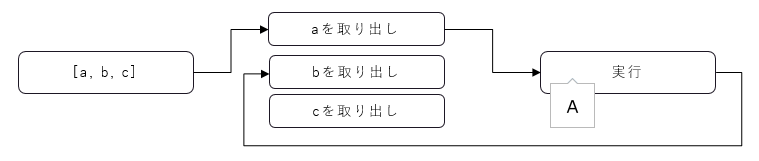
**Do While**

フローチャートでアクティビティを後ろに回すのとほぼ同じですが、次の繰り返しの前に条件文を判定して、判定結果が True であれば繰り返し実行します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/31/)

**For Each**

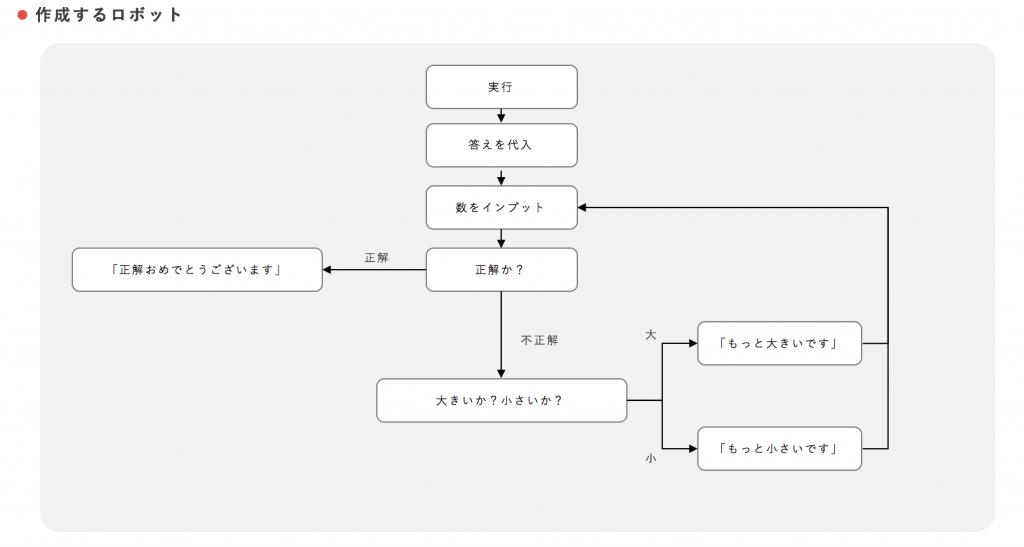
複数の要素が入っている配列など、繰り返し処理が可能な変数を受け取ってそれらの要素一つ一つに対して処理を実行します。エクセルデータを取り出して一行ずつ処理するなどしたい場合に使います。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/32/)

**練習問題**

正解するまで終わらない数当てゲームをロボットで作ってみましょう。

**数あてゲーム**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-recording/attachment/33/)

**復習**

新しい概念や操作が多く覚えきれないところもあると思います。

情報をここで復習してみましょう。

**レコーディングは、なにを記録するものですか？**

**異なるレコーディングの種類について**

* Basic、Desktop レコーディングの違いを説明してください
* Citrix レコーディングの方法を説明してください

**フロー制御について**

* ２種類の条件分岐、Decision フロー と If 文の違いを説明してください
* 変数への代入（Assign） と 繰り返し（Do While 文）について説明してください

ではまた次回。

UiPathではじめるRPA入門（４）〜 デバッグ編

* 2017年11月21日

* [RPA](https://tutorial.co.jp/category/rpa/)

今回の内容は、おそらく多くの開発者が多くの時間をさくであろうデバッギングについて触れます。

**ロボットの動作が期待と異なる時の対処方法〜デバッグ**

動作中のアクティビティを一つ一つ調べる（Debug実行モード、Step Into）  
特定の動作の直前で止める（Breakpoint）  
変数の中身を表示する（Log Message）  
予期せぬ致命的状態が発生した場合の対処法〜例外処理（Error Handling）

**エラーを補足する（Try Catch）**

こちらの記事では、期待した通りにロボットが動かなかったり、エラーが出て動作しない場合に対処する方法について学習します。[RPA入門（3）](https://tutorial.co.jp/rpa/2017/11/uipath-recording/)の記事で作成した「数当てゲーム」を用いてデバッグを学習します。

**ロボットが期待通りの動作をしない時**

作成したロボットが期待通りの動作をしない場合があります。

特に、自動レコーディングを用いて作ったロボットは値を変更した時などに期待通りの場所をクリックしなかったり、クリックすべき場所を発見できずにストールしてしまうことなどがよくみられます。

**ロボットが異常停止してしまう時**

期待通りの動作をしないという中には、ロボット自体が異常停止してしまうケースもあります。

例えば、ユーザーが入力するテキストの形式が期待と違う、Webサイトから取得してきたデータの形式が期待と違うなどです。

前回作成した、数当てゲームでは、入力が数字である必要がありますが、「おはよう」などの文字列も入力できてしまいます。この場合ロボットは、異常停止します。

**問題は、問題の場所が特定できないこと**

上述のように、原因がわかっていて動作しないというのであれば、すぐに修正ができます。しかし、ロボットを普通に実行していると、想定外の事態でどこで何が問題だったのかがわからないようなケースにもたくさん遭遇します。

そこで、ここでは、 **問題の場所発見に着目** してUiPath上でどのような機能が提供されているのかを学習してきます。

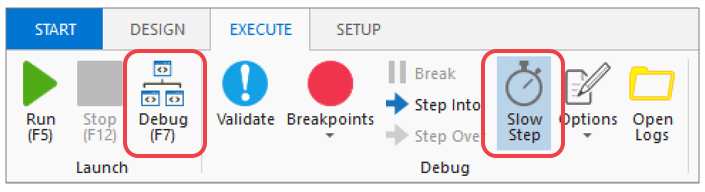
**デバッグ実行モード**

**どこを実行しているのかがわかる「デバッグ実行モード」**

リボン > EXECUTE > Debug でデバッグ実行モードでロボットを起動できます。

デバッグ実行中は、動作中の部分が設定ファイル上でハイライトされわかるようになっています。Slow StepというボタンをONにしておけば、動作もゆっくりになります。

とりあえず、Slow StepをONにして、デバッグ実行してみましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/1-2/)

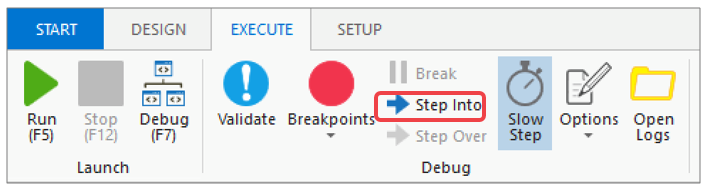
**Step into**

**一つ一つ確認しながら進められる Step into**

リボン > EXECUTE > Step into をクリックしてみましょう。このボタンは、何度もクリックすることになるので、（F11）で実行する方法も是非覚えてください。

早速実行してみましょう。先ほどのデバッグ実行モードをクリックの度に一つ一つ進めていく形になっていることがわかると思います。

入れ子になった部分から出て来るステップも詳細に確認できるので丁寧に手順を確認するにはおすすめの方法です。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/2-2/)

**Breakpoints**

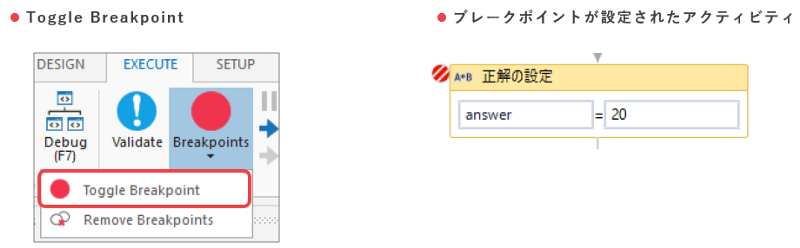
**特定の部分でロボットの動作を止める Breakpoints**

アクティビティを選択した状態で、リボン > EXECUTE > Breakpointsから、”Toggle Breakpoint” を選択してみましょう。

アクティビティに、ブレークポイントのマーク（斜め縞の赤丸）が出ていると思います。同じように “Toggle Breakpoint” を選択すると、今度はマークが消えます。

ブレークポイントを設定した状態でデバッグ実行をしてみましょう。ブレークポイントで処理が一時停止されます。

デバッグの効率のためにブレークポイントまで一気に処理を実行させたり、ブレークポイントの時点で変数の中身を見たいときなどに重宝します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/3-2/)

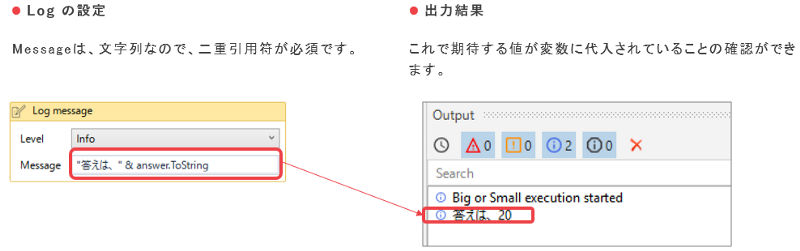
**Log Message**

**ログに内部状態を出力させる Log Message**

アクティビティより、Log Message を追加してみましょう。

数当てゲームの answer を設定したあとに、answer に正しい値が代入されているかをチェックしてみます。

Log の出力先は、テキストファイルか、実行中に登場する Output パネルになります。Log Messageは、デバッグ中じゃないものも記録できるので、実行中の情報を集めておきたいときにも重宝します。ただし、Log Messageの中にセキュリティ上問題になりそうなものを含めいないように注意しておきましょう。アクティビティには、Logに出力させないPrivateというプロパティもあります。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/4-2/)

**変数の表示**

**変数の現在の状態を表示する Locals**

デバッグ実行すると、Outputパネルの下に、”Locals”というタブが出てきます。

これを表示すると、ローカル変数の現在の値を参照することができます。

ブレークポイントと合わせて活用してください。

**変数の値を表示するLocals**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/5-2/)

**例外処理（Error Handling）**

**例外の発生**

ロボットが動作を継続できない状態に陥った時、ロボットは異常終了してしまいます。終了する前に、管理者へ通知を出したり、仕掛かりの仕事を元に戻したりしたい時、どうすればよいでしょうか。

この機能を提供しているのが、Error Handlingに関連するの一連のアクティビティです。

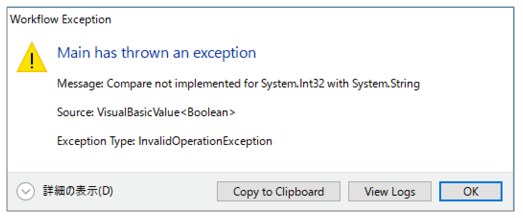
**試しに例外を発生させてみましょう**

前回作成した「数当てゲーム」の入力として、数字ではなく、「およよう」と文字列を入れてみてください。

**エラーメッセージ**

期待通りに例外が発生したところと思います。詳細を表示することもできますし、エラーメッセージをコピーすることもできます。理解できないエラーについては、そのままgoogleで検索すると解決策が見つかることもあります。

*UiPathでは、VB .NET を採用* していますので、VB .NET系の情報を探すのが良いですね。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/6-2/)

「文字列と数値の比較は実装されていない」という趣旨のメッセージが出ました。

例外が発生した場所は、先ほど紹介したデバッグ方法で確認します。

**例外処理**

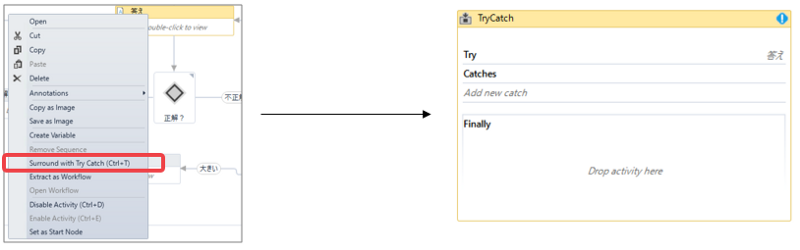
先ほどの例外を正しく処理するために、次のような方針で対処してみましょう。

**方針**

* 正解判定を別アクティビティにする（ここで例外を捕捉）
* エラー発生有無を変数で管理（Try Catch内でFlowchartの遷移を定義できないため）
* Flow Decision で発生していたら入力ダイアログに返す

**Try catch の配置の仕方**

例外処理は、Try catchアクティビティを使います。他のアクティビティのようにアクティビティパネルからドラッグで登録することもできますが、アクティビティを右クリックして、「Surround with Try Catch」というメニューが用意されているので、そちらを使ってみましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/7-2/)

**Try catch**

**概要**

Try catch は、Try（例外が発生するかもしれない実行したい処理）、Catch（発生した例外と後処理）、Finally（必ず実行する処理）の３つから構成されています。

**プロパティ**

Try catchは、表示名くらいしかプロパティがありません。他は、中身にアクティビティを定義して実装していきます。

**先ほどの例外の捕捉**

先ほど発生したエラーのタイプは、「InvalidOperationException」でした、ここでは、汎用的に例外を捕捉できる「System.Exception」を使い、これをCatchのところに加えます。発生する例外が予測できる場合は、必要に応じて補足する例外クラスを具体的に指定しましょう。より柔軟な対応が可能になります。

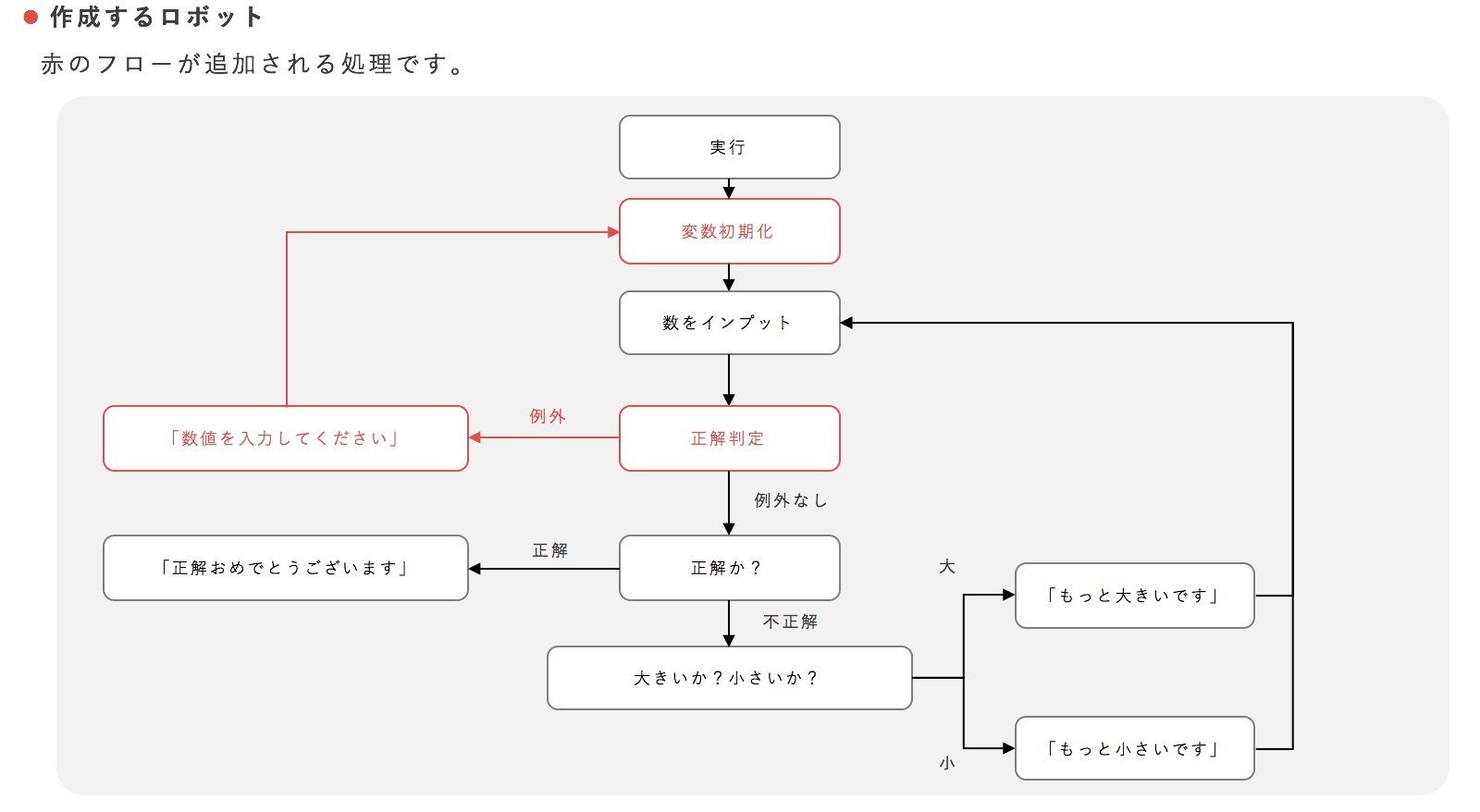
Catchの中身は、Message Box で、「数値を入力してください。」とし、必要に応じてエラーの内容を表示する Log Message を使ってください。例外発生の有無を判定するための hasError への値の代入を行います。

Finallyは、特にする処理がないので空白のままとします。

**例外処理エクササイズ**

数あてゲームに例外処理を追加してみましょう。

**完成イメージ**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-debug/attachment/8-3/)

**復習**

今回は、期待した通りにロボットが動かなかったり、エラーがでて動作しない場合に対処する方法、デバッギングの方法を学習しました。

**ロボットの動作が期待と異なる時の対処方法〜デバッグ**

* どんなときにデバッグが必要になるかせつめいしてください
* 動作中のアクティビティを確認する方法について説明してください
* 特定の動作の直前で止める方法について説明してください
* 変数の中身を表示する方法について説明してください

**予期せぬ致命的状態が発生した場合の対処法〜例外処理（Error Handling）**

* どんなときに例外処理が必要になるか説明してください
* 例外が発生した時に、正常系として処理する方法を説明してください

また次回。

UiPathではじめるRPA入門（５）〜 アプリケーション自動化編

* 2017年11月21日

* [RPA](https://tutorial.co.jp/category/rpa/)

こちらの記事では、アプリケーションとの連携について学習します。アプリケーションとは、例えばエクセル、ワード、PDF、E-mail、インターネットブラウザなどです。UIからの操作を記録させる方法でもできますので、効率の良い方法を見つけてください。

**今回の内容**

**エクセル書類からのデータの読み取りと書き込み**

* 読み込み、書き込み
* 繰り返し

**ブラウザを用いた自動化とデータの読み取り**

* 自動検索、結果表示

**ワード書類の読み取りと書き込み（文字列置換）**

* 書類の読み込み、検索、書き込み

**PDFからのデータ読み取り**

* 請求書データの読み取り

**Eメールの送受信**

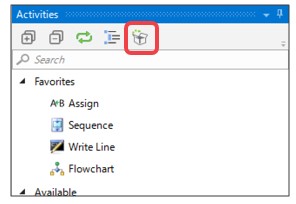
* Eメールの設定
* Eメールの受信
* 正規表現によるEメールの解釈
* Eメールの送信

**エクセルの自動化**

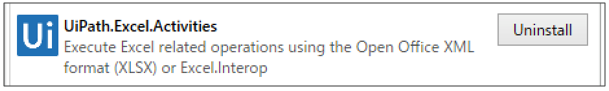
**セットアップ**

まず、UiPathからExcelを直接呼び出せるようにするための設定を行います。

アクティビティパネル > Manage Packages からPackage Managerを起動します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image7/)

Package managerから、UiPath.Excel.Activities をインストールします。

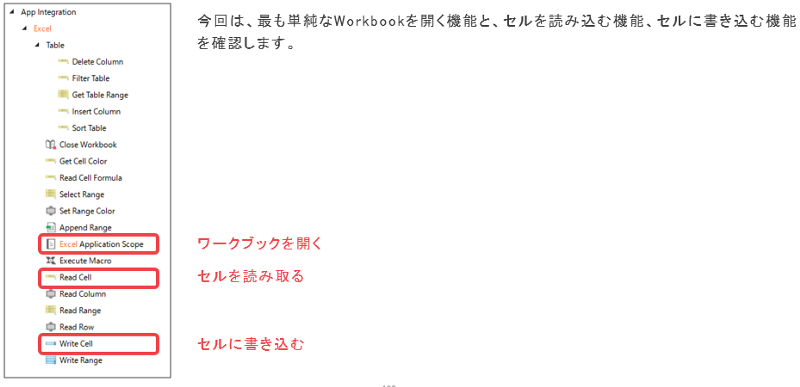
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image8/)

**テスト用ファイル**

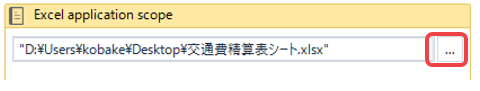
[交通費精算シート](https://wp.me/aaRKzM-aF)を使用してください

**エクセル関連のアクティビティ**

エクセルパッケージをインストールできたらエクセル関連のアクティビティが選択できるようになります。

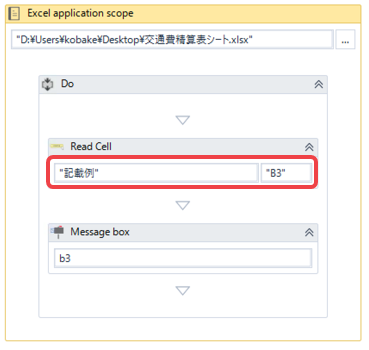
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image24/)

**ワークブックを開く**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image17/)

三点リーダ「…」をクリックして、開くべきワークブックを選択します

**セル B3 を読み込み Message Box で表示する**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image9/)

Read Cellアクティビティで、指定のシートの指定のセルを読み取る設定をします。

読み取った内容を Message Box で表示します。

**エクセルの自動化 〜 繰り返し**

**繰り返しの方針**

繰り返しの方針は２つあります。

* ロボット側で読み取るセルを一つずつずらす
* エクセルを読み込む時に配列で読み込みFor Eachで処理

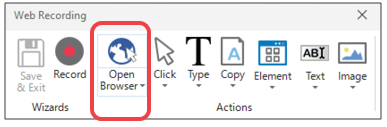
効率が良いのは、２番目の For Each を利用した方ですが、一つずつずらして読み取るというのも十分に使える方法です。

**ブラウザの自動化**

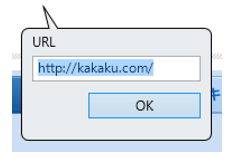
**Web レコーディング**

Recording > Web を選択します。

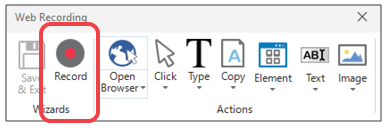
目的のサイトを開いた状態で、Open Browser でブラウザを開く指定をします。（Start Appと同じ要領です）

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image25/)

例えば、kakaku.com を開くとURLの確認がでるので、確認しOKをクリックします。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image18/)

あとは、自動レコーディングでやってみてください。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image10/)

**データスクレイピング**

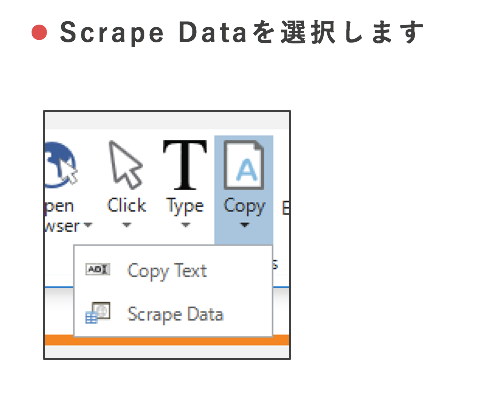
**データの取得**

データを取得する際は、Copy Text でもできますが、表のようなデータををまとめて取得することができます。

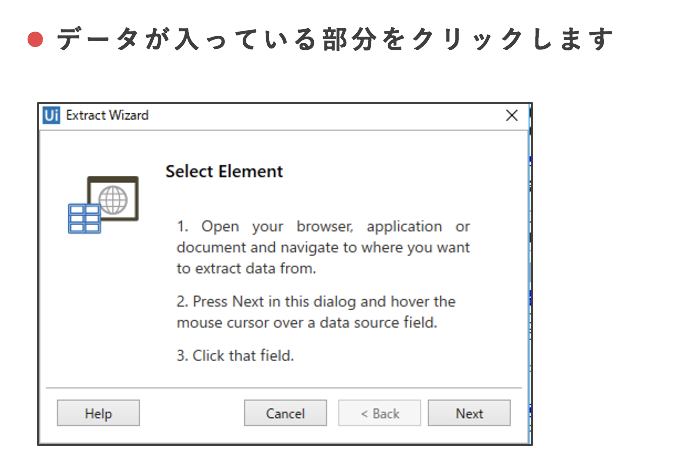
例えば、下のようなデータがあった場合、Copy > Scrape Data で 表形式でのデータを取得することができます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image26/)

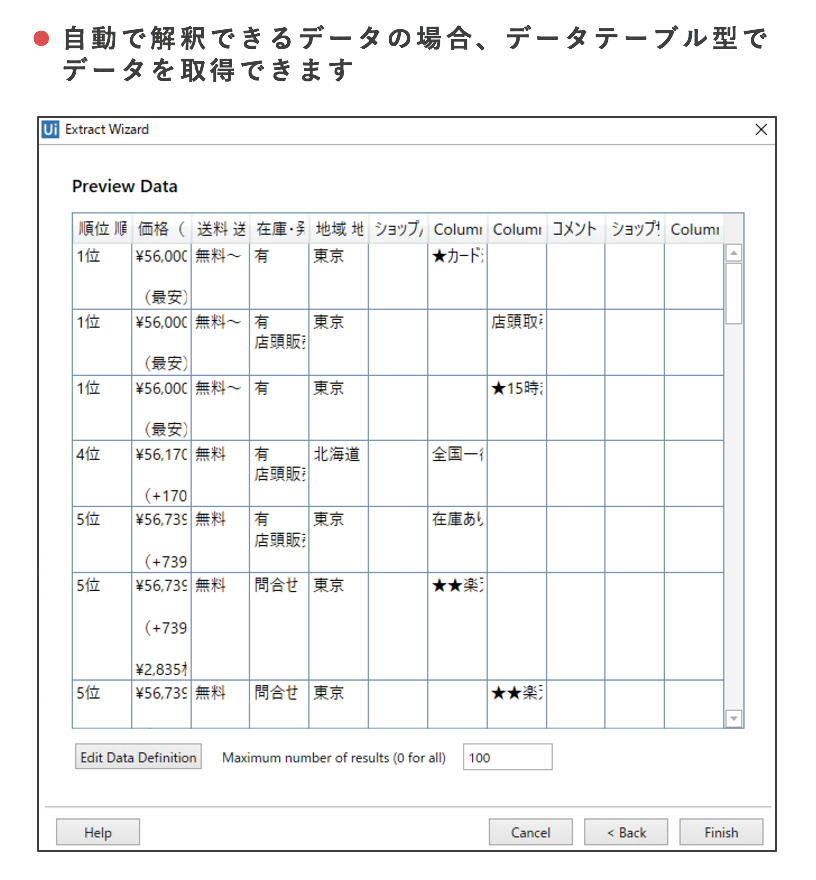
Scrape Dataを選択します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image35/)

すると、スクレイプするデータの位置を選択するようなプロンプトが表示されますので、欲しいデータがある部分をブラウザ上で選択します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image36/)

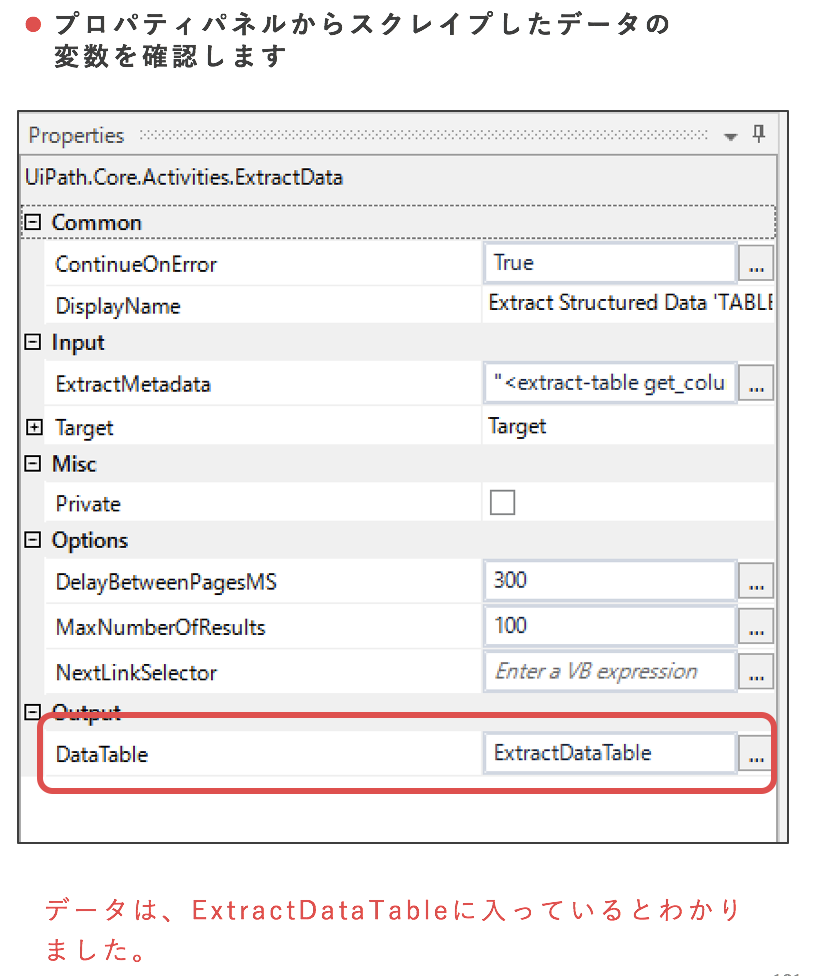
すると、自動的にデータを取得してフォーマットしてくれます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image0/)

**データの書き込み**

次に取得したデータをエクセルに書き込んでみましょう。

まずは、スクレイプしたデータがどこに格納されているのかをプロパティパネルから確認しましょう。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image32/)

ここで、取得したデータは、自動的に生成されたExtractDataTableに入っているとわかります。

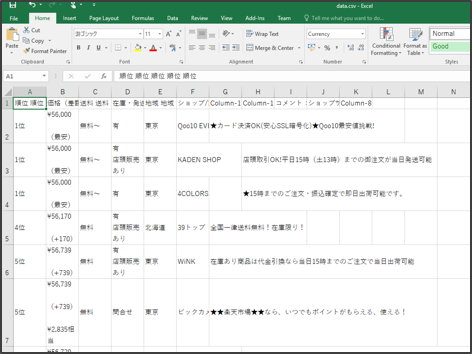
ExtractDataTableをCSVとして保存してみます。

Write CSV アクティビティを使って、ファイル名を指定します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image34/)

ロボットを実行後、ファイルが保存されていることを確認します。

ファイルを開くと、取得したデータがCSVになって保存されたことが確認できました。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image33/)

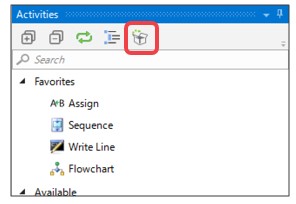
さて、次は、ワードの自動化です。

**ワードの自動化**

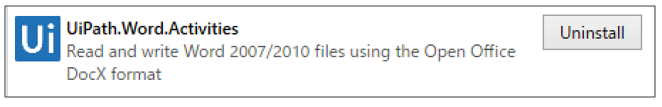
**セットアップ**

ワードにアクセスするために、エクセルと同様ワード用のパッケージをインストールします。

アクティビティパネル > Manage Packages からPackage Managerを起動します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image7-1/)

Package managerから、UiPath.Word.Activities をインストールします。

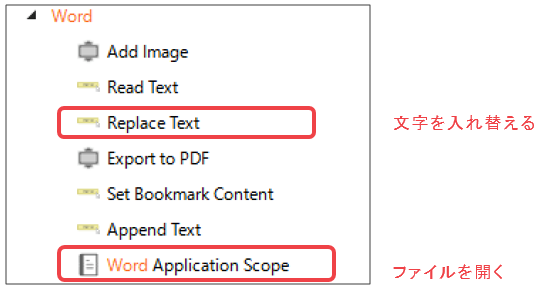
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image27/)

**テスト用ファイル**

[請求書サンプル](https://wp.me/aaRKzM-aJ)

**ワード関連のアクティビティ**

ワードパッケージをインストールできたらワード関連のアクティビティが選択できるようになります。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image19/)

**テキストを入れ替える**

例えば以下のような、ワード書類テンプレートがあったとした際に文字列を入れ替えてみましょう。テンプレートでは、[[Trihikisaki]]と実際にはありえなそうな記号で囲っているのが工夫のポイントになっています。

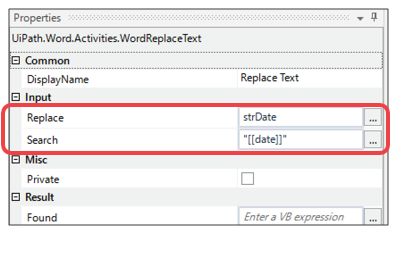
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image12/)

**Replace Text アクティビティ**

ワード書類の文字列を置換するアクティビティです。

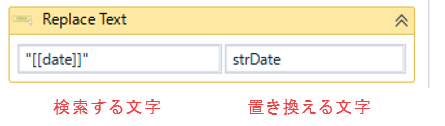
**プロパティ**

検索文字列と置換文字列があるだけのシンプルなものです。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image28/)

**アクティビティ**

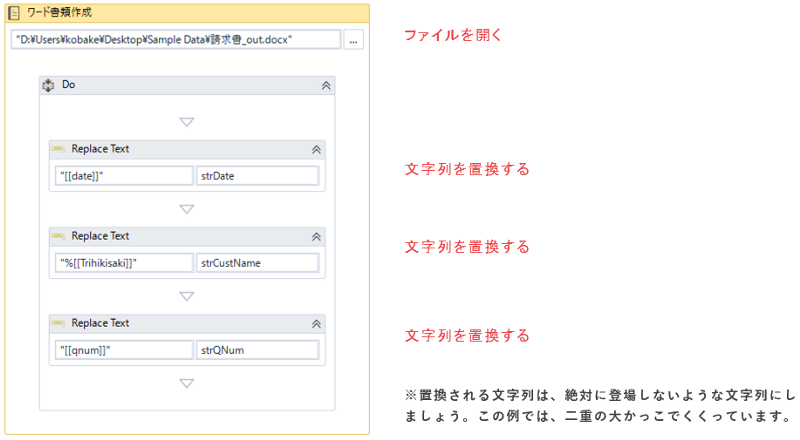
プロパティを開かずとも、アクティビティの中で設定することもできます。左側に検索する文字、右側に置き換える文字を入力します。どちらも変数を使用しても大丈夫です。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image20/)

**実際のフローチャート**

実際に置き換える処理は以下のようになります。

1. まず、ワードファイルを開きます
2. その中で、[[date]]を日付の文字列が入った変数で置換します
3. 次に、[[Trihikisaki]]に顧客名が入るように置換しています
4. 最後に、[[qnum]]に伝票番号が入るように置換しています。

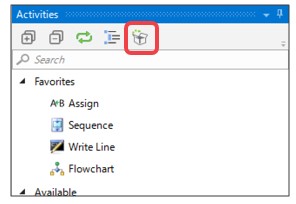
[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image13/)

実際に実行してみて置換されることを確認してください。

**PDFからのデータ読み取り**

**セットアップ**

アクティビティパネル > Manage Packages からPackage Managerを起動します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image7-2/)

Package managerから、UiPath.PDF.Activities をインストールします。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image21/)

**テスト用ファイル**

[2017qsample](https://wp.me/aaRKzM-aJ)をご使用ください。

**PDF関連のアクティビティ**

PDFパッケージをインストールできたらPDF関連のアクティビティが選択できるようになります。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image14/)

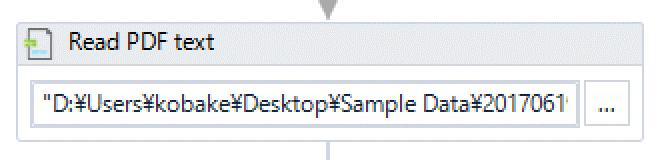
**OCRの種類と補足**

使えるOCRの種類は、GoogleとMicrosoftのものです。

* Google は、白背景で良い結果を出すので、白抜きの場合 Inverse オプションで精度が向上します
* Microsoft は、日本語にも対応しており、活字読み取りには使えますが、読み取り画像の倍率 Scale で調整が必要な場合があります

**テキストデータを読み取る**

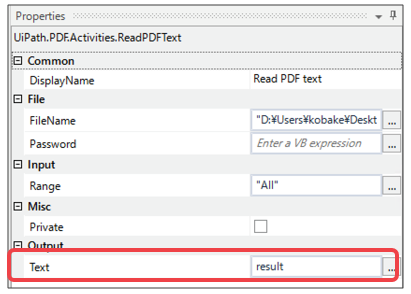
Read PDF Text アクティビティを用います。読み取るPDFファイルを指定してください。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image30/)

**読み取ったデータ**

プロパティパネルより、読み取ったデータを変数に格納します。

ここでは、result という変数に格納しています。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image22/)

**請求金額だけ読み取る**

所定の場所だけを読み取りたい場合、正規表現と呼ばれる方法を用いるのが最も正確です。

正規表現の詳細は、ここでは立ち入りませんが、以下の方法で請求金額だけを読み取ることができます。

**Matches アクティビティ**

プレゼンテーションレイヤーでの操作がないためアクティビティは簡素です。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image15/)

**Matches プロパティ**

Inputに検索対象の変数、Patternに検索パターンを入力します。

Patternには、「正規表現」を使います。この場合は、”\nご請求金額(.+)円[\r\n]”とします。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image23/)

**マッチしたパターンの返却は、IEnumerable**

マッチしたパターンの返却は、Matchオブジェクトを格納した、IEnumerableです。単純にこの中身を取り出すには、一つしかマッチしなくても、For Each を使い、itemのクラスは、Matchオブジェクトを指定します。もしくは、一番はじめのものだけを取り出します。

ここでは、For Eachを使ってみましょう。

**For Each の中身**

マッチした中身を取り出して、Message Box に表示させてみます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image31/)

Matchクラスについては、以下のURLを参照してください。

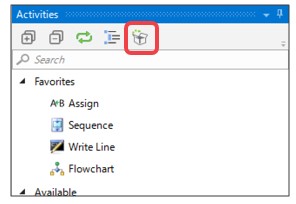
[https://medium.com/r/?url=https%3A%2F%2Fmsdn.microsoft.com%2Fja-jp%2Flibrary%2Fsystem.text.regularexpressions.match%28v%3Dvs.110%29.aspx](https://docs.microsoft.com/ja-jp/dotnet/api/system.text.regularexpressions.match?redirectedfrom=MSDN&view=netframework-4.7.2)

**Eメールの自動化**

**セットアップ**

まずは、Email用のパッケージをインストールします。

アクティビティパネル > Manage Packages からPackage Managerを起動します。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image7-3/)

Package managerから、UiPath.Mail.Activities をインストールします。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image1/)

**Eメール関連のアクティビティ（一部）**

メール関連のアクティビティは、ほぼメールクライアントと同じ動作をします。  
＊メール送受信はUIの自動化でも可能です

ここでは、SMTPでのメールの送信を試してみます。

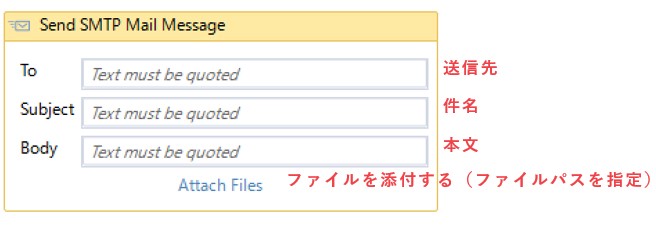
SMTPサーバーでセキュリティ設定がされているときに、この方法でうまくいかないときがありますので予めご承知ください。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image2/)

**Send SMTP Mail アクティビティ**

SMTPでメールを送信するためのアクティビティです。

送信先、件名、本文からファイルの添付もここで指定できます。メーラーでできることはほぼ全てカバーしているのでそれらは、プロパティから設定できます。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image3/)

**必須のプロパティ**

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image4/)

メールで送信するため、SMTPサーバーの

Port：SMTPサーバーのポート  
Server：SMTPサーバーのホスト  
Email：送信者ID（Eメール）  
Password：SMTPパスワード

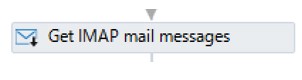
の設定が必須です。

**Eメールの受信**

**Get IMAP messages**

IMAPでメールを取得するアクティビティです。

送信と同様に、ポート、ホスト、ログイン情報が必須となります。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image5/)

**List**

取得したデータは、MailMessageオブジェクトを要素に持つListオブジェクトとなります。

MailMessageクラスについては、以下のリンクを参照してください。

URL: <https://msdn.microsoft.com/ja-jp/library/system.net.mail.mailmessage(v=vs.110).aspx>

**そのほかの細かいテクニック**

**右クリックコンテクストメニュー**

右クリックは、手動レコーディングです。

手動で右クリックをレコーディングしたあと、自動レコーディングにしF2で3秒間のレコーディングを中断を挟みその間にコンテクストメニューを表示します。

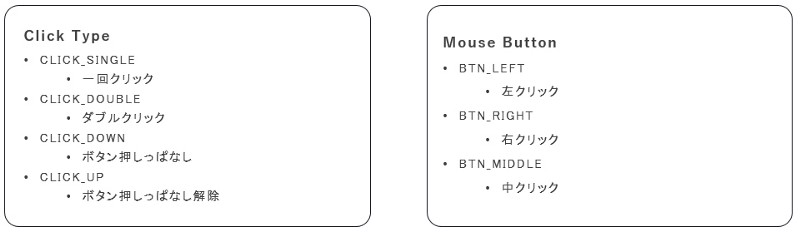
3秒の中断が終わると再び自動レコーディングが始まります。その最中にコンテクストメニューの操作を行います。

**ドラッグ＆ドロップ**

レコーディングの左クリックを用いてアクティビティを登録します。

クリックアクティビティには、次のプロパティがあります

ドラッグ＆ドロップは、CLICK\_DOWNHover CLICK\_UP でエミュレート可能です。

[](https://d1zzsytabgxvtl.cloudfront.net/rpa/2017/11/uipath-app/attachment/image6/)

**Silverlightの自動化**

Silverlight Extensionをインストールします。管理者権限が必要になります。

前提として、silverlight\_developer\_x64.exe がインストールされている必要があります。すでに、別のSilverlightがインストールされている場合は、アンインストールしてからdeveloper runtimeをインストールしてください。

Silverlight developer runtimeの参考ダウンロードリンク：  
<https://www.microsoft.com/ja-jp/download/details.aspx?id=29715>

**Javaの自動化**

Java Extensionをインストールします。管理者権限が必要になります。

AWT、SwingなどのGUIフレームワークを用いている場合、GUI要素のオブジェクトを取得できるようになります。

**復習**

今回は、アプリケーションの自動化について学習しました。

* エクセル書類からのデータの読み取りと書き込み方法を説明してください
* ブラザを用いた自動化とデータの読み取りは、どのようにしますか
* ワード書類の文字列の置き換えはどのようにしますか
* PDFからのデータ読み取り方法には、何がありますか
* Eメールの送受信では何を設定しなければなりませんか

また機会がありましたら、掲載させていただきます。