

Python Introduction

PYTHON INTRODUCTION



Python Introduction

In this session

- What is Python?
- What is Anaconda?
- Popularity rank
- Why use Anaconda in Machine Learning?
- Anaconda installation
- PyCharm installation
- Hello world
- Basic calculation
- Variable assignment
- Basic Operator
- Data Structure (Tuple, List, Matrix, Dictionaries)
- If Statement
- For Loop
- Function

Python Introduction

What is Python?

- Computer language
- Interpreter
- Multi-paradigm: (OOP, imperative, functional, procedural)
- Typing: dynamic
- Statistical and graphics
- Linear and nonlinear modelling
- Age 26 (C# 17)
- Free Software (GNU project)
- Linux, Windows and MacOS
- One of the most powerful ML language
- Tool for ML exploration
- Good for building a production model
- Supported in Azure ML Studio











Python Introduction

What is Anaconda?

- Anaconda is a python and R distribution
- Provide everything you need for data science "out of the box"
- The core python language
- 100+ python "packages" (libraries)
- Spyder and Jupyter ready
- conda: Anaconda's own package manager
- Microsoft Azure ML Studio fully supports

Python Introduction

Popularity rank

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9



Source: The 2016 Top Programming Languages

<http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages>

Python Introduction

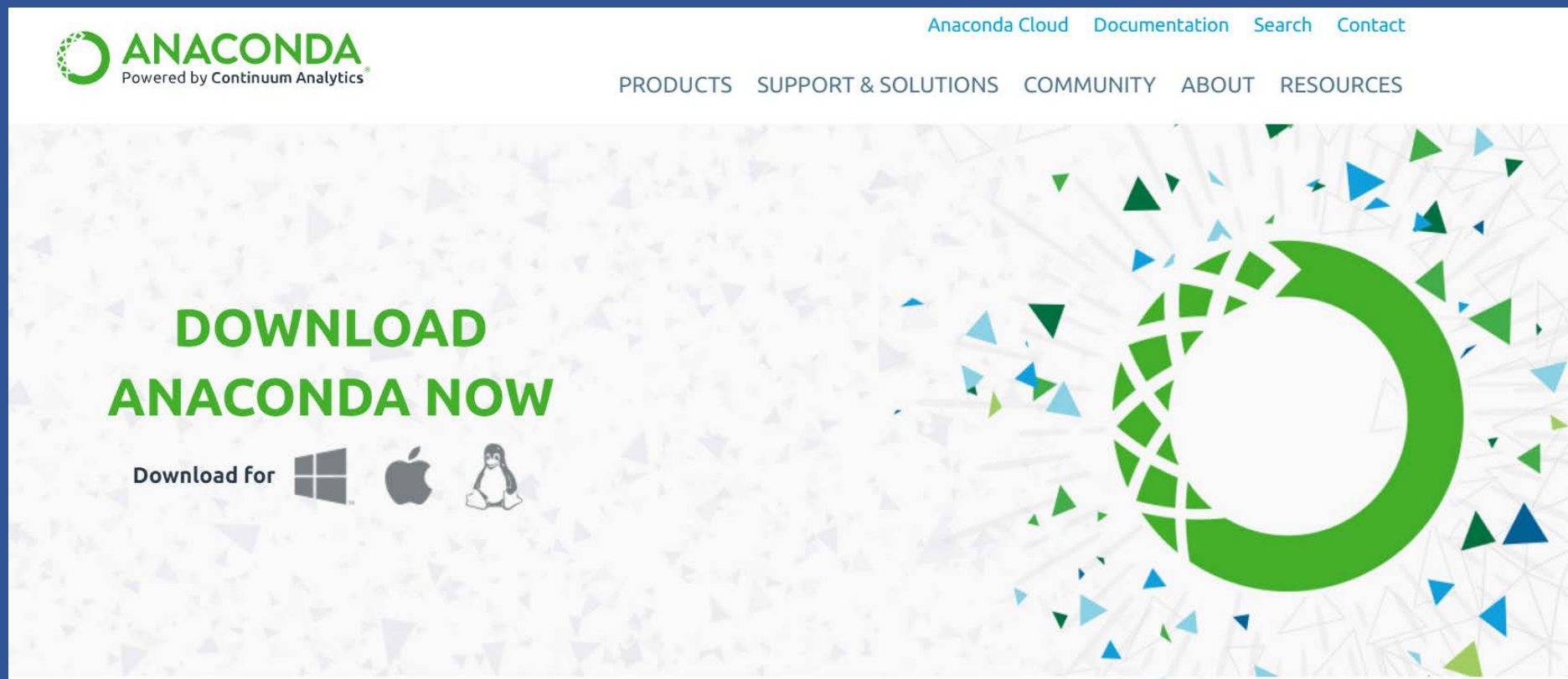
Why use Anaconda in Machine Learning?

- 720+ data science packages
 - visualizations
 - machine learning
 - deep learning
 - Big Data
 - tensor calculation
- Native access
 - HDFS
 - Amazon S3
- Distributed computing
- GPU supercharged
- Agile & fast experimentation data science
- Use Microsoft Excel® to perform predictive analytics
- Query and transform Big Data
- Easily deploy production
- Spyder & Jupiter build-in

Python Introduction

Anaconda installation

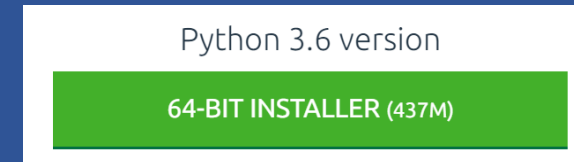
<https://www.continuum.io/downloads>



Python Introduction

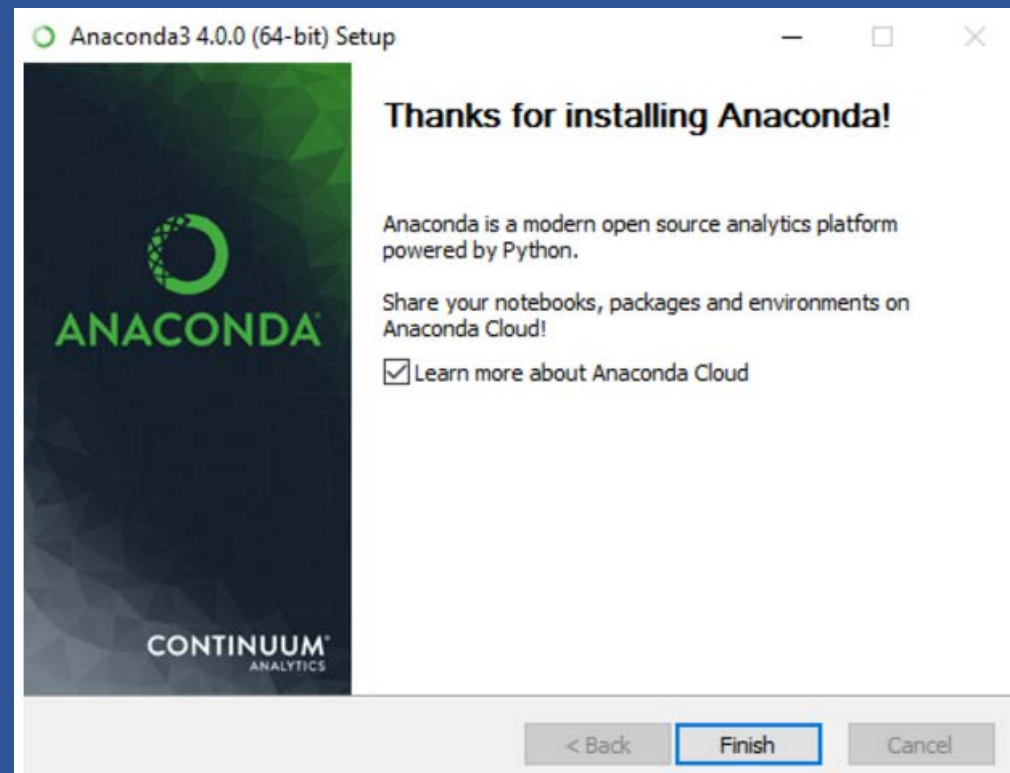
Anaconda installation

1. Click Python 3.6 version
2. Click Run to launch the installer.
3. Click Next.
4. Read the licensing terms and click I Agree.
5. Select an install for “Just Me”
6. Select a destination folder to install Anaconda and click Next.
7. Choose whether to add Anaconda to your PATH environment variable.
8. Choose whether to register Anaconda as your default Python 3.6.
9. Click Install.
10. Click Next.



Python Introduction

Anaconda installation

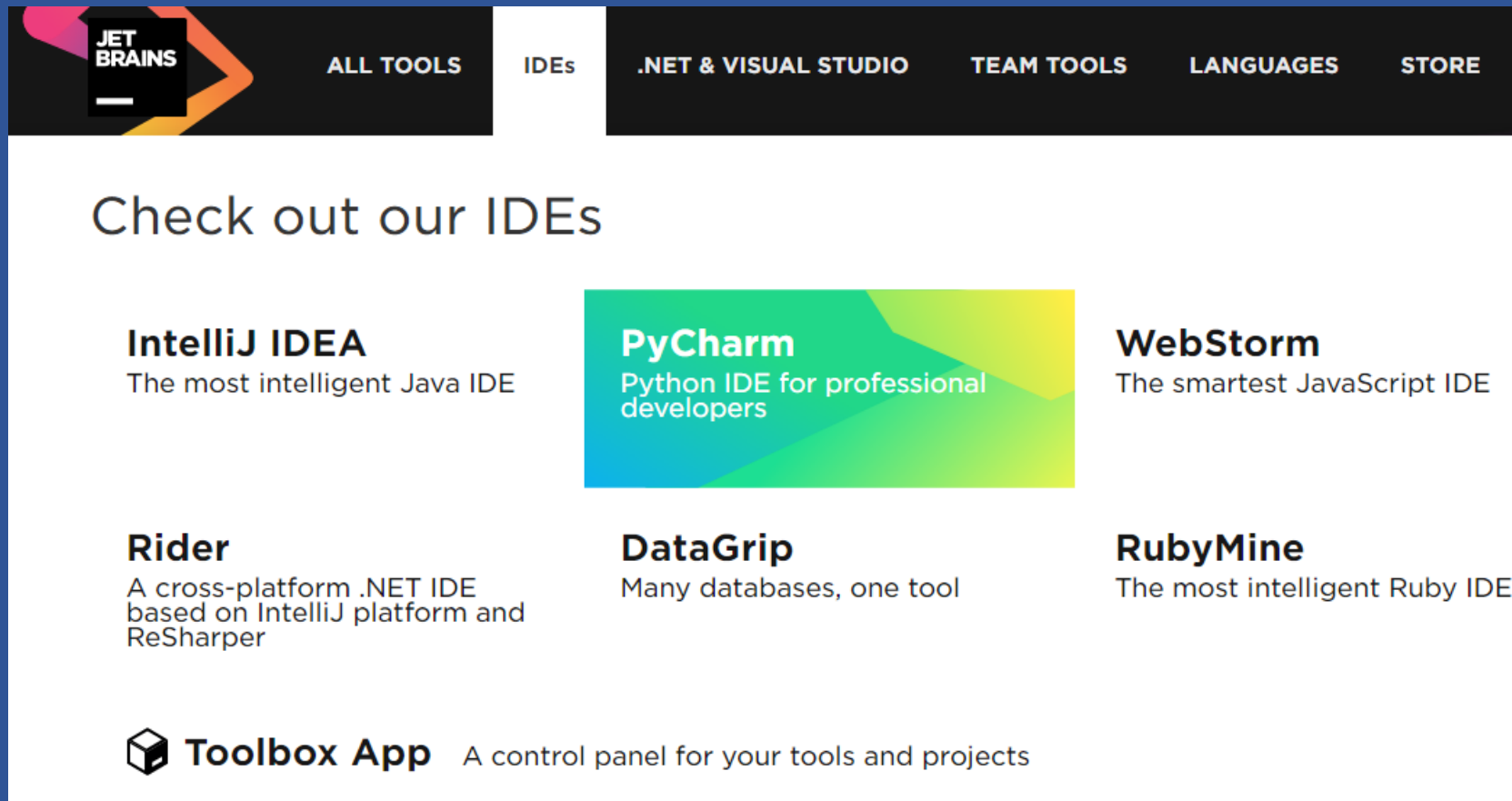


11. After a successful installation you will see the “Thanks for installing Anaconda” image:

Python Introduction

PyCharm installation

<https://www.jetbrains.com/>



The screenshot shows the JetBrains website's navigation bar and a section titled "Check out our IDEs". The navigation bar includes the JetBrains logo, "ALL TOOLS", "IDEs" (highlighted), ".NET & VISUAL STUDIO", "TEAM TOOLS", "LANGUAGES", and "STORE". The "Check out our IDEs" section features six IDEs: IntelliJ IDEA, PyCharm, WebStorm, Rider, DataGrip, and RubyMine, each with a brief description. At the bottom, there is a "Toolbox App" section.

JETBRAINS

ALL TOOLS IDEs .NET & VISUAL STUDIO TEAM TOOLS LANGUAGES STORE

Check out our IDEs

IntelliJ IDEA
The most intelligent Java IDE

PyCharm
Python IDE for professional developers

WebStorm
The smartest JavaScript IDE

Rider
A cross-platform .NET IDE based on IntelliJ platform and ReSharper

DataGrip
Many databases, one tool

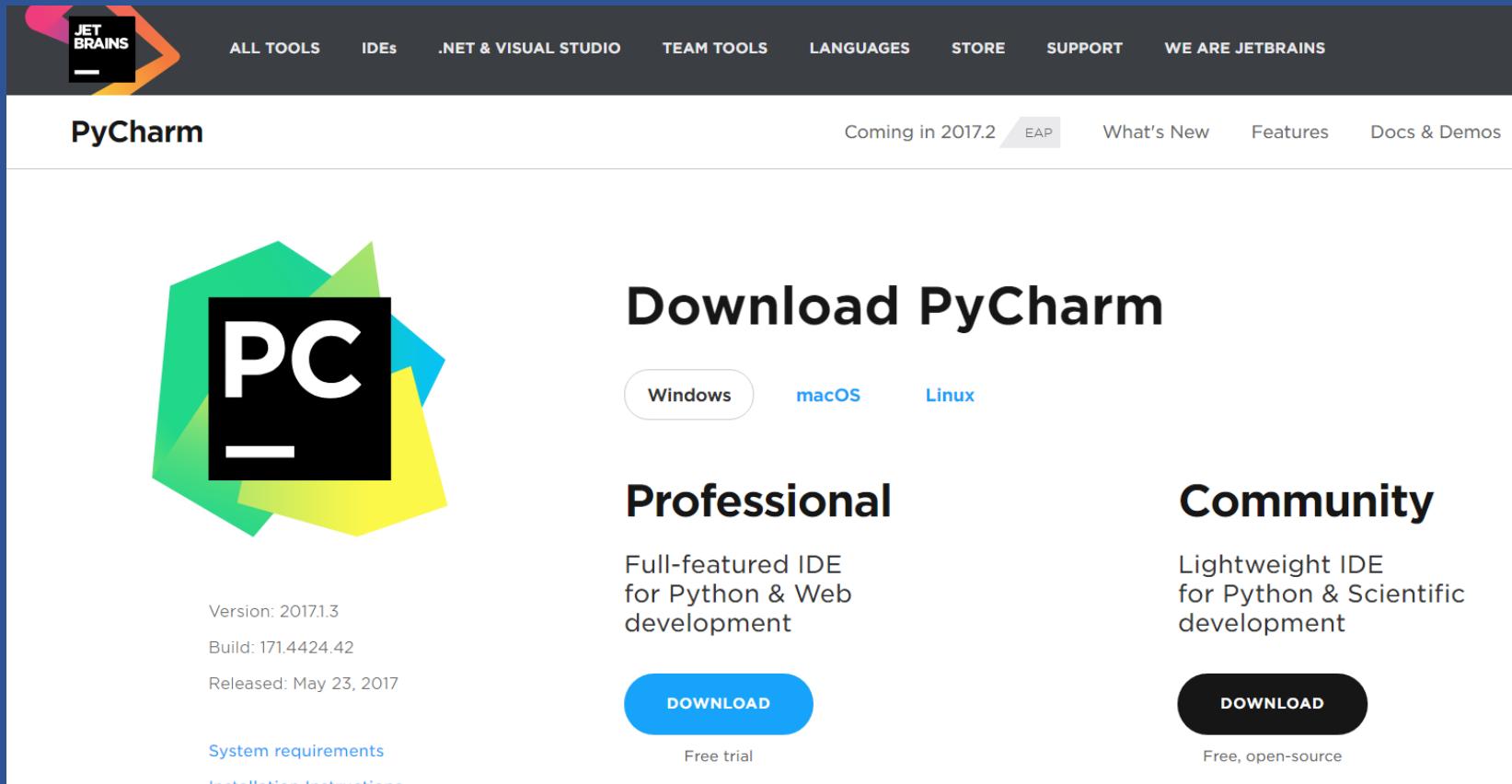
RubyMine
The most intelligent Ruby IDE

Toolbox App A control panel for your tools and projects

Python Introduction

PyCharm installation

Click Professional download



The screenshot shows the PyCharm download page. At the top is a dark navigation bar with the JetBrains logo and links: ALL TOOLS, IDEs, .NET & VISUAL STUDIO, TEAM TOOLS, LANGUAGES, STORE, SUPPORT, and WE ARE JETBRAINS. Below this is a white header with the PyCharm logo, a status bar indicating 'Coming in 2017.2 EAP', and links for 'What's New', 'Features', and 'Docs & Demos'. The main content area has a large 'PC' logo on the left. To its right is the heading 'Download PyCharm' with three tabs: 'Windows' (selected), 'macOS', and 'Linux'. Below the tabs are two columns: 'Professional' and 'Community'. The 'Professional' column describes it as a 'Full-featured IDE for Python & Web development' with a blue 'DOWNLOAD' button and 'Free trial' text. The 'Community' column describes it as a 'Lightweight IDE for Python & Scientific development' with a black 'DOWNLOAD' button and 'Free, open-source' text. On the left side, below the logo, are version details: 'Version: 2017.1.3', 'Build: 171.4424.42', and 'Released: May 23, 2017', followed by links for 'System requirements' and 'Installation Instructions'.

PyCharm Coming in 2017.2 EAP [What's New](#) [Features](#) [Docs & Demos](#)

Download PyCharm

[Windows](#) [macOS](#) [Linux](#)

Professional

Full-featured IDE for Python & Web development

[DOWNLOAD](#)

Free trial

Community

Lightweight IDE for Python & Scientific development

[DOWNLOAD](#)

Free, open-source

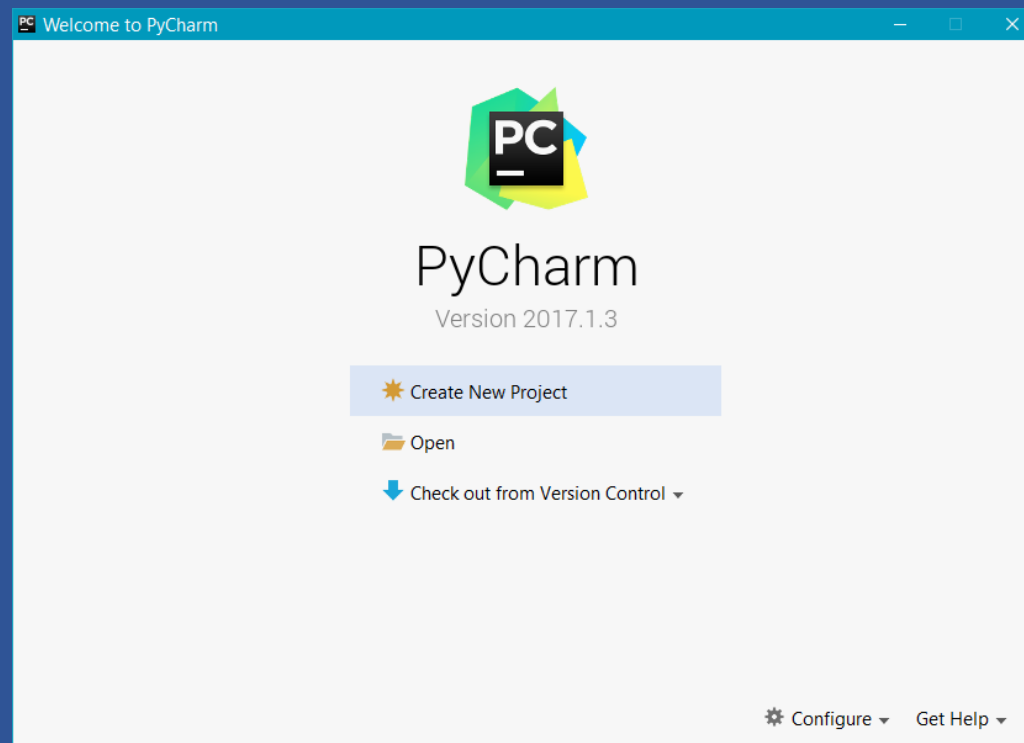
Version: 2017.1.3
Build: 171.4424.42
Released: May 23, 2017

[System requirements](#)
[Installation Instructions](#)

Python Introduction

PyCharm installation

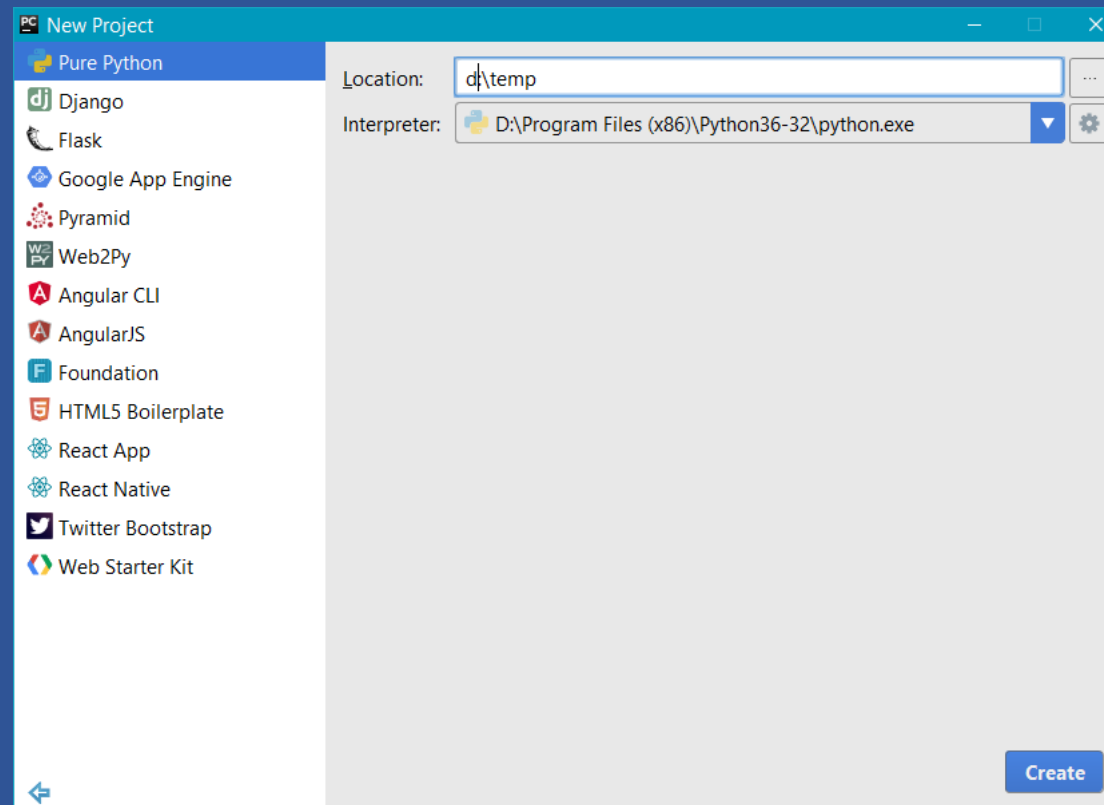
Install with all default setting and run



Python Introduction

Hello world

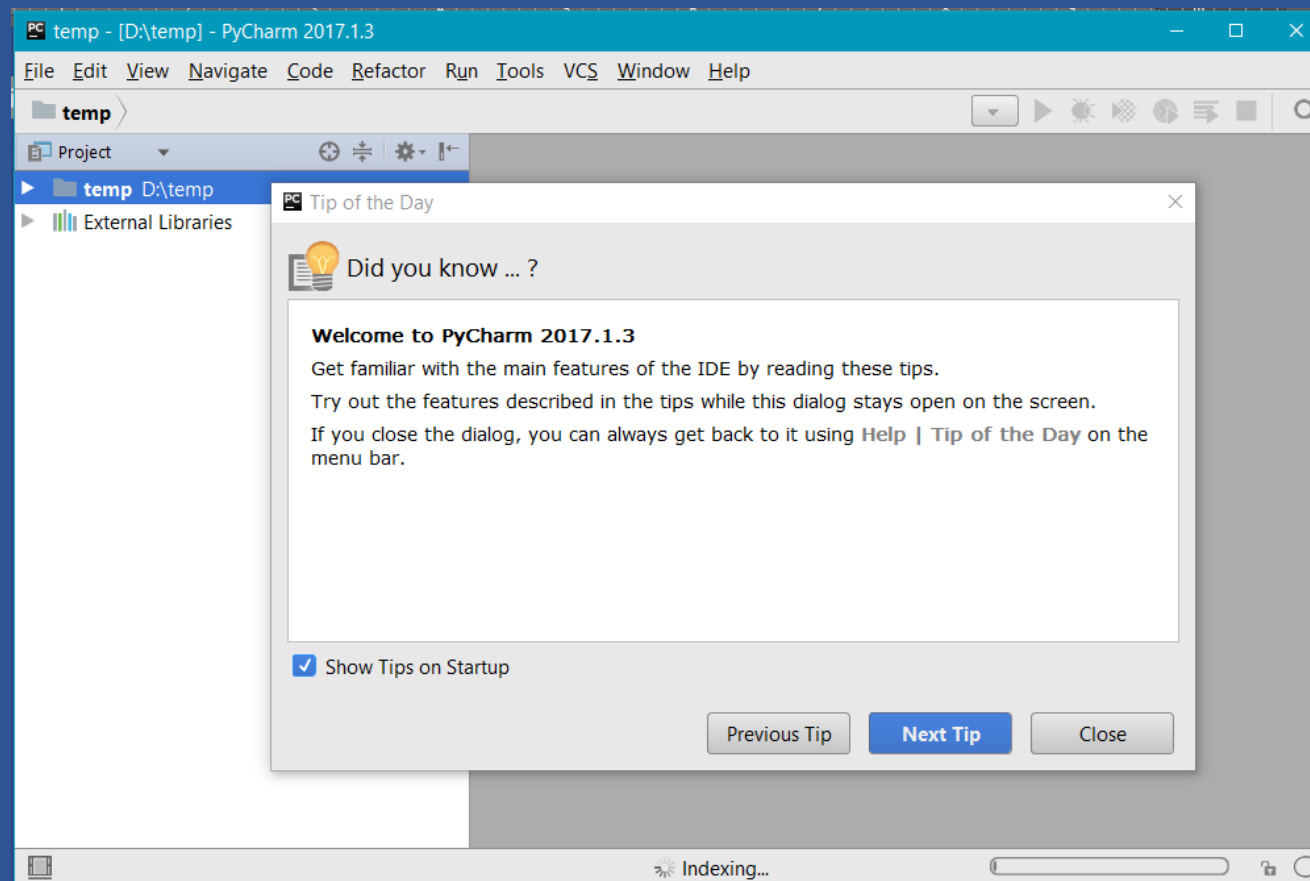
1. Create new folder c:\temp (or d:\temp)
2. In Location box type c:\temp (or d:\temp)



Python Introduction

Hello world

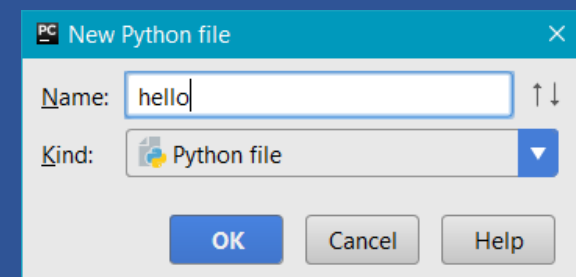
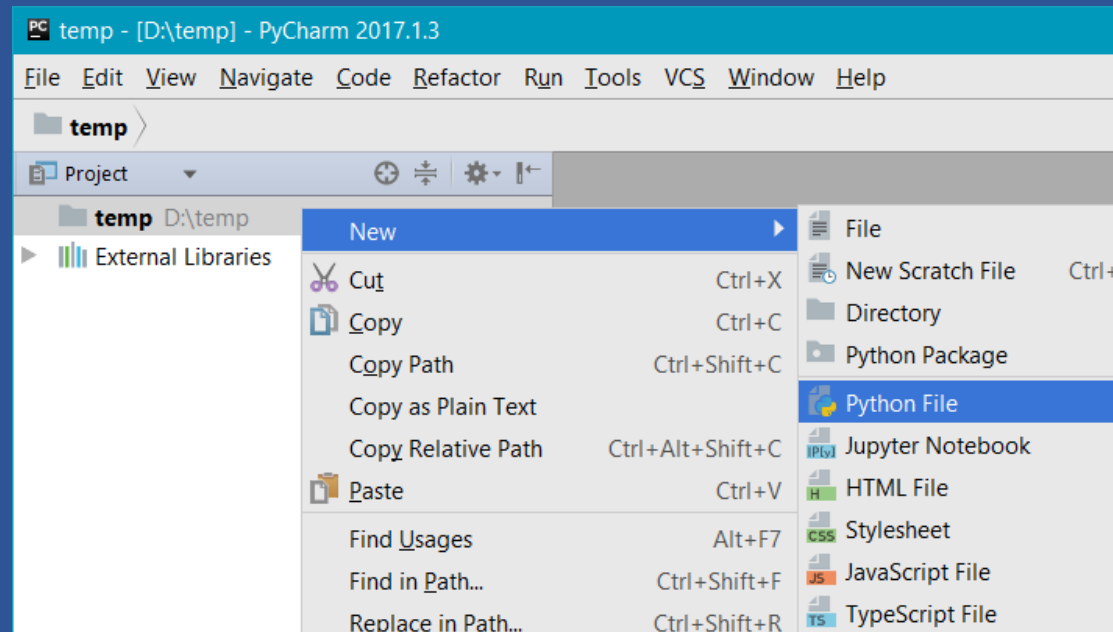
3. Click create



Python Introduction

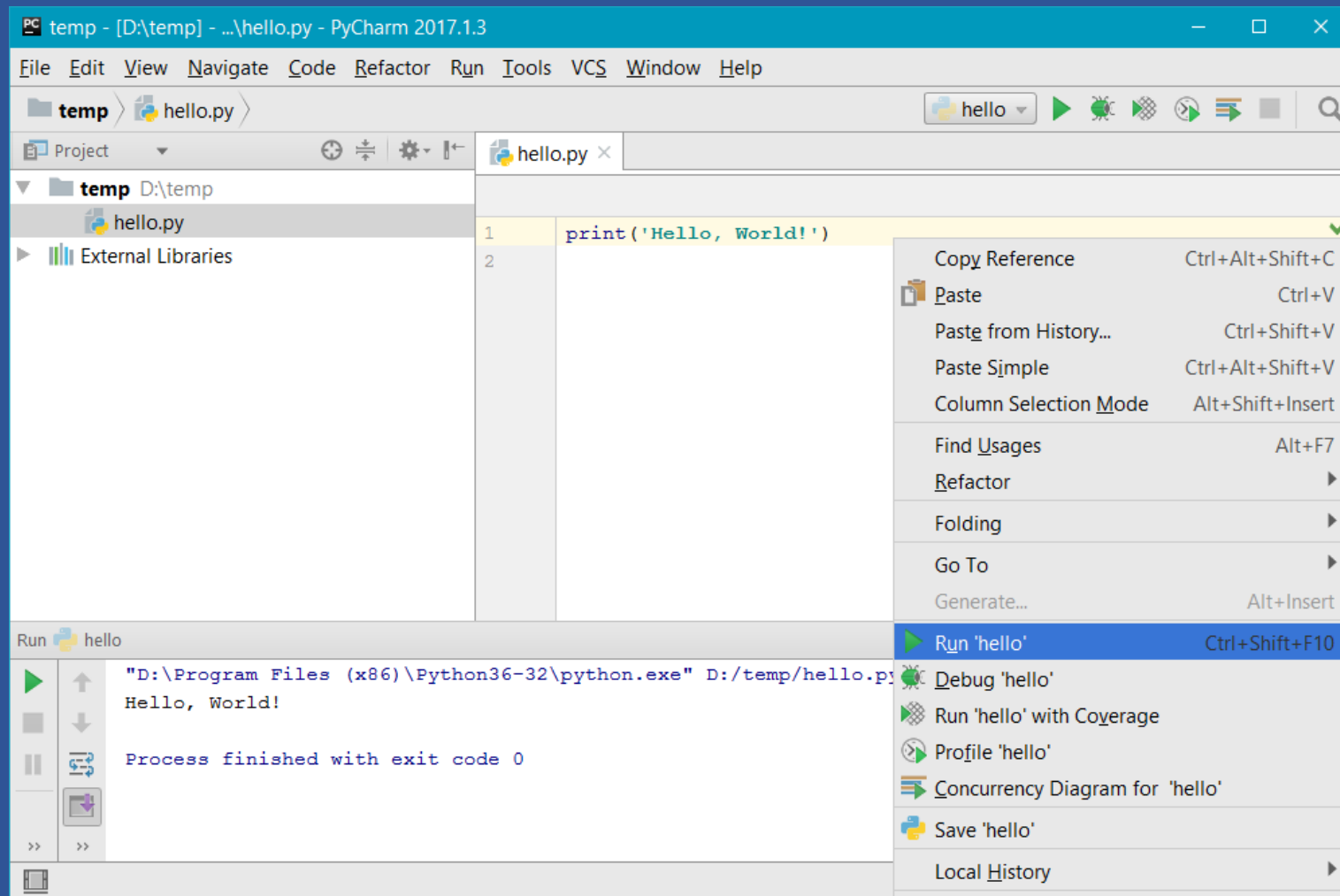
Hello world

4. Right click at temp D:\temp
5. Click New
6. Click Python File
7. In Name box type hello
8. Click OK



Python Introduction

Hello world



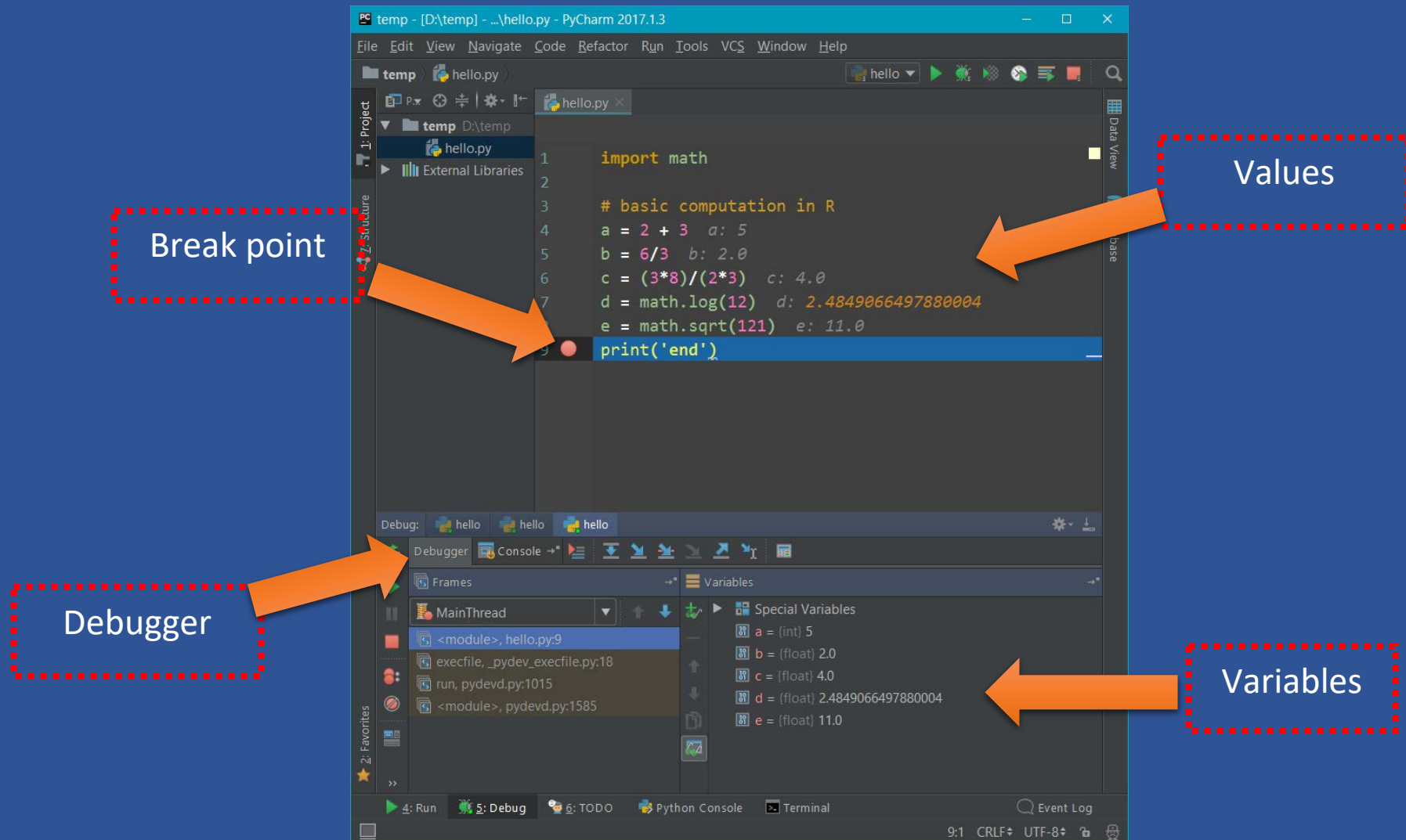
Python Introduction

Basic computation

```
1  import math
2
3  # basic computation in R
4  a = 2 + 3  a: 5
5  b = 6/3  b: 2.0
6  c = (3*8)/(2*3)  c: 4.0
7  d = math.log(12)  d: 2.4849066497880004
8  e = math.sqrt(121)  e: 11.0
9  print('end')
```

Python Introduction

Debugging in PyCharm



Python Introduction

Variable declaration

```
1      # variable assignment & declaration
2      a = 1
3      b = 2
4      c = a + b
5      k = (a * 2) + (b * 3)
6      o = 1; p = 'dog'; q = 3
7      x = y = z = 1          # multiple
8      l, m, n = 1, 2, 'cat'
9      print('end')
10     |
```

Python Introduction

Basic data type

```
1      # Basic data type
2      name = 'laploy'      # character
3      who = name + ' v.'  # string concat
4      price = 1500         # integer
5      kilo = 1.5           # floating point
6      x = price + kilo     # automatic type casting
7      yes = True          # bool
8      gen = 123            # integer
9      gen = 'hello'       # change to string
10     print('end')
```

Python Introduction

Basic Operator

```
1      # Basic Operators
2      # Arithmetic
3      a = (2 + 3) * 2  a: 10
4      x, y = 2, 3  x: 2  y: 3
5      b = (x + y) * 2  b: 10
6      c = (2 * 3) / 2  c: 3.0
7      d = 2 / (3 * 4)  d: 0.16666666666666666
8      e = 3 * (4/2)  e: 6.0
9      # tuple and operator +
10     v = (1, 2, 3)  v: <class 'tuple'>: (1, 2, 3)
11     t = (2, 2, 1)  t: <class 'tuple'>: (2, 2, 1)
12     f = v + t  f: <class 'tuple'>: (1, 2, 3, 2, 2, 1)
13     # relation operator
14     x1 = a > c  x1: True
15     x2 = a < d  x2: False
16     x3 = a == b  x3: True
17     x4 = a != b  x4: False
18     print('end')
```

Python Introduction

Data Structure : List

```
1  L1 = [] # An empty list L1: <class 'list'>: []
2  # Four items: indexes 0..3
3  L2 = [123, 'abc', 1.23, {}] L2: <class 'list'>: [123, 'abc', 1.23, {}]
4  # Nested sublist
5  L3 = ['Bob', 40.0, ['dev', 'mgr']] L3: <class 'list'>: ['Bob', 40.0, ['dev', 'mgr']]
6  L4 = list('spam') L4: <class 'list'>: ['s', 'p', 'a', 'm']
7  L5 = list(range(-4, 4)) L5: <class 'list'>: [-4, -3, -2, -1, 0, 1, 2, 3]
8  x = [1, 2, 3] x: <class 'list'>: [1, 2, 3]
9  y = [1, 2, 3] y: <class 'list'>: [1, 2, 3]
10 z = x + y z: <class 'list'>: [1, 2, 3, 1, 2, 3]
11 # List comprehensions
12 a1 = [v for v in 'SPAM'] a1: <class 'list'>: ['S', 'P', 'A', 'M']
13 a2 = [v * 4 for v in 'cat'] a2: <class 'list'>: ['cccc', 'aaaa', 'tttt']
14 print('end')
```


Python Introduction

Data Structure : Matrix

```
1      # Matrix
2      matrix = [[1, 2, 3], matrix: <class 'list'>: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
3              [4, 5, 6],
4              [7, 8, 9]]
5      a = matrix[1]    # get a row a: <class 'list'>: [4, 5, 6]
6      b = matrix[1][2]  # get row 1 column 2 b: 6
7      c = [[1, 2, 3], c: <class 'list'>: [[1, 2, 3], [12, 13, 14], [7, 8, 9]]
8              [4, 5, 6],
9              [7, 8, 9]]
10     c[1] = [12, 13, 14]
11     d = [[1, 2, 3], d: <class 'list'>: [[1, 2, 3], [], [7, 8, 9]]
12           [4, 5, 6],
13           [7, 8, 9]]
14     d[1] = []
15     print('end')
```

Python Introduction

Data structure: Dictionary

```
1      # Dictionary
2      a = {'name': 'loy', 'age': 20, 'gender': 'M'}  a: {'name'
3      b = a['name']  b: 'Loy'
4      c = len(a)  c: 3
5      d = 'name' in a  d: True
6      e = list(a.keys())  e: <class 'list'>: ['name', 'age', 'g
7      f = list(a.values())  f: <class 'list'>: ['loy', 20, 'M']
8      g = {'name': 'loy', 'age': 20, 'gender': 'M'}  g: {'name'
9      h = g['name'] = 'lap'  h: 'Lap'
10     i = {'name': 'loy', 'age': 20, 'gender': 'M'}  i: {'age':
11     # delete entry
12     del i['name']
13     j = {'name': 'loy', 'age': 20, 'gender': 'M'}  j: {'name'
14     # add entry
15     l = j['year'] = 3  L: 3
16     print('end')
```


Python Introduction

If statement

```
1      # If Statement
2      x = 1
3      if x == 1:
4          print('same')
5      elif x > 1:
6          print('bigger')
7      else:
8          print('smaller')
9
10     print('end')
```

Python Introduction

Loop statement

```
1      # For Loop statement
2      string = "Hello World"
3      s = ''
4      for x in string:
5          s += x + ' '
6      # while loop
7      start, end = 0, 5
8      while start <= end:
9          print('*')
10         start += 1
11      print('end')
```

Python Introduction

Function Basic

```
1      # Function
2      def add1(a1, b1):
3          r1 = a1 + b1
4          return r1
5
6      a = add1(2, 4)  a: 6
7
8      # anonymous function
9      add2 = lambda a2, b2: a2 + b2
10
11     b = add2(10, 20)  b: 30
12
13     print('end')
```

Python Introduction

More information

Official Python tutorial

<https://docs.python.org/3/tutorial/>

PyCharm 2017 Quick Start Guide

<https://www.jetbrains.com/help/pycharm/quick-start-guide.html>