NoteExpress2 过滤器制作指南

北京爱琴海软件中心 • 2009

目　录

1 前言

过滤器是 NoteExpress 中，处理导入数据的一种机制。一种过滤器，定义了针对某一种格式的数据，如何让 NoteExpress 可以识别它们。换言之，一个过滤器定义了一组数据解析的规则，通过过滤器，NoteExpress 可以导入并处理各种各样的参考文献数据。

NoteExpress 现在的过滤器的工作方式，如图1.1所示。

可以把整个解析的过程，分成如下几步：

1. 定义记录的分隔方式，把源文件的内容分成一块一块的。每一个块就包含一条题录的完整数据。
2. 定义标识的解析模式，继续把每一块的题录数据分成一条一条的。第一条包含一条字段的完整信息。
3. 从“题录类型”的那一条数据中，确定题录的类型。
4. 为每一种题录类型定义一套字段取值的规则。
5. 对每一条字段信息，套用定义好的取值规则，得到相应的字段值。

以一个简单点的例子来说明这个解析的过程。假如我们要导入的数据如下，不包括每行前面的行号：

```
1 type: book
2 title: how to define a filter
3 author: ne
4 year: 2009
5
6 type: article
7 title: the guide to NoteExpress
8 author: AegeanSoftware
9 year: 2009
```

那么，针对这一数据，整个解析的过程可以用图1.2表示。

一步一步地说：

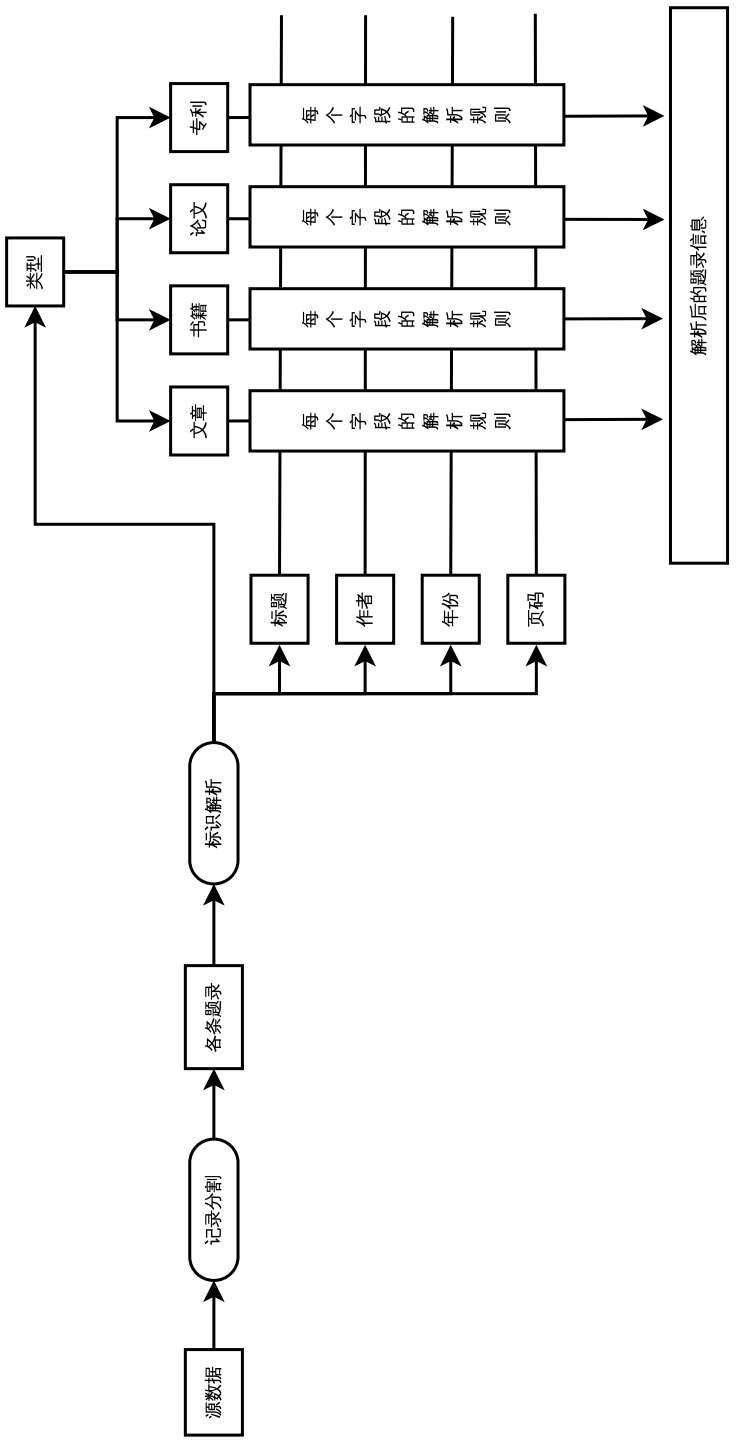

图 *1.1:* 过滤器工作的流程

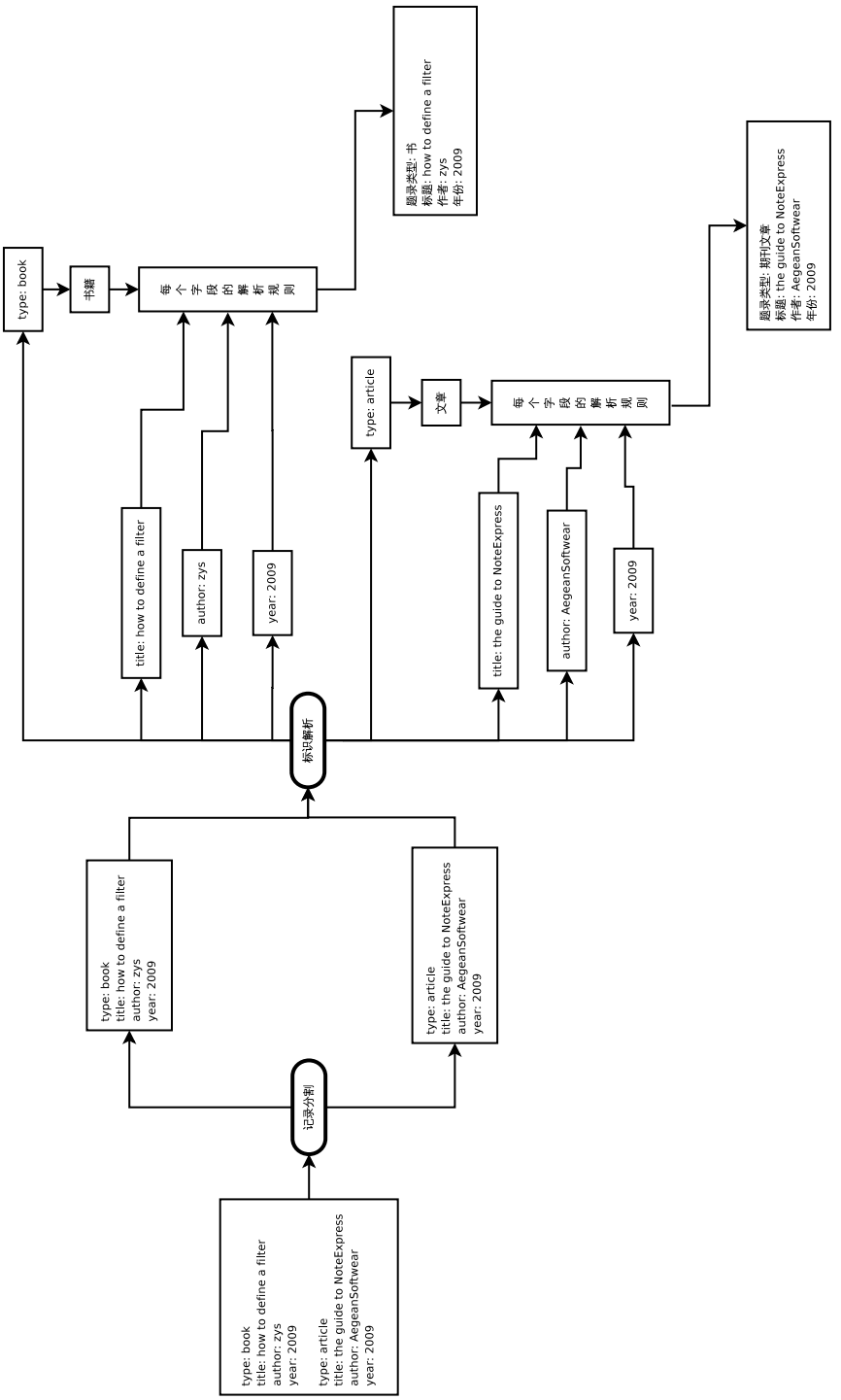

图 *1.2:* 举例数据的解析过程

1. 以空行为界，将源文件分成一块一块的，每一块是一条题录。
2. 以“xxxx: xxxxx”这种模式，把每一块数据再分成一条一条，每一条是一个字段。
3. 从“type: xxxx”的那一条数据中，确定题录的类型。
4. 为每一种题录类型定义一套字段取值的规则。
5. 对每一条字段信息，套用定义好的取值规则，得到相应的字段值。

好了，我想前面的东西你应该都明白了。不过，对于定义过滤器来说，重要是那些解析的“规则”应该如何写，好在，很多数据的结构都是简单明了的，不需要你过多地去折腾，你只要搞清楚过滤器是如何做出来的，然后写上几个字符就可以了。下面我们会有一个完整的，简单的例子。

2 简单完整的例子

这一章，我们会一步一步地建立一个过滤器，用于导入前面我们给出的样例数据，它是这样的（源数据不包括每行的行号）：

```
1 type: book
2 title: how to define a filter
3 author: ne
4 year: 2009
5
6 type: article
7 title: the guide to NoteExpress
8 author: AegeanSoftware
9 year: 2009
```

2.1 创建新的过滤器

先作一些准备工作，在 NoteExpress 中，点击“工具”-“过滤器”-“过滤器管理器”。如图2.1

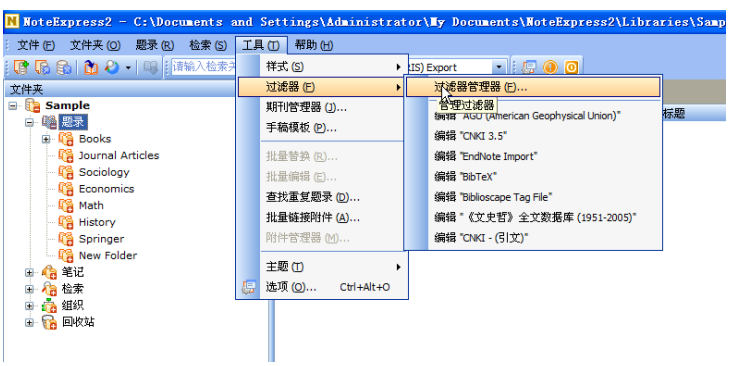

图 *2.1:* 打开“过滤器管理器”

然后会出现一个新的对话框，如图2.2所示。

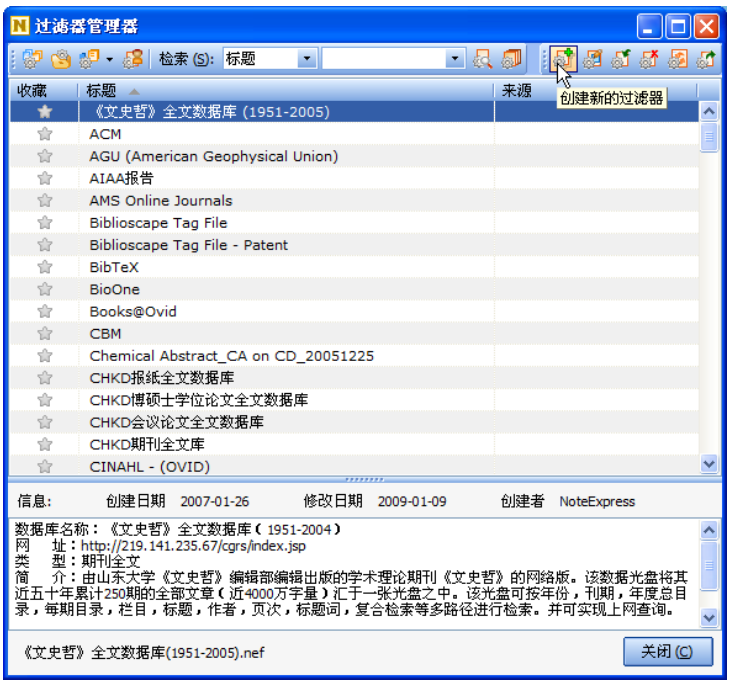

图 *2.2:* 创建新的过滤器

点击“创建新的过滤器”这个图标，进入过滤器的创建环节，如图2.3

先在图2.3的界面，填上一些必要的信息，结果如图

2.2 记录解析

这一步我们要确认，源数据中的每一条题录，是以什么规则来分隔。就是说，我们要制定一些规则来把源数据分成一块一块的，每一块是一条完整的题录信息。

因为我们的源数据格式很简单，显然，我们只需要使用“空行”就可以轻易地分隔每一条题录信息。当然，NoteExpress 的过滤器可以处理比这更复杂的一些情况，会在后面的章节详细介绍“记录解析”。

我们使用“空行”来分隔我们那可爱简单的源数据，如图2.5

2.3 标识解析

把记录分成一块一块的之后，就要继续把每一块数据，分成一条一条的，这每一条就是一个字段的信息。这就是“标识解析”所要做的工作。

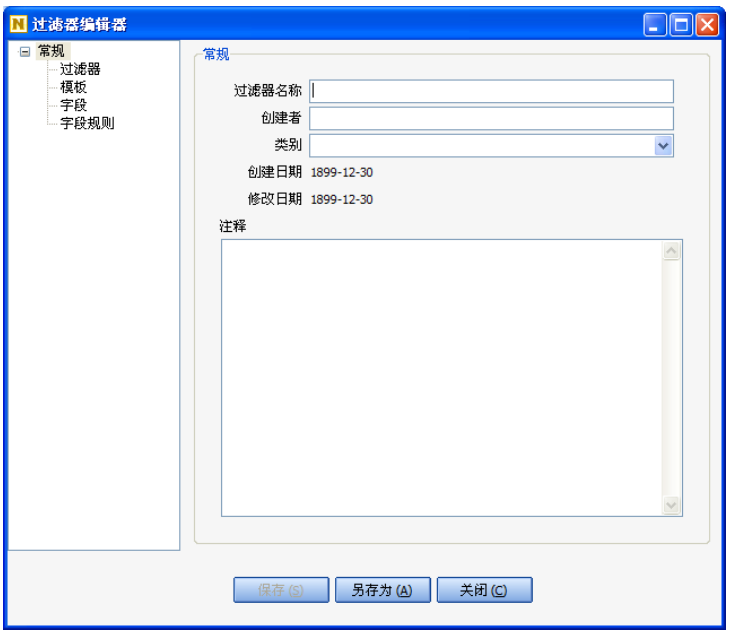

图 2.3: 过滤器编辑器

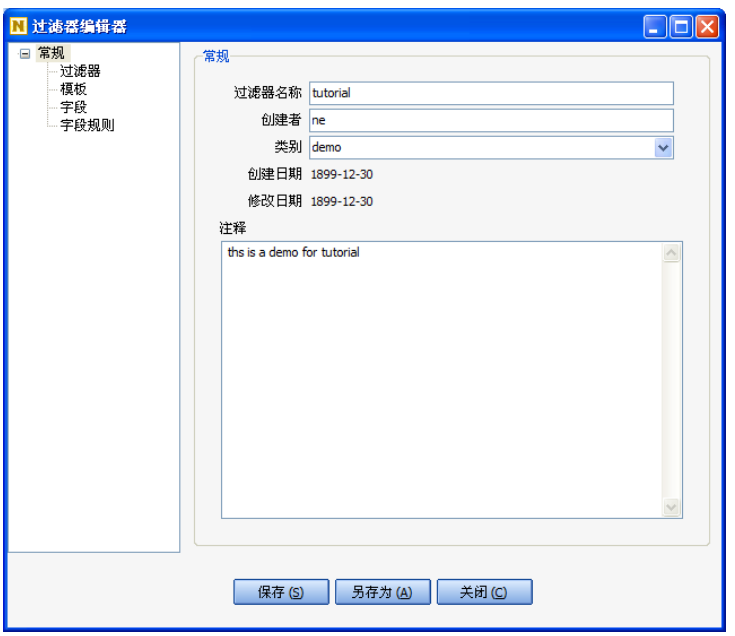

图 2.4: 填上必要的信息

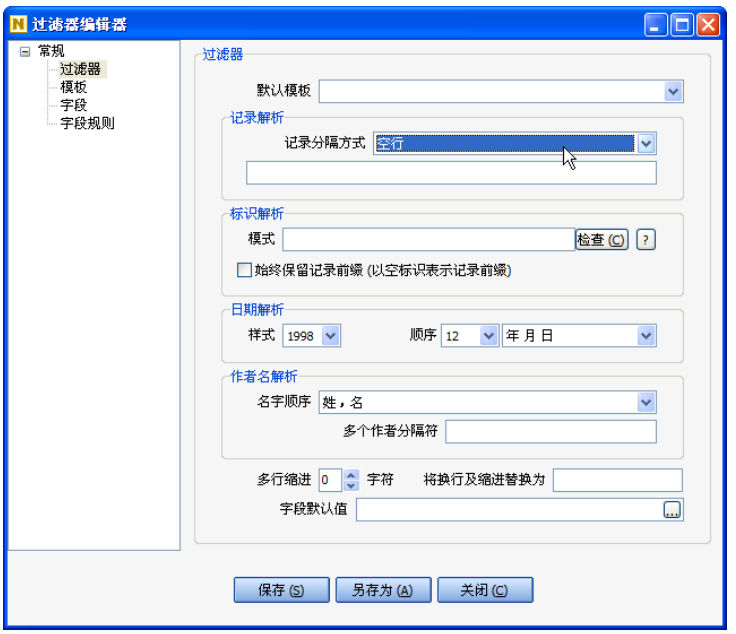

图 2.5: 记录解析

虽然我们的源数据已经是很简单的了，不过，这一步却不像上一步那么容易。我们还是必须要定义一个“模式”来匹配每一条的字段信息。在开始动手之前，有必要了解“模式”与“匹配”的概念。我们只是介绍一些简单的东西，然后来处理简单的问题。

对于一个字符串，或者说对于一串文字，都可以使用一些比较模糊的规则来描述它们。比如：

`abcdefg`	连续 7 个英文字母
`@35bcd`	以“@”开头，然后是 2 个数字，再是 3 个英文字母
`Note: FH*$%@2fd`	以多个字母开头，接一个冒号，然后是多个任意字符

然后，现在的问题是，我们如何用符号来“翻译”这些描述性的语言。NoteExpress 的过滤器中定义了一套这样的符号，你可以点击“?”来看到它们，如图2.6

我们使用这些符号来“翻译”我们之前的描述性的语言。

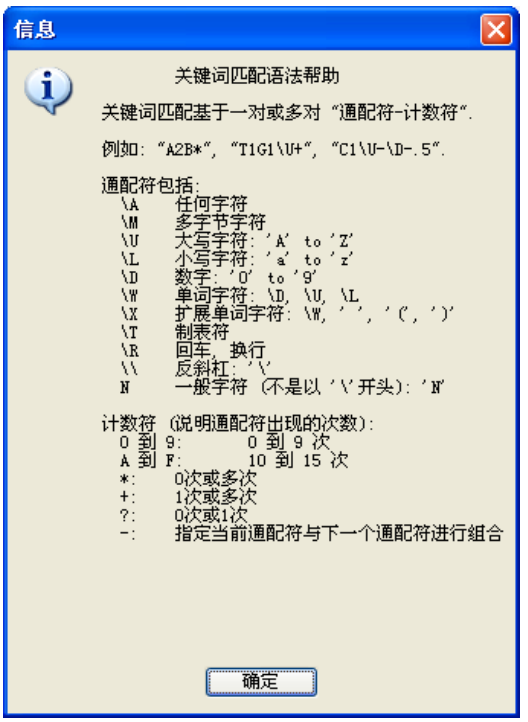

图 *2.6: NoteExpress* 过滤器中的通配符

连续 7 个英文字母	`\U-\L7`
以“@”开头，然后是 2 个数字，再是 3 个英文字母	`@1\D2\U-\L3`
多个字母开头，接一个冒号，然后多个任意字符	`\U-\L*:1\A*`

可以看出，使用符号来表示这些描述，只是使用表示特定集合的符号，再加一个表示数量的数字，按顺序写出原字符串即可，并且，我们必须从行首开始写，如果开头有空格或者制表符的话，也必须表示出来。

表示集合的通配符之间，可以使用“-”进行“合集”的运算。表示数量，可以使用明确的数字，也可以使用表示范围的符号，如“+”，“?”，“*”等。

对于标识解析的模式，会在后面章节作详细的介绍。

现在来看我们需要处理的数据：

```
1 type: book
2 title: how to define a filter
3 author: ne
```

```
4 year: 2009
```

显然，每一个标识，或者说每一个字段，它的形式类似于这样：

```
type: book      一个单词，接一个冒号或者还有一个空格，然后多个字符、汉字
```

结构单一，我们可以很容易写出匹配模式：`\U-\L+:1 ?`

注意，这个模式：`\U-\L+:1 ?`不是完整地匹配了一条字段的信息，只是匹配了前半部分，不过已经足够了。足够区分出每一条字段的信息就可以了。至于如何正确取值那是下一步要考虑的问题。

我们把`\U-\L+:1 ?`填入模式这一栏，如图2.7。

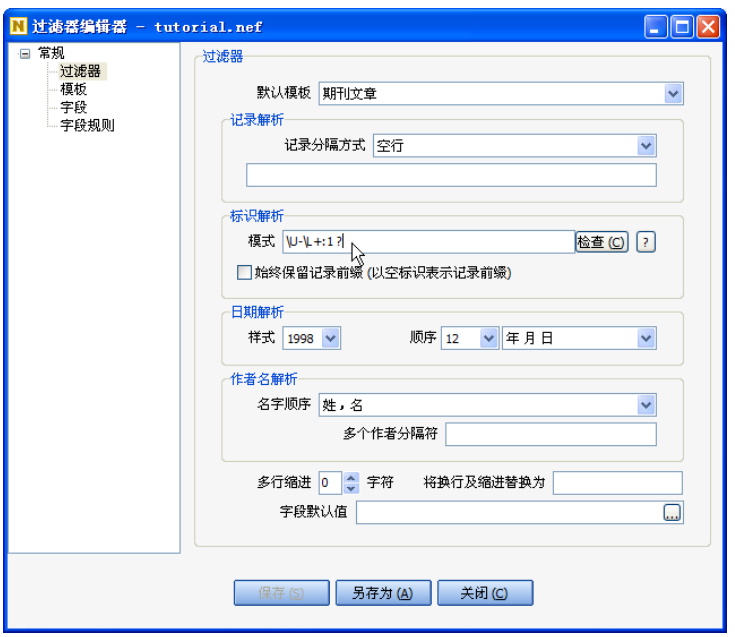

图 2.7: 标识解析的模式

2.4 区分题录类型

这一步，是要让 NoteExpress 可以明确地识别出各条题录的类型，这是为下一步：为各种题录建立一套解析规则，做作的必要准备。

在“模板”这一栏下，先新建一条“期刊”的模板，如图2.8。

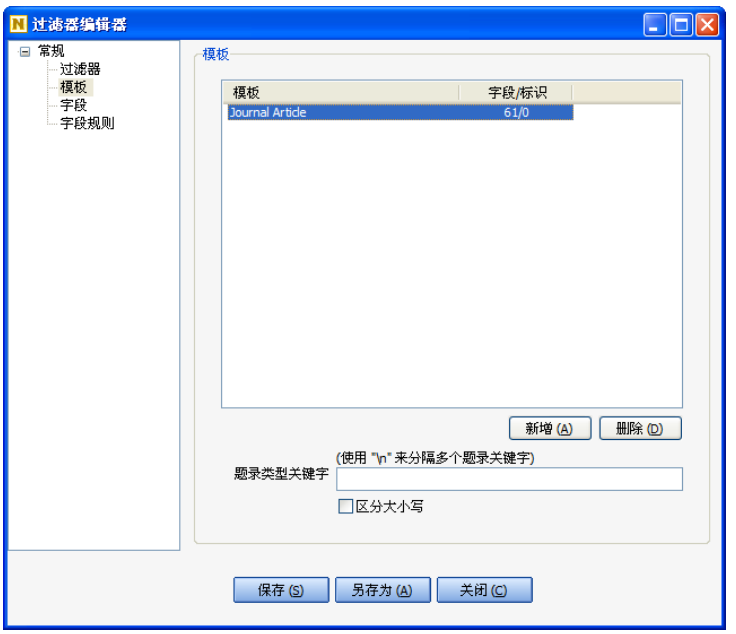

图 *2.8: 新建模板*

这里，先记住下面的“题录关键字”，它很重要，但是，现在我们还没有条件填上它。

在“字段”那一栏，“期刊文章”这一模板下，双击“题录类型”这一字段，然后添加一个规则，在“标识”这一栏添加上：`type:`。如图2.9。

解释这个`type:`，可以看成是，对所有按`\U-\L+:1 ?`这个模式，匹配出来的字段信息，这个信息实际上是两个`\U-\L+:1 ?`模式之间的所有内容，再去匹配`type:`这个标识，显然`type:`是符合`\U-\L+:1 ?`这个模式的。然后用整条字段的信息，如“`type: article`”，去掉“标识”信息，即去除“`type:`”后，剩下的字符串，就是“题录类型”这一字段，解析后的取值。即“`type: article`” – “`type:`” = “`article`”。

每一种题录类型，都是对应一个“题录类型”这一字段解析后的取值，如果取值为“article”，就是“期刊文章”。如果取值为“book”，就是“书”。这个对应关系，就是在前面提到过的“题录类型关键字”中标明。如图2.10。

图 *2.9:* 题录类型解析规则

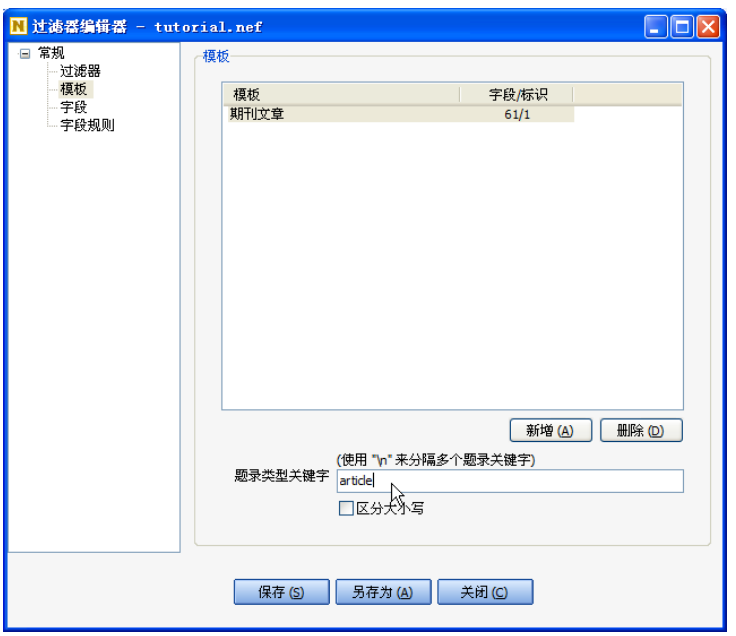

图 *2.10:* 题录类型关键字

2.5 添加其它字段的解析规则

同“题录类型”这一字段一样，我们再添加其它字段的解析规则。在“字段”这一栏，“期刊文章”这一模板下，双击“作者”，为“作者”字段添加一个规则，在“标识”栏填入`author:`，与“题录类型”一样。如图2.11。

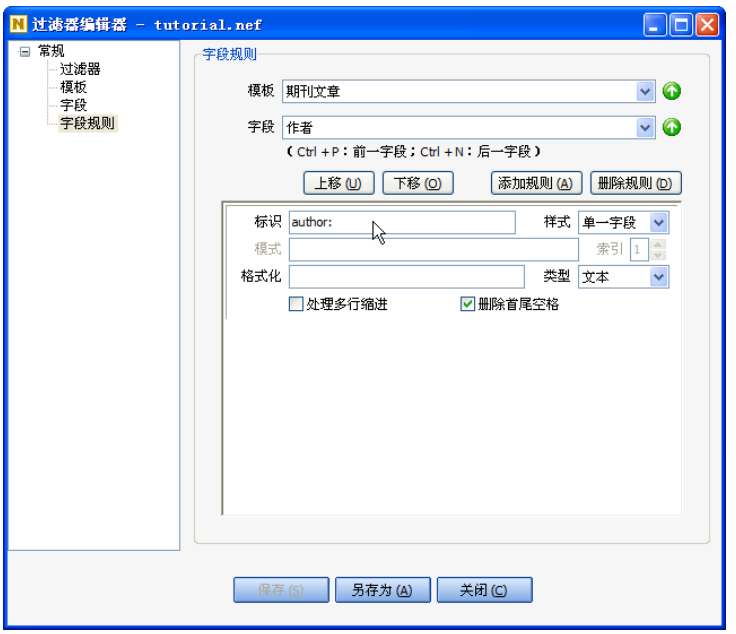

图 *2.11:* “作者”字段的解析

2.6 测试一下导入情况

好了，现在你可以点击“保存”，然后输入一个你喜欢的名字，如`tutorial`，保存当前正在编辑的过滤器。

把下面的框中的数据保存到一个文件中，如：`tutorial.txt`。

```
type: book
title: how to define a filter
author: ne
year: 2009

type: article
title: the guide to NoteExpress
```

```
author: AegeanSoftware
year: 2009
```

你可以直接复制后把这些数据保存到“记事本”中，如图2.12。

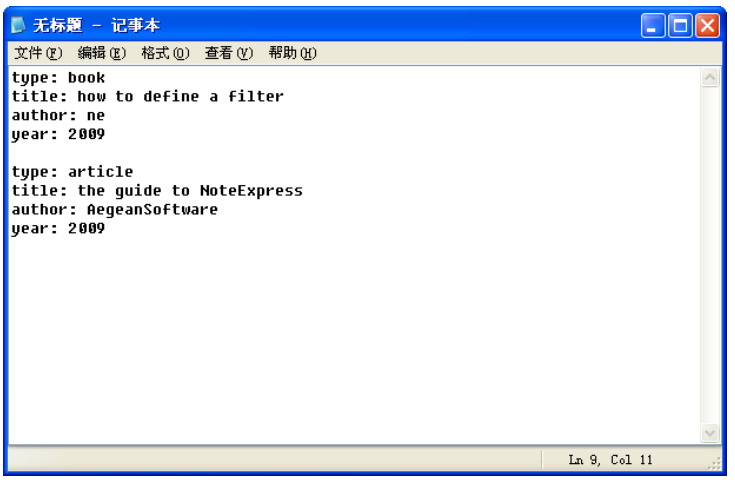

图 *2.12: 保存数据到 tutorial.txt*

现在在 NoteExpress 中点击“文件”-“导入题录”，文件选择`tutorial.txt`，过滤器选`tutorial`。这里在选择过滤器，注意打开“用户定义的过滤器”的页面再选择，如图2.13。

点击“开始导入”，不出意外的话，会提示“成功导入 2 条记录”！

看起来不错，不过我们的过滤器还没有制作完成，继续下面的内容吧。

2.7 添加其它字段的解析规则

到现在，我们只有“作者”和“题录类型”这二个字段的解析规则，但，一条完善的题录它包含的字段信息可是非常多的。我们需要一个一人地把这些字段的解析添加进去。方法就和“作者”这一字段是一样的了。

“年份”这一字段，添加一个规则，标识填：`year:`。

“标题”这一字段，添加一个规则，标识填：`title:`。

然后保存过滤器的设置，再试着导入一下`tutorial.txt`，是不是“年份”，“作者”，“标题”都有了？

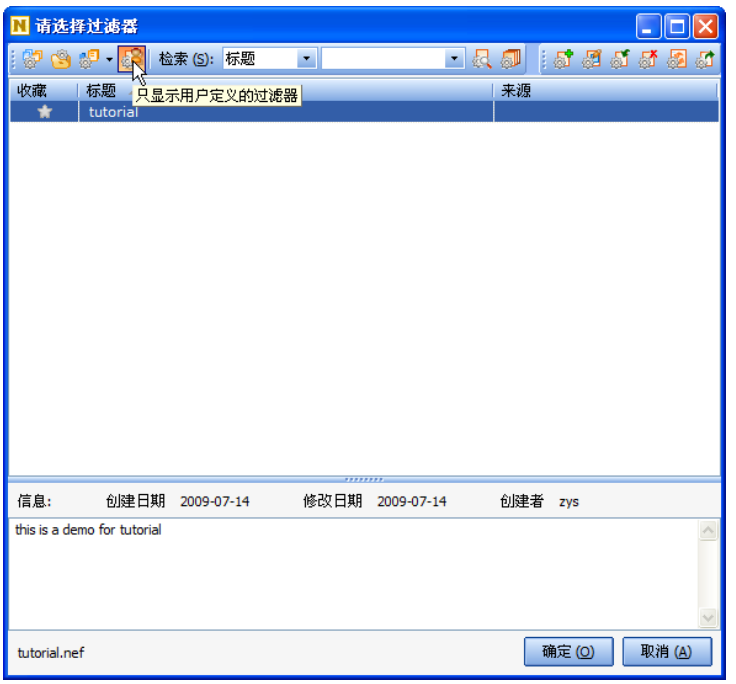

图 *2.13:* 用户定义的过滤器

2.8 添加其它类型的模板

至今为止，我们所做的工作都是为“期刊文章”这一模板做的，但我们需要导入题录类型可是有多种的。所以，还需要其它题录类型的模板。不同的模板，很有可能对各个字段的解析规则是相同的，所以，我们可以复制“期刊文章”的模板。如图2.14。我们基于做好的“期刊文章”的模板，来新建“书”的模板。

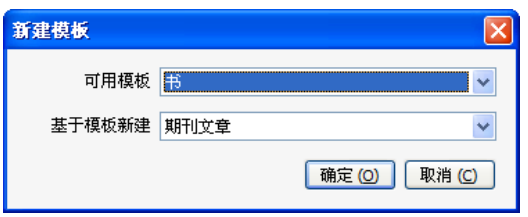

图 *2.14:* 基于模板新建

如果“书”的各个字段的解析规则没有什么变化，我们就不需要去作更改了，只需要修改一下“题录类型关键字”即可。如图2.15。

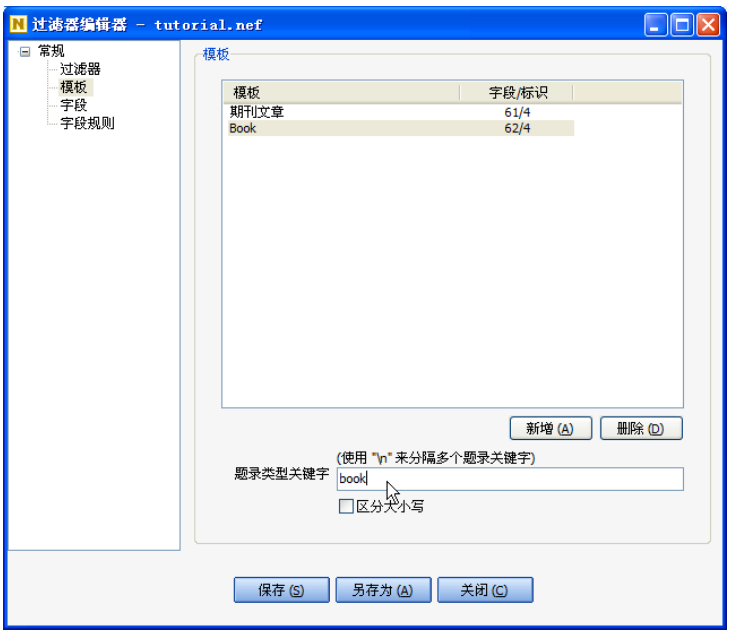

图 *2.15:* 修改“题录类型关键字”

2.9 完成

现在保存过滤器，导入`tutorial.txt`，已经可以完整地解析出所有字段内容了，如图2.16。

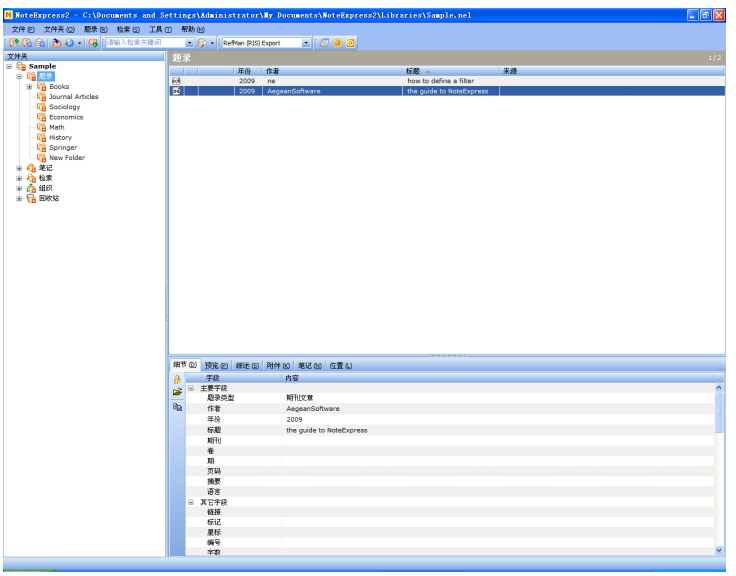
NoteExpress2 - C:\Documents and Settings\Administrator\My Documents\NoteExpress2\Libraries\Sample.nel
文件 (F) 文件夹 (O) 题录 (R) 检索 (S) 工具 (T) 帮助 (H)
RefMan (RIS) Export
文件夹
Sample
题录
Books
Journal Articles
Sociology
Economics
Math
History
Springer
New Folder
笔记
检索
组织
回收站
题录
年份 作者 标题 来源
2009 ne how to define a filter
2009 AegeanSoftware the guide to NoteExpress
细节 (D) 预览 (P) 引注 (S) 附件 (A) 笔记 (N) 位置 (L)
字段 内容
主要字段
题录类型 期刊文章
作者 AegeanSoftware
年份 2009
标题 the guide to NoteExpress
期刊
卷
期
页码
摘要
语言
其它字段
链接
标记
星标
编号
字数

图 *2.16:* 过滤器完成

3 记录解析

记录解析，就是规定从源数据中，以什么规则来分隔每一条题录。NoteExpress 提供了“起始标识”、“结束标识”、“空行”、“分隔字符串”、“分隔模式串”这五种分隔方式。下面我们会一一介绍。

3.1 起始标识与结束标识

起始标识与结束标识，这两种分隔方式与在下方“标识解析”中定义的“模式”有关。用于分隔的标志，其结构与其它字段相同的情况。

考虑以下例子：

```
record: 1
type: book
title: how to define a filter
author: ne
year: 2009
record: 2
type: article
title: the guide to NoteExpress
author: AegeanSoftware
year: 2009
```

显然，每一条题录，可以用`record:`来分隔开，这与每一条字段的解析模式`\U-\L+:1 ?`是一致的。这种情况，就适合使用“起始标识”或“结束标识”的方式进行记录分隔。如图3.1。

例子稍作修改：

```
type: book
title: how to define a filter
author: ne
year: 2009
```

图 *3.1:* 起始标识分隔

```
 5 record: 1
 6 type: article
 7 title: the guide to NoteExpress
 8 author: AegeanSoftware
 9 year: 2009
10 record: 2
```

这种情况适用“结束标识”分隔。如图3.2。

3.2 空行

这是最简单的一种分隔方式了。前面的 tutorial 就是使用的这种方式。

3.3 分隔字符串

这个很好理解，它是一种与“模式”无关的分隔方式。定义一个分隔用的字符串，只要源数据中一行的开始与这个字符串相同，即认为此处是一个分隔处。

还是考虑这个例子：

图 3.2: 结束标识分隔

```
1  type: book
2  title: how to define a filter
3  author: ne
4  year: 2009
5  record: 1
6  type: article
7  title: the guide to NoteExpress
8  author: AegeanSoftware
9  year: 2009
10 record: 2
```

这里，我们可以定义分隔用的字符串为“rec”，对，三个字母就可以了。因为每到rec: xx 这一行时，前三个字母都会匹配到这个分隔用字符串，所以每个rec: xx 都会被认为是一个分隔处。如图3.3。

图 3.3: 分隔字符串分隔

3.4 分隔模式串

“分隔模式串”可以看成是对“分隔字符串”的一种扩展，适合用于以特定的字符串分隔，但是这个字符串是变化的，其模式又与其它标识的模式不同的情况。

考虑这样一个例子：

```
1  type: book
2  title: how to define a filter
3  author: ne
4  year: 2009
5  s1
6  type: article
7  title: the guide to NoteExpress
8  author: AegeanSoftware
9  year: 2009
10 s2
```

在这个例子中，可以看出，每条题录的信息，可以用“s1”或“s2”这样的字符

串来分隔，但是，字符串又不是固定的一个，同时，它的结构也和其它字段的模式不同。这时，就要用到“分隔模式串”的方式来分隔各条题录信息了。

对于“s1”或“s2”这样的字符串，很容易以 NoteExpress 给的一些符号写出它的模式：`s1\D+`，即一个“s”后跟了多个数字。这样的话，每一个位于行首的，符合这个模式的字符串，都会被认为是一个分隔处。

注意：“模式”总是从行首开始起作用。

如图3.4所示，即可正确分隔给的例子数据。

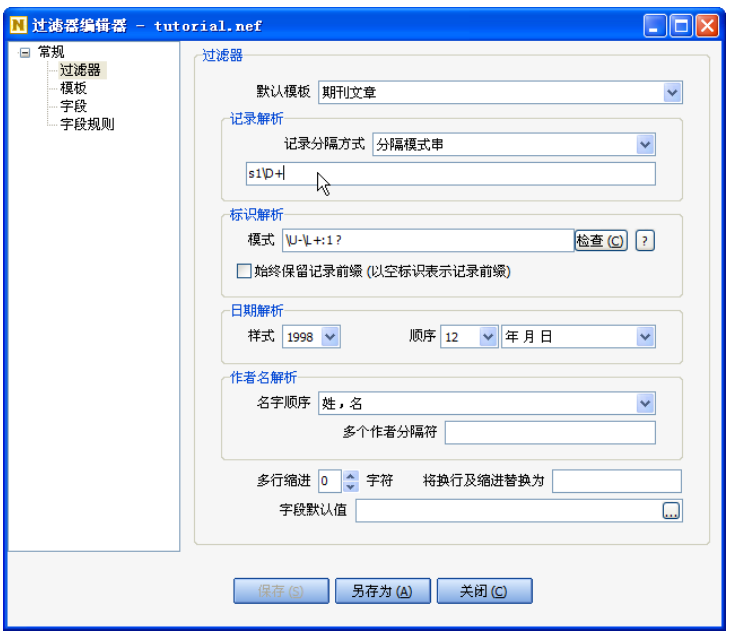

图 3.4: 分隔模式串分隔

4 标识解析

标识解析，即是提取源文件中，每一个题录中的每一个字段中的内容。因为，对于一个参考文献的格式化文档来讲，它存储各字段的信息的格式一般是统一的，这种统一的格式我们可以用一个“模式”表达式来表示它，凡是匹配这个“模式”的内容，就可以识别出来，然后作下一步的处理。

4.1 模式

如果你曾经接触过“正则表达式”的话，会对“模式”这东西很熟悉了。NoteExpress 自己有一套通配符系统，其语法也和普通的正则表达式有所区别。其基本的语法格式是：字符 计数。

下面列出了 NoteExpress 定义的通配符：

A	任何字符，包括换行
M	多字节字符，如汉字
U	大写字母
L	小写字母
D	数字
W	大写字母，小写字母，数字
X	大写字母，小写字母，数字，空格符，“(”，“)”
T	制表符
R	回车，换行
\\	“\”
-	并集操作

下面列出了 NoteExpress 定义的计数符：

0 - F	*0* 到 *15* 次，十六进制表示法
*	*0* 次或多次
+	*1* 次或多次
?	*0* 次或 *1* 次

再次提醒，“模式”总是从行首开始匹配的。

下面举出一些“模式”的例子：

%X	%1X1
␣␣␣␣@article{	␣*@1a1r1t1c1l1e1{1
item0:	i1t1e1m1\D+:1 1
--type--	-2\L+-2
(author)	(1\W+)1

注意，@article{ 那一行的前面，是带有空格符的。

4.2 记录前缀

记录前缀指的是，用于记录分隔的字符串，与第一条标识之间的所有内容。如以下源数据：

```
1 !!this is prefix
2 type: book
3 title: how to define a filter
```

如果我们以“!!”作为“分隔字符串”（见第3章），以 \U-\L+:1 1 作为标识的模式，那么，从“!!”到“type: ”之间的内容：“this is prefix”就是所谓的记录前缀。

我们在“字段规则”，把某一字段的“标识”这行留空，那么这个字段的值就是记录前缀。如图4.1。

同时，我们也可以在“字段”这一栏看到，“标识”这一列显示的是“记录前缀”。如图4.2。

前面说过，记录前缀指的是，分隔字符与第一个标识之间的字符串，但是，如果一个标识都没有呢？考虑以下的例子：

```
1 [1] (t)the guide to NoteExpress. (ty)article. (y)2009. (a)A.
2 [2] (t)how to define a filter. (ty)book. (y)2009. (a)B.
```

显然，我们可以使用“分隔模式串”（见第3章），模式为[1\D+]1 1 ，来分隔各条题录信息。

图 4.1: “标识”留空就是处理记录前缀

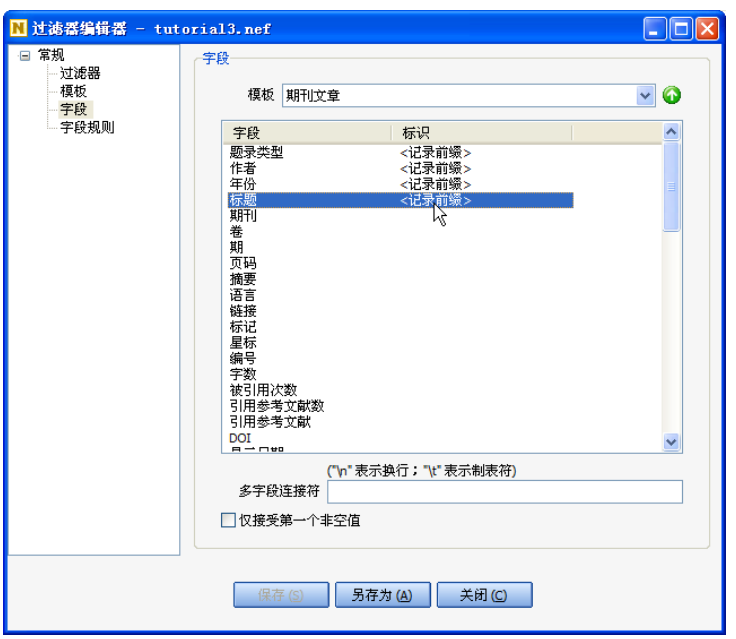

图 4.2: “标识”处显示“记录前缀”

然后，问题来了。数据中，整条题录的各个字段信息都在一行中，没有分成一条一条的。我们无法以一个“模式”来定义各个字段的信息。这种情况，我们就可以勾选“始终保留记录前缀”，如图4.3。这个选项会在没有一个标识的情况下，把两个分隔字符串之间的内容，作为记录前缀处理。

图 4.3: 始终保留记录前缀

现在对于一条题录，只有一行数据，我们要从中解析出各个字段的信息。使用一个匹配整行字符串的正则表达式，分组后对于每个字段再分别利用“索引”取值就可以了，如图4.4。

图 4.4: 正则表达式匹配记录前缀

5　特殊字段类型解析

此章内容与第7章有关，你或许应该看完了第7章，再回来看这里。

在 NoteExpress 中，有一些特殊的字段的值，是可以被特殊处理，有“日期”、“作者名”、“多行值”。对于“日期”这种类型的值而言，NoteExpress 需要能正确识别“年”、“月”、“日”。对于“作者名”这一字段的来说，NoteExpress 需要知道是否是多个作者，每个名字，哪部分是“名”，哪部分是“姓”。“多行值”的话，NoteExpress 可以被设置成是否需要把多行的内容合并成一行。

5.1　日期解析

日期解析涉及两部分的操作，一部分是在“过滤器”中设置日期解析的规则，如图5.1所示。第二部分是在“字段规则”中，要指定某一字段的值的类型是“日期”，如图5.2。

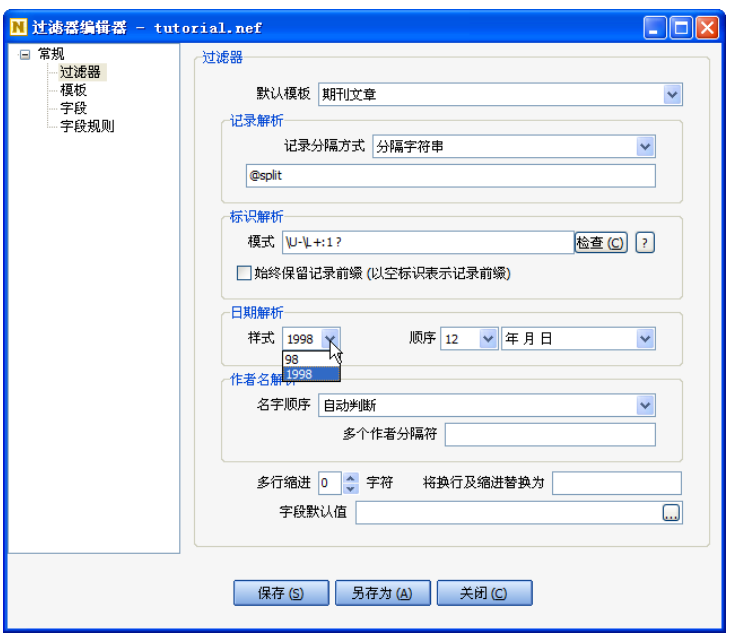

图 *5.1:* 设置日期解析

图 5.2: 设置字段值的类型为“日期”

考虑如下数据:

```
1 type: article
2 title: the guide to NoteExpress
3 author: AegeanSoftware
4 date: 05-07-2005
5 year: 2009
```

源数据中有一个“date”的字段，它的值是“05-27-2005”，很明显地是以“月”、“日”、“年”的顺序排列的。默认地，在 NoteExpress 中，对于像“日期”这种值的类型是日期的字段，都会按“xxxx-xx-xx”这种模式去识别它。自然地，对于像“05-27-2005”这种“xx-xx-xxxx”的模式，默认情况下 NoteExpress 是不能正确识别的，这就需要我们作一些设置。

第一步，在“顺序”这一栏，把样式设置成“月日年”，如图5.3。

第二步，在“字段规则”中，设置相应字段的值的类型为“日期”，如图5.2。

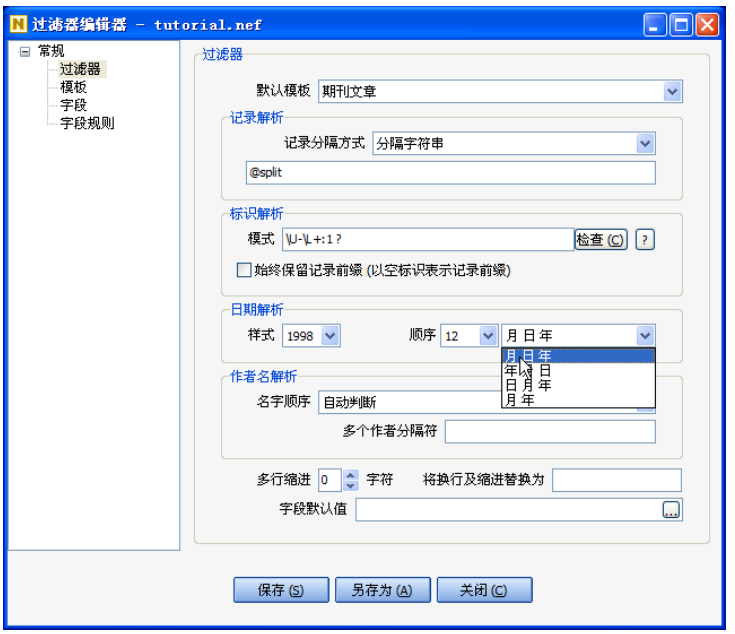

图 5.3: 设置日期值的顺序

5.2 作者名解析

作者名解析，有二个功能。一是识别多个作者，NoteExpress 自己存储数据时，是以分行来区别多个作者的。二是对于“姓”与“名”的识别，因为某些样式的输出要求明确哪部分是“姓”，哪部分是“名”。

实现作者名解析的步骤与日期解析是一样的，即先在“过滤器”中设置好规则，然后在“字段规则”中设置某个字段的值的类型是“姓名”。

考虑如下数据：

```
1 type: article
2 title: the guide to NoteExpress
3 author: A and B
4 date: 05-07-2005
5 year: 2009
```

在“author”这个字段中，我们有两个作者名，“A”和“B”。默认情况下，如果把“作者”这个字段的值以“文本”来对待，那么得到的结果就是“A and B”，并

没有把两个作者分开，这样的话，在一些输出样式中就可能碰到麻烦了。正确地，应该是让 NoteExpress 识别出，“作者”是“A”和“B”这两个人。

第一步，在“过滤器”中设置“多个作者分隔符”为“and”，如图5.4。

图 5.4: 设置多个作者分隔符

第二，在“字段规则”中，把相应字段的值类型设置成“姓名”，如图5.5。

这样的话，数据导入 NoteExpress 后，作者就可以被正确地识别为是两个，如图5.6。

下面看下“姓”与“名”的问题，一般，区别“姓”与“名”的情况针对英文作者名比较多。考虑如下数据：

```
1 type: article
2 title: the guide to NoteExpress
3 author: givename family-name
4 date: 05-07-2005
5 year: 2009
```

NoteExpress 中，存储姓名的格式是“姓, 名”。上面的数据，如果在“名字顺

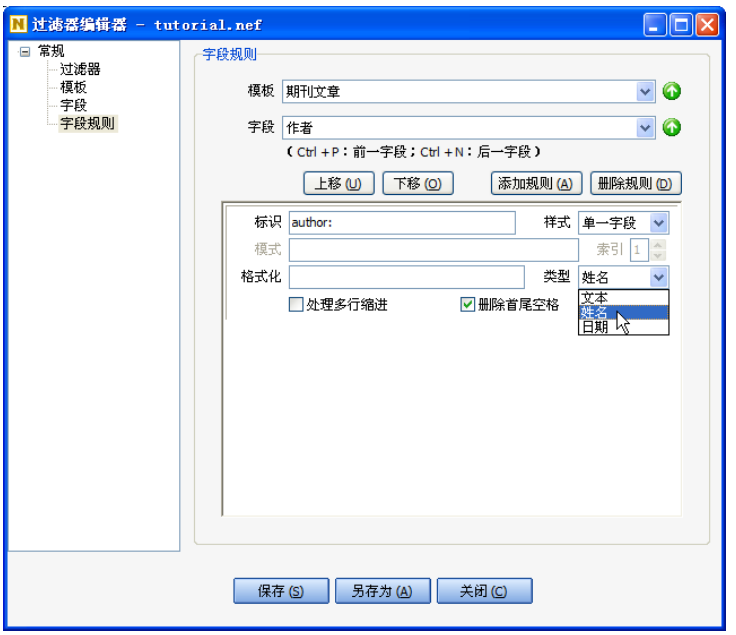

图 5.5: 设置字段的值类型为“姓名”

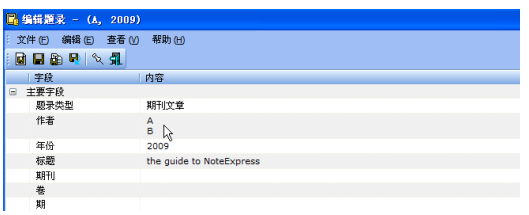

图 5.6: 导入后是两个作者

序”中使用“自动判断”的话，最后“作者”这个字段得到的是值是“givename family-name”，这与事实是相背的。我们应该定义“名字顺序”为“名姓”，如图5.7。

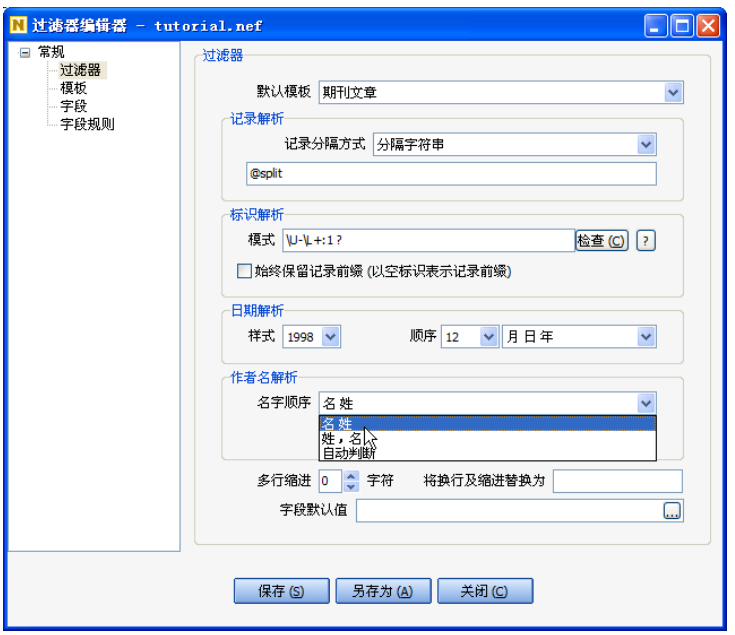

图 5.7: 名字顺序

这样最后“作者”这个字段得到的结果就是“family-name, givename”。

5.3 多行值解析

多行值的处理，一般出现在其值是比较长的字符串的字段中，比如“摘要”，它的结果可能有多行。

考虑如下数据：

```
1 type: article
2 title: the guide to NoteExpress
3 author: givename family-name
4 date: 05-07-2005
5 abstract: first line
6   second line
7   third line
```

```
8 year: 2009
```

其中的“abstract”字段的值，就是一个多行的值。默认情况下，不做额外的处理的话，导入 NoteExpress 后，仍然是一个多行的字符串。但是 NoteExpress 有将多行的字符串合并为单行的机制，不管你是否认为这是个好主意，但 NoteExpress 可以做的。

首先，还是在“过滤器”中作对合并多行时应该如何处理的设置，如图5.8。

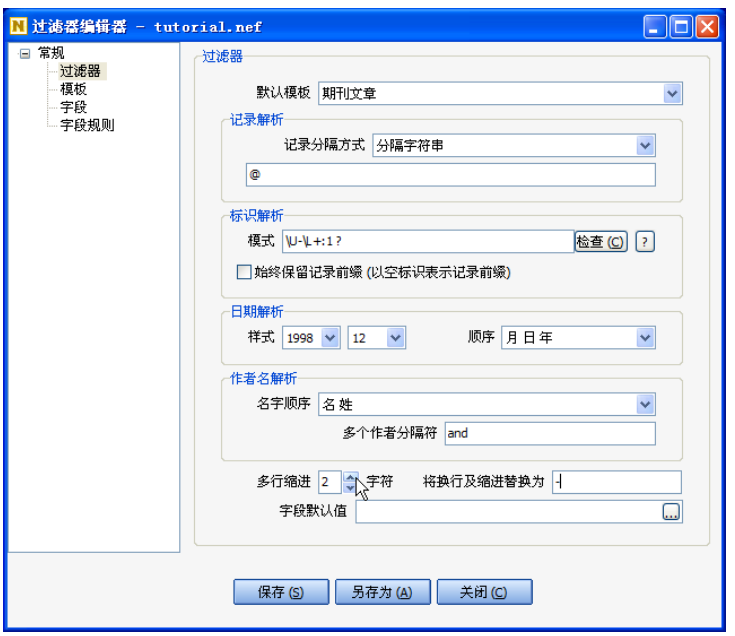

图 5.8: 多行合并的设置

在这里，有两个可以设置的选项。

第一个是“多行缩进 X 字符”，实际上应该理解成“删除行首 X 个字符”。如例子中，我们“abstract”字段，第二行和第三行的数据前都有两个空格，那么这里我们就可以设置成多行缩进 2 个字符，这样，合并成一行后，在中间就不会有多余的空格出现了。同理，如果设置成了缩进 3 个空格，那么位于最前面的字符，第二行的“s”和第三行的“t”就会被删掉了。

第二个是“将换行及缩进替换为”，这个好理解，就是多行之间以什么符号连接起来，留空的话，就直接合并。图中，我们设置了一个“-”来连接多行的字符串。

下面要做的，就是在“字段规则”中指明某个字段的值要进行多行处理，如图5.9。

图 5.9: 指明要进行多行处理

经过这些操作，我们最后得到的“摘要”这个字段的值，就是“first line-second line-third line”。

6 字段的默认值

源数据中，可能存在数据不完整的情况。在这种情况下，如果相关字段没有信息，则导入 NoteExpress 后该字段会留空。你可以为某些字段设置一个默认值，它可以在源数据没有该字段信息时，自动使 NoteExpress 以默认值填充。也可以始终以默认值填充，无视源数据的内容。

操作方法如下。

首先，点击最下端的，“字段默认值”右侧的图标，如图6.1。

图 *6.1:* 设置字段默认值

然后，添加一些字段后，再在下方填入希望的默认值即可，如图6.2。

如你希望直接用某个值填充某个字段，而不管源数据里的信息是怎么样的，可以勾选下方的可选项，如图6.3。

设置完成后，过滤器编辑窗口的下端，“字段默认值”处会显示出来，你设置好的有默认值的字段及默认值，如图6.4。

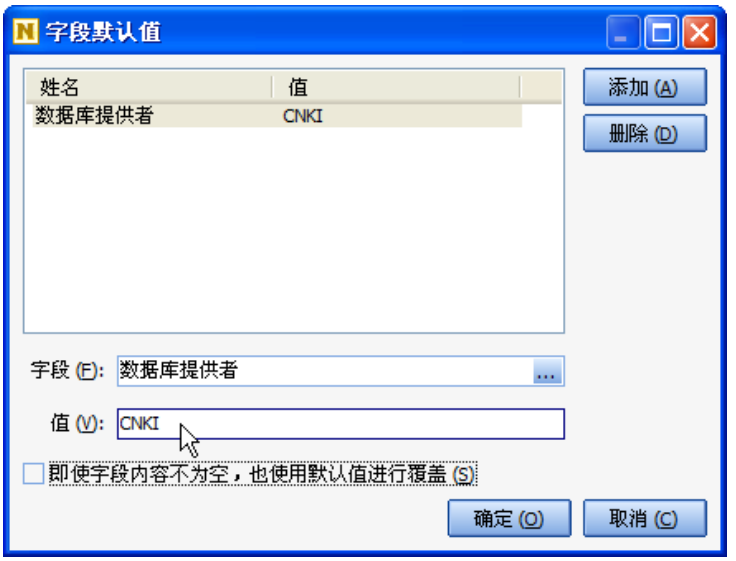

图 6.2: 填入字段的默认值

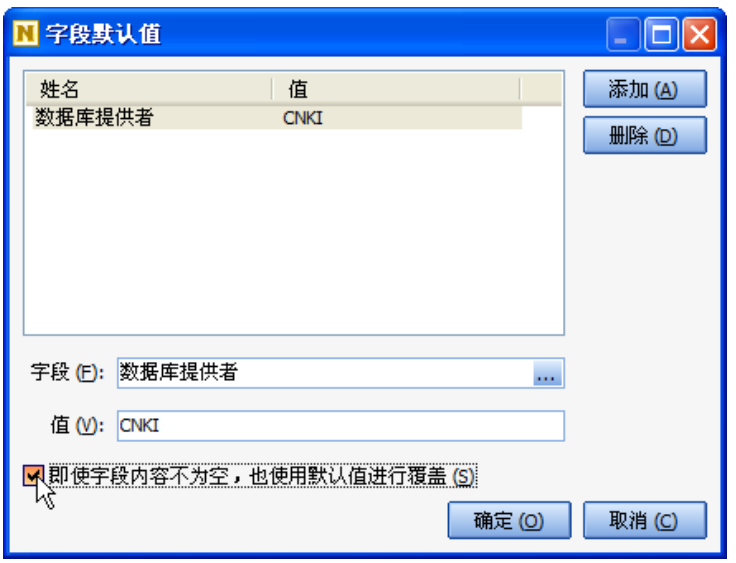

图 6.3: 始终以默认值填充

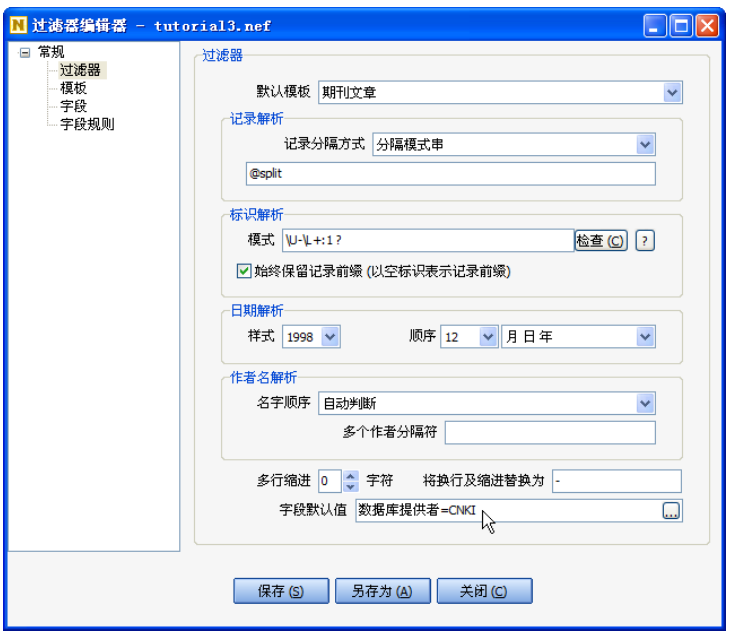
过滤器编辑器 - tutorial3.nef
常规
过滤器
模板
字段
字段规则
过滤器
默认模板
期刊文章
记录解析
记录分隔方式
分隔模式串
@split
标识解析
模式
\U-\L+:1?
检查(C)
?
始终保留记录前缀(以空标识表示记录前缀)
日期解析
样式
1998
顺序
12
月 日 年
作者名解析
名字顺序
自动判断
多个作者分隔符
多行缩进
0
字符
将换行及缩进替换为
-
字段默认值
数据库提供者=CNKI
保存(S)
另存为(A)
关闭(C)

图 6.4: 字段默认值显示

7 字段规则

前面的介绍的一个完整的例子中，我们只是在一种最简单的情况下，直接取得了各字段的值。即，前面说过的，先解析出一条字段的完整的信息，这里指的完整信息，是指两个标识解析（见第4章）结果之间的所有内容。然后对于这条完整信息再去匹配我们给的标识（见图2.9），匹配成功后，用完整的信息，去掉标识，剩下的字符串就是取值，这个取值返回给我们可以进一步用“正则表达式”处理。

考虑如下的例子：

```
1 type: article
2 title: {the guide to NoteExpress}
3 author: AegeanSoftware
4 year: 2009
```

按以前我们做的工作，对于“title”这个字段，取出的值是“{the guide to NoteExpress}”。显然，那两个碍眼的大括号是我们不需要的。

下面要做的事，要求你必须掌握了“正则表达式”。

要把那一对多余的大括号去除，使用“正则表达式”是十分容易做到的事，第一步，写一个正则表达式去匹配这个取得的值，并对匹配结果分组。第二步，引用分组，在这里你也可以添加额外的字符。

如图7.1所示：

图7.1中，把“样式”设置成“正则表达式”，然后在“模式”中写上正则表达式，这个例子，我们应该写成`{(.*)}`，并且用小括号分组。再在“替换”那一栏，填上最后需要的分组，使用`$1`，`$2`之类的标记，就可以正确提取相应的值了。

右边有一个索引，是可以直接引用某个分组的值，“索引”为2，即相当于“替换”处写的`$2`。

再看一个复杂点的例子：

```
1 type: article
```

图 *7.1:* 使用正则表达式解析提取的字段值

```
2 title: {[main]the guide to NoteExpress [sub]first version}
3 author: AegeanSoftware
4 year: 2009
```

假设对于“title”这个字段，我们想得到的是形如“the guide to NoteExpress (first version)”这样的结果。

那些，这里的正则表达式就应该写成`{\[main\](.+)\[sub\](.+)}`，然后在“替换”处写`$1($2)`，如图7.2。

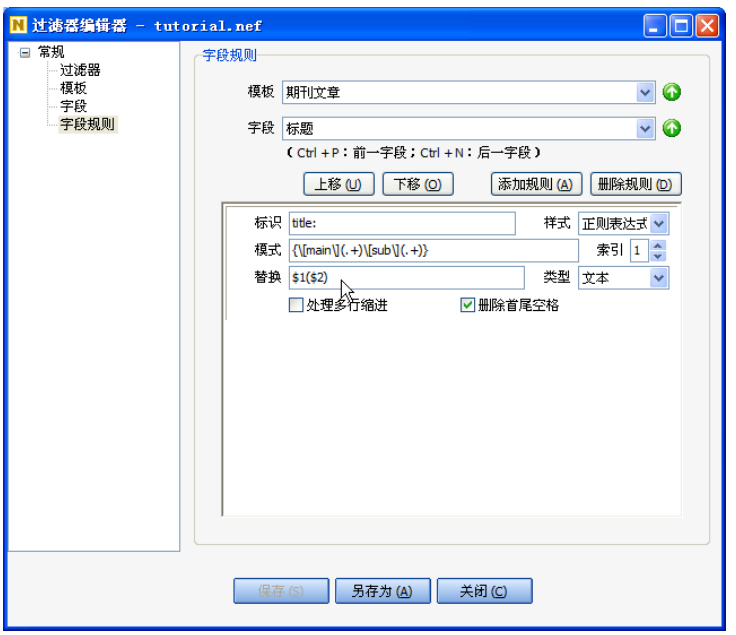

图 7.2: 正则表达式的分组引用