# 第1章：一起走进python3

## 1.1为什么学习python

原因1：【python简单且强大】

Python的语法是目前最贴近英语自然语言的一种计算机，所以有人戏称python为“说人话的语言”，尤其是对于程序员新手来说，只要你学过初中的英语，那么学习我们本套教程足够了，跟着教程认真学习能入门初级python工程师，找到这方面工作是不成问题的。并且python的免费开源的，拥有简单强大的标准库和第三方库，（关于库的概念我们后期有详细的讲解）让开发人员用起来更加方便快捷，在很短的时间内就可以完成程序的设计。

原因2：【人工智能时代要求】

人工智能是时代发展的趋势，在最近的10年间，大部分劳动力已经被工厂取代，工厂追求智能化，无人化，未来也将会有很多职业被人工智能取代， 顺应趋势的发展就需要不断学习掌握新技能，我们国家不仅从成人职业培训市场对人工智能进行布局，而且还从青少年时代开始人工智能的布局，并颁发了一系类政策比如：国务院发布《新一代人工智能发展规划》，明确指出在中小学阶段设置人工智能相关课程，逐步推广编程教育。浙江已经率先将编程纳入高考等，这些现象都足够证实人工智能时未来发展的必然趋势。

原因3：【三大主流语言之一】

根据TIOBE公开的编程语言排名中，python凭借着语法简单、功能强大等优势，已经成为继Java、C++之后的第三大编程语言，在将近40年的时间里python一直默默无闻地充当脚本语言使用，能一跃成为三大主流语言充分证实了python各方面的优势。

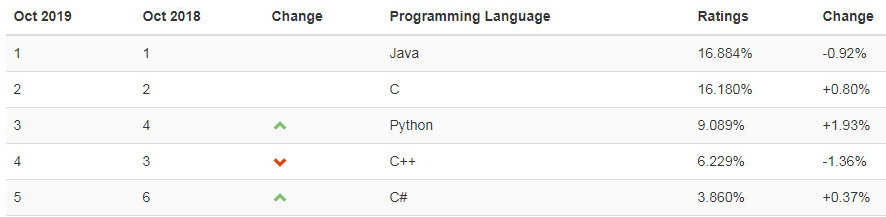


图1.1 2019年10月编程语言排行榜

原因4：【python的发展方向广】

Python可谓是多面手，学习了python之后从事很多IT类工作的方向有很多：python开发工程师、web全栈开发工程师、爬虫工程师、数据挖掘工程师、人工智能工程师、机器学习工程师、python自动化运维等，所以我们学习好python有着非常广阔的发展前景。

## 1.2什么是python语言

### 1.2.1python语言的发展历程简介

Python编程语言的作者，Guido von Rossum，他是荷兰人，在1982年，Guido从阿姆斯特丹大学获得了数学和计算机硕士双学位，并且已经寄出过Pascal、C、Fortran等语言，而这些计算机语言的设计其实都是让计算机更快速的运行，在80年代，苹果刚掀起个人电脑浪潮，并且个人电脑配置非常低，每一k内存都非常宝贵，程序员恨不得用手榨取计算机每一寸能力，内存自动管理、面向对象就更不用提了，会让电脑陷入瘫痪，这种编程方式让Guido非常苦恼，1989年，Guido为了打发无聊的圣诞节，开始尝试编写python语言编译器，python这个名字中文翻译过来是大蟒蛇的意思，它其实来自Guido所钟爱的一部电视连续剧，Monty Python’s Flying Circus（巨蟒剧团的飞行马戏团），他认为python这个名字更能符合他的理想，功能全面、简单易学，可拓展的编程语言。

1991年，第一个python编译器诞生，它使用c语言进行实现的，可以调用c语言的库文件。Python的语法和c有着天然的联系，但是又有ABC语言的基因。如果你接触过c或者是ABC语言你就会深有体会，python就好像是玩乐高积木一样，规定好大框架之后，你就可以在大框架中进行自由的拓展或更改，而你不需要了解每一个积木块是如何制造的。这就让程序员花费时间在程序的逻辑上，不是具体实现，所以有一句话特别流行“人生苦短，我用python”。

目前我们已经进入到互联网的时代，计算机的性能大幅提升，硬件足以满足个人电脑的需求，在这种情况下计算机性能已经不再是编程语言的绊脚石，语言的易用性被大多数人认可，python语言从而得到了大家的认可，已经成为当前三大主流语言之一。很多没有任何计算机编程基础的学员也可以轻松入门，甚至小学三年级的学生都能够把python基础学得特别扎实，所以大家不用担心python学不会。



图1.2 python语言创造者Guido

### 1.2.2python编程语言版本

|  |  |  |
| --- | --- | --- |
| Python 1.0- 1994年1月 | Python 2.0- 2000年10月 | Python 3.0 – 2008年12月 |
| Python 1. 2 - 1995年4月2日  Python 1.3- 1995年10月12日  Python 1.4- 1996年10月25日  Python 1. 5-1997年12月31日  Python 1.6- 2000年9月5日 | Python 2.1 – 2001年4月15日  Python 2.2 – 2001年12月21日  Python 2.3 – 2003年7月29日  Python 2.4- 2004年11月30日  Python 2.5- 2006年9月19日  Python 2.6 - 2008年10月1日  Python 2.7 – 2010年7月4日 | Python 3. 1- 2009年6月27日  Python 3. 2 2011年2月20日  Python 3.3 2012年9月29日  Python 3.4- 2014年5月16日  Python 3.5- 2015年9月13日  Python 3.6- 2016年12月23日  Python 3.7- 2018年6月27日 |

从上述的所罗列的python历史版本中我们可以看出，python一共经历的3代，python3是目前主流的版本，python2已经有很少人在使用了，并且python2和python3本身是不兼容的。根据官方发布的消息，到2020年，对python2就不在进行维护，所以我们直接学习python3即可，目前官网的最新版是python第3代第7版，也就是python3.7。

在软件版本管理规定中，软件版本的命名为X.Y.Z，其中X表示重大类软件更新；Y表示增强类软件更新；Z表示纠正类软件更新；那么你了解了软件版本管理规定，你就很容易理解python之间的版本关系。

## 1.3 python集成开发环境搭建

### 1.3.1什么是python集成开发环境

我们原始的计数方式是使用阿拉伯数字0-9表示的十进制，大家都知道十进制是逢十进一，不管什么国家什么地区、什么历史、什么文化水平都是使用十进制进行表示数字。这是对于我们人类来讲的，但对于计算机呢？它是使用二进制，也就是只有0和1两个数码进行表示的，至于为什么使用0和1表示，在这里我们不会讲解太深奥，你只需要知道计算机只认识0和1，就可以了，比如要表示十进制的122，计算机的二进制表示方式就是：1111010，二进制只能让计算机识别，我们人类没有可读性，那么我们人类如何和计算机进行交流呢？如果我们将编写的程序通过某种方式编译为计算机可以读懂的二进制数码是不是就可以和计算机进行交流沟通了，这就是编程语言的编译器。其实不管什么编程语言，我们安装JAVA，C，Python等等这些编程软件，主要就是安装的这个编程语言的“编译器”。

那么什么是解释器呢？无论是编译器和解释器所实现的功能都是一样的，编译器是先整体编译在执行，解释器是边编译边执行。在这里可以给大家举一通俗例子说明一下，比如我们去餐厅吃一大餐，点了满汉全席，编译器的方式就是厨师把所有的菜给你全部做好，一块端上来，至于你如何吃、在哪里吃随便。而解释器的方式则是厨师做好一道菜，给你端上来一道菜，你吃一道菜，且你必须在饭店吃。



图1.3 python语言执行原理

关于python解释器你清楚了之后，我们再来了解另外一个概念：python的集成开发环境, 拥有编辑、编译、调试、运行等多种功能的集成工具。集成开发环境才是我们编写程序之前需要搭建任务，通俗将就是安装python软件。Python的集成开发环境有很多，常用的有以下几个：python IDLE、Eclipse、sublime Text、pychram、Atom等，其中python IDLE和pycharm是我们所学习的重点，所以我们仅介绍这两款开发工具，其他开发工具，大家如果有兴趣可以在网上查阅。

python IDLE这是python官网提供的一款，界面简洁友好、上手非常简单、是编程新手学习python语言非常好用的集成开发环境。也是我们课程所学习的重点。

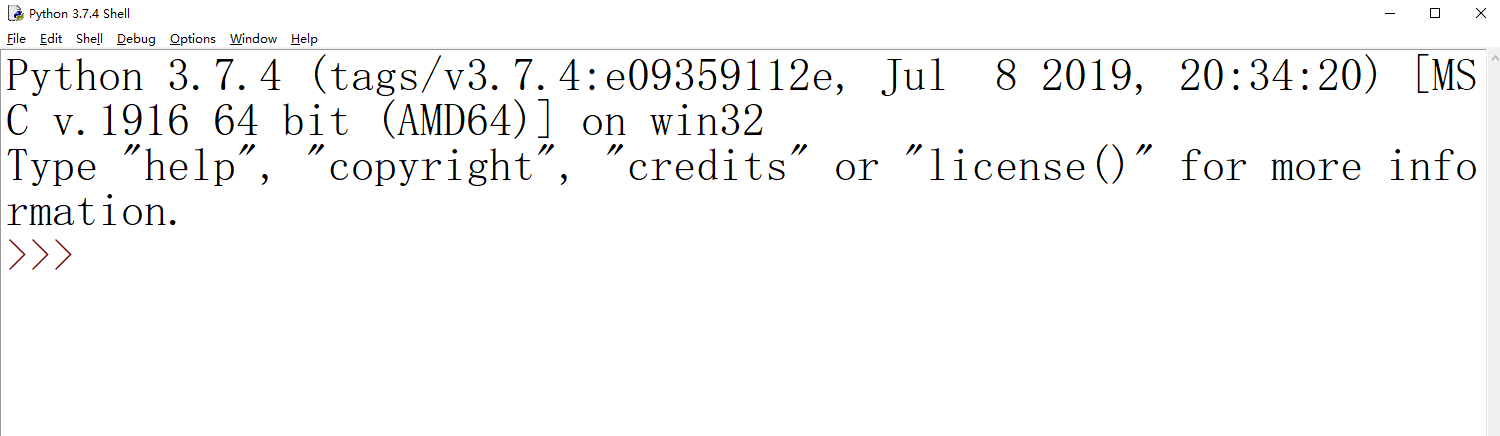


图1.4 python IDLE集成开发工具

pyChram 它是一套可以帮助用户在使用python语言开发时提升效率的开发工具，它拥有调试、语法高亮显示、Project管理、代码跳转、智能提示、自动完成、版本控制等功能，python语言专业人员使用的一款工具，分为社区版和专业版，社区版免费、专业版收费，本课程后面的章节内容讲解使用的是社区免费版，完全能实现我们课程所讲的所有功能，所以大家不必花钱购买专业版。

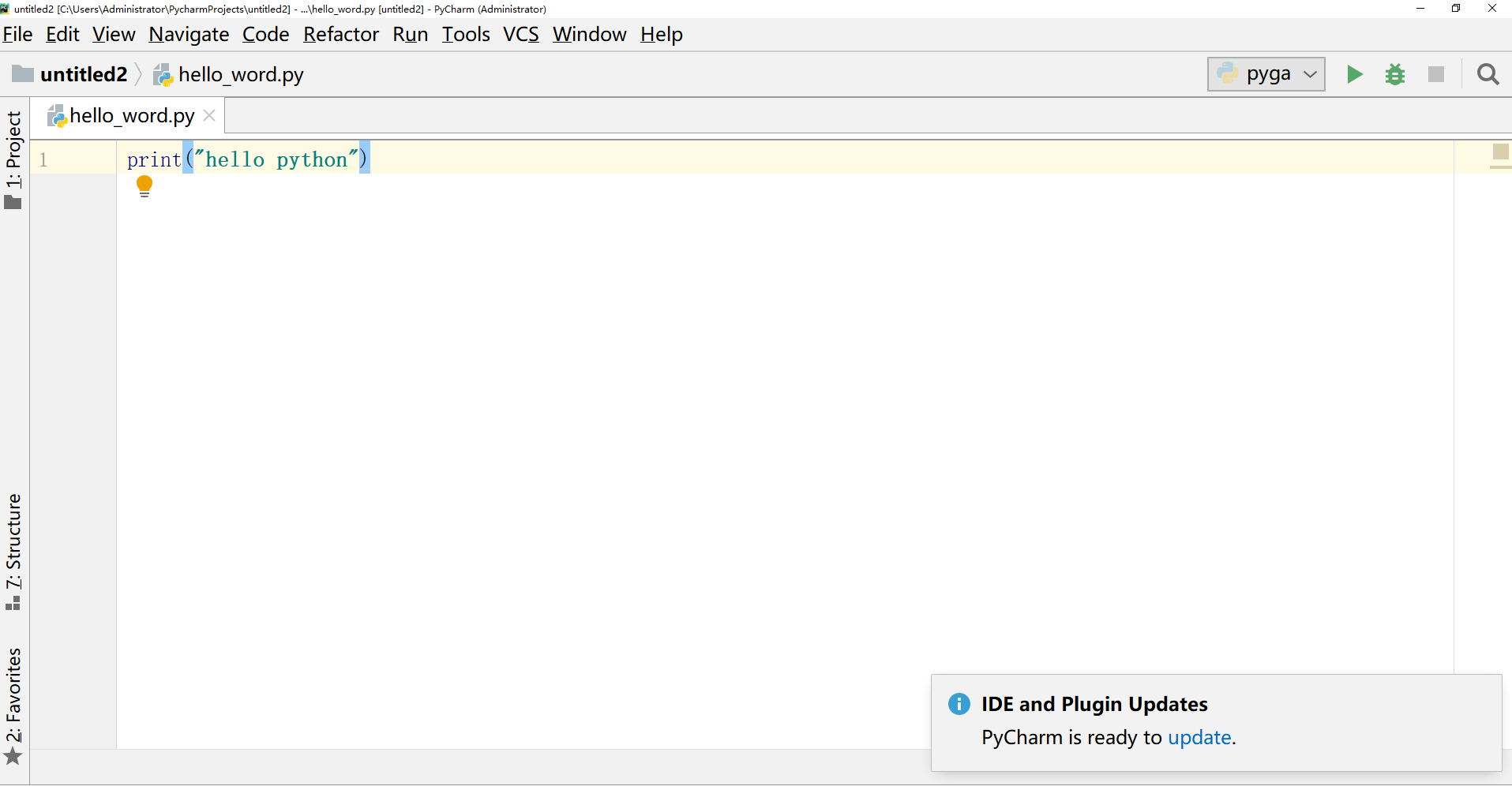


图1.5 pyChram集成开发工具

|  |  |  |
| --- | --- | --- |
| Python 1.0- 1994年1月 | Python 2.0- 2000年10月 | Python 3.0 – 2008年12月 |
| Python 1. 2 - 1995年4月2日  Python 1.3- 1995年10月12日  Python 1.4- 1996年10月25日  Python 1. 5-1997年12月31日  Python 1.6- 2000年9月5日 | Python 2.1 – 2001年4月15日  Python 2.2 – 2001年12月21日  Python 2.3 – 2003年7月29日  Python 2.4- 2004年11月30日  Python 2.5- 2006年9月19日  Python 2.6 - 2008年10月1日  Python 2.7 – 2010年7月4日 | Python 3. 1- 2009年6月27日  Python 3. 2 2011年2月20日  Python 3.3 2012年9月29日  Python 3.4- 2014年5月16日  Python 3.5- 2015年9月13日  Python 3.6- 2016年12月23日  Python 3.7- 2018年6月27日 |