insource

【事例集】
Python でできることの具体例 5 選
～現場で役立つ便利な技術・テクニック

insource

目次

1. Python でできること① Excel 操作の自動化

Python でできることの 1 つ目は、Excel 操作の自動化です。

総務や営業事務などのバックオフィス業務では、Excel によるデータ管理、操作、集計等が多数行われています。
恐らく皆さんの会社や業務でも、一切 Excel を使っていないということは稀かと思います。
したがって、Excel 操作を自動化・効率化できるということが、バックオフィス業務にどれだけのインパクトを与えられるかということについては、イメージがしやすいでしょう。

○「Excel 操作の自動化」に用いるライブラリ

Python で Excel 操作を自動化できるライブラリはたくさんありますが、一般的によく使われるのは「openpyxl」というライブラリです。
読み方としては、オープンパイエクセルかオープンパイエックスエルなどがあります。

※ライブラリとは、一定の処理をひとまとまりにしておいて、後から使いやすいようにしておく仕組みのことです。Python では第三者が作成したライブラリがたくさん公開されており、複雑な機能を初心者でも簡単に実装することができるのが特徴的です。

openpyxl は Excel ファイルへの繰り返し処理が簡単に行うことができるというメリットがあります。
ある 1 行に対して処理を行い、それを終えると次の 1 行に対して処理を行い、最後の行まで繰り返すといったイメージです。
Excel 操作でできることの多くは openpyxl を使用して実現することができます。

○「Excel 操作の自動化」によって実現できることの例

Excel 操作を自動化することで、以下のような業務を自動化・効率化することができます。

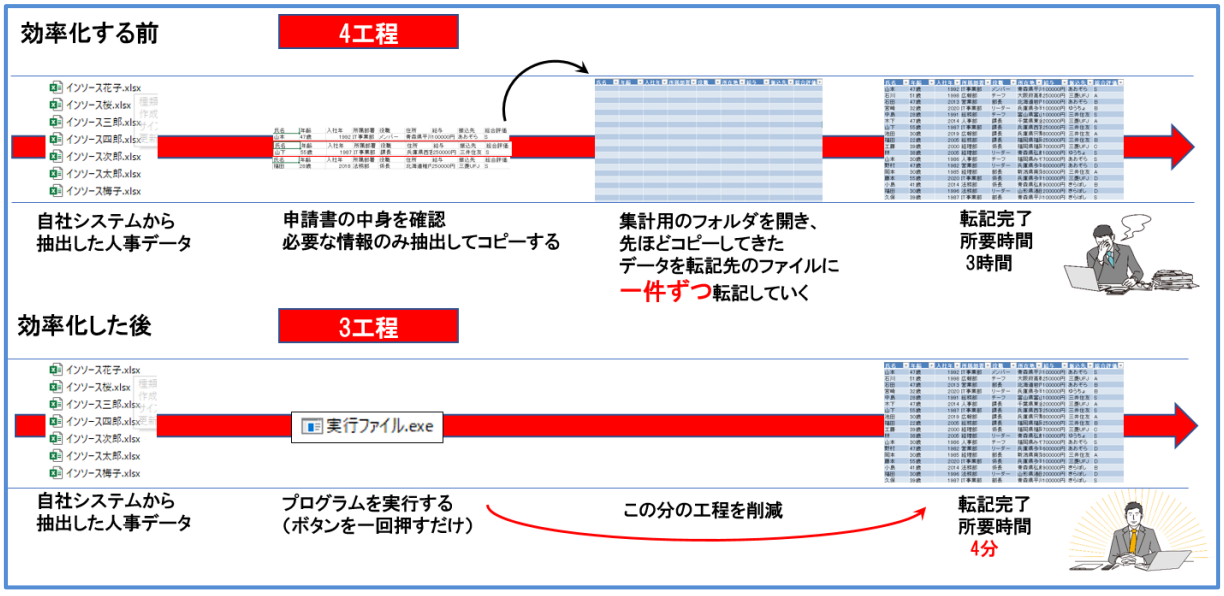
・複数の Excel ファイルからデータを抽出し、1 つのファイルにまとめる
・フォルダ内の複数の経費申請ファイルを1つのファイルにまとめ、経費精算業務を効率化する
・顧客ごとのおすすめ商品提案書を自動作成する

このような例を見ると、あなたの業務の中でも自動化・効率化できそうな部分があることに気づくことができるのではないでしょうか。

○「Excel の自動化」の業務活用実例

【事例①】人事データの作成、集計プログラム

類型	□大量のExcelファイルの中からデータの抽出・転記・集計の自動化
導入部署	□人事部
導入の背景	□期末ごとの人事情報の管理が大変。 一人一人のExcelファイルから必要な情報をコピー＆ペーストして、 400人分をまとめた集計用のファイルに転記している。 単純だが辛い作業。 定型作業だからこそ自動化したい。
導入効果	□98%の時間削減(3時間→4分)
導入後の感想	□こんな便利のものがあったのかと驚いた。また、新入社員が1から作ったということにも価値を感じている。 □これで、自分が悩むべき「人事戦略、施策」に時間をかけることができ生産性が高まった
開発時間	1時間



2. Python でできること② スクレイピング

Python でできることの 2 つ目は、スクレイピング（Web 上からの情報収集）です。
スクレイピングを一言で表すと、「Web ページから大量の情報を自動で収集し、活用することができる技術」となります。

皆さんの会社の中でも、Web 上で情報を検索して、その情報をどこかに転記するなどの業務は行われているかと思います。
Python を使えば、Web 上からの情報収集についても自動化することができます。

「Web 上で検索して、必要な部分をコピーし、ファイルに転記をする」
こういった業務はどうしても単調になりやすく、業務の面白さはなかなか感じられにくいことに加え、人為的なミスも発生しやすい分野です。
Python を使ってプログラムにお任せしてしまえば、「人間以上のスピード」「人間以上の正確性」をもって、業務を進めることができます。

そうやって手が空いた人員に、もっと価値を生み出せる業務を割り振ることで、会社が成長していくことができるようになります。

○「スクレイピング」に用いるライブラリ

Python でスクレイピングを行う際に用いられるライブラリはたくさんありますが、一般的なところとしては「Requests」ライブラリと「BeautifulSoup」ライブラリがあります。

Web ページから情報を取得する流れは以下の通りです。
①情報を取得したい対象のページに対して、Requests ライブラリを使って Web ページ全体の情報を取得する
②取得した Web ページの情報から HTML 部分を抽出する
③取り出した HTML 部分から、さらに必要な部分だけを抽出する

※HTML とはハイパーテキスト・マークアップ・ランゲージの略称です。Web ページの文章や画像などの構成をコンピュータに指示するために使います。

スクレイピングとは Web 上から必要な情報を取得し、加工することができる技術のことです。情報をそのまま取得するのではなく、必要な部分だけを抜き出したり、加工したりすることが目的になります。

Python を使ってプログラムを作っておくことで、何度も手作業でコピー＆ペーストする手間を省くことができたり、Web 上から無数のデータを取得することができたりする、というメリットがあります。

○「スクレイピング」によって実現できることの例

Python でスクレイピングを実装することで、以下のようなことが実現できるようになります。

・自社 HP の検索エンジンでの検索順位を、毎日自動取得する
・交通費申請書の乗り降り駅情報を元に、乗換案内サイトで最安の運賃を調べ
　申請金額と差異がないかを自動で調べる
・特定商品の価格を複数サイトから自動で取得して、どこか一番安いか把握する

具体例を見ていただくと、あなたの会社の中でも似たような業務を手作業で取り組んでいる、なんてケースを発見できるかもしれません。

3. Pythonでできること③ Webブラウザの自動操作

Pythonでできることの3つ目は、Webブラウザの自動操作です。
この「Web ブラウザの自動操作」を実装することができると、自動化・効率化することができるようになる業務の幅がかなり広がります。
シンプルに言ってしまえば、2次元から3次元への変化くらいのインパクトがあります。

これまでに出てきた「Excel 操作の自動化」や「スクレイピング」についても、この Web ブラウザの自動操作を組み合わせることで、業務に与えるインパクトは別次元となります。

例えばWebブラウザで何か調べものをする際、サイトによってはログインが必要な場合や、フォームに何か値を入力し、検索結果から情報を取得したい場合があります。
1つ前でてきたRequestsライブラリとBeautifulSoupライブラリは、こういった「動き」があるWebサイトから情報を取得することが不得意です。
スクレイピングの効果を最大限活かすために、このWeb ブラウザの自動操作ができるようになっておくと非常に便利です。
Pythonは「Web ブラウザの自動操作」と非常に相性が良いのが特徴的です。

○「Webブラウザの自動操作」に用いるライブラリ

Pythonで Webブラウザの自動操作を行う際には、一般的に「Selenium」ライブラリが用いられます。

Selenium を用いることで、フォームへの値入力や画面のスクロール、ボタンのクリック、画面キャプチャを撮って保存するなどといったことができます。
Pythonでこの「Selenium」が使えるようになると、業務の自動化の幅が広がります。

「Web ブラウザの自動操作」は、単体で用いるというよりは、他の技術と組み合わせることで幅広い業務の自動化・効率化に活用する、といったイメージです。

○「Web ブラウザの自動操作」によって実現できることの例

Python で Web ブラウザを自動操作できるようになることで、以下のようなことが実現できます。

・自社システム上で指定の検索を行い表示される情報を収集し、CSV ファイルに出力する
・ログインが必要な自社システム、もしくは外部の Web サイトから情報を収集する
・あるサイト上で検索結果が 1,000 件ヒットし、全ての情報を取得するためにページ遷移が必要な場合、自動でページ移動のボタンを押すことで移動し、全ての情報を取得する

「Web ブラウザの自動操作」を実装することで、Web 上で動きが必要な場面に対応することができるようになります。
できることの幅がかなり広がる、という意味が伝わったのではないでしょうか。

○「スクレイピング」と「Web ブラウザの自動操作」を組み合わせた業務活用実例

【事例②】定期申請の経理処理自動化プログラム

類型	□Excelファイルの情報をもとに、WEB自動操作を行う。 WEBから取得した情報をExcelに転記する。
導入部署	□経理部
導入の背景	□期末ごとの人事情報の管理が大変。 一人一人のExcelファイルから必要な情報をコピー＆ペーストして、 400人分をまとめた集計用のファイルに転記している。 単純だが辛い作業。 定型作業だからこそ自動化したい。
導入効果	□94%の時間削減(8時間→30分)
導入後の感想	□毎回毎回クリック＆ペーストを繰り返す作業に嫌気が来ていた。 さらに、「コンピューターの動作を阻害されずに」プログラミング が動くため、自分の業務を止めずに行えるので実質時間はかかって いない。 □手作業で行うよりも、疲れなどによる「事務ミス」も発生しづらく 手戻りにかかる時間も減っている。
開発時間	4時間

4. Python でできること④ データ分析

Python でできることの 4 つ目はデータ分析です。
社内で保管しているデータ、また WEB 上から収集したデータなどを用いてデータ分析を
行うことで、データから新たな示唆・価値を創出することができます。
Python にはデータ分析に便利な専用ツールが用意されているため、データ分析を容易に行うことができるという特徴があります。

○「データ分析」に用いるライブラリ

Pythonには、データ分析を行うための便利なライブラリが豊富に存在しますが、特によく使われているライブラリが「Pandas」です。
Python の基本文法を習得できれば、Pandas を用いることで高度なデータ分析でも簡単に実現することができます。

Pandas は、データ分析を行う上で以下 3 点のメリットがあります。
①様々な形態のデータを読み込める
②分析結果を手軽にグラフ化できる
③短いコードで複雑なデータ分析ができる

○「データ分析」によって実現できることの例

Python でデータ分析を実装することができるようになると、以下のようなことが実現できるようになります。

・社内システムから吐き出した CSV ファイルを基に、自動的に発表用グラフを作成する
・Web 上から収集したデータから売れ筋商品の傾向を把握して、商品開発に活用する
・社内データベースから部署ごとの売上と経費を読み込み、部門別損益計算書を自動作成する

Pandas を使用すると、データ分析だけでなくグラフや表の作成までを自動化することができるため、かなり幅広い業務を効率化することができます。
特に時間がかかりやすい業務が多い分野ですから、一度自動化してしまうとかなり多くの時間を節約することができます。

5. Python でできること⑤ メールの自動送信

Python でできることの 5 つ目は、メールの自動送信です。
日常業務で Python を使う機会は数多くあると思いますが、それを自動化できるかもしれないと考えると、かなりワクワクしませんか。

○「メールの自動送信」に用いるライブラリ

Python でメールの自動送信を行う際に用いることができるライブラリは数多く存在します。
具体的には使用するメールサービスによって、どのようなライブラリを使用すればいいかは変わってきます。
例えば Microsoft の Outlook を自動送信する場合は、「win32com.client」というライブラリを使用することが一般的です。
また Gmail を自動送信する場合は、「email.mime.text」というライブラリを使用することが多いです。

○「メールの自動送信」によって実現できることの例

メールの自動送信を実装することができるようになると、以下のような業務を効率化・自動化することができるようになります。

・その日の行動量や売上をデータベースから取得した後、社員全員にメールで通知する
・あらかじめ用意しておいた顧客リストに対して、定型メールの宛名部分を差し替えて、提案書を添付して送信する
・在庫量が一定ラインを下回ったときに、アラート代わりの通知を行うメールを送信する

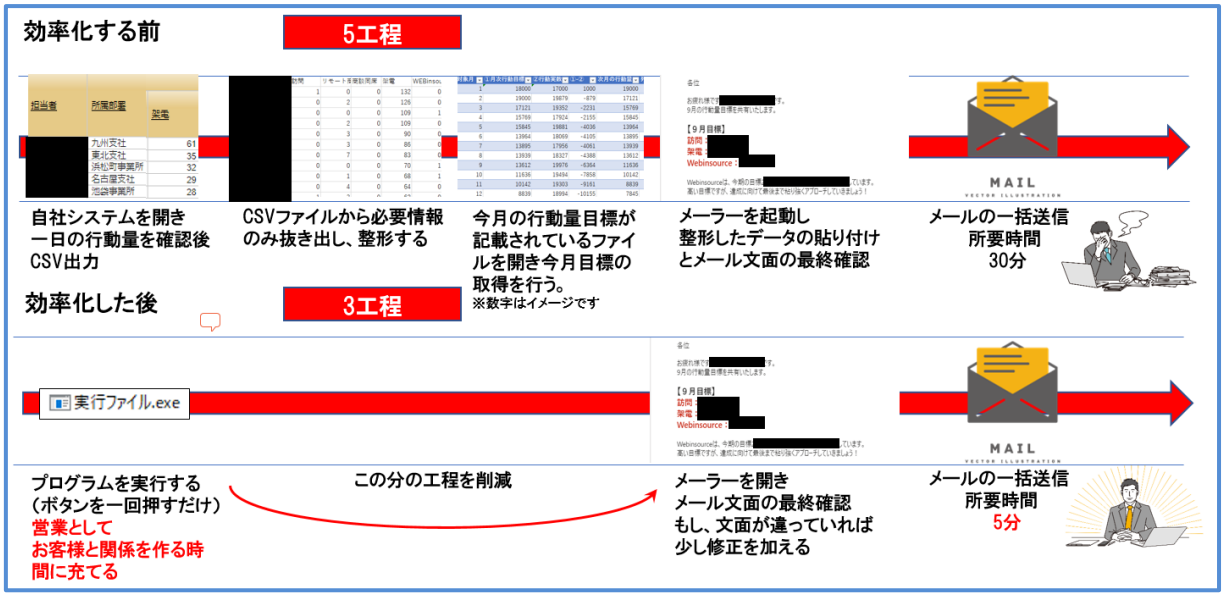
ご覧いただいたように、定期的に行うメール送信や、通知代わりのメール送信、大量の顧客への定型メール送信など、幅ひろい業務に活用することができます

ここまで Python でできることについて 5 つ見てきました。
次は現場で役立つ便利な技術・テクニックについて 3 つご紹介します。
ご紹介する技術を活用することで、より高度な自動化・効率化を実現できます。

○「メールの自動送信」の業務活用実例

【事例③】毎日発生する定型メールの自動送信プログラム

類型	□CSVファイルの必要データの抽出 メール文面への転記と自動送信
導入部署	□営業部
導入の背景	□営業の行動量を社内周知するために毎日17時にメール送信を行っている。 システムの同じ場所から毎日データを引っ張ってくるだけの作業。 □毎日決まった時間に同じ作業で拘束されてしまう。その時間を自動化し お客様にのために使いたい。 □業務自体は簡単だが、全社に送られるメールなので 「ミスが許されない」というプレッシャーも強く時間がかかっていた。
導入効果	□83%の時間削減(30分→5分) 年間でおよそ100時間の削減
導入後の感想	□デスクトップのボタンを一つ押せば、1分ほどで正しい情報がメールで 届くのでとても便利。 人為的な集計ミスが起こらないことで安心でき、確認時間も減った。
開発時間	20時間

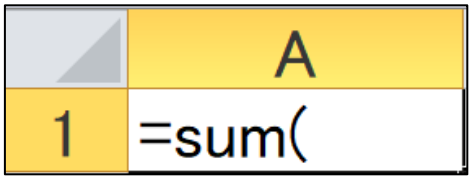


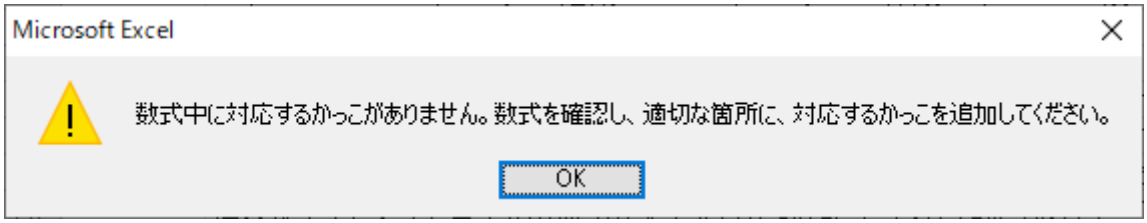
6. 現場で役立つ便利な技術・テクニック①
～ダイアログの活用

(1)ダイアログとは

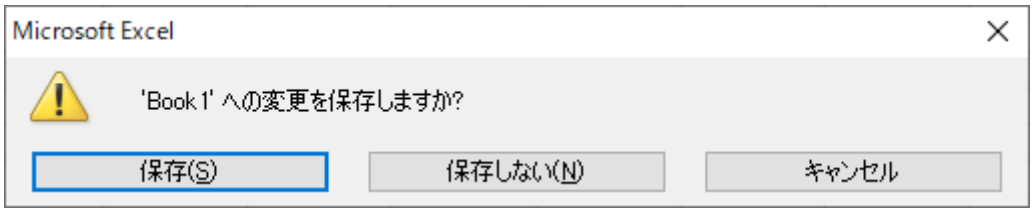
ダイアログボックス(略称:ダイアログ)とは、プログラムの実行中に一時的に開かれる、小さなウインドウのことです。
コンピュータ上で何か操作をしている時に、よく出てきます。

例えば Excel のセルで以下のように実行すると、**メッセージボックス**が表示されます。

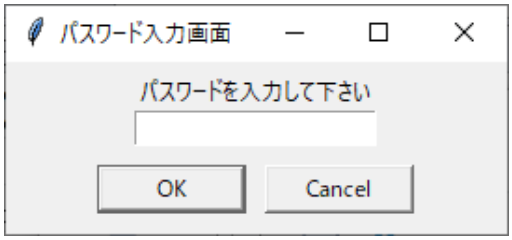




Excel を保存せずに終了しようとすると、以下のような**選択ダイアログ**が表示されます。



また、ユーザーからの入力を受け付けて、入力内容を元にプログラムの処理内容を変える**入力ダイアログ**もよく見ます。



様々な場面でよく出てくるこの「ダイアログ」ですが、Python でも表示をすることができます

(2)ダイアログの種類

ダイアログは用途に応じて様々なものを使い分ける必要があります。
以下、よく使うダイアログを 4 つご紹介します。

① メッセージボックス

■主な用途

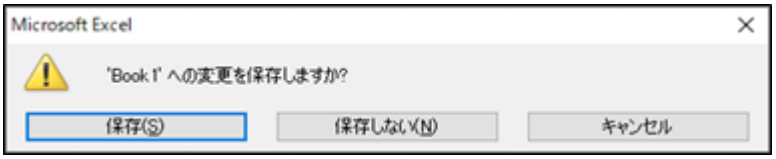
ユーザーに対して何らかのメッセージを伝える

■活用例

・フォーム送信を受け付けるプログラムで、入力値に不備があった時に
警告メッセージを出す

・自動化プログラムを実行して、異常終了した時にエラーメッセージを出す

② 選択ダイアログ



■主な用途

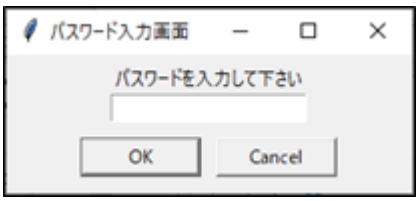
複数のボタンを表示して、ユーザーに何かしらの判断を行ってもらう

■活用例

・処理を続行するか、「はい・いいえ」の二択ダイアログを出す

・ファイルを保存するか、「保存・保存しない・キャンセル」の三択ダイアログを出す

③ 入力ダイアログ



■主な用途

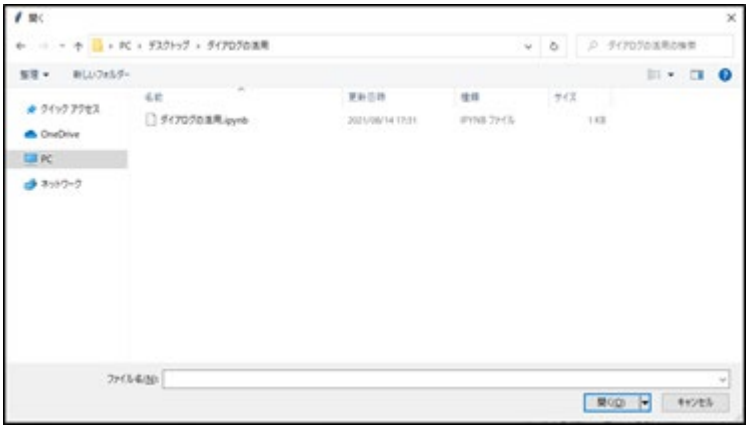
入力フィールドを表示して、ユーザーから何らかの値を受け取る

■活用例

・システムにログインするためのパスワードを入力してもらう

・自動配信メールの送信者名を入力してもらう

④ ファイル選択ダイアログ



■主な用途

処理対象のファイルを選択してもらう

■活用例

・顧客リストに記載の各顧客に対してはがきを自動生成する際に、
使用するひな型ファイルを選択する

・Web サイトにアップロードするファイルを選択する

Python におけるダイアログの表示には、標準ライブラリの「tkinter」を利用します。

7. 現場で役立つ便利な技術・テクニック②
～実行ファイル作成

(1)実行ファイルとは

「実行ファイル」といっても、様々な解釈・定義の仕方があります。
ここでは Windows の拡張子が「.exe」のファイルのような、ダブルクリックすることでそのファイル単体でプログラムが実行されるものを「実行ファイル」と呼びます。

Windows をお使いの方は、拡張子が「.exe」のファイルをよく見ることと思います。
この「exe」というのは、「実行する」という意味の英単語「execute」の略です。

ここまで動作させてきた Python プログラムは、実行していたファイル単体で動作を 全て行っていたわけではありません。
PC 内の様々なファイルとデータをやり取りし、動いていました。
exe ファイルのような「実行ファイル」は、そのファイル単体で動作が完結します。

(2)なぜ実行ファイルを作成する必要があるか

Jupyter Notebook 上で作成したプログラムは、開発者の PC 上では動きますが、他の PC 上では動かない場合が多いです。

なぜ今 Python のプログラムを動かせているかというと、それは事前に Python のインストールを済ませ、またプログラムの動作に必要なライブラリをインストール済みであるからです。

例えば、Excel 操作を自動化するためのプログラムを作成し、それを組織内の他の方たちにも使えるようにしたいとします。

組織内の他の方たちにも自分の PC 上に Python をインストールしてもらい、またプログラムの動作に必要となるライブラリをインストールしてもらえれば、他の PC の Jupyter Notebook 上でも動作するようになりますが、開発者でもない利用者にそこまでの作業を依頼するのは大変です。

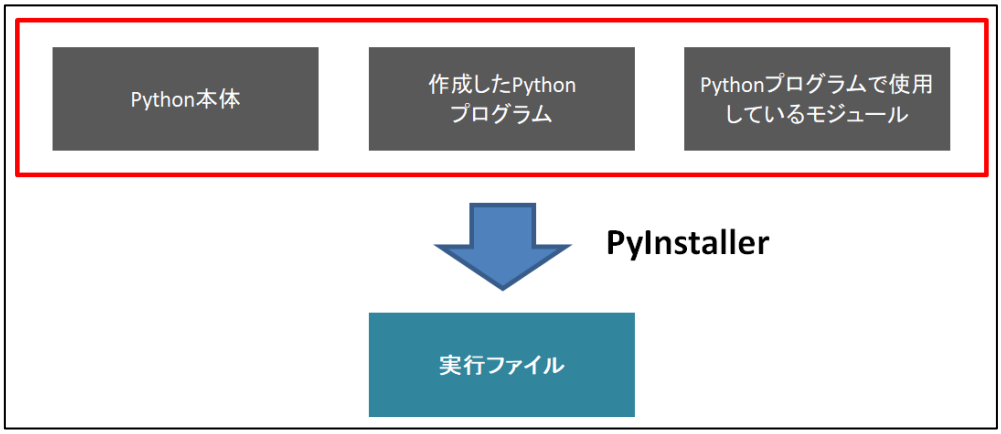
そこで、Python で作成したプログラムを実行ファイルに変換し、実行ファイルを組織内に配布することで、Python を動作させるための環境がなくても、Python のプログラムを動作させることができるようになります。

実行ファイルを受け取った相手は、自分の PC 上で実行ファイルをダブルクリックするだけで、Python の自動化プログラムを実行することができ、非常に便利です。

(3)PyInstaller について

実行ファイルの変換には「PyInstaller」を使います。

PyInstaller を使うことで、
・Python 本体
・作成した Python プログラム
・Python プログラムで使用しているモジュール
をまとめて実行ファイルにすることができます。



また、PyInstaller で作成した実行ファイルは、**OS 依存**となります。
Windows で作成した実行ファイルは Windows 上でのみ動作し、Mac で作成した実行ファイルは Mac 上でのみ動作します。

(4)PyInstaller による実行ファイル作成の手順

①PyInstaller のインストール
②拡張子「.ipynb」のファイルを「.py」のファイルに変換する
③実行ファイルを作成する
④動作確認する

8. 現場で役立つ便利な技術・テクニック③
～マウス・キーボードの自動操作

RPA ツールでできることは、Python でもできます。
「RPA」とは「Robotic Process Automation/ロボティック・プロセス・オートメーション」の略であり、人間が行っている単純・定型作業などを自動化するためのツールです。

本研修で紹介してきた Excel や Web ブラウザの自動操作以外にも、あらゆるソフトウェアなどを用いた操作を自動化したい場面はあります。

そのような時に、RPA で代表的な技術とされる「マウス・キーボードの自動操作」ができるようになると、業務自動化の幅が広がります。

ここではコードの解説は行いませんが、「Python でもマウス・キーボードの自動操作は実現可能である」ということは押さえておきましょう。

Python でマウス・キーボードの自動操作を行うためには、「PyAutoGUI」という外部ライブラリをインストールする必要があります。

まとめ

ここまで「Python でできる 5 つのこと」「現場で役立つ便利な技術・テクニック」を見てきましたが、皆さんの業務の中でも自動化・効率化できそうなことは思いつきましたでしょうか。
「こんなことまで自動化できるのか」と興味が湧きましたら、ぜひ Python に取り組んでみてもらえれば、大変うれしく思います。
最後までご覧いただき、ありがとうございました。

Python 関連のお役立ち情報を配信しているメールマガジンもございます。
ご興味がある方は下記 URL からご登録くださいませ。
https://www.mrc-s.com/insource/python_mail.html