Pycharm的基础设置

[file]--[Setting]/[Defaut Settings]

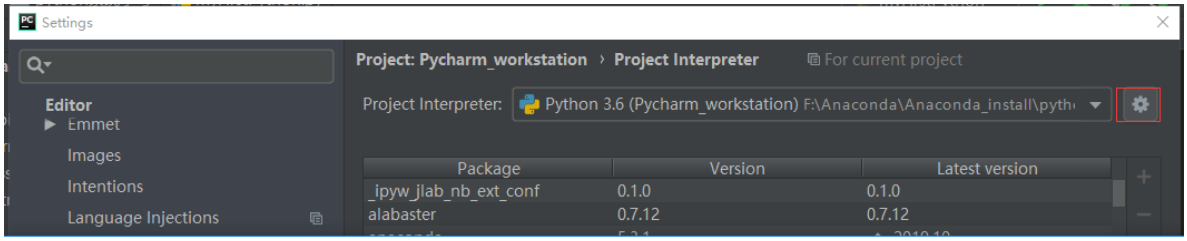
修改代码文字格式

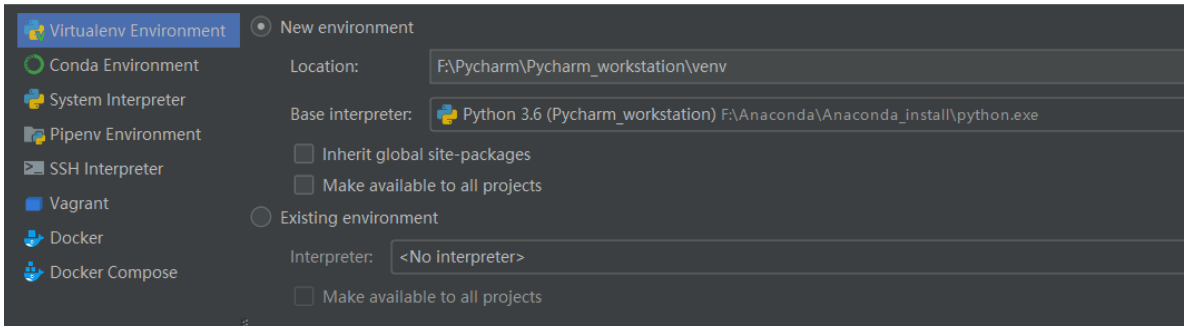
[Editor]--[Font]

- Font：修改字体
- Size：修改字号
- Line Spacing：修改行间距

修改解释器

[Project:项目名称]--[Project Interpreter]--[设置图标]--[Add]--浏览到目标解释器--[OK]--[OK]

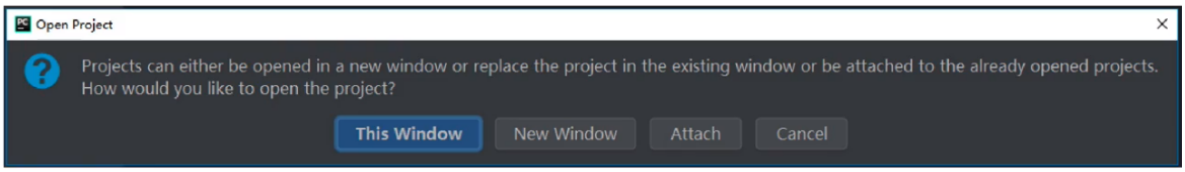




项目管理

[File]--[Open]--浏览选择目标项目根目录--[ok]--选择打开项目方式

打开项目的方式三种：



1、This Window

覆盖当前项目，从而打开目标项目

2、New Window

在新窗口打开，则打开两次Pycharm,每个pycharm负责一个项目

3、Attach

一个窗口下打开多个项目，也就是多个项目重叠（本人比较喜欢这种，一眼尽收眼底）

项目关闭： [File]-[Close Project]/[Close Project in current window]

一、Python基础语法

1.1 注释

```
第一种(快捷键：ctrl+/)：   #
第二种：
        """
        """
```

1.2 变量

定义变量

```
变量名 = 值
```

> 变量名自定义，要满足标识符命名规则

标识符

标识符命名规则是 Python中定义各种名字的时候的统一规范，具体如下

- 由数字、字母、下划线组成
- 不能数字开头 不能使用内置关键字
- 严格区分大小写

```
False      None     True     and      as       assert   break    class
continue   def      del      elif     else     except   finally  for
from       global   if       import   in       is       lambda   nonlocal
not        or       pass     raise    return   try      while    with
yield
```

命名习惯：

- 见名知义。
- 大驼峰：即每个单词首字母都大写，例如：`MyName`
- 小驼峰：第二个（含）以后的单词首字母大写，例如：`myName`
- 下划线：例如：`my_name`

使用变量：

```
my_name = "jiajikang"
```

1.3 认识bug&Debug工具

所谓bug，就是程序中的错误。如果程序有错误，需要程序员排查问题，纠正错误。

Debug工具是PyCharm IDE中集成的用来调试程序的工具，在这里程序员可以查看程序的执行细节和 流程或者调解bug。

Debug工具使用步骤：

1. 打断点
2. Debug调试

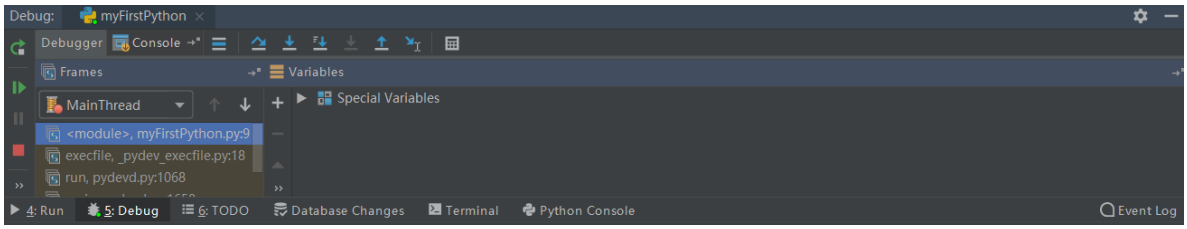
1.3.1 打断点

断点位置：目标要调试的代码的第一行代码即可，即第一个断点。

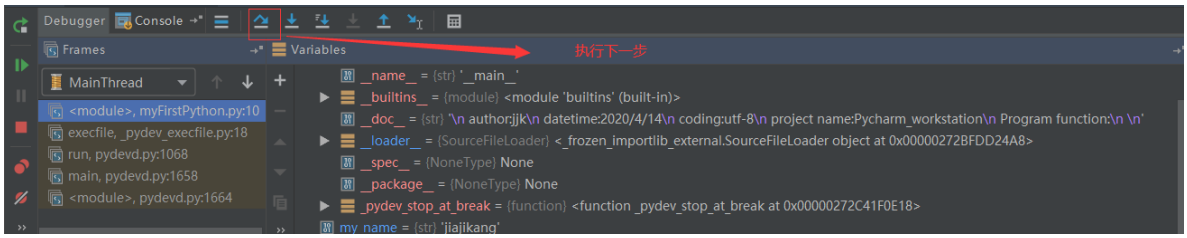
打断点的方法： 单击目标代码的行号右侧空白位置

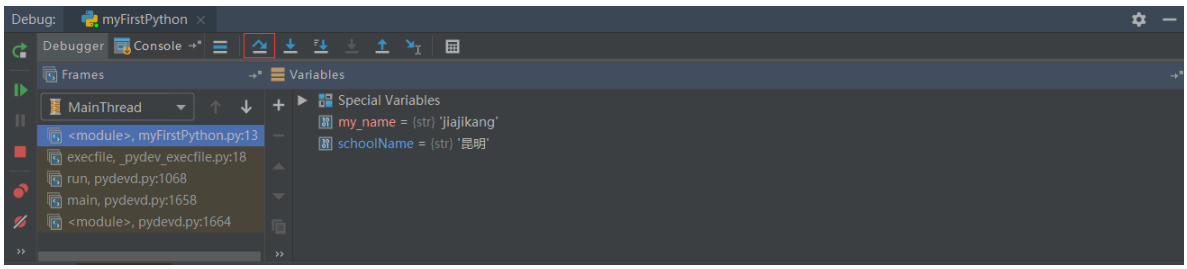
1.3.2 Debug调试

第一步：Debug运行



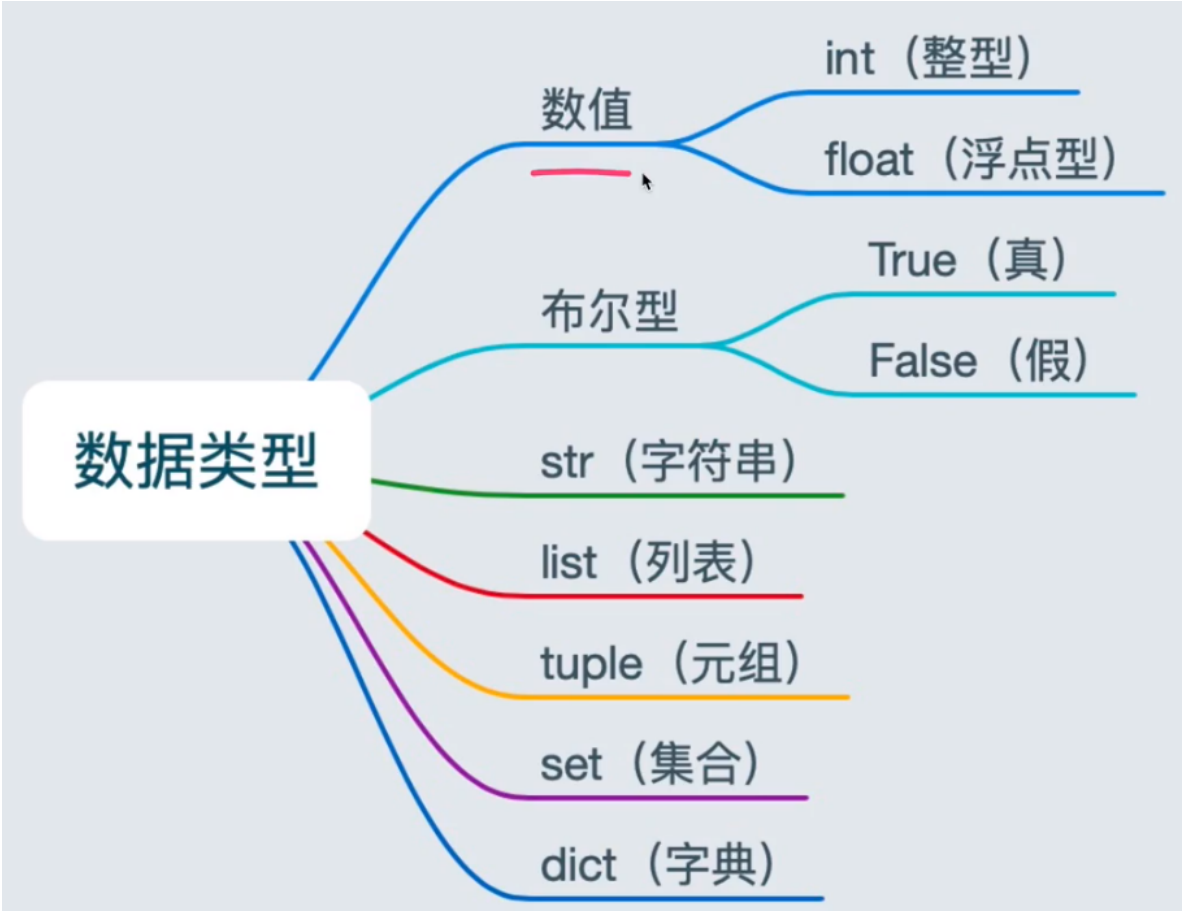
第二步：





1.4 数据类型

在 Python里为了应对不同的业务需求，也把数据分为不同的类型（xmind制作，可了解这个思维导图软件奥，用习惯了，你会比爱你老婆还喜欢这个，哈哈哈）。



> 说明：使用`type()`函数实现查看数据具体的类型

1.5 变量章节总结

- 定义变量的值

```
变量名 = 值
```

- 标识符

  由数字、字母、下划线组成

  不能数字开头

  不能使用内置关键字

  严格区分大小写

- 数据类型

  整型：int

  浮点型：float

  字符串：str

  布尔型：bool

  元组：tuple

  集合：set

  字典：dict

1.6 输出

- 格式化输出

  格式化符号

  f-字符串
- print的结束符

```
print('hell')
age = 18
print(age)
```

1.6.1 格式化输出

格式符号	转换
%s	字符串
%d	有符号的十进制整数
%f	浮点数
%c	字符
%u	无符号十进制整数
%o	八进制整数
%x	十六进制整数（小写ox）
%X	十六进制整数（大写OX）
%e	科学计数法（小写'e'）
%E	科学计数法（大写'E'）
%g	%f和%e的简写
%G	%f和%E的简写

技巧:

- %06d，表示输出的整数显示位数，不足以0补全，超出当前位数则原样输出
- %.2f，表示小数点后显示的小数位数。

1.6.2 输出_格式化基础

所谓的格式化输出即按照一定的格式输出内容。

格式化符号

```python
# 格式化符号输出数据
age = 18
name = "jiajikang"
weight = 120.3
stu_id = 1
print('%d岁' % age)
print('%s' % name)
print('%.2f' % weight)# 小数点后面保存2位
```

1.6.3 输出_格式化高级使用

```python
print('%d' % stu_id)
# 例如学号001
print('%03d' % stu_id) # %06d，表示输出的整数显示位数，不足以0补全，超出当前位数则原样输出

print('名字%s, 今年年龄%d' %(name,age) )
print('名字%s, 明年年龄%d' %(name,age+1) )
print('名字%s,年龄%d,体重%f,学号%d' % (name,age,weight,stu_id))
```

1.6.4 输出_拓展格式化字符串

```python
name = 'tom'
age = 13
weight = 12.3

print('名字%s，年龄%s， 体重%s' % (name,age,weight)) # 都可以使用%s

```

1.6.5 输出_f-格式化字符串

格式化字符串除了%s，还可以写成：`f{表达式}`

```python
age = 23
name = 'tom'
print('名字%s，年龄%s， 体重%s' % (name,age,weight)) # 都可以使用%s

# 语法：f{表达式}
print(f'名字是{name}, 年龄{age}') # 比%s更高效一点
```

1.6.6 输出_转义字符

- `\n`：换行
- `\t`：制表符，一个tab键（4个空格）距离

```python
print('hell \n python') # 换行
print('\tabcd') # 四个制表符
```

1.6.7 输出_print结束符

```python
print('输出的内容',end='\n')
print('hello',end='\t')
print('word')
print('hello',end='...')
```

> 在 Python中， print()，默认自带end="\n"这个换行结束符，所以导致每两个print直接会换行展示，用户可以按需求更改结束符。

1.6.8 输出_总结

- 格式化符号

  %s：格式化输出字符串

  %d：格式化输出整数

  %f：格式化输出浮点数

- f-字符串

  f'{表达式}'

- 转义字符

  \n：换行

  \t：制表符

- print结束符

  ```python
  print('内容',end="")
  ```

1.7 输入

在 Python中，程序接收用户输入的数据的功能即是输入。

目标：

- 输入功能的特点
- 输入input的特点

输入语法：

```python
input('提示信息')
```

输入的特点：

- 当程序执行到 `input` ，等待用户输入，输入完成之后才继续向下执行。
- 在python中，`input` 接收用户输入后，一般存储到变量，方便使用。
- 在python中，`input` 会把接收到的任意输入的数据当做**字符串**处理。

输入功能的实现：

```python
password=input('请输入您的密码：')
print(f'您输入的密码是{password}')
print(type(password)) # str
```

1.8 转换数据类型

- 数据类型转换的必要性
- 数据类型转换常用方法

转换数据类型的作用：

问：input()接收用户输入的数据都是字符串类型，如果用户输入1，想得到整型该如何操作

答：转换数据类型即可，即将字符串类型转换成整型

转换数据类型的函数：

函数	说明
int(x [,base])	将x转换为一个整数
float(x)	将x转换为一个浮点数
complex(real [,imag])	创建一个复数，real为实部，imag为虚部
str(x)	将对象 x 转换为字符串
repr(x)	将对象 x 转换为表达式字符串
eval(str)	用来计算在字符串中的有效Python表达式,并返回一个对象
tuple(s)	将序列 s 转换为一个元组
list(s)	将序列 s 转换为一个列表
chr(x)	将一个整数转换为一个Unicode字符
ord(x)	将一个字符转换为它的ASCII整数值

```
num = input('请输入数字：')
print（num）
print(type(num)) # str
print(type(int(num))) # int

num1 = 1
str1 = '10'
print(type(float(num1))) # float
print(float(num1)) # 1.0

print(float(str1)) # 10.0

#数据转换成字符串
print(type(str(num1)))

# 3、tuple()将一个序列转换成元组
list1 = [10,20,30]
print(type(tuple(list1))) #

#将一个序列转换成列表
t1 = (100,200,300)
print(list(t1)) # [100,200,300]

```

```python
# 计算在字符串中有效python表达式，并返回一个对象
str2 = '1'
str3 = '1.1'
str4 = '(100,200,300)'
str5 = '[100,200,300]'
print(type(eval(str2))) # int
print(type(eval(str3))) # float
print(type(eval(str4))) # tuple
print(type(eval(str5))) # list

```

1.9 转换类型总结

- 转换数据类型常用的函数

- [ ] int()
- [ ] float()
- [ ] list()
- [ ] tuple()
- [ ] eval()

1.10 pychrm交互式开发

```
左下角：python Console
关闭交互式开发环境：1、右侧“-”；2、file-close project
```