# TableOutput的说明

## 背景

在主从表模式下，要决定从表的展示方式，原有的方式就在MetaData定义的时候，修改ListStyle属性，通常都是Table方式。如果从表字段比较多，那么可以采用NormalTable的方式进行布局。

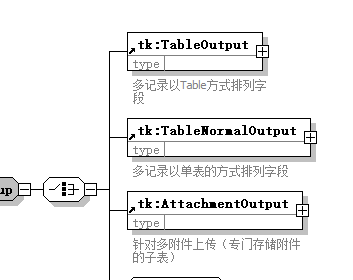
上述两种方式是固定的，如果想用自定义的表布局，在Tk5版本时，就必须做出整体的Razor重载。在Tk55中，RazorData中引入了TableDisplay节点，可以通过razor重载一张表的html输出。不过这个节点的内容，由于没有元数据，无法通用，只能针对特定的例子。而且配置相对麻烦。

在业务处理中，经常有附件表，任务很单一，就是存储多个附件，它的UI无论是Table和NormalTable两种布局，都显得交互不太友好，自定义并且能够重用的表布局就显得比较重要。

## 新的设计

Tk5增强了配置插件的功能，因此使用配置插件来替代原有的枚举是最方便的。

新的Xml定义如下：

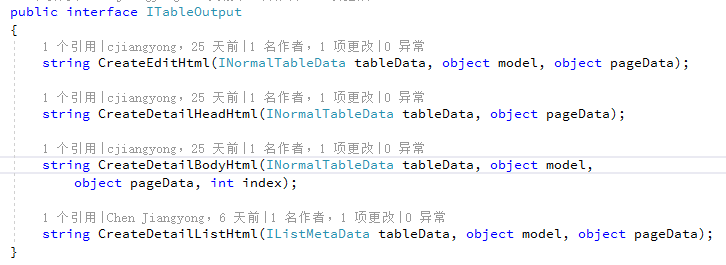


由于采用配置插件的方式，也就意味着，如果业务需要，可以自行扩展表的输出。（考虑到兼容性，如果在配置上没有配置相应的Output节点，那么会自动根据原有的ListStyle属性自动生成对应TableOutput或者TableNormalOutput）

## ITableOutput

如果要实现一个自定义的表布局输出，必须实现ITableOutput接口。

ITableOutput接口定义如下：



CreateEditHtml是负责生成编辑页面的表布局。

而详情页面比较复杂，如下图：



详情通常有两部分构成，一部分是表头（Head），一部分是内容（Body）。而内容通常情况下，只是放了一个URL链接，而具体的内容通过Ajax额外获取。

因此，CreateDetailHeadHtml用来创建表头的html，CreateDetailBodyHtml是创建内容的html，如果是通过Ajax获取额外内容，那么需要完成CreateDetailListHtml。如果CreateDetailBodyHtml把所有的数据都填了，不需要Ajax获取，那么就不需要实现CreateDetailListHtml了。当前系统，由于是来自原有系统，所以都需要Ajax获取。不需要Ajax获取，这个特性可能会在以后的版本中实现。

## RazorTableOutput

由于系统大多数的文本输出，都采用razor的方式。因此，系统提供了RazorTableOutput的实现，只要有提供对应的razor文件，就可以完成对应的函数实现。文件名必须在系统当前使用的模板库目录下。