## **Python vs Javascript**

## 李艳生

## 湖北师范大学

| 项目     | Python   | Javascript  |
|--------|--|---|
| 语言特点   | 1.解释类型的脚本语言<br>2.弱类型的动态语言<br>3.面向对象编程语言<br>4.开源跨平台   | 1.解释类型的脚本语言<br>2.弱类型的动态语言<br>3.面向对象编程语言<br>4.函数式编程语言  |
| 开发工具   | 1.Jupyter<br>2.PyCharm<br>1.由字母、数字、下划线组成   | <ul><li>5.浏览器标准语言</li><li>1.VSCode</li><li>2.WebStorm</li><li>1.由字母、数字、下划线、美元符号\$组成</li></ul>   |
| 命名规则   | 2.第一个字母不能为数字 3.区分大小写 case sensitive  | 2.第一个字母不能为数字<br>3.区分大小写 case sensitive  |
| 注释<br> | 1.单行#<br>2.多行******或******************************   | 1.单行//<br>2.多行/* */   |
| 常量     | 1.关键字:无         2.常量名习惯全部大写  | 1.关键字: const         2.常量名习惯全部大写  |
| 变量     | 1.关键字: <mark>无</mark><br>2.习惯采用蛇型命名 sneak case   | 1.关键字: let<br>2.习惯采用驼峰命名 camel case   |
| 基本数据类型 | <ol> <li>数字:int, long, float,complex</li> <li>字符串: ''、"", """"""", '""</li> <li>布尔: True, False</li> <li>None</li> </ol> | <ol> <li>数字: int, long, float</li> <li>字符串: '', "", ``````</li> <li>布尔: true, false</li> <li>null</li> <li>undefined</li> <li>Symbol</li> </ol> |
| 复杂数据类型 | 1. 元组: (v1,v2,,vn) 2. 列表: [v1,v2,,vn] 3. 字典: {'k1': v1,, 'kn':vn} 4. 集合: {v1,v2,,vn}                                     | 1. 数组: [v1,v2,,vn]<br>2. 对象: {k1:v1,,kn:vn}   |
| 算术运算   | 1. 加: +<br>2. 减: -<br>3. 乘: *<br>4. 除: /<br>5. 整除: //<br>6. 求余: %<br>7. 幂: **  | 1.加: +<br>2.减: -<br>3.乘: *<br>4.除: /<br>5 求余: %<br>6.幂: **  |

| 关系运算  | 1. 小于: < 2. 大于: > 3. 小于等于: <= 4. 大于等于: >= 5. 等于: == 6. 不等于: !=   | 1. 小于: < 2. 大于: > 3. 小于等于: <= 4. 大于等于: >= 5. 等于: ==, === 6. 不等于: !=, !==  |
|-------|--|---|
| 逻辑运算  | 1. 与: and<br>2. 或: or<br>3. 非: not   | 1. 与: &&<br>2. 或:   <br>3. 非: !   |
| 位运算   | 1. 与: &<br>2. 或:  <br>3. 非: ~<br>4. 异或: ^<br>5. 左移: <<<br>6. 右移: >>>                                       | 1. 与: &<br>2. 或:  <br>3. 非: ~<br>4. 异或: ^<br>5. 左移: <<<br>6. 右移: >>><br>7. 无符号右移: >>>>  |
| 赋值运算符 | 1. 赋值: = 2. 加法赋值: += 3. 减法赋值: -= 4. 乘法赋值: *= 5. 除法赋值: /= 6. 取模赋值: %= 7. 幂赋值: **= 8. 整除赋值: //= 9. 海象运算符: := | 1. 赋值: = 2. 加法赋值: += 3. 减法赋值: -= 4. 乘法赋值: *= 5. 除法赋值: /= 6. 取模赋值: %= 7. 左移赋值: <<= 8. 右移赋值: >>>= 9. 无符号右移赋值: >>>>= 10.位与赋值: &= 11.位或赋值:  = 12.异或赋值: ^= 13.逻辑与赋值: &&= 14.逻辑或赋值:   = |
| 分支语句  | if expr1:     Statement elif expr2:     Statement elif expr3:     Statement else:     Statement            | <pre>if(expr1){     Statement; } else if(expr2){     Statement; } else if(expr3){     Statement; } else{     Statement; }</pre>   |

|         |  | <pre>switch(expr){   case label1: statement;break;   case label2: statement;break;    case labeln: statement;break;   default: statement; }</pre>                                |
|---------|--|--|
| 循环语句    | while expression:     Statement  for n in range(10):     Statement | <pre>while(expression){     Statement; }  do{     Statement; } while(expression);  for(let i = 0; i &lt; 10; i++){     Statement; }  for(let n of list) {     Statement; }</pre> |
| 中止语句    | break #中止当前循环 continue #中止本次循环                                     | break; //中止当前循环<br>continue; //中止本次循环  |
| 复合语句    | 用缩进表示 if expression:     Statement1     Statement2      Statementn | 用{}表示 if(expression){     Statement1;     Statement2;      Statementn; }   |
| 空语句     | Pass   | ;  |
| with 语句 | with expression as target: statement                               | with (expression) {     Statement; }   |

```
try:
                                                           try{
                   Statement
                                                               Statement;
              except e1:
                                                           }
                   Statement
                                                           catch(e1){
              except e2:
                                                               Statement;
                   Statement
                                                           }
              else:
                                                           catch(e2){
                   Statement
                                                               Statement;
异常处理
              raise Exception()
                                                           finally{
                                                               Statement;
                                                           }
                                                           throw new Exception()
              1. 定义
                                                           1. 定义
              def fname(a1, a2=default, *list, **map):
                                                           function fname(a1,a2=default, ...a){
                   Statement
                                                               Statement;
                   return expression;
                                                               return expression;
  函数
                                                           }
              2. 调用
                                                           2. 调用
              fname()
                                                           fname();
              def func(a1, a2):
                                                           function func(a1, a2){
                   Return a1 + a2
                                                               return a1 + a2;
Lambda
              func = lambda a1, a2:a1+a2
                                                           let func = (a1, a2) \Rightarrow a1 + a2;
              1. 定义
                                                           1.定义
              class CName:
                                                           class CName{
                   def init (self, name):
                                                               constructor(name){
                        self.name = name
                                                                    this.name = name;
                   def sayHi(self):
                                                               }
                        print(self.name)
                                                               sayHi(){
              2. 创建
                                                                    console.log(this.name);
  对象
              c = CName("liva")
                                                               }
              c.sayHi()
                                                           2.创建
                                                           let c = new CName("liva");
                                                           c.sayHi();
```

```
1.单继承
             1. 多继承
             2.定义
                                                    2.定义
             class EName(CName):
                                                    class EName extends CName{
                 def init (self, name, age):
                                                         constructor(name, age){
                      #CName. init (self,name)
                                                             super(name);
                      super(). init (name)
                                                             this.age = age;
                      self.age = age
                                                         sayHi(){
  继承
                 def sayHi(self):
                                                             super.sayHi();
                      #CName.sayHi();
                                                             console.log(this.age);
                      super().sayHi()
                                                         }
                      print(self.age);
                                                    }
             3.创建
             e = EName("liva", 40)
                                                    3.创建
             e.sayHi()
                                                    let e = new EName("liva", 40);
                                                    e.sayHi();
             1. 定义
                                                    1. 定义
                                                    将自定义的变量,函数,类放到 mod.js,用
             将自定义的变量,函数,类放到 mod.py
                                                      export 关键字导出变量,函数,类
             文件中即可
             #mod.py
                                                    //mod.js
             def sayHi():
                                                    export function sayHi(){
  模块
                 print("Hello")
                                                         console.log("Hello");
                                                    }
                                                    2. 引用
             2. 引用
             import mod
                                                    import {sayHi} from './mod.js';
             mod.sayHi()
                                                    sayHi();
             str = "hello"
                                                    let str = "Hello";
             It = iter(str) #创建迭代器
                                                    let iterator = str[Symbol.iterator]();//迭代器
 迭代器
             next(it)
                         #迭代
                                                    iterator.next(); //迭代
                         #迭代
             next(it)
                                                    iterator.next(); //迭代
             def gen(): #返回迭代器的函数
                                                    function* gen(){
                 yield 1
                                                         yield 1;
                 yield 2
                                                         yield 2;
                 return 3
                                                         return 3;
 生成器
             f = gen() #调用生成器, 返加迭代器
                                                    let f = gen();
             next(f)
                     #迭代
                                                    f.next();
             next(f)
                     #迭代
                                                    f.next();
             json.dumps() #对数据编码,序列化
                                                    JSON.stringify(); //将对象转为 JSON 字符串
  JSON
                          #对数据解码, 反序列化
                                                    JSON.parse(); //将 JSON 字符串转为对象
             ison.loads()
             r'pattern'
                                                    /pattern/
正则表达式
             re.compile(pattern[, flags])
                                                    let re = new RegExp("pattern", "flags");
```

| async def f(): #返回 coroutine 协程对象 return 1;  f.send(None) #调用  f.send(None) #调用  f().then(alert); //调用  f().then(alert); //调用  f().then(alert); //调用  f().then(alert); //调用  //await 只能用在 async 函数中 async def await _f():     result = await f()     print(result)  fthreading     threading     thread  Socket  Mysql  Mongodb  Django  flask  Web  Django  flask  Web  Angular  React  Vue  Layui      Awaypthon     pyQt    Numpy     Scipy     Pandas  Matplottib     Scikit-learn     Tensorflow    Augular   Electron   Electron   Fetch   Electron   Electron   Electron   Fetch   Electron   Electron   Fetch   Electron   Electron   Fetch   Electron   Electron   Fetch   Electron   Fetch   Electron   Electron   Fetch   |             | W W 40                            | *                                     |
|---|-------------|-----------------------------------|---------------------------------------|
| retrun 1; f.send(None) #週用 f().then(alert); //週用 f().then(alert); //』 f() |             | 单线程实现多任务,用于 IO 密集型任务              | 单线程实现多任务,用于 IO 密集型任务                  |
| # f.send(None) #调用  # f().then(alert); //调用  # await 只能用在 asyne 函数中 asyne def await f():     result = await f()     print(result)  # worker  # threading     thread   |             | async def f(): #返回 coroutine 协程对象 | async function f(){ //返回 promise 异步对象 |
| f.send(None) #调用  f().then(alert); //调用  #await 只能用在 async 函数中 async def await f():     result = await f()     print(result)  **Sétat**  **Interest thread  |             | retrun 1;                         | return 1;                             |
| #await 只能用在 asyne 函数中 asyne def await f():     result = await f()     print(result)   多线程     threading     thread      socket     Mysql     Mongodb  Web Sorage     flask  Web  Web  Web Sorage  Mangular  React  Vue  Layui   |             |                                   | }                                     |
| #await 只能用在 asyne 函数中 asyne def await_f():     result = await f()     print(result)   多线程  threading     thread  socket  Méb web storage     Indexdb     Mongodb  Django flask  Web  Web  Web  Web  Angular React Vue Layui   游戏  pygame  Numpy Scipy Pandas  Matplotlib Scikit-learn Tensorflow  //await 只能用在 asyne 函数中 asyne function await_f() {     let result = await f();     alert(result); } let result = await f();     let result = await f();     alert(result); }  Web Socket  WebSocket Fetch XMLHttpRequest  Web storage Indexdb mongodb  Express.js Kio Angular React Vue Layui  Flectron pyQt  Numpy Scipy Pandas  Matplotlib Scikit-learn Tensorflow   |             | f.send(None) #调用                  | f().then(alert); //调用                 |
| async def await_f():     result = await f()     print(result)  多线程  threadingthread  socket   | async/await |                                   |                                       |
| result = await f() print(result)   **State**  **Intered ing thread  **Socket**    WebSocket**   Numlib  |             | #await 只能用在 async 函数中             | //await 只能用在 async 函数中                |
| ### Print(result)  ### print(result)  ### alert(result);  ### Worker  ### WebSocket  ### WebSoc |             | async def await_f():              | <pre>async function await_f(){</pre>  |
| ### Style   |             | result = await f()                | <pre>let result = await f();</pre>    |
| ### Style ### S |             | print(result)                     | alert(result);                        |
| System  |             |                                   | }                                     |
| thread  | 夕处印         | threading                         | Worker                                |
| 网络       httplib urllib       Fetch XMLHttpRequest         Sqlite       Web storage         Mysql       Indexdb         Mongodb       mongodb         Django       Express.js         flask       Kio         Angular       React         Vue       Layui         Layui       Three.js         GUI       wxPython pyQt         Numpy       Electron         Scipy Pandas       Tesorflow.js         Matplotlib Scikit-learn Tensorflow       Tensorflow   | <b>多</b> 线性 | _thread                           |                                       |
| wrliib XMLHttpRequest  Sqlite Web storage Mysql Indexdb Mongodb mongodb  Django Express.js flask Kio Angular React Vue Layui  游戏 pygame Three.js  GUI wxPython pyQt  Numpy Scipy Pandas  Matplotlib Scikit-learn Tensorflow   |             | socket                            | WebSocket                             |
| 数据库 Sqlite Web storage Indexdb Mongodb mongodb Express.js  Django Express.js Kio Angular React Vue Layui  游戏 pygame Three.js  GUI wxPython Electron pyQt Echarts Scipy Pandas Matplotlib Scikit-learn Tensorflow  | 网络          | httplib                           | Fetch                                 |
| 数据库 Mysql Indexdb mongodb Django Express.js flask Kio Angular React Vue Layui  游戏 pygame Three.js  GUI wxPython pyQt Numpy Scipy Pandas Matplotlib Scikit-learn Tensorflow  I Express.js Kio Angular React Vue Layui  Three.js  Electron  Electron  Tesorflow.js  |             | urllib                            | XMLHttpRequest                        |
| Mongodb mongodb Django Express.js flask Kio Angular React Vue Layui  mygame Three.js  wxPython pyQt  Numpy Scipy Pandas Matplotlib Scikit-learn Tensorflow  Express.js Kio Angular React Vue Layui  Three.js  Electron  Three.js  Tesorflow.js  |             | Sqlite                            | Web storage                           |
| Django  | 数据库         | Mysql                             | Indexdb                               |
| Hask Kio Angular React Vue Layui  pygame Three.js  wxPython Electron pyQt  Numpy Echarts Scipy Pandas Matplotlib Scikit-learn Tensorflow  Kio Angular React Vue Layui  Elextron  Three.js  Three.js  Three.js   |             | Mongodb                           | mongodb                               |
| Web React Vue Layui  游戏 pygame Three.js  GUI wxPython Electron pyQt Echarts Scipy Tesorflow.js  Matplotlib Scikit-learn Tensorflow  |             | Django                            | Express.js                            |
| React Vue Layui  游戏  pygame  Three.js  GUI  wxPython pyQt  Numpy Scipy Pandas Matplotlib Scikit-learn Tensorflow  React Vue Layui  Electron  Frace of the series of the s |             | flask                             | Kio                                   |
| React Vue Layui  游戏  pygame  Three.js  GUI  wxPython pyQt  Numpy Scipy Pandas Matplotlib Scikit-learn Tensorflow  React Vue Layui  Electron  Flectron  Three.js  Electron  Flectron  Three.js   | ****        |                                   | Angular                               |
| Layui   | Web         |                                   | React                                 |
| 游戏 pygame Three.js  GUI wxPython Electron pyQt Numpy Echarts Scipy Tesorflow.js Pandas Matplotlib Scikit-learn Tensorflow   |             |                                   | Vue                                   |
| MX WxPython Electron  GUI WxPython Electron  Numpy Echarts Scipy Tesorflow.js  和器学习 Matplotlib Scikit-learn Tensorflow  |             |                                   | Layui                                 |
| MX wxPython Electron  GUI wxPython Electron  Numpy Echarts Scipy Tesorflow.js  和器学习 Matplotlib Scikit-learn Tensorflow  | 724 - D     | pygame                            | Three.js                              |
| GUI pyQt Echarts Numpy Echarts Scipy Tesorflow.js Pandas 机器学习 Matplotlib Scikit-learn Tensorflow  | 游戏          |                                   |                                       |
| Numpy Echarts Scipy Tesorflow.js 和器学习 Matplotlib Scikit-learn Tensorflow  | GUI         | wxPython                          | Electron                              |
| Scipy Tesorflow.js Pandas 机器学习 Matplotlib Scikit-learn Tensorflow   |             | pyQt                              |                                       |
| 和器学习 Pandas Matplotlib Scikit-learn Tensorflow  | 机器学习        | Numpy                             | Echarts                               |
| 机器学习 Matplotlib Scikit-learn Tensorflow   |             | Scipy                             | Tesorflow.js                          |
| Scikit-learn Tensorflow   |             | Pandas                            |                                       |
| Tensorflow  |             | Matplotlib                        |                                       |
|   |             | Scikit-learn                      |                                       |
|   |             | Tensorflow                        |                                       |
| keras   |             | keras                             |                                       |

项目地址: <a href="https://github.com/liva2008/pythonvsjavascript">https://github.com/liva2008/pythonvsjavascript</a>