Apache Flink Python API 现状及规划

孙金城 (金竹) Apache Flink PMC/阿里巴巴 高级技术专家

Aache Flink 直播教程 2019.08.06

Who

2019
- Apache Flink Python API
- Apache Flink PMC

2017
- Apache Flink Table/SQL API
- Apache Flink Committer

2011
- 云转码/阿里郎/OPLog/云代码...
- 计算平台/搜索/信息平台/技术平台

https://enjoyment.cool

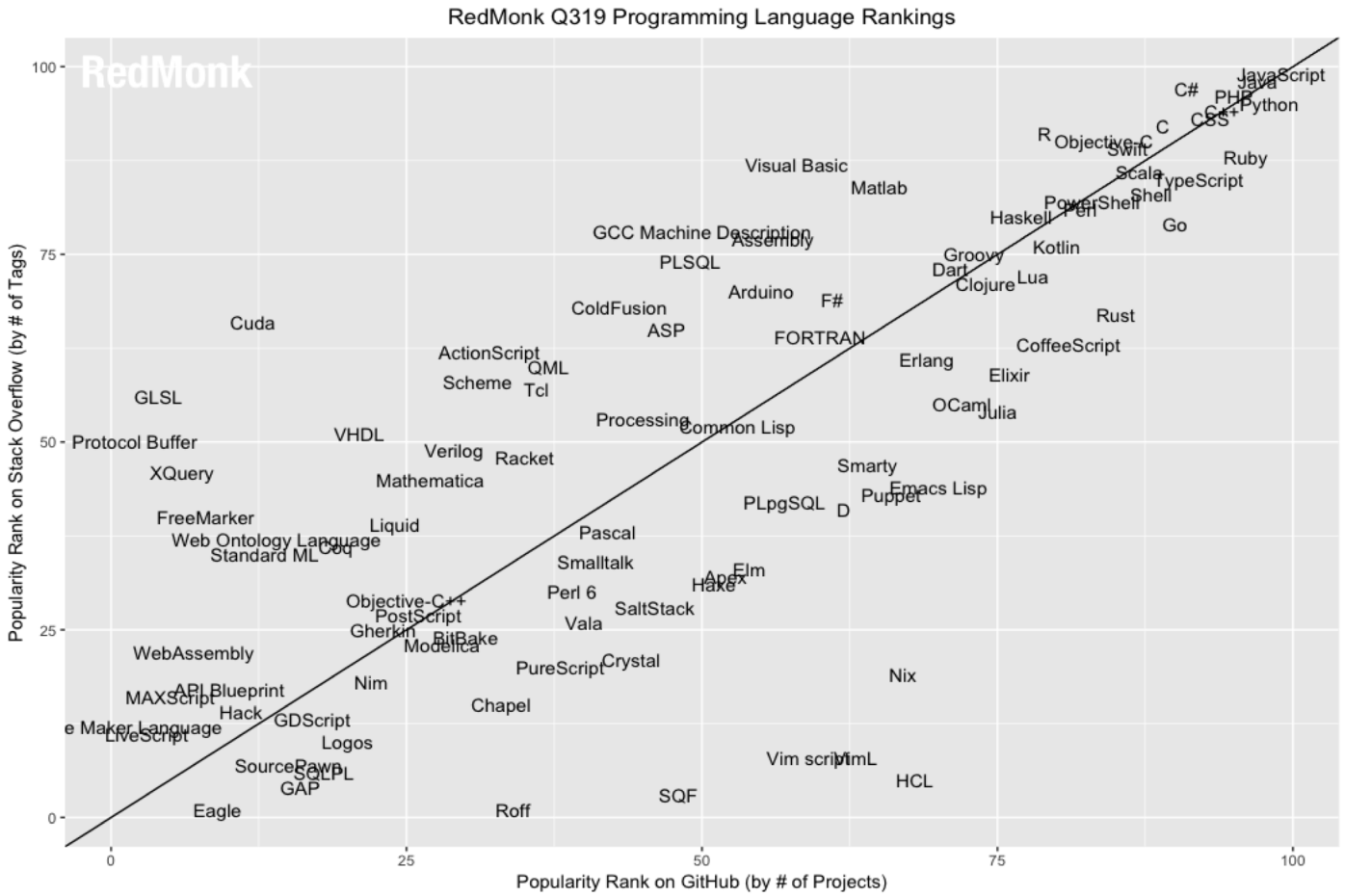
CONTENT

目录 >>

01

Flink Python API的前世今生和未来发展

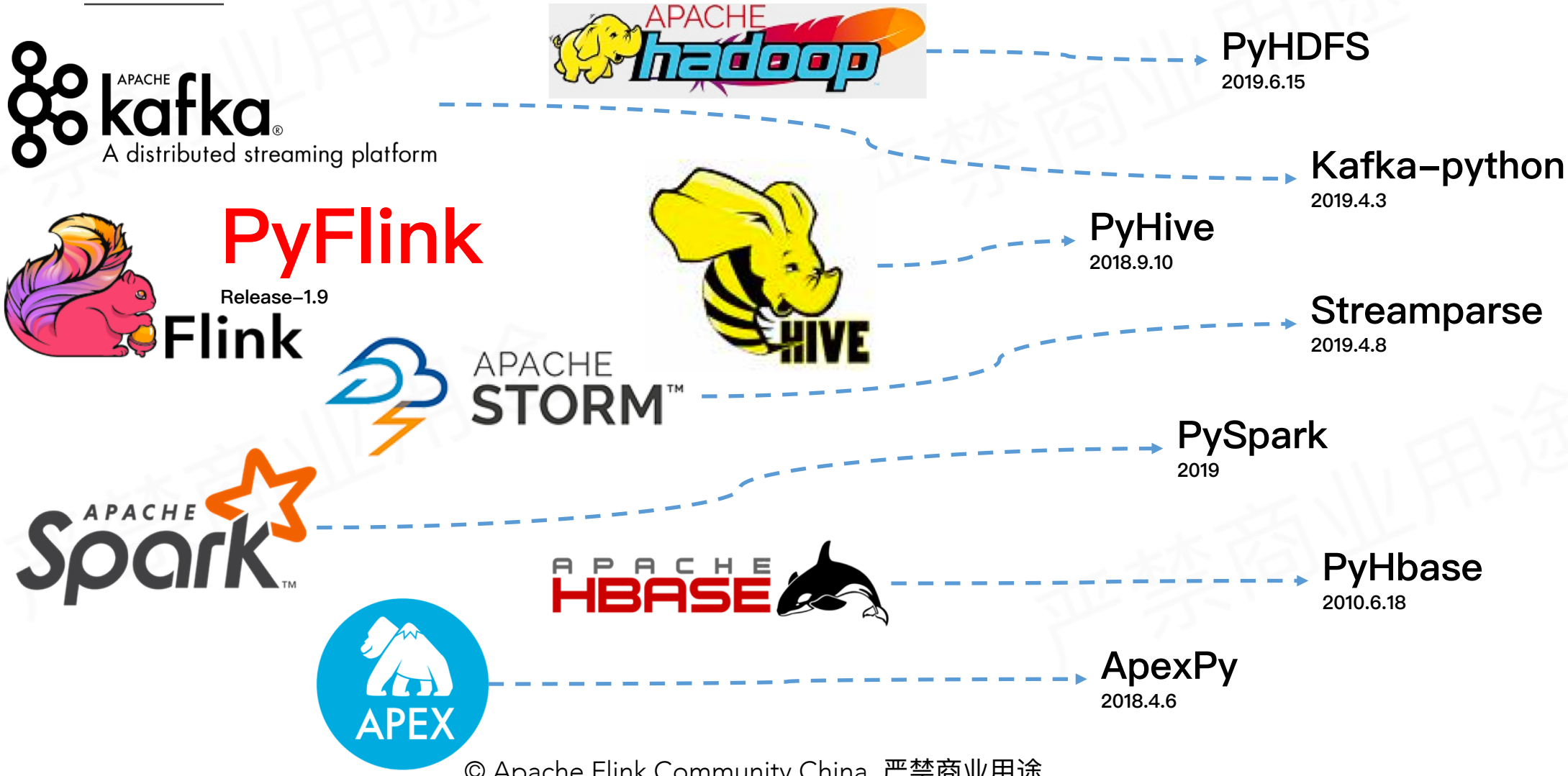
Why Python API — 最流行的开发语言



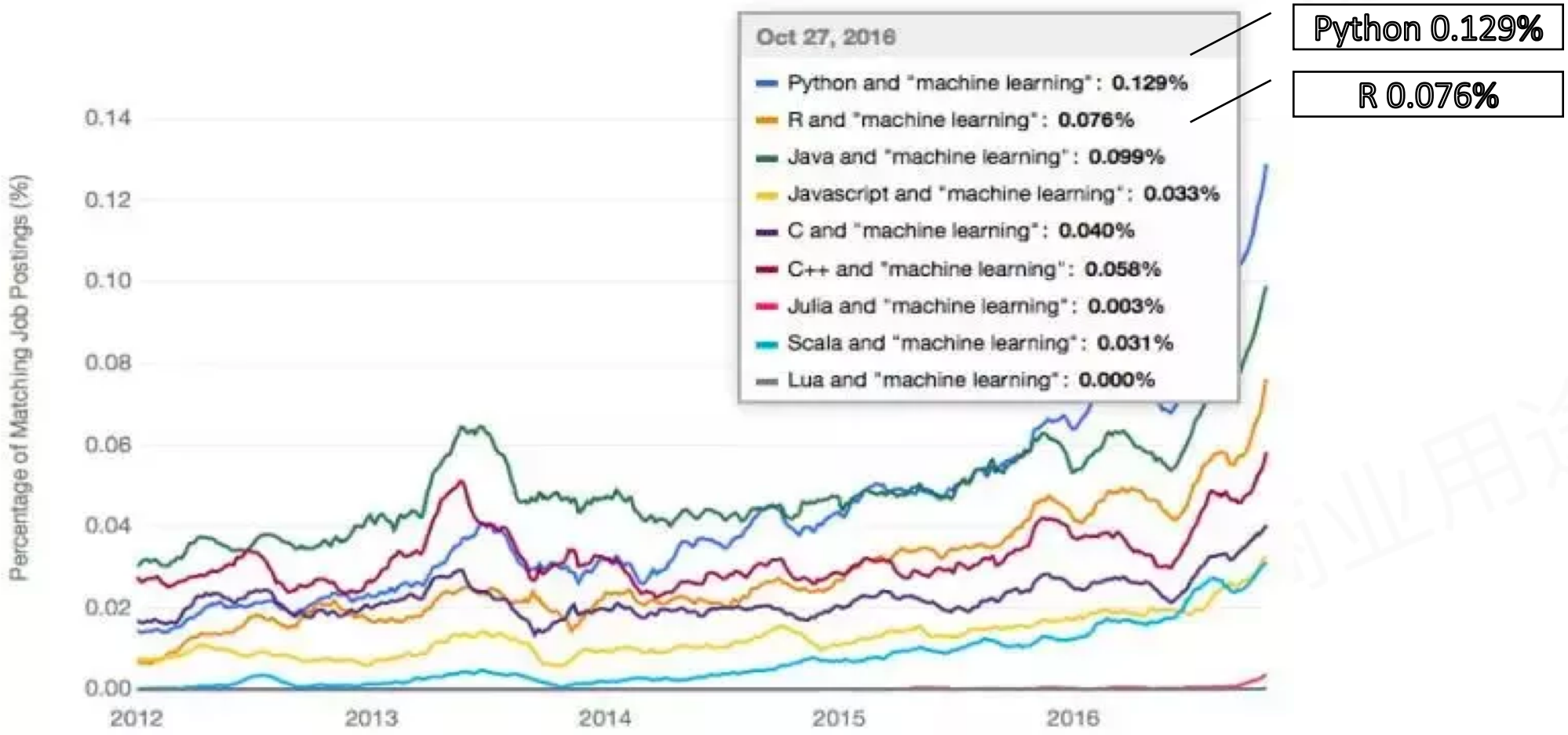
- JavaScript
- Java
- **Python**
- PHP
- C++
- C#
- CSS
- Ruby
- C
- TypeScript

统计数据来源于RedMonk 2019.06月

Why Python API — 众多开源项目支持



Why Python API — ML青睐的语言



Python API – RoadMap

1.9

2019.8
- Python Table API
- Java UDFs

1.10

2019.12
- Support UDFs
- Pandas

1.11

2020.3
- DataStream API
- ML API

1.12

?

https://enjoyment.cool

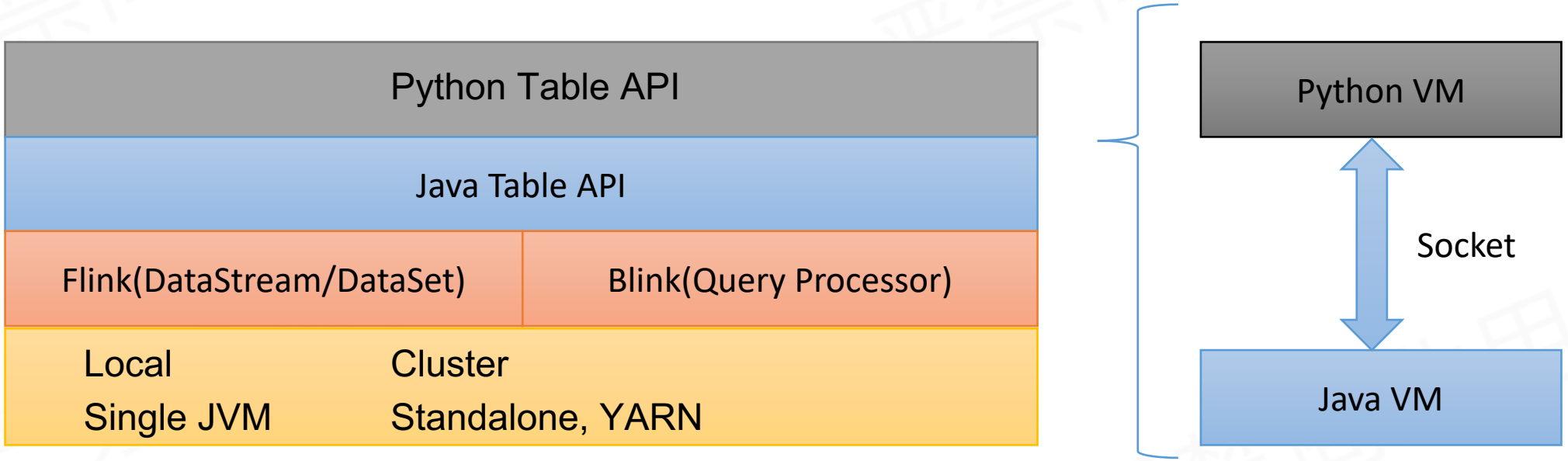
- ...

02

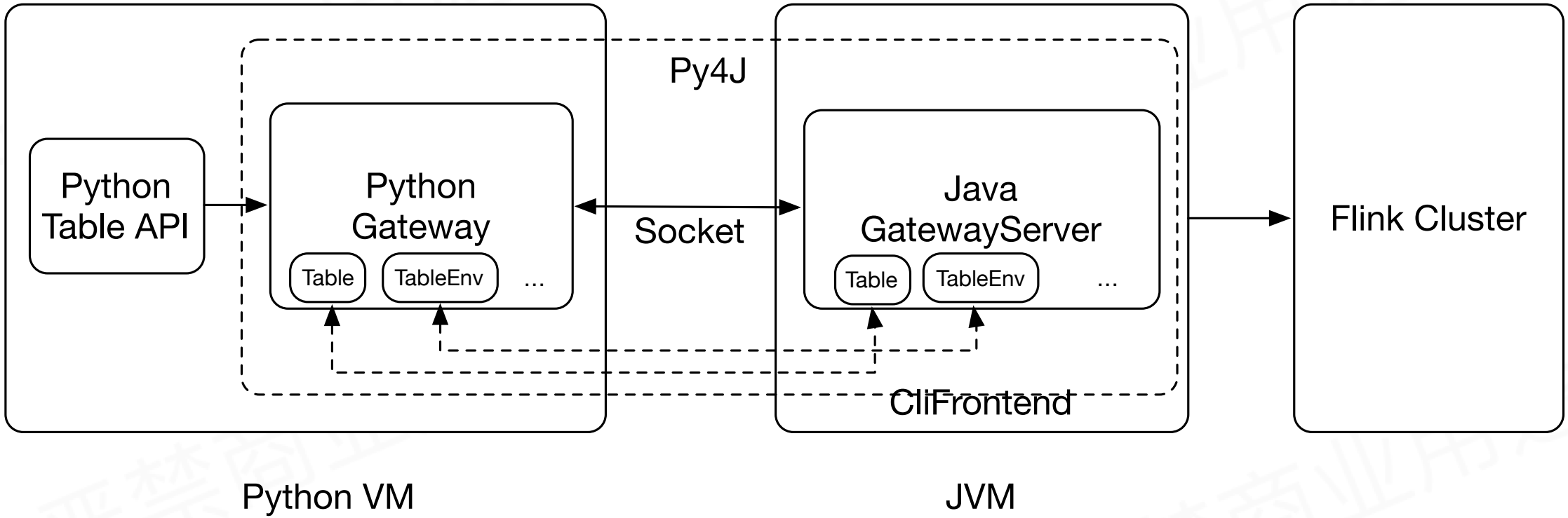
Python API架构及开发环境搭建

Python Table API 架构

Apache Flink 1.9 新架构

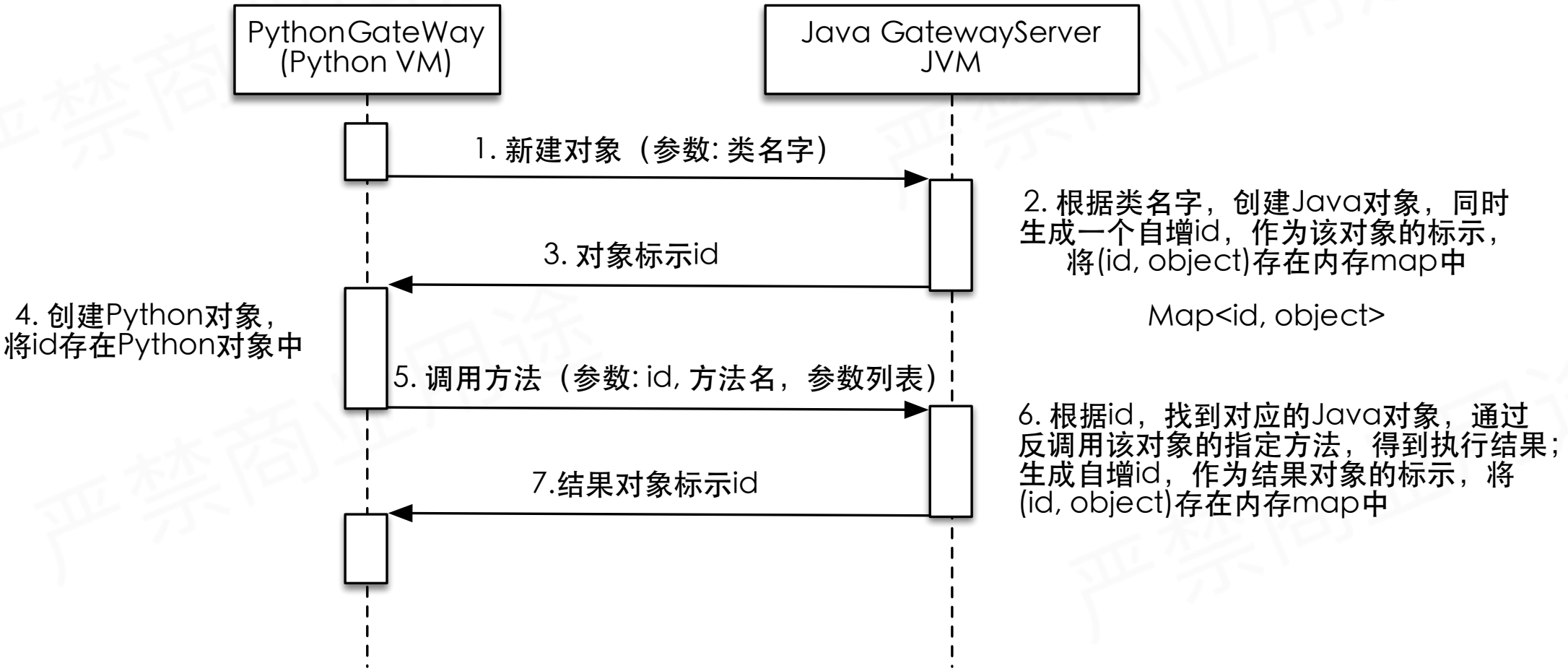


Python Table API 架构



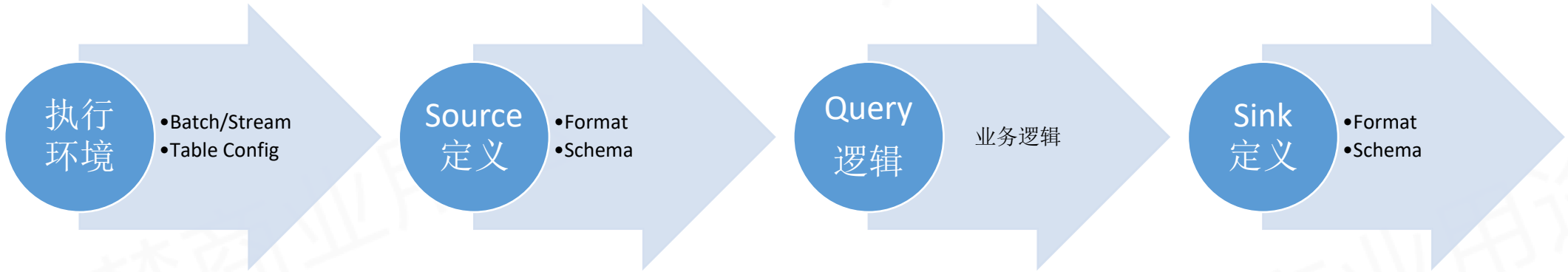
- Python object is just a wrapper of the corresponding Java object
- Leverage the QO and QE framework of Flink Table module automatically

Python Table API 架构



Python Table API — Job开发

Table API 开发核心部分



Python Table API – Job开发

创建执行环境

```
# 创建执行环境
exec_env = ExecutionEnvironment.get_execution_environment()

# 创建配置对象(IdleState TTL, NULL check, timezone 等)
t_config = TableConfig()

# 创建一个Table ENV
t_env = BatchTableEnvironment.create(exec_env, t_config)
```

Python Table API – Job开发

创建数据源

```
# 创建数据源表
t_env.connect(FileSystem().path(source_file)) \
    # Csv/Json/Avro
    .with_format(OldCsv()
                 .line_delimiter(',')
                 .field('word', DataTypes.STRING())) \
    # 定义字段名和类型
    .with_schema(Schema()
                 .field('word', DataTypes.STRING())) \
    # 注册Source
    .register_table_source('mySource')
```

word
enjoyment
Apache Flink
cool
Apache Flink

Python Table API – job开发

创建结果表

```
# 创建结果表
t_env.connect(FileSystem().path(sink_file)) \
    # Csv/Json/Avro
    .with_format(OldCsv()
                 .field_delimiter(',')
                 .field('word', DataTypes.STRING())
                 .field('count', DataTypes.BIGINT())) \
    # 定义字段名和类型
    .with_schema(Schema()
                 .field('word', DataTypes.STRING())
                 .field('count', DataTypes.BIGINT())) \
    .register_table_sink('mySink')
```

word	count

Python Table API – Job开发

编写业务逻辑和执行

```
#  word_count计算逻辑
# 读取数据源
t_env.scan('mySource') \
    # 按 word 进行分组
    .group_by('word') \
    # 进行count计数统计
    .select('word, count(1)') \
    # 将计算结果插入到结果表
    .insert_into('mySink')

# 执行Job
t_env.execute("wordcount")
```

word	count
Apache Flink	2
cool	1
enjoyment	1

https://github.com/sunjincheng121/enjoyment.code/blob/master/myPyFlink/enjoyment/word_count.py

Python Table API — 环境搭建

环境依赖检查

- JDK 1.8+ (1.8.0_211) java –version
- Maven 3.x (3.2.5) mvn –version
- Scala 2.11+ (2.12.0) scala –version
- Python 2.7+ (2.7.16) python –V
- Git 2.20+ (2.20.1) git version
- Pip 19.1+ (pip 19.1.1) pip -V

Python Table API – 环境搭建

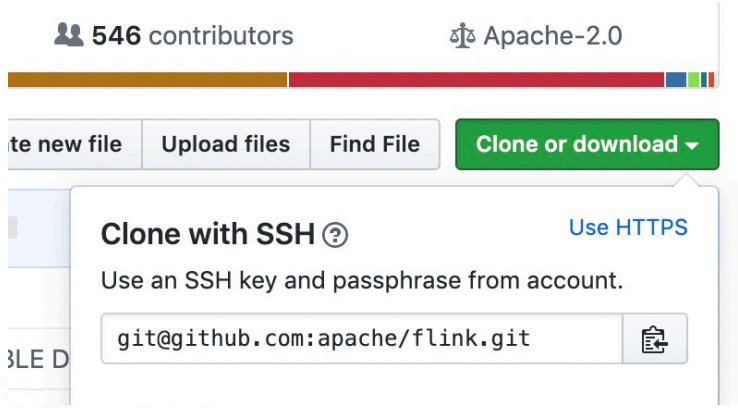
构建Java二进制发布包

我们要想利用Apache Flink Python API 进行业务开发， 需要将PyFlink发布包进行安装。目前PyFlink并没有发布到Python仓库, 所以我们需要从源码构建。

```
# 下载源代码
git clone https://github.com/apache/flink.git

# 拉取1.9分支
cd flink; git fetch origin release-1.9
git checkout -b release-1.9 origin/release-1.9

#构建二进制发布包
mvn clean install -DskipTests -Dfast
```



Python Table API – 环境搭建

构建Python发布包

一般情况我们期望以pip install的方式安装python的类库，我们要想安装PyFlink的类库，也需要构建可用于pip install的发布包。执行如下命令：

cd flink-python; python setup.py sdist

```
...
copying pyflink/util/utils.py -> apache-flink-1.9.dev0/pyflink/util
Writing apache-flink-1.9.dev0/setup.cfg
creating dist
Creating tar archive
removing 'apache-flink-1.9.dev0' (and everything under it)
```

在dist目录的apache-flink-1.9.dev0.tar.gz就是我们可以用于pip install的PyFlink包.

Python Table API — 环境搭建

安装PyFlink

接下来我们将PyFlink安装到Python的环境中：

pip install dist/*.tar.gz

用pip命令检查是否安装成功： pip list

```
IT-C02YL16BJHD2:flink-python jincheng.sunjc$ pip list
Package         Version
--------------- -----------
apache-flink    1.9.dev0
configparser    3.7.4
entrypoints     0.3
enum34          1.1.6
flake8          3.7.8
```

Python Table API – 环境搭建

验证PyFlink

我们可以运行刚才开发的word_count.py，以验证环境的正确性：

https://github.com/sunjincheng121/enjoyment.code/blob/master/myPyFlink/enjoyment/word_count.py

https://github.com/sunjincheng121/enjoyment.code/blob/master/myPyFlink/enjoyment/source.csv

Git clone https://github.com/sunjincheng121/enjoyment.code.git

cd enjoyment.code; python word_count.py

Python Table API — 环境搭建

IDE配置

Apache Flink 官网
https://ci.apache.org/projects/flink/flink-docs-master/flinkDev/ide_setup.html#pycharm

注意的细节，我们边操作，边说明。

同时我也整理到了Blog ： http://1t.click/6Nf

Python Table API — 作业提交

CLI方式提交

1. 启动或者配置Apache Flink 集群，我们这里启动一个本地集群：

./bin/start-cluster.sh

启动之后查看启动日志，在文件中找到

Rest endpoint listening at localhost:8081

的信息，证明集群启动正常，我们可以访问 Apache Flink Dashboard

http://localhost:8081

Python Table API – 作业提交

CLI方式提交

2. 提交已经写好的Job到集群运行，我们用如下命令：

```
./bin/flink run -py
~/training/0806/enjoyment.code/myPyFlink/enjoyment/word_count_cli.py
```

详细命令说明请查阅： https://ci.apache.org/projects/flink/flink-docs-master/ops/cli.html

- -py 指定python文件
- -pym 指定python的module
- -pyfs 指定python依赖的资源文件
- -j 指定依赖的JAR包

Python Table API – 作业提交

Python-Shell 方式提交

Python shell 是很方便进行研究性开发的，我们可以方便的在Python REPL 中进行开发Flink Python Table API.

详细文档参考： https://ci.apache.org/projects/flink/flink-docs-master/ops/python_shell.html

接下来我们以Local和Remote两种方式，体验一下Python-Shell。

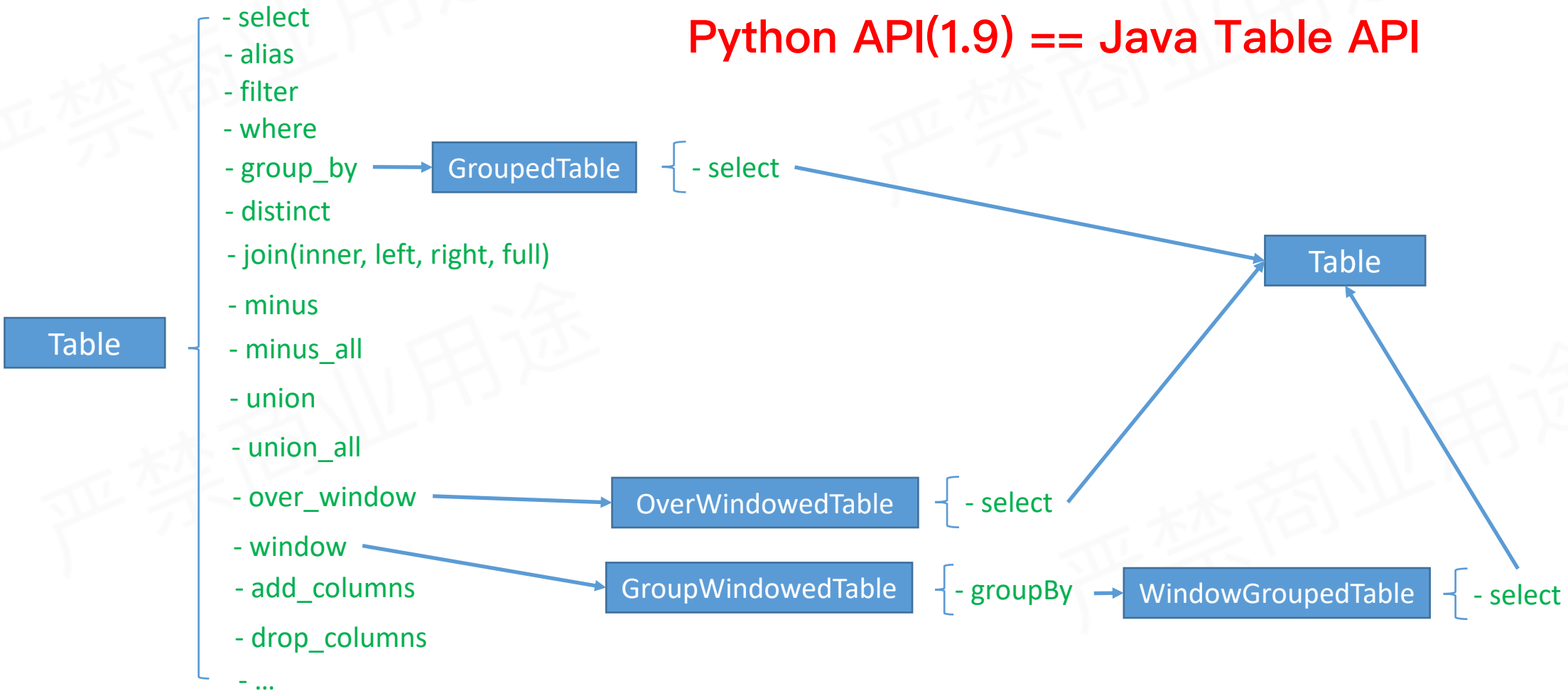
- Local
  bin/pyflink-shell.sh local (会启动一个mini Cluster)

- Remote
  bin/pyflink-shell.sh remote 127.0.0.1 4000 (需要一个已经存在的Cluster)

03

Flink Python API 核心算子介绍及应用

Python Table API 算子



Python Table API 算子–Watermark定义

```
.with_format(
  Json()
  .fail_on_missing_field(True)
  .json_schema(
    "{"
    "  type: 'object',"
    "  properties: {"
    "    a: {"
    "      type: 'string'"
    "    },"
    "    time: {"
    "      type: 'string',"
    "      format: 'date-time'"
    "    }"
    "  }"
    "}"
  )
```

```
.with_schema(
    Schema()
    .field("rowtime", DataTypes.TIMESTAMP())
    .rowtime(
      Rowtime()
      .timestamps_from_field("time")
      .watermarks_periodic_bounded(60000))
    .field("a", DataTypes.STRING())
)
```

Watermark原理介绍，
查阅我的博客：http://1t.click/7dM

Python Table API — Java UDF

Java UDF

虽然我们在Flink-1.9中没有支持Python的UDF，但在Flink 1.9版本中我们可以使用Java UDF。

1. 创建Java项目，并配置pom依赖如下：

```
<dependency>
  <groupId>org.apache.flink</groupId>
  <artifactId>flink-table-common</artifactId>
  <version>1.9-SNAPSHOT</version>
  <scope>provided</scope>
</dependency>
```

Python Table API – Java UDF

Java UDF

2. 编写一个计算字符串长度的函数UDFLength

```
package org.apache.flink.udf;
import org.apache.flink.table.functions.ScalarFunction;

public class UDFLength extends ScalarFunction {
  public int eval(String str) {
    return str.length();
  }
}
```

注册和使用：

```
t_env.register_java_function("len", "org.apache.flink.udf.UDFLength")
…
.select("word, len(word), count(1) as count")
```

Python Table API — Java UDFs

Java UDFs

开发Python Job，并使用上面自定义的UDFLength函数：

https://github.com/sunjincheng121/enjoyment.code/blob/master/myPyFlink/enjoyment/word_count_udf.py

提交Python Job，并上传UDF JAR包：

```
./bin/flink run -py word_count_udf.py  -j  <PATH>/flink-udf-1.0.0.jar
```

Python Table API – Java UDF

Java UDFs

- Scalar Function

```
t_env.register_java_function("len", "org.apache.flink.udf.UDFLength")
…
.select("word, len(word), count(1) as count")
```

- Table Function

```
t_env.register_java_function("split", "com.pyflink.table.Split")
tab.join_lateral("Split(a) as (word, length)").select("a, word, length")
```

- Aggregate Function

```
t_env.register_java_function("wAvg", "com.pyflink.table.WeightedAvg")
tab.group_by("a").select("a, wAvg(b) as d")
```

Python Table API 常用链接

- Python Table API文档
https://ci.apache.org/projects/flink/flink-docs-master/api/python/

- Python Table API IDE开发环境
https://cwiki.apache.org/confluence/display/FLINK/Setting+up+a+Flink+development+environment

- Python Shell
https://ci.apache.org/projects/flink/flink-docs-master/ops/python_shell.html

- Python Table API Tutorial
https://ci.apache.org/projects/flink/flink-docs-master/tutorials/python_table_api.html

- 我的个人博客
https://enjoyment.cool/

关注 Apache Flink 社区微信公众号

Ververica，由 Apache Flink Community China 运营管理，旨在联合国内的 Flink 大 V，向国内宣传和普及 Flink 相关的技术。

持续输出 Flink 最新社区动态：

- Release 发布及新特性解读
- 入门教程
- Meetup
- 应用案例
- 源码解析

THANKS