# 一、模板规则说明

html邮件模板和html打印模板的解析都是由一个统一的字符串模板解析类来处理的，因此html邮件模板和html打印模板的模板规则是一致的；其具体的模板规则如下（主要就是3点规则，加一点附加说明）：

（注：我们没有采用第三方的模板引擎，主要是因为不想把模板规则弄的太复杂，增加大家的学习成本，我们的需求主要以显示为主，也不需要太复杂的计算和逻辑规则）

1. **对于输出的统一都使用 <%= XXXX%>的方式**

XXXX则是KeyValueVariables的key或者KeyTableVariables里的某个DataTable中的列名（嵌套在某个循环里的话）；在代码中，只需要传入“XXXX”即可，如keyValues.Add**(**"XXXX"**,** "要替换的真实数据内容"**)；**

如果XXXX没有设置，那么解析模板不会报错，只是<%= XXXX%>就是输出string.Empty了；

模板中用<%=XXXX%>、 <%=    XXXX%>与<%=XXXX        %>效果一样，也就是XXXX前后可以有空格不影响解析，但是%和=之间、%与<之间、%与>之间不能有任何空；

1. **支持循环输出，使用2个固定标记（<%TABLE\_ROWS\_BEGIN ID= YYYY%>和<%TABLE\_ROWS\_END ID = YYYY %>）来包住需要循环输出的html部分**

如下：

<talbe>

    <%TABLE\_ROWS\_BEGIN ID= OrderItems%>

    <tr>

        <td><%=ProudctID %></td>

        <td><%=ProductName %></td>

        <td><%=UnitPrice%></td>

        <td><%=Count%></td>

    </tr>

    <%TABLE\_ROWS\_END ID=OrderItems %>

</talbe>

标记TABLE\_ROWS\_BEGIN / TABLE\_ROWS\_END和ID之间至少要有一个空格，也可以多个空格，不影响解析；TABLE\_ROWS\_BEGIN / TABLE\_ROWS\_END与前面的%之间也可以有0个、1个或多个空格；ID与=之间也可以有0个、1个或多个空格；=与变量名OrderItems之间也可以有0个、1个或多个空格；变量名OrderItems与%之间也可以有0个、1个或多个空格；但是%与<之间、%与>之间不能有任何空；

其中OrderItems为对应KeyTableVariables的key或者KeyTableVariables里的某个DataTable的列名（嵌套在某个循环里的话），其对应的value数据必须为DataTable类型，否则模板解析就会抛出相关异常；

而循环部分的ProudctID、ProductName、UnitPrice、Count都为value的DataTable的列名，将会被替换成DataTable中对应的单元格的值；这里需要注意的是，循环标签所包含的html内容中出现的<%=XXXX%>符号，其XXXX都只会从DataTable的列中寻找了，也就是数据源会切换为该循环标记的ID所对应的DataTable；

如果OrderItems没有设置，那么解析模板不会报错，只是所包含的html部分都不会输出了（或者说全部被替换成string.Empty）;

1. **支持IF判断，使用2个固定标记（<% IF\_BEGIN ID= ZZZZ%>和<% IF\_END ID = ZZZZ %>）来包住需要判断才能决定输出的html部分**

可以根据变量的赋值转换成bool的情况，来控制所包含的内容是否输出，如下：

<%IF\_BEGIN ID=showFoot %>

<div>

    copyright....

</div>

<%IF\_END ID=showFoot %>

IF\_BEGIN和IF\_END的标记中空格的处理，和第2点的标记对空格的处理方式一样；

其中showFoot为变量名，对应KeyValueVariables的key或者KeyTableVariables的DataTable的列名（嵌套在某个循环里的话），其value将被尝试转换为bool值，然后用作判断；如果没有设置变量showFoot，或者在设置的value为null或者无法成功转换为bool的值，那么都将被认为是false，也就是所包含的html部分都不会被输出（或者说全部被替换成string.Empty）；

1. **一点附加说明**

第2点和第3点的标签可以嵌套使用，支持无限层次的嵌套，如：

<%IF\_BEGIN ID=showOrderDetail %>

<talbe>

    <%TABLE\_ROWS\_BEGIN ID= OrderItems%>

    <tr>

        <td><%=ProudctID %></td>

        <td><%=ProductName %></td>

        <td><%=UnitPrice%></td>

        <td><%=Count%></td>

        <td>

            <%IF\_BEGIN ID=HasGift %>

            <div>

                <ul>

                    <%TABLE\_ROWS\_BEGIN ID= GiftList%>

                    <li>

                        <div><%=GiftProudctID %></div>

                        <div><%=GiftProudctName %></div>

                   </li>

                    <%TABLE\_ROWS\_END ID= GiftList%>

                </ul>

            </div>

            <%IF\_END ID=HasGift %>

        </td>

    </tr>

    <%TABLE\_ROWS\_END ID=OrderItems %>

</talbe>

<%IF\_END ID=showOrderDetail %>

这里需要注意的一点是，关于寻找ID的数据源的切换，如果是被IF\_BEGIN/IF\_END所包含的html部分，那么不会切换数据源，如果是被TABLE\_ROWS\_BEGIN/TABLE\_ROWS\_END所包含的html部分，其数据源会切换为该TABLE\_ROWS的ID所对应的DataTable。

# 二、模板对应的标准数据结构

对于模板对应的标准数据结构，主要有2种：

1. **构建Key-Value 替换变量**

KeyValueVariables keyValueList= **new** KeyValueVariables**();**

keyValueList.AddKeyValue**(**"#替换符1#"**,** "替换内容1"**);**

keyValueList.AddKeyValue**(**"#替换符2#"**,** "替换内容2"**);**

**……**

**……**

**……**

1. **构建Key-Table 替换变量**

KeyTableVariables keyTableList= **new** KeyTableListVariables**();**

DataTable tableList = new DataTable();  
tableList.Columns.AddRange(new DataColumn[]

{

new DataColumn("ProductID"),

new DataColumn("ProductName"),

new DataColumn("Price"),

new DataColumn("Quantity"),

new DataColumn("Amount"),

new DataColumn("Discount", typeof(DataTable)),

});

DataTable discoutTable = new DataTable();

discountTable.Columns.AddRange(new DataColumn[]

{

new DataColumn("ID"),

new DataColumn("Discount")

});

discoutTable.Rows.Add(12, 10.00);  
tableList.Rows.Add("Item177292", "XXX电视机", 1270.00, 1, 127.00, discoutTable);

keyTableList.Add("ProductList", tableList);

注意：DataTable的某个列可以嵌套一个DataTable，从而实现嵌套循环；