北京中航易购信息服务有限公司



【“运价云”接口说明】

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 创建时间 | 变更类型  （增**\***删**\***改） | 修改人 | 修改内容 |
| V1.0 | 2015-11-16 | 增 |  | 初始版 |

目录

[1 易购云 Open API概述/说明 3](#_Toc436224627)

[1.1 Open API作用及优点 3](#_Toc436224628)

[2 准备 3](#_Toc436224629)

[2.1 接口调试工具 3](#_Toc436224630)

[2.2 XML工具 3](#_Toc436224631)

[2.3 XPATH语法 4](#_Toc436224632)

[3 一次接口调用实例 4](#_Toc436224633)

[3.1 Open API接口的引用 4](#_Toc436224634)

[3.2 Open API接口统一方法XmlSubmit 4](#_Toc436224635)

[3.3 Open API调用实例（C#） 5](#_Toc436224636)

[4 登陆（身份认证） 6](#_Toc436224637)

[4.1 DoLogin\_2\_0登陆身份认证接口（加密接口） 6](#_Toc436224638)

[4.1.1 请求参数说明 6](#_Toc436224639)

[4.1.2 返回参数说明 6](#_Toc436224640)

[5 租车查询模块 7](#_Toc436224641)

[5.1 QueryCarRentalCitys\_1\_0获取租车城市列表接口 7](#_Toc436224642)

[5.1.1 请求参数说明 7](#_Toc436224643)

[5.1.2 过滤参数说明 7](#_Toc436224644)

[5.1.3 返回参数说明 8](#_Toc436224645)

[5.2 QueryCarRentalList\_1\_0获取城市车型接口 8](#_Toc436224646)

[5.2.1 请求参数说明 8](#_Toc436224647)

[5.2.2 过滤参数说明 8](#_Toc436224648)

[5.2.3 返回参数说明 9](#_Toc436224649)

[5.3 QueryCarRentalAddress\_1\_0 地址联想接口 9](#_Toc436224650)

[5.3.1 请求参数说明 9](#_Toc436224651)

[5.3.2 过滤参数说明 9](#_Toc436224652)

[5.3.3 返回参数说明 10](#_Toc436224653)

[5.4 GetCityPickupLocation\_1\_0 城市接送地点获取接口 10](#_Toc436224654)

[5.4.1 请求参数说明 10](#_Toc436224655)

[5.4.2 过滤参数说明 11](#_Toc436224656)

[5.4.3 返回参数说明 11](#_Toc436224657)

[5.5 QueryCarRentalPrice\_1\_0价格预估接口 12](#_Toc436224658)

[5.5.1 请求参数说明 12](#_Toc436224659)

[5.5.2 过滤参数说明 13](#_Toc436224660)

[5.5.3 返回参数说明 13](#_Toc436224661)

# 易购云 Open API概述/说明

易购云Open API简称“易购云开放应用接口”。请求数据为标准XML格式，响应数据支持标准XML格式和JSON(JavaScript Object Notation)格式。

## Open API作用及优点

通过Open API 的XmlSubmit接口方法可完成机票航班、运价相关功能的开发，一次引用即可完成权限允许范围内的功能请求，

调用简单，灵活，所有的调用请求只通过一个入口XmlSubmit（不同请求XML结构）既可完成。任何种程序语言只要能访问WebService 就可调用开发，不需要插件。

如果您是第一次调用Open API接口作相关开发，请按手册目录顺序自上而下查看本手册。

“调用方法”里描述了如何运用Open API接口作开发。

“功能详解”里描述了调用Open API可以完成的功能，并说明了每一个功能的详细请求参数。

# 准备

## 接口调试工具

推介工具：TestWSDL.exe

工具说明：.NET自带，小而实用，不需要安装使用简单。

使用心得：在调用开发前可以对接口作调用测试以了解熟悉接口参数及返回数据，在开发过程中如遇到接口返回错误信息，可以用WSDL工具作相同的调用操作并查看结果，灵活使用有助于在开发过来程中的异常问题。

## XML工具

推介工具：XML Spy

工具说明：XML Spy 是一个集成了编辑、检验、预览等多项功能的商业性 XML 开发软件，支持 XML 文档所见即所得的编辑方式、XPATH语法查询、支持 UNICODE 、多字符集，支持格式良好的和有效的 XML 文档，同时提供了强有力的样式表设计。

使用心得：因为Open API接口参数和返回数据都为XML结构，所以有一个强大实用的XML工具对开发和解决问题有大帮助，它对XML数据结构有检验、XPATH检索、视图查看等功能很方便。

## XPATH语法

XPATH简介：XPath是一门在 XML 文档中查找信息的语言。其有语法、运算符、函数等，是 W3C XSLT 标准的主要元素。使用路径表达式来选取 XML 文档中的节点或者节点集。使用XPATH读取XML很灵活，读取大量XML数据的性能、效率和代码量都优于一般XML处理方法。

于由篇幅较长，且本文档着重介绍接口，详细XPATH语法介绍请到：[w3school](http://www.w3school.com.cn/xpath/index.asp)参考。

C#很好的集成了XPATH，using System.Xml;

JAVA需要javax.xml.xpath包，详细请参考：[IBM:Java 语言的 XPath API](http://www.ibm.com/developerworks/cn/xml/x-javaxpathapi.html)

# 一次接口调用实例

## Open API接口的引用

以WebServices方式引用，

地址：http://api服务地址/aois/ysta.asmx

引用接口方法：XmlSubmit。

## Open API接口统一方法XmlSubmit

接口方法：public string XmlSubmit(string identity, string request, string filter)

参数：

Identity：身份认证信息

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<Identity\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **Operator** | 登录操作工号 | 字符 |  | Y |
| **Pwd** | 密码 | 字符 |  | Y(Pwd与PassportID必填其中一项) |
| **PassportID** | 身份凭证号 | 字符 |  | Y(通过PassportID认证模式，不需填入Pwd) |
| **Terminal** | 终端号 | 字符 |  | N |
| **UserType** | 登陆身份类型 | 字符 | 固定值：Platform | Y |
| **样例数据** | <Identity\_1\_0><Operator>admin</Operator><PassportID>FDSAGFGFJHGLKKJHLREWQREWQTYRIYUYTOUY</PassportID><Terminal></Terminal><UserType>Platform</UserType></Identity\_1\_0> | | | | |

<Identity\_1\_0><Operator>admin</Operator><PassportID>FDSAGFGFJHGLKKJHLREWQREWQTYRIYUYTOUY</PassportID><Terminal></Terminal>**<UserType>Platform</UserType>**</Identity\_1\_0>

request：结构化查询条件，请参考功能接口各方法说明

filter：过滤条件，请参考功能接口各方法说明

## Open API调用实例（C#）

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

YeeGo.OpenAPI.FareCloud api = new YeeGo.OpenAPI.FareCloud();

//身份XML

StringBuilder identity = new StringBuilder();

identity.Append("<?xml version=\"1.0\" encoding=\"utf- 8\"?>");

identity.Append("<Identity\_1\_0>");

identity.Append("<Operator>UserName</Operator>"); //登录账号

identity.Append("<Pwd>PassWord</Pwd>"); //登录密码

identity.Append("<Terminal></Terminal>"); //终端号

identity.Append("<UserType>Platform</UserType>");

identity.Append("</Identity\_1\_0>");

//请求XML 这里我们调用1.3版航班查询QueryWebFlights\_1\_3

StringBuilder request = new StringBuilder("<?xml version=\"1.0\"?>");

request.Append("<QueryWebFlights\_1\_3>");

request.Append("<From>PEK</From>"); // 起飞机场三字代码

request.Append("<Arrive>DLC</Arrive>"); // 到达机场三字代码

request.Append("<Date>2014-03-30</Date>"); // 航班日期

request.Append("<Carrier></Carrier>"); // 航空公司两字代码

request.Append("<Time>0000</Time>"); // 起飞时间

request.Append("<StopType>D</StopType>"); // D:直飞（默认） N:所有

request.Append("<Option/>"); // 系统保留

request.Append("<CmdShare>0</CmdShare>"); // 是否查询共享航班,0:不查询共享航班

request.Append("<PrdType>R</PrdType>"); // 产品类型

request.Append("</QueryWebFlights\_1\_3>");

//过滤条件XML

StringBuilder filter = new StringBuilder("<Filter\_1\_0><Compress>N</Compress></Filter\_1\_0>");//生产环境建议Compress=Y

string resStr = api.XmlSubmit(identity.ToString(), request.ToString(), filter.ToString()); //访问接口并返回请求结果resStr

//resStr为一个XML描述，如果根节点为ErrorInfo\_1\_0则为错误描述，否则为请求参数request XML的根节点，也就是此例中的QueryWebFlights\_1\_3

XmlDocument xmlDoc = new XmlDocument();

xmlDoc.LoadXml(resStr);

//判断是否返回错误

if (xmlDoc.DocumentElement.Name == "ErrorInfo\_1\_0" && xmlDoc.Name != "")

{

//错误描述

XmlElement rx = xmlDoc.DocumentElement;

string errMsg = rx["Content"].InnerText.Replace("\r", "").Replace("\n", "").ToString();

//输出错误信息

throw new Exception("10001：查询航班出错，详细：" + errMsg);

}

else

{

//接口请求成功

//处理航班查询结果

}

# 登陆（身份认证）

## DoLogin\_2\_0登陆身份认证接口（加密接口）

登录(身份认证)接口,加密登录密码信息。

说明：也可使用普通认证模式，登录身份信息拼写方法参见”3.3 OpenAPI调用实例”。

### 请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<DoLogin\_2\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Operator>** | 登录操作工号 | 字符 |  | Y |
| **<Terminal>** | 终端号 | 字符 |  | N |
| **<UserType>** | 登陆身份类型，填写：Platform | 字符 |  | Y |
| **<Authenticator>** | 认证码 | 字符 |  | Y |
| **样例数据** | <?xml version="1.0" encoding="utf-8"?><DoLogin\_2\_0><Operator>admin</Operator><Terminal/><UserType>Platform</UserType><Authenticator>DSAFDSAFDSAFDSAFEWQR</Authenticator></DoLogin\_2\_0> | | | | |

**认证码取值算法：**

Authenticator = DES(key, Operator + "$" + Base64Encrypt(Password))

加密算法使用DES, key = Password，如key长度不足8位，末尾以空格补至8位。

### 返回参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<DoLogin\_2\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Operator>** | 登录操作工号 | 字符 |  | Y |
| **<PassportID>** | 身份凭证号 | 字符 |  | Y |
| **样例数据** | <DoLogin\_2\_0><Operator>admin</Operator><PassportID>JLKJLKJLJLDSAFDSAFDSAFDSAFEWQR</PassportID></DoLogin\_2\_0> | | | | |

如失败见“标准错误结果文档”。

# 租车查询模块

## QueryCarRentalCitys\_1\_0获取租车城市列表接口

此接口用于查询租车的城市列表。

### 请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<QueryCarRentalCitys\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<CitySearchValue>** | 搜索城市的条件，可以是城市名称，也可以是城市区号，为空时返回所有的城市 | 字符 | 20 | 否 |
| **样例数据** | <QueryCarRentalCitys\_1\_0>  <CitySearchValue>010</CitySearchValue>  </QueryCarRentalCitys\_1\_0> | | | | |

### 过滤参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<Filter\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Compress>** | 是否返回压缩格式数据(deflate压缩格式)  N 返回原始结果(默认)  Y 返回压缩结果(推荐) | 字符 | 1 | N |
| **样例数据** | <Filter\_1\_0><Compress>Y</Compress></Filter\_1\_0> | | | | |

### 返回参数说明

|  |  |
| --- | --- |
| **根节点** | { |
| **子节点** | "CityData": [  [城市ID ,"城市名称","城市区号"],  ] |
| 城市列表格式，例子："{CityData":[[1,"北京市","010"],[1,"北京市","010"]]} |
| **样例数据** |  |

## QueryCarRentalList\_1\_0获取城市车型接口

此接口用于查询对应城市的车型列表。

### 请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<QueryCarRentalList\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<CityCode>** | 城市代码,对应上面5.1接口返回的城市ID，如[[1,"北京市","010"]中的1 | 数字 | 3 | 否 |
| **<RuleType>** | 计价模型分类：  201(专车)  202(包车套餐)  301(快车)  默认为201 | 字符 | 3 | 否 |
| **样例数据** | <QueryCarRentalList\_1\_0>  <CityCode>1</CityCode>  <RuleType>201</RuleType>  </QueryCarRentalList\_1\_0> | | | | |

### 过滤参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<Filter\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Compress>** | 是否返回压缩格式数据(deflate压缩格式)  N 返回原始结果(默认)  Y 返回压缩结果(推荐) | 字符 | 1 | N |
| **样例数据** | <Filter\_1\_0><Compress>Y</Compress></Filter\_1\_0> | | | | |

### 返回参数说明

|  |  |
| --- | --- |
| **根节点** | { |
| **子节点** | "CarsData": [  ["车型ID ","车型名称",起步价,每公里单价],  ] |
| 车型列表格式，例子："{CarsData":[["200","豪华型",22, 4.6]，["200","豪华型",22, 4.6]]} |
| **样例数据** |  |

## QueryCarRental[Address](https://es.api.xiaojukeji.com/v1/common/Address/getAddress)\_1\_0 地址联想接口

此接口用于通过指定的搜索关键词，返回对应的地点名称及精确坐标

### 请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<QueryCarRentalAddress\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<CityName>** | 城市名称，如北京市 | 字符 | 10 | 是 |
| **<SearchValue>** | 搜索词 | 字符 | 20 | 是 |
| **样例数据** | <QueryCarRentalAddress\_1\_0>  <CityName>北京市</CityName>  <SearchValue>西站</SearchValue>  </QueryCarRentalAddress\_1\_0> | | | | |

### 过滤参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<Filter\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Compress>** | 是否返回压缩格式数据(deflate压缩格式)  N 返回原始结果(默认)  Y 返回压缩结果(推荐) | 字符 | 1 | N |
| **样例数据** | <Filter\_1\_0><Compress>Y</Compress></Filter\_1\_0> | | | | |

### 返回参数说明

|  |  |
| --- | --- |
| **根节点** | { |
| **子节点** | "AddressData": [  [  城市ID,  "北京市",  "首都机场",  "详细地址",  "经度",  "纬度"  ]  ] |
| **样例数据** | {"AddressData": [  [  1,  "北京市",  "首都机场",  "",  "116.59686597618",  "40.082321212787"  ], [  1,  "北京市",  "首都机场",  "",  "116.59686597618",  "40.082321212787"  ]  ]} |

## GetCityPickupLocation\_1\_0 城市接送地点获取接口

此接口用于通过指定的城市名称，返回对应的城市中所有机场和火车站地点名称及精确坐标

### 请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<GetCityPickupLocation\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<CityName>** | 城市名称，如北京市 | 字符 | 10 | 是 |
| **样例数据** | <GetCityPickupLocation\_1\_0>  <CityName>北京市</CityName>  </GetCityPickupLocation\_1\_0> | | | | |

### 过滤参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<Filter\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Compress>** | 是否返回压缩格式数据(deflate压缩格式)  N 返回原始结果(默认)  Y 返回压缩结果(推荐) | 字符 | 1 | N |
| **样例数据** | <Filter\_1\_0><Compress>Y</Compress></Filter\_1\_0> | | | | |

### 返回参数说明

|  |  |
| --- | --- |
| **根节点** | { |
| **子节点** | "AddressData": [  [  城市ID,  "北京市",  "首都机场",  "详细地址",  "经度",  "纬度"  ]  ] |
| **样例数据** | {"AddressData": [  [  1,  "北京市",  "首都机场",  "",  "116.59686597618",  "40.082321212787"  ], [  1,  "北京市",  "首都机场",  "",  "116.59686597618",  "40.082321212787"  ]  ]} |

## QueryCarRentalPrice\_1\_0价格预估接口

此接口用于根据指定地点推算车型价格。

### 请求参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<QueryCarRentalPrice\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<FromLat>** | 出发地纬度 | 字符 | 30 | 是 |
| **<FromLng>** | 出发地经度 | 字符 | 30 | 是 |
| **<ArriveLat>** | 目的地纬度 | 字符 | 30 | 是 |
| **<ArriveLng>** | 目的地经度 | 字符 | 30 | 是 |
| **<MapType>** | 地图类型，暂只支持soso | 字符 | 10 | 否 |
| **<CarType>** | 车型代码（专车如：100、200等；快车如：600等） | 数字 | 3 | 否 |
| **<RuleType>** | 计价模型分类:  201(专车)  202(包车套餐)  301(快车) | 数字 | 3 | 是 |
| **<CityID>** | 出发城市ID | 数字 | 3 | 是 |
| **<Date>** | 出发时间(不传或者为空表示当前时间，格式例如：2015-06-16 12:00:09) | 字符 | 20 | 否 |
| **样例数据** | <QueryCarRentalPrice\_1\_0>  <FromLat>34.24728</FromLat>  <FromLng>108.97127</FromLng>  <ArriveLat>34.177297</ArriveLat>  <ArriveLng>108.918465</ArriveLng>  <MapType/>  <CarType/>  <RuleType>201</RuleType>  <CityID>10</CityID>  <Date/>  </QueryCarRentalPrice\_1\_0> | | | | |

### 过滤参数说明

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **根节点** | **<Filter\_1\_0>** | | | | |
| **子节点** | **节点名称** | **说明** | **数据类型** | **长度** | **是否必填** |
| **<Compress>** | 是否返回压缩格式数据(deflate压缩格式)  N 返回原始结果(默认)  Y 返回压缩结果(推荐) | 字符 | 1 | N |
| **样例数据** | <Filter\_1\_0><Compress>Y</Compress></Filter\_1\_0> | | | | |

### 返回参数说明

|  |  |
| --- | --- |
| **根节点** | { |
| **子节点** | "PriceData": [  [  "单车类型名称",  "单车类型的对应码（100舒适型，400商务型, 200豪华型, 500优选型）",  "价格 单位：元",  "起步价格 单位：元",  "每公里单价 单位：元"  ]  ] |
| **样例数据** | {"PriceData": [  [  "优选型:卡罗拉、科鲁兹等",  "500",  "31",  "13",  "2.3"  ], [  "优选型:卡罗拉、科鲁兹等",  "500",  "31",  "13",  "2.3"  ]  ]} |