“高端语言”？

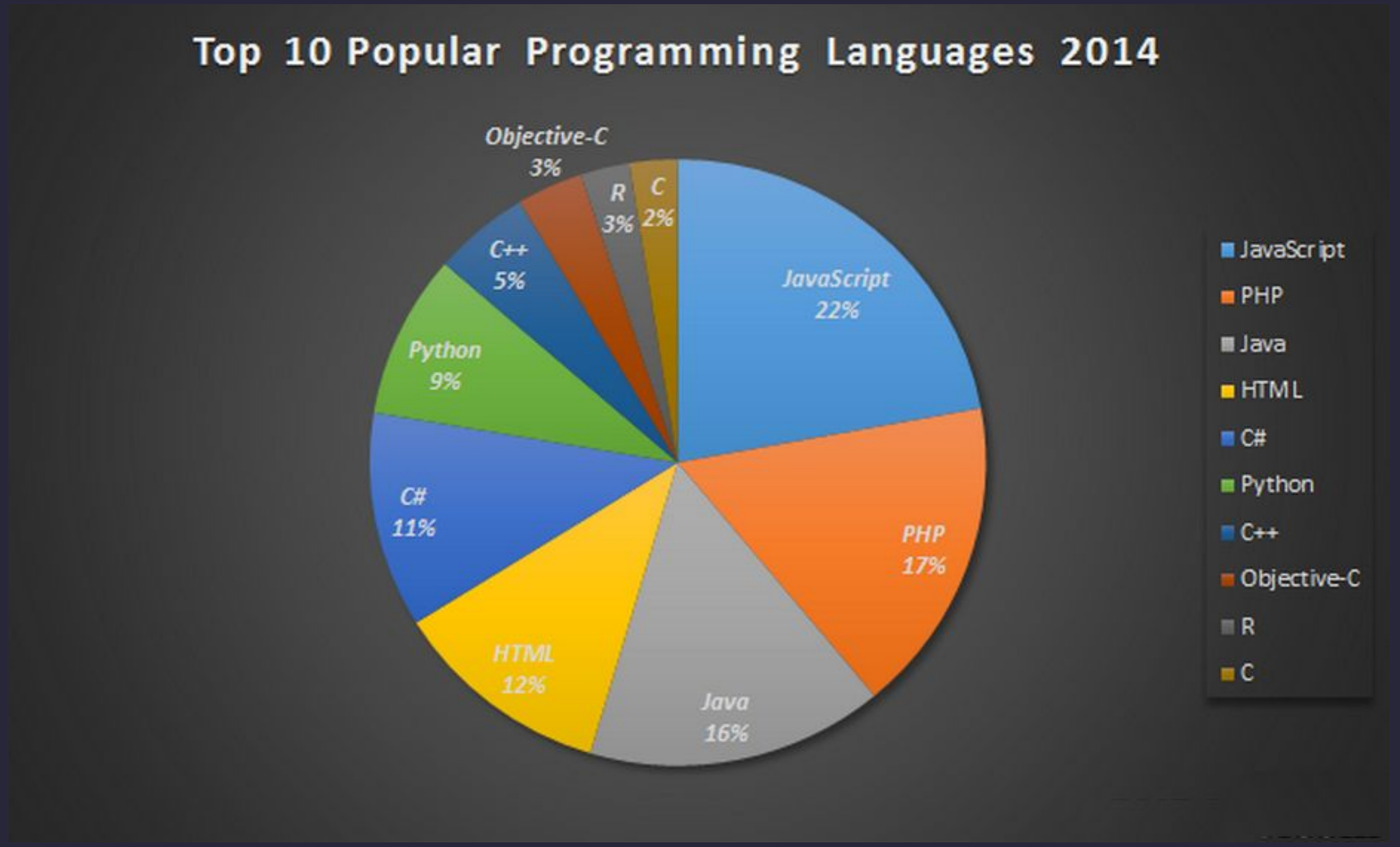
@k1ic

chenkai116@gmail.com

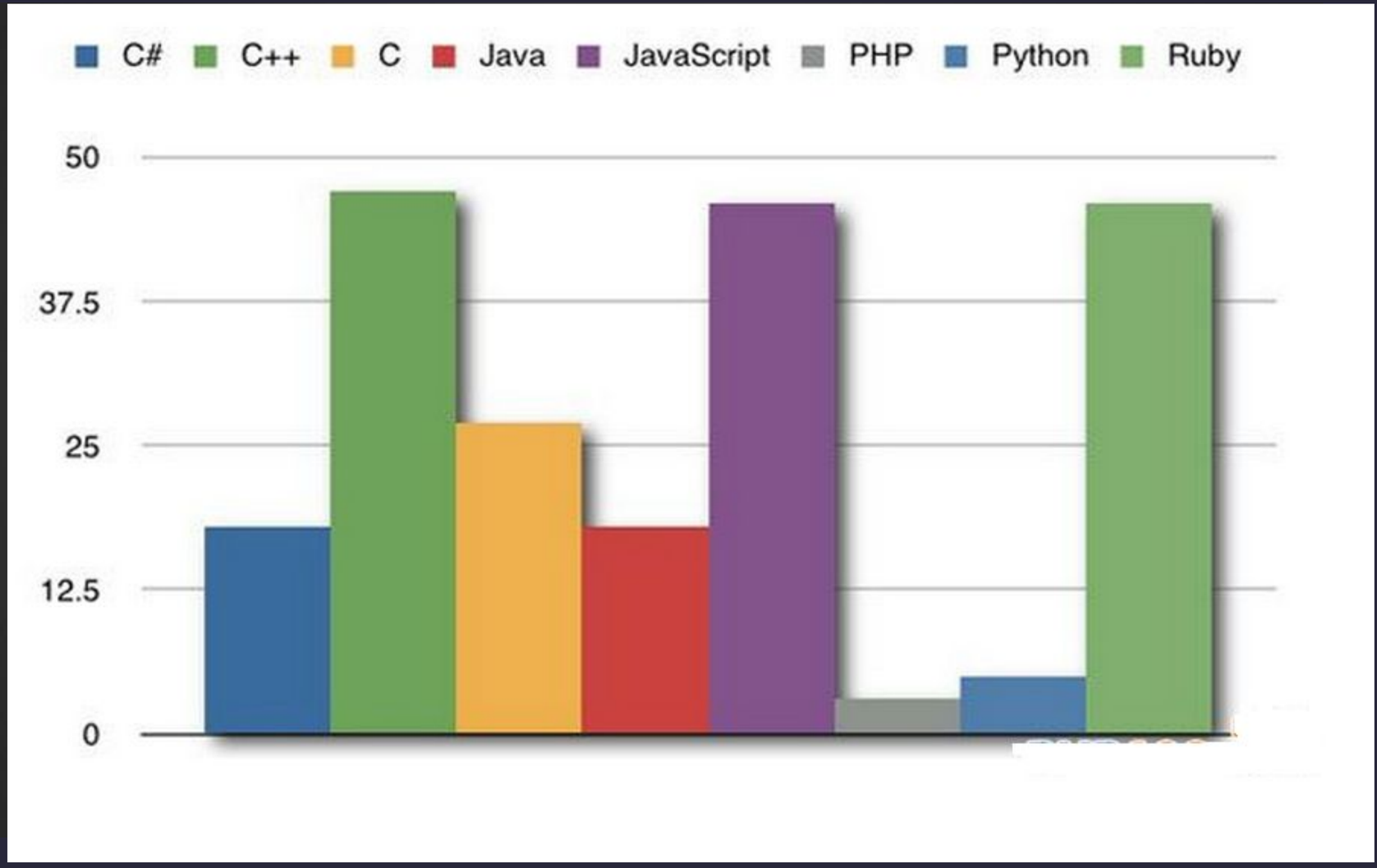
缘起

- 搞Java的不要转C++
- 第一门编程语言选什么
- Ruby、Python、JavaScript初学者抱怨语言太低端

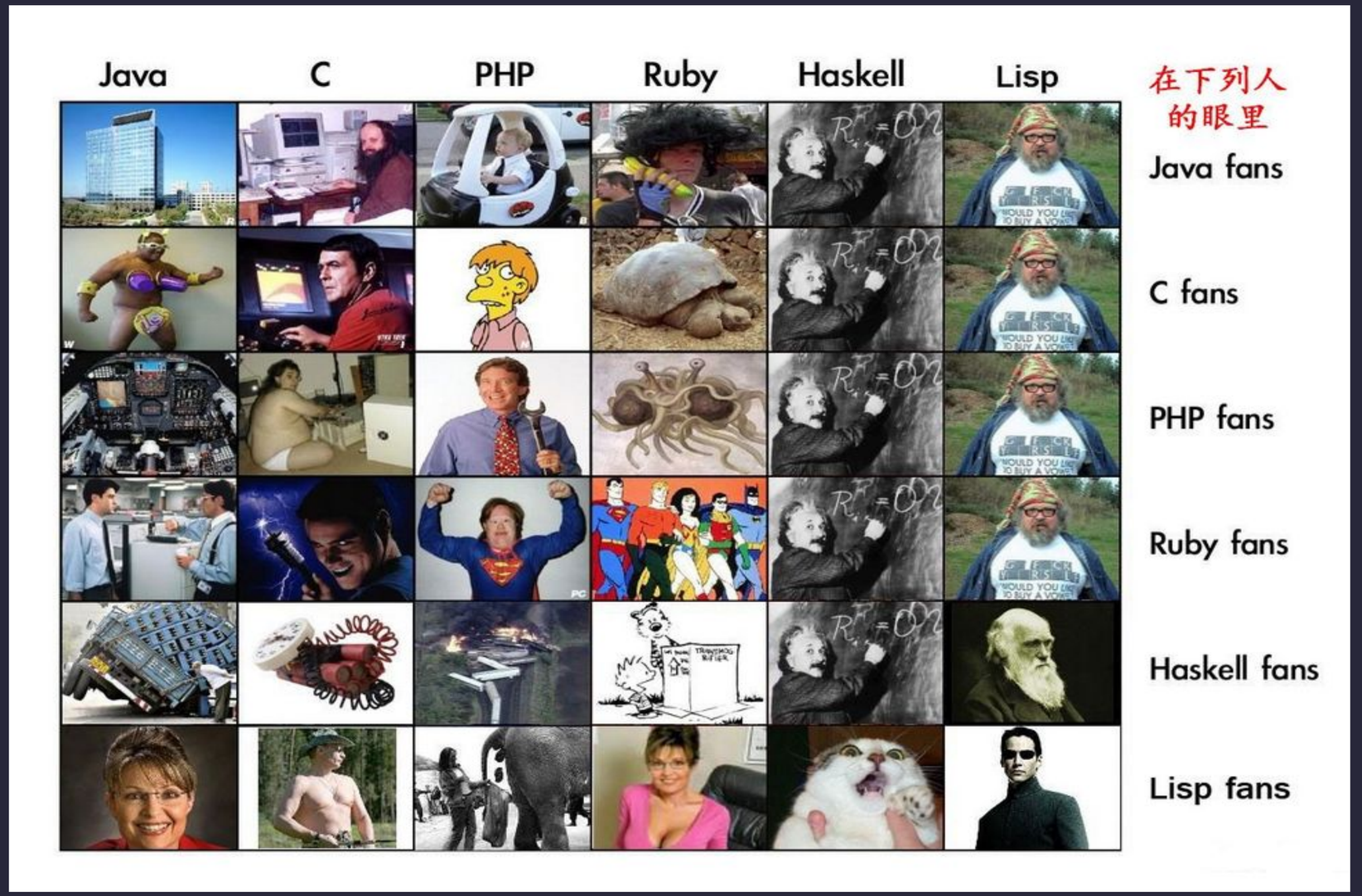
2014年被讨论最多的语言 JavaScript



2014年被吐槽最多的语言 C++



大家都很自负


Java
C
PHP
Ruby
Haskell
Lisp
在下列人的眼里
Java fans
C fans
PHP fans
Ruby fans
Haskell fans
Lisp fans

到底有没有“高端”语言？

编程语言面面观

电频层面

市场需求

性能

执行效率

设计思想

扩展库

主要面向开发人员

行业人才储备

可移植性

开发效率

社区成熟度

项目复杂度

学习成本

执行效率

执行效率 之 执行机制

解释执行 vs 编译执行

解释执行

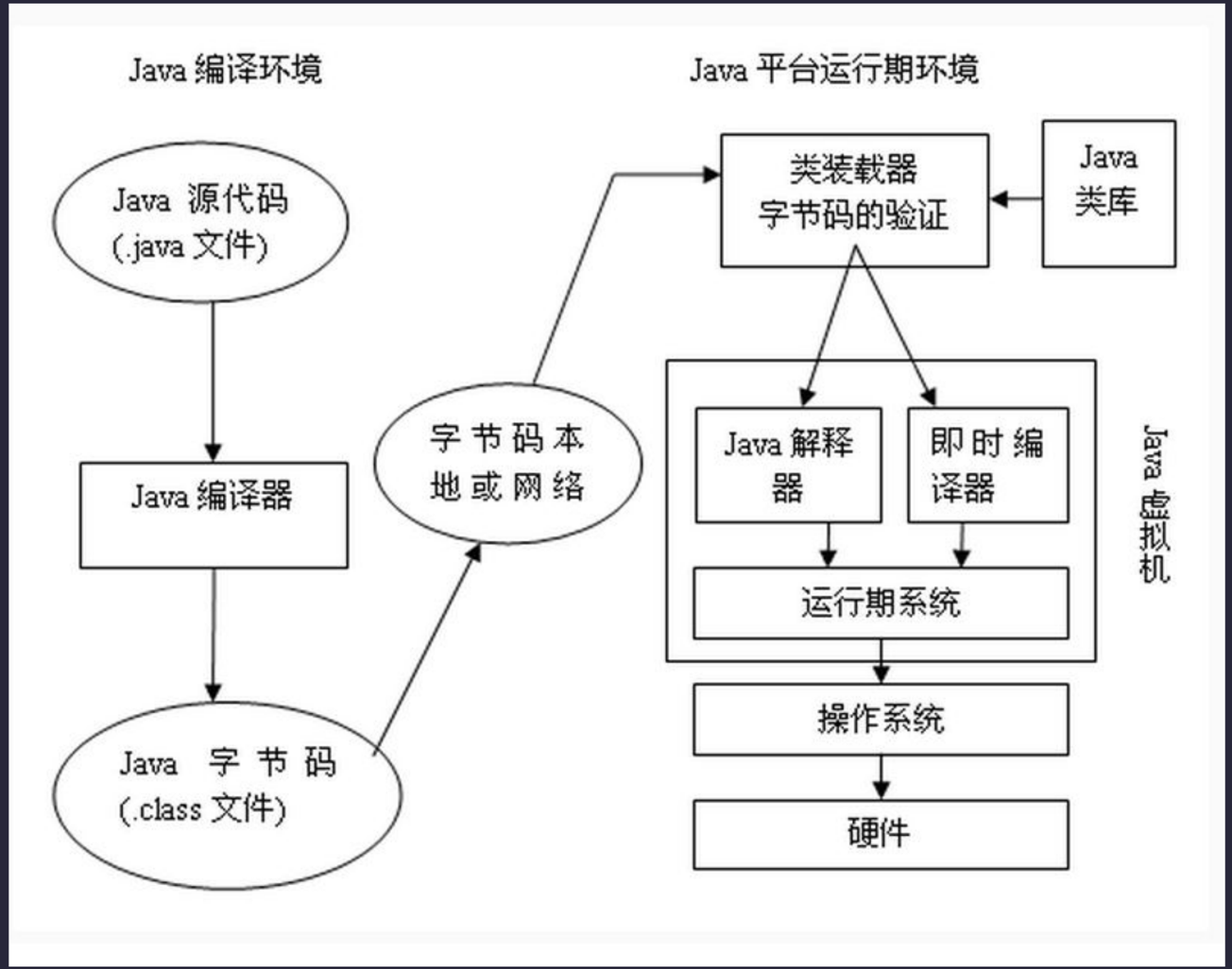
PHP - 一次编译，一次执行

- 调用zend_compile_file()函数完成词法分析、语法分析（输入php源文件，输出op_array即op code）

- 调用zend_execute()函数，执行op code（op命令共150种）
  zend引擎命令分发方式：call（默认）、switch、goto
  效率：goto > switch > call

编译执行

JAVA - 一次编译，处处运行



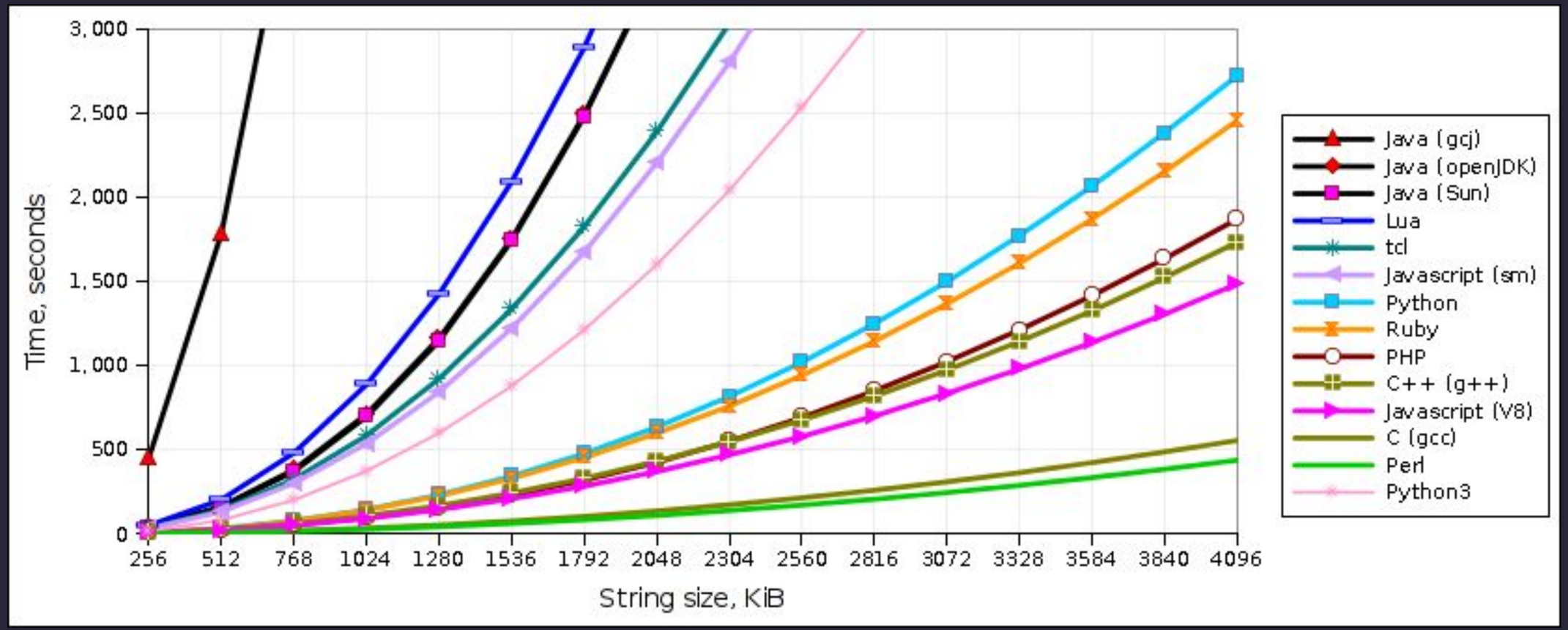
对比（同一个请求运行100次）

- JAVA：一次编译重复执行
  1次编译 + 100次执行

- PHP：每次都要编译 + 执行
  100次编译 + 100次执行

执行效率 之 文本处理

编程语言文本处理速度对比



执行时间易度量，象C一样“快速”的语言

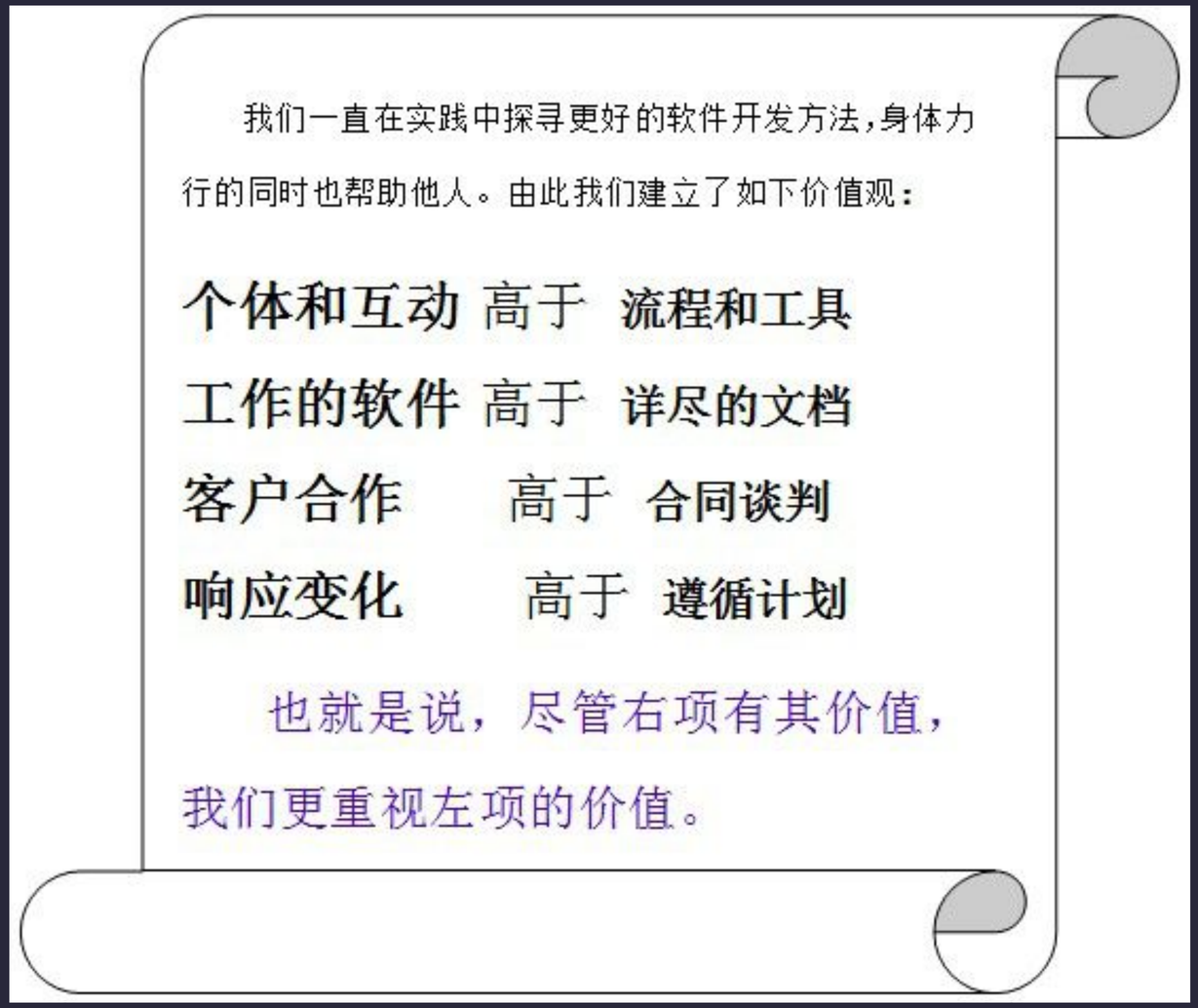
被认为是“高端”语言，但这是片面的

C以及更底层语言，在计算机硬件发展迟缓且系统资源受限的时代，是不可或缺的

然而，当下计算能力是充足且廉价的。软件开发的瓶颈在于软件工程管理和开发人员的时间和精力

更少的代码做更多的事 --> 敏捷宣言

敏捷宣言


我们一直在实践中探寻更好的软件开发方法，身体力行的同时也帮助他人。由此我们建立了如下价值观：
个体和互动 高于 流程和工具
工作的软件 高于 详尽的文档
客户合作 高于 合同谈判
响应变化 高于 遵循计划
也就是说，尽管右项有其价值，我们更重视左项的价值。

设计思想

设计思想 之 语法特色

pipe vs chaining

pipe

```
/* elixir pipe "|>" */
list_data = Store.get_host(host)
map = to_map(list)
formatted_output = tabularize(map)
IO.puts(formatted_output)

host
|> Store.get_host
|> to_map
|> tabularize
|> IO.puts

================================================

/* shell pipe "|" */
log_name=${res_log_pre}"_push_msg_create_raw.log";
raw_data="cat "${log_names}" | grep push/message/create | grep FS_INFO_API | grep -v \"^$\" > "${raw_

log_name=${res_log_pre}"_push_msg_create_raw.log";
raw_data="cat "${log_names}"
            | grep push/message/create
            | grep FS_INFO_API
            | grep -v \"^$\" > "${raw_log_name};

```

chaining

```
27
28 /* jQuery chaining "." */
29 $.ajax("test.html").done(function(){ alert("success");} ).fail(function(){ alert("error"); } ).done(f
30
31 $.ajax("test.html")
32     .done(function(){ alert("success");} )
33     .fail(function(){ alert("error"); } )
34     .done(function(){ alert("second callback");} );
35
```

表述方式类似，但设计思想不同

- pipe针对数据，chaining针对对象
- pipe返回数据，chaining返回对象
- 部分chaining操作有顺序依赖

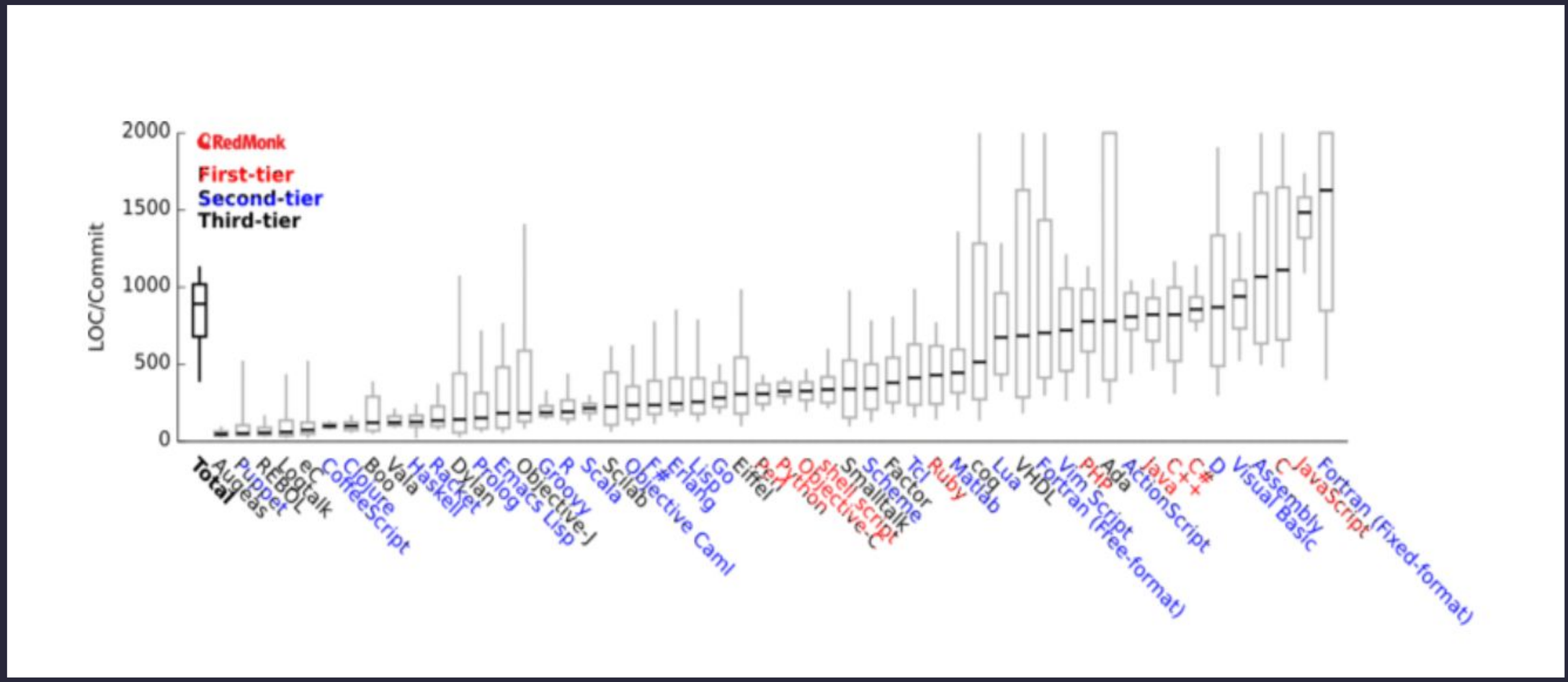
设计思想 之 设计目标

- C：接近机器语言，良好移植性
- C++：高执行效率，直接操作硬件而不引入额外开销
- PHP：Web开发、网页胶水？（Swoole）
- sh：OS跟HardWare交互的命令集
- js：供浏览器执行，处理前后端交互及页面特效

多样化需求  多样化设计目标

复杂度

编程语言代码复杂度统计



C++的语法较PHP更严谨

但C++的代码复杂度、开发周期稍逊于PHP

学习门槛低 == 低端 ？

学习门槛低 == 低端 ？

学习曲线陡峭 == 高端 ？

学习门槛低 == 低端 ？

学习曲线陡峭 == 高端 ？

编程语言只是解决问题的工具

学习门槛低 == 低端 ？

学习曲线陡峭 == 高端 ？

编程语言只是解决问题的工具

喝水，何必在乎杯子的奢华~

六种语言读数据库

PHP

```php
<?php
    $mysql_server_name="localhost";
    $mysql_username="root";
    $mysql_password="123456";
    $mysql_database="test";

    $connection = mysql_connect($mysql_server_name, $mysql_username,$mysql_password);
    if(!$connection) {
        echo "connection failed!";
        return;
    }
    mysql_set_charset("gbk",$connection);
    mysql_select_db($mysql_database, $connection);
    $sql="select * from test";
    $result=mysql_query($sql, $connection);
    while($row = mysql_fetch_array($result)) {
        echo "|".$row["id"]."|".$row["name"]."|\n";
    }
    mysql_close($connection);
?>
```

Python

```python
# coding=utf-8
import MySQLdb
import sys

host = 'localhost'
user = 'root'
password  = '123456'
db    = 'test'


if __name__ == '__main__':
    connection = MySQLdb.connect(host,user,password,db);
    try:
        connection.ping()
    except:
        print ('failed to connect MySQL.')
    sql = 'select * from test'
    cursor = connection.cursor()
    cursor.execute(sql)
    for row in cursor:
        print ("|" + str(row[0]) + "|" + row[1] + "|")
    cursor.close()
    connection.close()
    sys.exit()
```

C

```c
#include "c_mysql.h"

#define HOST "localhost"
#define USERNAME "root"
#define PASSWORD "123456"
#define DATABASE "test"

int main()
{
    char *sql = "select * from test";
    execute_sql(sql);
    return 0;
}

void execute_sql(char* sql)
{
    MYSQL connection;
    MYSQL_RES *result_pointer;
    MYSQL_ROW result_row;
    int result, row, column, i, j;
    mysql_init(&connection);
    if (NULL == mysql_real_connect(&connection, HOST, USERNAME, PASSWORD, DATABASE, 0, NULL, CLIENT_FOUND_ROWS))
    {
        printf("Error:connection failed!\n");
        return;
    }
    mysql_query(&connection, "set names gbk");
    result = mysql_query(&connection, sql);
    if (result)
    {
        printf("Error:query failed!\n");
        mysql_close(&connection);
        return;
    }
    result_pointer = mysql_store_result(&connection);
    if (result_pointer)
    {
        row = mysql_num_rows(result_pointer);
        for (i = 1; i < row + 1; i++)
        {
            result_row = mysql_fetch_row(result_pointer);
            printf("|%s|%s|\n", result_row[0] ,result_row[1]);
        }
    }
    mysql_close(&connection);
    system("pause");
}
```

```c
#ifndef C_MYSQL_H_
#define C_MYSQL_H_

#include <stdio.h>
#include <stdlib.h>
#include <winsock2.h>
#include <windows.h>
#include <mysql.h>

void execute_sql(char* sql);

#endif

```

C++

```cpp
#include "c++_mysql.h"

#define HOST "localhost"
#define USERNAME "root"
#define PASSWORD "123456"
#define DATABASE "test"

int main()
{
    const SQLString sql = "select * from test";
    execute_sql(sql);
    return 0;
}

void execute_sql(const SQLString sql)
{
    mysql::MySQL_Driver *driver;
    Connection *connection;
    Statement *statement;
    ResultSet *result_set;
    driver = mysql::get_mysql_driver_instance();
    connection = driver->connect("tcp://localhost:3306", "root", "123456");
    statement = connection->createStatement();
    statement->execute("use test");
    statement->execute("set names gbk");
    result_set = statement->executeQuery(sql);
    while(result_set->next())
    {
        cout << "|" << result_set->getInt("id") << "|" << result_set->getString("name") << "|" << endl;
    }
    delete statement;
    delete connection;
    system("pause");
}

```

```cpp
#ifndef C___MYSQL_H_
#define C___MYSQL_H_

#include <iostream>
#include <mysql_connection.h>
#include <mysql_driver.h>
#include <statement.h>
using namespace sql;
using namespace std;

void execute_sql(const SQLString sql);

#endif

```

C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using MySql.Data.MySqlClient;

namespace CSMysql
{
    class Program
    {
        static void Main(string[] args)
        {
            MySqlConnection connection = new MySqlConnection("Database='test';Data Source='localhost';User Id='root';Password='123456';charset='utf8';pooling=true");
            MySqlCommand command = new MySqlCommand();
            command.Connection = connection;
            command.CommandText = "select * from test";
            try
            {
                command.Connection.Open();
                MySqlDataReader reader = command.ExecuteReader();
                while (reader.Read())
                {
                    Console.WriteLine("|" + reader.GetInt32("id") + "|" + reader.GetString("name") + "|");
                }
                Console.ReadLine();
            }
            catch (Exception)
            {
                Console.WriteLine("query failed!");
            }
            finally
            {
                command.Connection.Close();
            }
        }
    }
}

```

JAVA

```java
package cn.zxl.jmysql;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class JMysql {

    private static final String DRIVER = "com.mysql.jdbc.Driver";
    private static final String URL = "jdbc:mysql://localhost/test";
    private static final String USERNAME = "root";
    private static final String PASSWORD = "123456";
    private static final String SQL = "select * from test";

    public static void main( String[] args ) {
        Connection connection = null;
        Statement statement = null;
        ResultSet resultSet = null;
        try {
            Class.forName(DRIVER);
            connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);
            statement = connection.createStatement();
            resultSet = statement.executeQuery(SQL);
            while (resultSet.next()) {
                System.out.println("|" + resultSet.getString("id") + "|" + resultSet.getString("name") + "|");
            }
        } catch (Exception e) {
            System.out.println("query failed!");
        } finally {
            try {
                resultSet.close();
                statement.close();
                connection.close();
            } catch (Exception e) {
                throw new RuntimeException(e);
            }
        }
    }

}
```

结论

“高端语言”是个伪命题

“高端语言”是个伪命题

问题的答案取决于你的需要

“高端语言”是个伪命题

问题的答案取决于你的需要

Web开发者需要像PHP或Python那样注重开发效率的语言，满足快速增长业务需求

统计学家和数据分析师需要像R语言那样擅长统计计算和作图的语言，完成统计分析

发散

后端程序猿三座高峰

后端程序猿三座高峰

高并发、海量存储、实时更新

正确的学习姿势

- 语法背后的思想，设计背后的初衷
- 基础知识（进程、网络、内存、文件系统、OOP...）
- 工程化思维、结构化思维、产品意识、商业意识...

Q&A

THE END