阿里云接口使用说明

# 1、获取 APPCODE

在阿里云市场购买相应的 api 接口后，登陆阿里云控制台，左侧菜单最下方有云市场，选择相应的 api 点击使用 api，即可查看自己的 appcode


# 2、请求次数的说明

## 2.1 什么是调用次数

阿里云上的价格表是 XX 元/1000 次 这个次数是什么？
每个接口有独立的请求地址与参数，对应返回不同的结果。
那么使用 Appcode 调用某个接口一次 就算一次的次数。

## 2.2 如何预估需要购买多少次

一般我们会把 API 接口封装到我们的网站或者 APP 中，那么有多少人在使用我们的网站或者 APP 呢？这个大家肯定有一定的统计。
假如我们的网站每天有一千个人访问，每个人访问 10 分钟，我们需要同时调用 2 个接口，而且这 2 个接口每秒更新，那么计算公式如下：
1000 人 乘以 10 分钟 乘以 60 秒 乘以 2 个=120W 次
那么我们每天就需要 120 万次的调用次数 一个月就是 3600W 次
一年大约是 4 亿次

## 2.3 如何减少我的使用次数？

可根据业务需求，进行存储或者缓存，减少接口请求次数。

# 3、GET/POST 方法

代码地址：https://www.kancloud.cn/wangjikeji/wangjiapi/435496
代码说明：无论使用哪种语言，方法都是根据接口地址 url，请求方式 GET/POST，请求参数 params，用户密钥 appcode 来完成一次请求，用户可自行根据接口说明实现 GET/POST，请求时需将 appcode 写入请求 headers 中，格式为：
`'Authorization':'APPCODE 你的 APPCODE'`

## 3.1 ajax 代码

```
$.ajax({
    type: "GET",
    url:"接口请求地址和路径组合链接",
    dataType: "json",
    data: {"key":"value"},//接口的参数
   headers : {'Authorization':'APPCODE 你的APPCODE'},
    beforeSend: function(xhr) {
        xhr.setRequestHeader("Authorization", "APPCODE 你的APPCODE");
    },
    success: function(data){ console.log(data);},//获取到数据，打印到控制台
    error: function(data){ console.log(data); } ,
});
```

## 3.2Python3 代码

```python
import requests

headers = {"Authorization":"APPCODE 您的appcode"}
#定义请求地址
host = ' http://stock.api51.cn '
#定义路径
path = '/trend'
 # 定义参数
querys = 'prod_code=000001.SS'
url = host + path + '?' + querys

datas = requests.get(url=url, headers=headers, timeout=15).content
print(datas)
```

## 3.3java 代码

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.URL;
import java.net.URLConnection;
import java.util.List;
import java.util.Map;

public class chart {
    /**
     * 向指定URL发送GET方法的请求
     *
     * @param url
     *            发送请求的URL
     * @param param
     *            请求参数，请求参数应该是 name1=value1&name2=value2 的形式。
     * @return URL 所代表远程资源的响应结果
     */
    public static void main(String[] args) {
        //发送 GET 请求
        //实时数据
          String s=real.sendGet("http://finance.api51.cn/real", "en_prod_code=UKOIL");
        System.out.println(s);
        //k线数据
        String s=chart.sendGet("http://finance.api51.cn/chart", "candle_period=1&data_count=8&prod_code=UKOIL")
        System.out.println(s);
        /*
        //发送 POST 请求
        String sr=HttpRequest.sendPost("http://localhost:6144/Home/RequestPostString", "key=123&v=456");
        System.out.println(sr);
        */
    }
    public static String sendGet(String url, String param) {
        String result = "";
        BufferedReader in = null;
        try {
            String urlNameString = url + "?" + param;
```

```
            URL realUrl = new URL(urlNameString);
            // 打开和URL之间的连接
            URLConnection connection = realUrl.openConnection();
            // 设置通用的请求属性
            connection.setRequestProperty("accept", "*/*");
            connection.setRequestProperty("connection", "Keep-Alive");
            connection.setRequestProperty("user-agent",
                    "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;SV1)");
            connection.setRequestProperty("Authorization", "APPCODE "+"您的appcode");

            // 建立实际的连接
            connection.connect();
            // 获取所有响应头字段
            Map<String, List<String>> map = connection.getHeaderFields();
            // 遍历所有的响应头字段
            for (String key : map.keySet()) {
             //   System.out.println(key + "--->" + map.get(key));
            }
            // 定义 BufferedReader输入流来读取URL的响应
            in = new BufferedReader(new InputStreamReader(
                    connection.getInputStream()));
            String line;
            while ((line = in.readLine()) != null) {
                result += line;
            }
        } catch (Exception e) {
            System.out.println("发送GET请求出现异常！" + e);
            e.printStackTrace();
        }
        // 使用finally块来关闭输入流
        finally {
            try {
                if (in != null) {
                    in.close();
                }
            } catch (Exception e2) {
                e2.printStackTrace();
            }
        }
        return result;
    }
```

```
    /**
     * 向指定 URL 发送POST方法的请求
     *
     * @param url
     *            发送请求的 URL
     * @param param
     *            请求参数，请求参数应该是 name1=value1&name2=value2 的形式。
     * @return 所代表远程资源的响应结果
     */
    public static String sendPost(String url, String param) {
        PrintWriter out = null;
        BufferedReader in = null;
        String result = "";
        try {
            URL realUrl = new URL(url);
            // 打开和URL之间的连接
            URLConnection conn = realUrl.openConnection();
            // 设置通用的请求属性
            conn.setRequestProperty("accept", "*/*");
            conn.setRequestProperty("connection", "Keep-Alive");
            conn.setRequestProperty("user-agent",
                    "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;SV1)");
            // 发送POST请求必须设置如下两行
            conn.setDoOutput(true);
            conn.setDoInput(true);
            // 获取URLConnection对象对应的输出流
            out = new PrintWriter(conn.getOutputStream());
            // 发送请求参数
            out.print(param);
            // flush输出流的缓冲
            out.flush();
            // 定义BufferedReader输入流来读取URL的响应
            in = new BufferedReader(
                    new InputStreamReader(conn.getInputStream()));
            String line;
            while ((line = in.readLine()) != null) {
                result += line;
            }
        } catch (Exception e) {
            System.out.println("发送 POST 请求出现异常！"+e);
            e.printStackTrace();
        }
```

```
        //使用finally块来关闭输出流、输入流
        finally{
            try{
                if(out!=null){
                    out.close();
                }
                if(in!=null){
                    in.close();
                }
            }
            catch(IOException ex){
                ex.printStackTrace();
            }
        }
        return result;
    }
}
```