

在线分销产品
自动废票服务用户手册（非订单）

目录

- 1. 概述..... 1
- 2. 版本..... 1
- 3. 业务范围..... 1
- 4. 功能范围..... 1
- 5. 接口说明..... 2
 - 5.1 接口结构..... 2
 - 5.1.1 请求结构..... 2
 - 5.1.2 响应结构..... 2
 - 5.2 请求参数说明..... 3
 - 5.3 响应参数说明..... 3
- 6. 错误说明..... 4
 - 6.1 http 协议错误..... 4
 - 6.2 接口错误..... 4
- 7. 参考样例..... 5
 - 7.1 非订单废票..... 5
- 8.接口调用..... 6
 - 8.1 接口申请..... 6
 - 8.2 接口认证..... 6
 - 8.3 接口地址..... 6
 - 8.4 JAVA 调用示例..... 6
 - 8.5.NET 调用示例..... 9

1. 概述

自动废票服务是在线分销平台上的一个自动化操作环节的重要服务，提供销售环节的服务功能。

自动废票服务(AirTicketVoid)提供的服务是：在航信在线分销平台上，根据客户请求，对已出票的 PNR 进行废票的功能，主要涉及到废票。

2. 版本

主要内容	自动废票				
参考文档					
创建部门					
批准单					
版本号：					
批准人		批准人签字		批准日期	
文档控制					
版本	修改日期	修改内容描述	作者	批准人	批准日期
V1.0.0	2013.03.25	创建	王宏文		
V1.0.1	2013.04.22	修改	王文亚		

3. 业务范围

本接口实现自动废票的功能。

4. 功能范围

本版本支持的自动废票场景是：

1、非订单直接废票（pnr 号+1 张票号）。

5. 接口说明

5.1 接口结构

5.1.1 请求结构

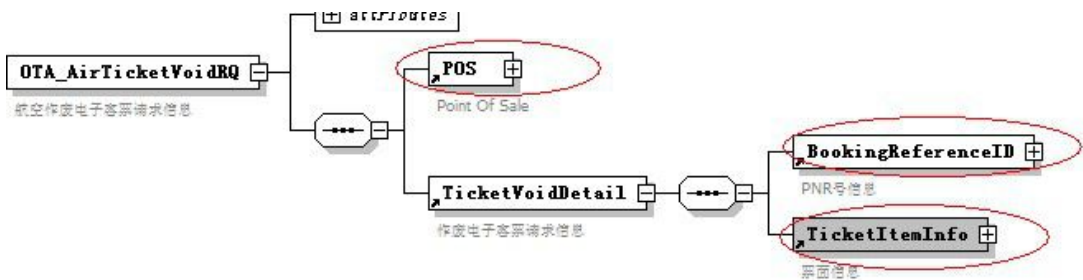


图 5-1

其中标识红色椭圆线的节点为废票请求时的输入项，说明如下：

POS 节点：销售方信息，具体输入信息见下，参数说明列表；

BookingReferenceID 节点：pnr 号信息，具体输入信息见参数说明列表；

TicketItemInfo 节点：主要输入票号，具体输入信息见参数说明列表。

5.1.2 响应结构

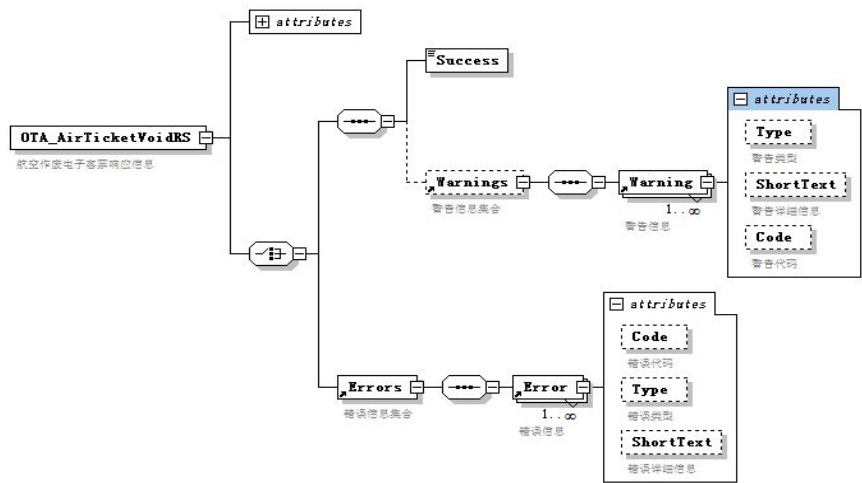


图 5-2

说明如下：

Success 节点：成功后返回该节点即可。

Error 节点：存入操作失败的客票信息。

Warnings 节点：存入部分失败操作后的失败的客票信息。

5.2 请求参数说明

元素名称	说明	Rep	类型	默认值	备注
OTA_AirTicketVoidRQ		1			
POS		1			
Source	资源信息	1			
@PseudoCityCode	Office 号	1	String		
@OtherId	打票机号	1	String		
TicketVoidDetail	废票信息	1			
BookingReferenceID	订单号或 PNR 号信息	1			
@ID	订单号或 PNR 号	1	String		
@ID_Context	类别	1	String	PNR	
TicketItemInfo	票号信息	1	String		
@TicketNumber	票号	1	String		

上表对请求输入时的常用属性字段进行说明。其他字段可参考 XSD。

5.3 响应参数说明

下表是对响应输出中常用的属性字段进行说明，其他字段可参考 XSD。

元素名称	说明	Rep	类型	默认值	备注
OTA_AirTicketVoidRS		1			
Errors	错误信息	0..1			至多 1 个
Error		0..n			可输出多个
@Code	错误代码	0..1	string		
@Type	错误类型	0..1	string		
@ShortText	错误详细信息	0..1	string		
Warnings		0..1			至多 1 个
Warning		0..n			可输出多个

@Code	警告代码	0..1	string	
@Type	警告类型	0..1	string	
@ShortText	警告详细信息	0..1	string	
OTA_AirTicketVoidRSSquence				
@Success	成功信息	0..1	string	至多 1 个

6. 错误说明

6.1 http 协议错误

由于接口为 HTTP 接口，HTTP 协议会返回 403 错误：

http 错误代码	说明
403	用户名、密码、IP 错误或无权限

6.2 接口错误

<Errors>节点记录了错误信息。错误信息分为两类：业务错误、系统错误。
详细见下表：

系统错误

错误代码	错误描述	说明
-303	Inner service error!	内部服务错误

业务错误

错误代码	错误描述	说明
-400	format of xml string is not valid	xml 内容格式错误
-402	input error	输入错误
-500	office code is empty	office 代码为空
-560	pnr locator is empty	pnr 号为空
-561	printer number is empty	打印机号为空
-563	ticket number is empty	票号为空
-790	format of ticket number is not valid	票号格式错误
-10060	pnr does not exist	pnr 不存在
-50011	status of printer is not normal	打票机状态错误
-50020	status of ticket is not valid	客票状态错误
-9999	unknown error	未知错误

7. 参考样例

输入请求的模板说明如下，其他信息可根据业务需要对应到相应的节点中，各个节点说明可参考 OTA_AirTicketVoid.xsd。自动废票的请求输入：

```
<OTA_AirTicketVoidRQ>
  <POS>
    <!-- PseudoCityCode = Office 号   OtherID = 打票机号-->
    <Source PseudoCityCode="bjs187" OtherID="60" />
  </POS>
  <!-- 票号,只有一个 TicketVoidDetail 节点-->
  <TicketVoidDetail>
    <!-- pnr 号 , 必输-->
    <BookingReferenceID ID="HQRHX7" ID_Context="PNR" />
    <!-- 票号 , 必输-->
    <TicketItemInfo TicketNumber="999-3944318760" />
  </TicketVoidDetail>
</OTA_AirTicketVoidRQ>
```

以下提供多种类型的请求和相应的返回样例，供参考。

7.1 非订单废票

RQ:

```
<OTA_AirTicketVoidRQ>
  <POS>
    <Source PseudoCityCode="bjs187" OtherID="60" />
  </POS>
  <TicketVoidDetail>
    <BookingReferenceID ID="HQRHX7" ID_Context="PNR" />
    <TicketItemInfo TicketNumber="999-3944318760" />
  </TicketVoidDetail>
</OTA_AirTicketVoidRQ>
```

RS:

```
<OTA_AirTicketVoidRS>
  <Success/>
</OTA_AirTicketVoidRS>
```

8.接口调用

自动废票服务提供基于 XML 文档规范的标准 HTTP 协议接口，通过在标准 HTTP 通道上传 XML 格式的请求和数据来完成服务调用。服务的输入输出定义在标准的 XSD 文档中，用户根据自有系统的实现环境结合 XSD 文档进行数据到对象间的转换。在线分销提供服务接口 URL,用户使用 HTTP 协议按照服务定义正确构造请求输入，正确解析数据输出实现服务的调用。

8.1 接口申请

请与您所在城市的中国航信分支机构联系，资质审查通过后签署试用协议，即可获得试用接口。

8.2 接口认证

授权用户调用试用接口时需要使用已分配的账号和密码、并使用申请时的 IP。如果已变化 IP 请联系业务人员申请变更 IP。

8.3 接口地址

接口描述位置：<http://espeed.travelsky.com/develop/xml/AirTicketVoid?xsd>

接口试用地址：<http://espeed.travelsky.com/develop/xml/AirTicketVoid>

8.4 JAVA 调用示例

Java 调用 http 接口示例如下：

其中该调用示例程序只是为了展示如何接入使用

```
import java.io.BufferedReader;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.zip.GZIPInputStream;
import java.util.zip.GZIPOutputStream;

import org.apache.commons.httpclient.Credentials;
import org.apache.commons.httpclient.DefaultHttpMethodRetryHandler;
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.HttpState;
import org.apache.commons.httpclient.HttpStatus;
```



```
import org.apache.commons.httpclient.UsernamePasswordCredentials;
import org.apache.commons.httpclient.auth.AuthScope;
import org.apache.commons.httpclient.methods.ByteArrayRequestEntity;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.methods.RequestEntity;
import org.apache.commons.httpclient.params.HttpMethodParams;

/**
 * http 客户端调用示例(数据压缩版)
 */
public class HttpClientDemo {

    public static void main(String[] args) {

        //实际使用时，请设置正确的变量值
        //用户名
        String username = "用户名";
        //密码
        String pwd = "密码";
        //服务地址
        String serviceUrl = "服务地址";
        //请求 XML
        String reqStr = "请求 xml";

        // 构造 HttpClient 的实例
        HttpClient httpClient = new HttpClient();
        //调用验证信息
        HttpState state = new HttpState();
        Credentials credentials = new UsernamePasswordCredentials(username, pwd);
        state.setCredentials(AuthScope.ANY, credentials);
        httpClient.setState(state);

        // 创建 POST 方法的实例
        PostMethod postMethod = new PostMethod(serviceUrl);
        // 使用系统提供的默认的恢复策略
        postMethod.getParams().setParameter(HttpMethodParams.RETRY_HANDLER,
            new DefaultHttpMethodRetryHandler());
        try {
            // 请求参数的数据压缩
            ByteArrayOutputStream out = new ByteArrayOutputStream();
            GZIPOutputStream gzip = new GZIPOutputStream(out);
            if (reqStr != null && !"".equals(reqStr)) {
                gzip.write(reqStr.getBytes());
            }
        }
```

```

        gzip.close();
        RequestEntity requestEntity = new ByteArrayRequestEntity(out
            .toByteArray());
        postMethod.setRequestEntity(requestEntity);
        postMethod.addRequestHeader("Content-Type",
            "text/html;charset=UTF-8");
        postMethod.addRequestHeader("accept-encoding", "gzip");
        postMethod.addRequestHeader("content-encoding", "gzip");

        // 执行 getMethod
        int statusCode = httpClient.executeMethod(postMethod);
        if (statusCode != HttpStatus.SC_OK) {
            throw new Exception("Invoke Get Method Failed, HttpStatus = "
                + statusCode);
        }

        // 返回结果的数据解压
        InputStream is = postMethod.getResponseBodyAsStream();
        BufferedReader br = new BufferedReader(new InputStreamReader(
            new GZIPInputStream(is)));

        StringBuffer respStr = new StringBuffer();
        String line = null;
        while ((line = br.readLine()) != null) {
            respStr.append(line);
        }

        // 打印结果
        System.out.println(respStr.toString());

        out.flush();
        out.close();
        is.close();
    } catch (Exception e) {
        // 发生致命的异常，可能是协议不对或者返回的内容有问题
        e.printStackTrace();
    } finally {
        // 释放连接
        postMethod.releaseConnection();
    }
}
}

```

8. 5. NET 调用示例

.Net 调用 http 接口示例如下：

其中该调用示例程序只是为了展示如何接入使用

(1)建立一个 cs 文件,与服务器通信的类：

// 定义 xml 与服务器的通信

```
class OTA_XMLHTTP
```

```
{
```

```
    string user;    //用户名
```

```
    string pwd;    //密码
```

```
    string url;    //服务地址
```

```
    public OTA_XMLHTTP(string _user,string _pwd,string _url)
```

```
    {
```

```
        user = _user;
```

```
        pwd = _pwd;
```

```
        url = _url;
```

```
    }
```

```
    // 获取服务器返回的字符串
```

```
    public string GetResponse(string requestXml)
```

```
    {
```

```
        string xmlString = "";    //要返回的 xml 字符串
```

```
        bool a= requestXml.Contains("<?xml version='1.0' encoding='UTF-8'?>");
```

```
        requestXml = requestXml.Replace("<?xml version='1.0' encoding='UTF-8'?>\n", "");
```

```
        try
```

```
        {
```

```
            //制定服务器地址
```

```
            HttpWebRequest request = (HttpWebRequest)HttpWebRequest.Create(url);
```

```
            request.Method = WebRequestMethods.Http.Post; //设定 http 的传递方式
```

```
            request.ContentType = "application/x-www-form-urlencoded";
```

```
            //设定 http 的 header
```

```
            if (user != null && pwd != null)
```

```
            {
```

```
                string user_pwd = user + ":" + pwd;
```

```
                byte[] authBytes = Encoding.UTF8.GetBytes(user_pwd.ToCharArray());
```

```
                request.Headers.Add("Authorization", "Basic " + Convert.ToBase64String(authBytes));
```

```
                request.Headers.Add("Content-Encoding", "gzip");
```

```
                request.Headers.Add("Accept-Encoding", "gzip");
```

```
            }
```

```
            //将数据写入流中
```

```
            string para = "request";
```

```
            ASCIIEncoding encoding = new ASCIIEncoding();
```

```
            byte[] data = encoding.GetBytes(para + " = " + requestXml);
```

```

        request.ContentLength = data.Length;
        Stream stream = request.GetRequestStream();
        stream.Write(data, 0, data.Length);
        stream.Close();
        //建立获取 http 返回的 response
        HttpResponseMessage response = (HttpResponseMessage)request.GetResponse();
        //读取 http 返回的字符串
        Stream responseStream = response.GetResponseStream();
        if (response.ContentEncoding.ToLower().Contains("gzip"))
            responseStream = new GZipStream(responseStream, CompressionMode.Decompress);
        else if (response.ContentEncoding.ToLower().Contains("deflate"))
            responseStream = new DeflateStream(responseStream, CompressionMode.Decompress);
        StreamReader streamReader = new StreamReader(responseStream, Encoding.UTF8);
        xmlString = streamReader.ReadToEnd();
        streamReader.Close();
    }
    catch (WebException e)
    {
        xmlString = e.Message;
    }
    return xmlString;
}
}

```

(2) 调用上面的文件

调用方法:

```

string user="用户名";
string pwd="密码";
string url="服务地址";
OTA_XMLHTTP xmlHttp = new OTA_XMLHTTP(user, pwd, url);
PonseXmlBox.Text = xmlHttp.GetResponse(requestXml);

```