

Xeterm 软件开放接口

接入 规范

(版本 1.0)

成都四象互动科技有限公司
2017 年 5 月

目录

Xeterm 软件开放接口.....	1
一、 功能简介	3
二、 应用场景	3
三、 接入准备	4
3.1 用户服务器端的接入准备.....	4
3.1.1 接口的准备	4
3.1.2 数据的接收	4
3.1.3 服务器代码的处理.....	4
3.2 Xeterm 的参数设置.....	5
四、 参数说明	5
五、 附注	6

一、 功能简介

Xeterm 提供开放接口，该接口允许 Xeterm 与第三方的软件系统对接。对接后，用户在 Xeterm 上的预订、出票、取消编码等操作，Xeterm 会实时捕获并通过接口调用地址通知第三方软件系统，实现航信 PNR 数量及状态与第三方软件系统的统一同步。

二、 应用场景

1、Xeterm 对接订单系统用户在 Xeterm 上的预订、出票、取消编码等操作，Xeterm 会通知订单系统，订单系统接收到 Xeterm 的通知消息后，生成订单或者修改订单，实现 PNR 状态和订单系统状态的同步；

2、Xeterm 对接监控系统用户在 Xeterm 上的预订、出票、取消编码等操作，Xeterm 会通知监控系统，监控系统接收到 Xeterm 的通知消息后，存储或修改编码信息，进行 PNR 的统一管理，规避违规订座风险。

三、 接入准备

3.1 用户服务器端的接入准备

3.1.1 接口的准备

Xeterm 作为主动通知方，通知数据是通过 POST 方式，参数传递信息完成交互。对于 B/S 架构的系统，需要准备一个页面来接收参数；对于 C/S 架构的系统，可以通过 Raw Socket 接收网络数据并解析得到参数。

3.1.2 数据的接收

以 B/S 系统为例，第三方系统在接收到 POST 请求后，获取相应的参数即可。

3.1.3 服务器代码的处理

- 对于预订操作，如果该编码不存在，则执行数据表的记录插入并运用自己的逻辑；如果该编码存在，则执行数据表的记录修改，并运用自己的逻辑；
- 对于出票操作，则执行数据表的记录修改，并运用自己的逻辑；
- 对于取消编码操作，则执行数据表的记录修改或删除，并运用自己的逻辑；
- 对于提取编码操作，则执行数据表的记录修改，并运用自己的逻辑。

3.2 Xeterm 的参数设置

以 B/S 系统为例，第三方系统在接收到 POST 请求后，获取相应的参数即可。Xeterm 软件仅需要设置服务器的接口地址，设置方法如下：

在 Xeterm 上点击菜单设置 - 绑定业务系统，弹出如下页面。设置完毕后，重新运行 Xeterm 软件生效。

Xeterm

www.xeterm.com

快乐的旅途，从这里出发！

用户：[头像]

公司首页Windows桌面版IOS移动版安卓移动版关于我们

我的资料

我的资料

修改资料

修改密码

我的工作组

绑定移动设备

绑定业务系统

登录帐号：

业务系统通知地址：

http://

验证码：

0863

保存

◇ Xeterm经过简单的业务系统绑定，就完成了两个系统的对接；

◇ 对接后，Xeterm可作为业务系统的黑屏操作前端，业务系统的PNR信息，可保持与Xeterm同步；

◇ 回调地址填写范例：www.mysystem.com/XetermCallback.aspx

◇ Xeterm回调参数及规范请参考《Xeterm开放接口接入规范》，[点击下载](#)。

◇ 应用场景：

说明：

1. Xeterm对接订单系统

用户在Xeterm上的预订、出票、取消编码的操作，Xeterm会通知订单系统，

订单系统接收到Xeterm的通知消息后，生成订单或者修改订单，实现PNR状态和订单系统状态的同步；

2. Xeterm对接监控系统

用户在Xeterm上的预订、出票、取消编码的操作，Xeterm会通知监控系统，

监控系统接收到Xeterm的通知消息后，存储或修改编码信息，进行PNR的统一管理，规避违规订座风险。

四、 参数说明

参数名	描述	参数取值	备注
account	Xeterm 登录帐号		
eterm_name	用户使用的PID 别名		
action	执行的动作	booking_pnr: 预订 etdz_pnr: 出票 xe_pnr: 取消编码 rt_pnr: 提取编码	仅有此四个值
time	执行动作的时间	yyyy-MM-dd HH:mm:ss	
pnr	6 位 PNR 编码 -		
data	黑屏截屏内容 编码		解码方法见附

	后的黑屏内容		注
over	表示参数已经接收完毕 1		

五、 附注

//黑屏截图内容的解码方法（以C#.net 为例）：

```
public static string DecodeToString(string str)
{
    string result = ""; int length = str.Length; int step = 0; string temp = "";
    try
    {
        while (step < length)
        {
            if ((step + 2) <= length)
            {
                temp = str.Substring(step, 2);
            }
            else
            {
                temp = str.Substring(step);
            }
            step += 2;
            if (string.CompareOrdinal(temp, "80") > 0)
            {
                if ((step + 2) <= length)
                {
                    temp += str.Substring(step, 2);
                }
                else
                {
                    try
                    {
                        temp += str.Substring(step);
                    }
                    catch
                    {
                        throw new ArgumentException("输入字符串不正确 ");
                    }
                }
                step += 2;
            }
        }
    }
}
```

```
int a = int.Parse(temp, System.Globalization.NumberStyles.AllowHexSpecifier);

if (a < 256)
{
    byte[] byteArray = new byte[1];
    byteArray[0] = (byte)a;
    Encoding gb = System.Text.Encoding.GetEncoding("gb2312");
    result += gb.GetString(byteArray);
}
else
{
    byte[] byteArray = new byte[2];
    byteArray[0] = (byte)(a / 256);
    byteArray[1] = (byte)(a % 256);
    Encoding gb = System.Text.Encoding.GetEncoding("gb2312");
    result += gb.GetString(byteArray);
}
temp = "";
System.Threading.Thread.Sleep(0);
}
}
catch (Exception ex)
{
    result = ex.Message;
}
return result;
}
```