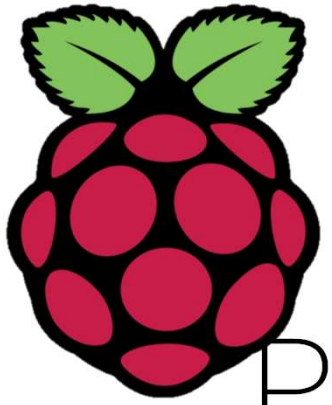


Python Thonnyの使い方

https://www.indoorcorgielec.com/resources/raspberry-pi/raspberry-pi-python/

Pythonとは

Pythonはプログラミング言語の１つですが、特徴をあげると以下の３つです。

- スクリプト言語
- 多彩な用途
- インデント

「スクリプト言語」とは、C言語やJAVAのようにコンパイルやビルドが不要で、
プログラムを書いたら即実行が可能という意味です。

Pythonは、今回のようにディープラーニングや機械学習などに
多く使われていますが、組込みやWebアプリケーションにも使われています。

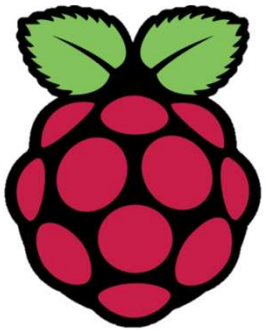
構文のまとまりにカッコを使わず、インデント（字下げ）で表現するのが特徴的です

Thonny

- プログラミングのコードを記述するためのエディタは様々なものがありますが、ここでは最新のRaspberryPiに標準でインストールされている"Thonny"というGUIエディタを使います。

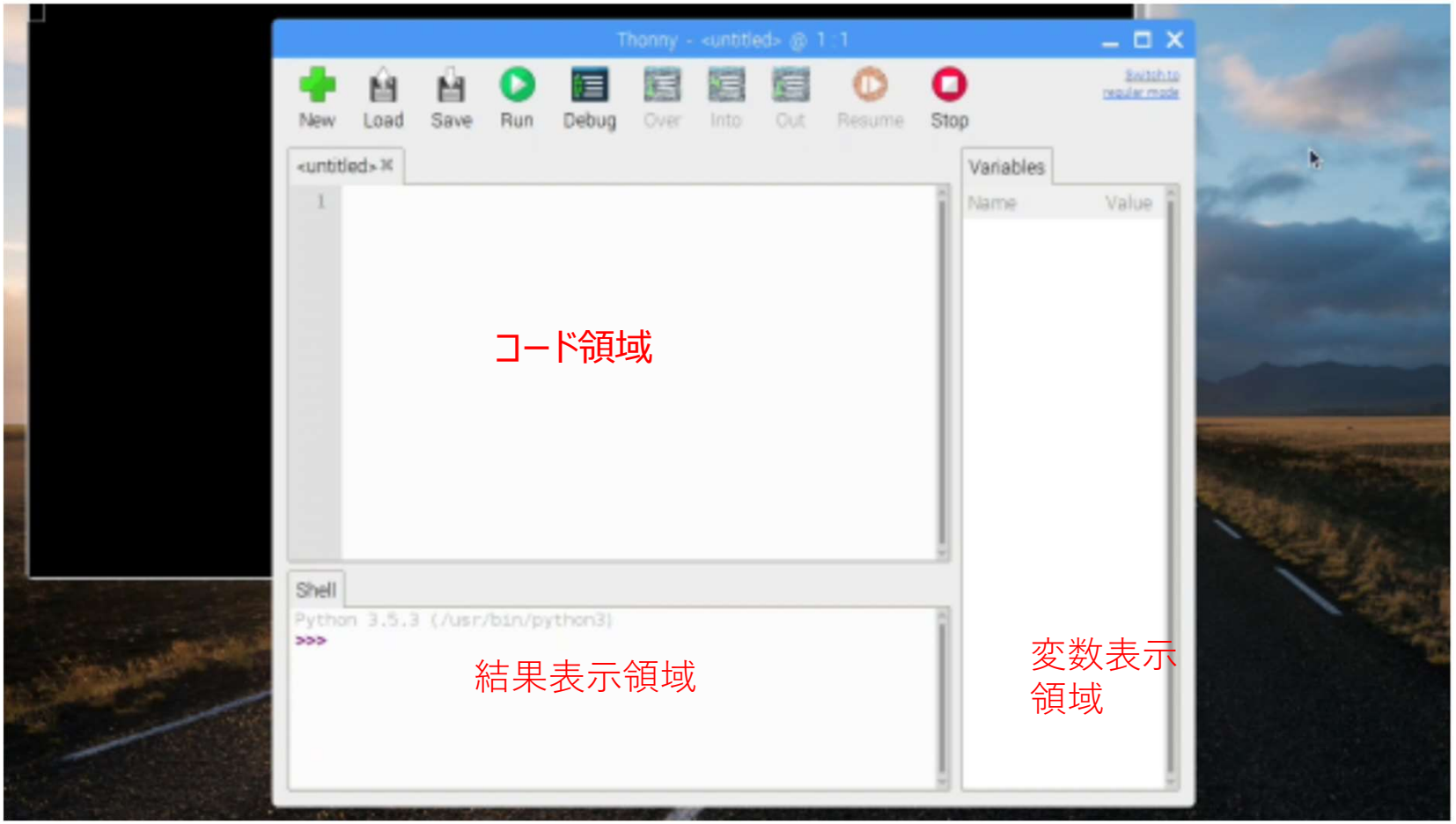
- 始めにコードを保存するための作業場所を作成しましょう。ユーザーのホームディレクトリの直下にworkspaceというフォルダを作成します。ファイルマネージャー（左上のフォルダアイコン）を開いて、右クリックで新規作成＞フォルダで簡単に作成できます。

Thonny　起動方法



「ファイル」→「プログラミング」→ 「Thonny」

Thonnyの画面

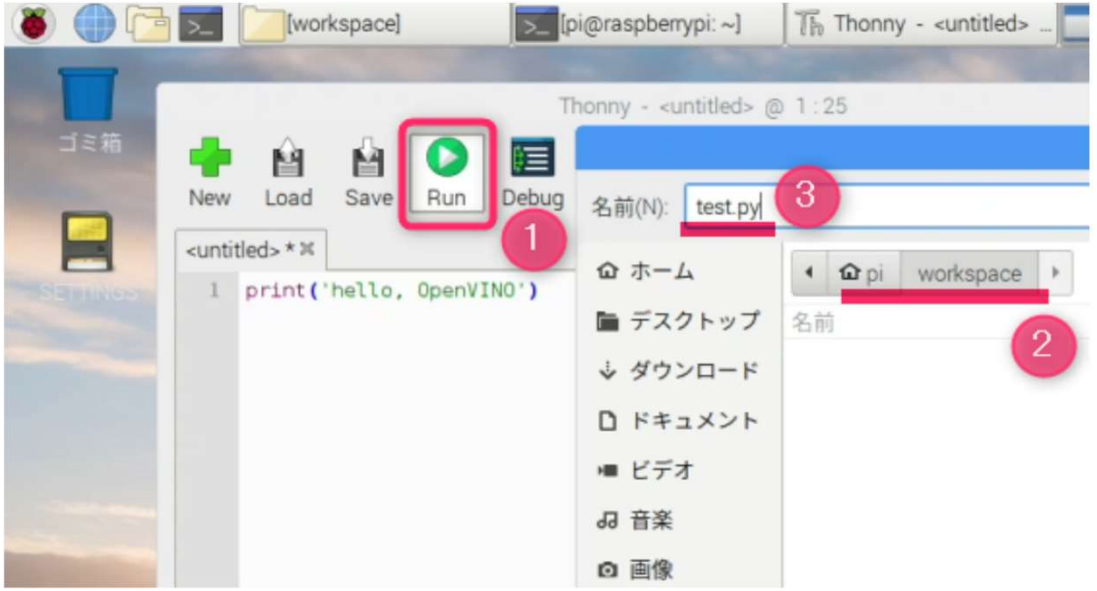

Thonny - <untitled> @ 1 : 1
New
Load
Save
Run
Debug
Over
Into
Out
Resume
Stop
Switch to regular mode
<untitled>
1
Variables
Name
Value
Shell
Python 3.5.3 (/usr/bin/python3)
>>>
コード領域
結果表示領域
変数表示領域

メニュー

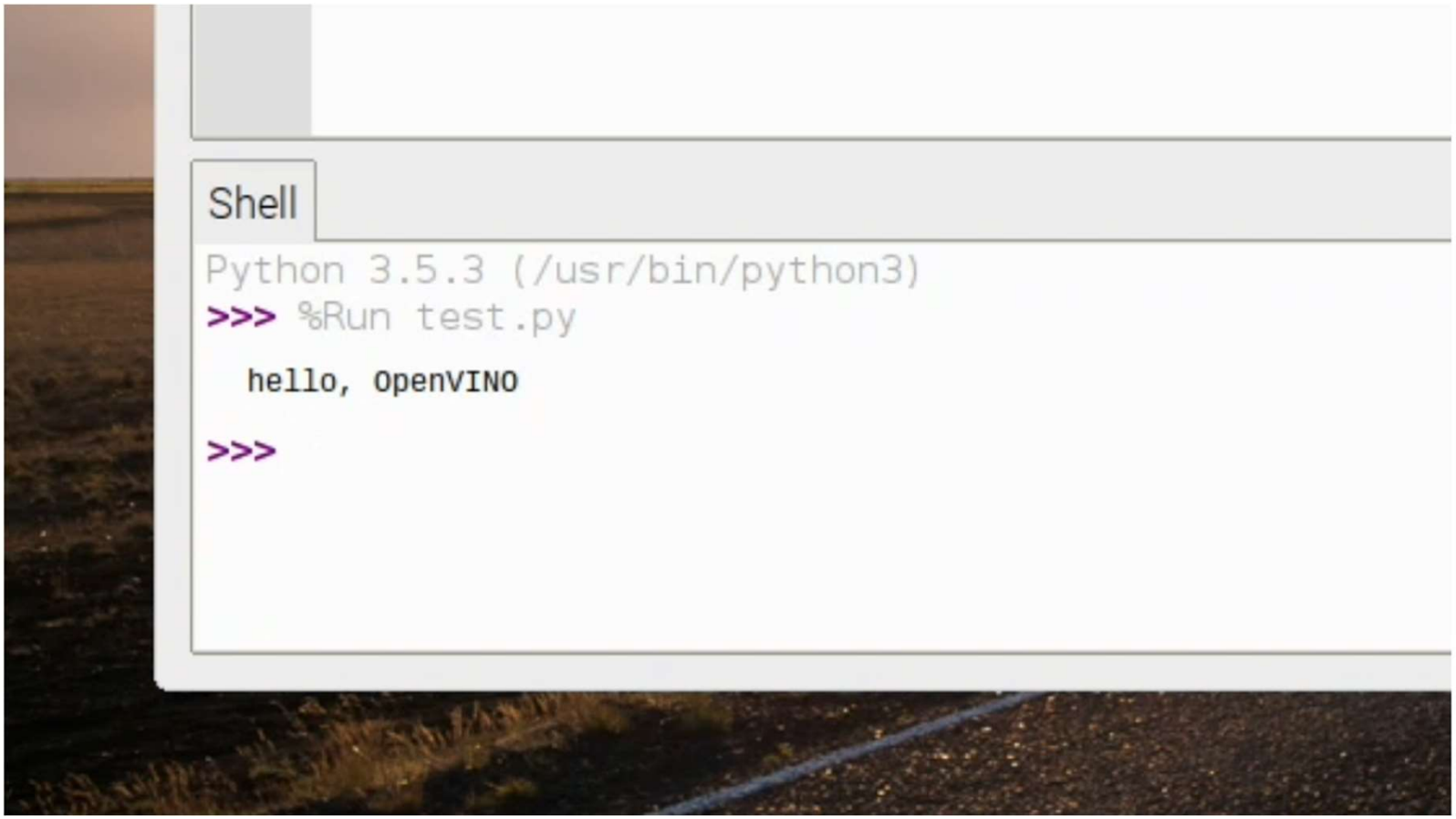
- 左上がコードを記述する領域、左下が結果を表示する領域、右側は変数の値を表示する領域です。右側の領域は今回は特に使わないので、狭くしたりクローズしたりしても構いません。よく使うメニューは５つだけです。
- New：新規作成
- Load：既存のコードを開く
- Save：上書き保存
- Run：プログラム実行
- Stop：プログラム強制停止

試しにコードを書いて実行してみましょう

- print('hello, OpenVINO')
- 書いたらRunボタンを押します。
- そうすると保存先を求められるので workspaceを選んで、テキストボックスにtest.pyと記入して右下のOKボタンを押すかEnterキーを押すと実行されます。



実行結果



コメント

#記号を書くことで、それ以降から行の終わりまでがコメント行になります。
コメント行とは、プログラム実行時に無視されるということです。主にプログラムに関する注釈を記述するために使用します。
#こんな感じで書きます
なお、複数行をコメントしたい場合は様々な方法がありますが、Thonnyでは複数行を選択してからCtrl + #キーを押すことで一気にコメント化とコメント解除が可能です。

文字列

- 文字列はシングルクォート ' もしくは ダブルクォート " で囲みます。クラゲはシングルクォートで統一しようと思います。文字列を表示するには以下のような形でprintを使います。
- print('hello jellyfish')
-

変数・代入・演算

- 変数の型宣言は不要です。整数も小数も文字列も同じように扱えます。代入は他のプログラミング言語同様 = を用いて、左側に右側の値を代入します。

```
a = 2019
b = 3.14
c = 'jellyfish'

print(a)
print(b)
print(c)
```

変数・代入・演算

- 記号 +, -, *, / を用いれば四則演算可能です。

```
a = 100 + 200
b = 24 - 8
c = 10 * 3.14

print(a)
print(b)
print(c)
```

変数･代入･演算

- 太字の式のような書き方も良く使われます。

自分自身に100を加え、結果的に600になります

```
a = 500
a = a + 100
print(a)
```

- 「ある変数に数値を足して、結果を同じ変数に代入する」という書き方は良く使われますので覚えておいてください。
- a = a + 1 など 1 を足す場合は "インクリメント" と呼ばれます。

型変換

- 型を変更させる必要がある場合は、型変換を行う必要があります。
  例えば、文字列から数値の小数に変更させる例です。floatを使います。

```
a = '3.14’
b = 10 * float(a)
print(b)
```

- こちらは小数から整数に変更させる例です。intを使います。

```
a = 3.14
b = 10 * int(a)
print(b)
```

-

if文を使って　**条件分岐**

Pythonでの注意点はコロンとインデントです。コロンはifの最後に書きます。
インデントとは字下げのことで、以下のコードの場合はprint文の行です。
TABキーを使ってスペースを空けます。
以下のコードは、変数time_of_sleepingの値を見て、8より大きかった場合は
print文で表示を行い、それ以外の場合は何もせずプログラムを終了します。

```
time_of_sleeping = 10

if time_of_sleeping > 8:
        print('よく寝ました')
```

if文を使って　**条件分岐**

- ifと半角スペースの後に書かれているtime_of_sleeping > 8は条件式と呼ばれる部分です。
- 条件式が合っていれば、その下のインデントされたブロックが実行されます。
  条件式には>などの比較演算子が使われます。比較演算子は他に、< <= >= == != などがあります。
- 以下の例でtime_of_sleepingの値を変更してみて、条件式が合っているとき・合っていないときで、
- どのブロックが実行されているのか確認してください。

```
time_of_sleeping = 6
if time_of_sleeping < 8:
        print('しっかり睡眠を取りましょう！')
        print('クラゲからのアドバイスでした。')
print('以上')
```

if文を使って　**条件分岐　ブール型**

- ブール型はTrueとFalseのどちらかしか持てない変数のことです。以下のコードでは、ブール変数sleepyの値がTrueのときにprint文を表示しています。

```
sleepy = False
if sleepy == True:
    print('仮眠しよう！')
```

リスト　（配列）

- リストは他のプログラミングでいうところの配列にあたります
  一つの値ではなく、一連の値を格納できる変数です。
  代表的な使い方は、

- **リストの初期化**
- このように各要素をカンマで区切り、カッコ[ ]内に記述します。
- a = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

リストの参照

- リスト名[n]でリストの要素にアクセスできます
  Pythonでは要素番号は 0 から始まります。
  例えば n が 5 の場合は 6番目 の要素をアクセスすることになります。

```
a = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
b = a[5]
    print(b)
```

- 今回は、リストの中身を整数値にしましたが、小数や文字列など他の変数も扱うことも可能です。

辞書

- Pythonの辞書は一言で言うと「キーワード付きのリスト」です。
- 以下は果物の値段を変数に格納して参照する例です。リストで取り扱うことも可能ですが、辞書を使うことにより分かりやすくなります。

```
# 辞書の初期化
a = {'apple':148, 'banana':99, 'orange':358}
# 辞書の参照
b = a['orange’]
# 表示
print(b)
```

- 特徴的なのは、初期化のときに{ }を使うことと、キーワード:値というセットで格納することです。
  リスト同様、キーワードと値は整数値以外も取り扱うことが可能です。
- ただし、キーワードに関しては文字列を使うことが一般的です。

タプル

- タプルとは要素を変更できないリストです。
- 初期化時のカッコの形が( )であり、リストや辞書と異なります。

```
a = (10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
b = a[5]
print(b)
```

- 座標（X・Y）、画像の大きさ（幅・高さ）、色情報（青・緑・赤）などに使われます。