Adding New Channels to Novobot

Table of Contents

Channel Definition Files

Novobot uses XML to define rules for extraction of headlines from the source text (XML or HTML). These XML rule files are called **channel** definition files. They have extension `.nov` or `.novx` and are displayed in Windows Explorer like this:

Types of Supported Channels

Novobot currently supports two types of channels:

- XML-based – they can be of different formats, the most often used ones being Moreover syndication format (see http://www.moreover.com) and RSS format, or RDF (Resource Definition Format, MyNetscape format);
- HTML-based – web pages of any layout that can be parsed ("scraped") in order to extract news headlines and relevant links.

While XML formats usually give guaranteed results because of strictly defined tags and their meanings, HTML-based channels often produce mixed results. This is because there is indefinite variety of web page designs on the internet, and it is very difficult to extract the necessary information from every possible page.

Advice	For best results, especially if you have no HTML knowledge, use XML-based channel sources (RDF, RSS, Moreover, UserLand) to create your channels – it is much easier than make Novobot scrape plain HTML web pages. Look for RSS RSS, XML XML or other syndication links on web sites for feed URLs.

Creating a New Channel

Prerequisites

You will have a better chance of succeeding if you are familiar with XML and/or HTML file formats. It also helps if you have programming experience. Do not be discouraged if you don't have these skills though, because these formats, at least to the extent necessary for Novobot, are quite easy to learn.

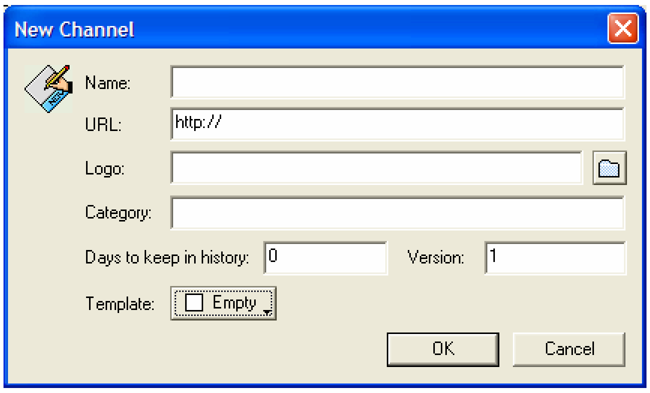
NovEdit

Use NovEdit tool to create and edit channels. To run NovEdit, click on the Start menu, select Novobot program group, and click on NovEdit icon. A dialog box will appear which is the main window of NovEdit application.

You can also select an existing channel file from Novobot *parsedef* directory (by default it is `C:\Program Files\Novobot\parsedef`), click right mouse button on it to display its context menu, and then select Edit from the menu. NovEdit will start and load the selected channel definition file for editing.

Creating a Channel Stub

In NovEdit, click New... or press Alt+N. The new channel setup dialog will appear (Figure 1).


New Channel

Name:

URL: http://

Logo:

Category:

Days to keep in history: 0 Version: 1

Template: Empty

Figure 1: New Channel Setup Dialog

Selecting channel name

In the Name field, enter human-readable name for your new channel. It is a good idea to name the channel after the web site from which you are going to get the headlines.

Entering channel URL

In the URL field, enter the link to your channel data. This link must produce the actual source text for the headlines. If the site you are adding as a channel provides syndicated news feed, enter link to that feed here.

Since the rules will be stored in XML format, you need to observe some precautions when entering URLs. For example, if your URL contains specific symbols, as long URLs usually do, you need to replace them with the appropriate text to keep your resulting XML file valid. Refer to the following table for some replacement rules.

Symbol	Replace with
&	`&amp;`
>	`&gt;`
<	`&lt;`

Choosing channel logo

This step is optional, but your channel will look more attractive in Novobot window if provided with a relevant logo. The best choice is to get a button or other picture from the site itself. To do this, go to the web site and look for an image on it that looks like a small standalone rectangle with some relevant graphics. Usually this is what webmasters provide for other webmasters to link to their sites. Look under headings like “Link to us” or similar. Right-click on the image in your browser and save it to a local file as Windows Bitmap. Note the location of the saved image.

Note	Be sure to save the image in Windows Bitmap format, or else Novobot will not be able to process it.

In NovEdit, click the button on the right of the Logo field. In the file dialog that appears, select the saved bitmap image file and click Open. The image you saved will be imported into the channel definition file. You can delete the image file once your channel has been saved at least once.

Selecting channel category

The channel category serves as subdirectory name under *parsedef* root. Therefore it can contain only characters allowed in filenames, and cannot contain any of the following characters: `/\:*?><|`. Try to categorize your channel as precisely as possible.

Separate your category names with backslash. When Novobot installs your channel automatically under the *parsedef* root, it will place the channel definition file in a subdirectory that matches the channel category. Novobot will create the subdirectory if it does not exist.

For example, if the site you are creating the channel for is about online music downloads, you can set its channel category to `Entertainment\Music`.

Setting number of days in history

The Novobot's history feature prevents duplicate headlines from being displayed. Novobot stores each retrieved headline in its history cache for the number of days specified separately for each channel. This way headlines from channels that are updated less often are stored in the history for a longer time, preventing old headlines from being displayed again.

To set number of days headlines from your new channel are going to be kept in the history cache, you should have an idea on how often the channel (or the web site) is updated. By "updated" I mean not just several new headlines added, but ALL headlines on the site changed so that no old headlines are left. For heavy use sites such as Slashdot (http://www.slashdot.org for those who lived in the jungle for the last several years) the turnaround period is probably a couple of days maximum. For other not-so-often updated sites this may reach several days or weeks.

If you set the number of days too big for your channel, the size of the cache file (which is `HISTORY.NOVH` file under *parsedef* root directory) can grow considerably. This is not a big problem, because once the file grows to its optimum size, it will stay that size due to old headlines being removed constantly as they expire. If you set this number too low, users of your channel may see headlines they already saw, which is not desirable.

Advice	It is better to set number of history days for a channel to be more than less, because the purpose of the history cache is to prevent old headlines from being displayed, and with the day number too low this may not work.

To set number of history days for headlines in your channel, enter the desired number into the Days to keep in history field. You can always change this number later if you discover it is not optimal.

Setting channel version

Channel version is a number from 1 to 65534 that tells Novobot if the channel being installed is newer than the one that already exists. When you release you modified channel definition file, you should increment its version number, so that it is installed correctly.

Enter version number in the Version field. Start with version 1. There is no need to increment the version every time you edit the channel while developing it–you only need to do this once the channel is ready for distribution.

Choosing channel template

To simplify channel development, NovEdit provides several predefined templates for the most popular channel formats. Using Template control, choose one of the following:

- Netscape RDF – use this template for a MyNetscape-style channel, or if your headline provider says the feed is in RSS format.
- Moreover XML – use this template for a XML feed provided by Moreover (http://www.moreover.com)

- Empty – for all other types of feeds, and for HTML web page scraping, choose this option, which really does not provide any template and leaves you on your own.

Creating and saving initial channel definition

Once you have set all the options in the New Channel dialog, click OK button, and NovEdit will create an initial channel definition XML code for you. In the main NovEdit window, click on Save button and save the channel file, noting its location for later access.

Now that you have the initial channel definition in XML format, let's see how to make it work.

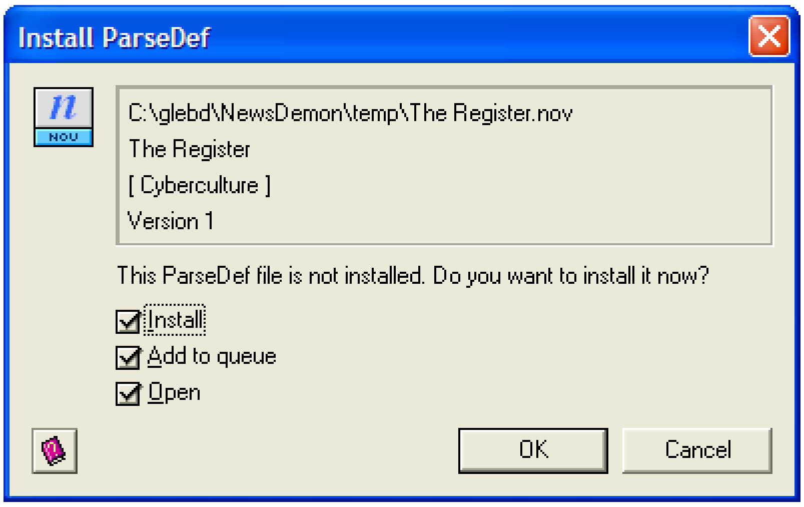
Testing the Channel

After you saved your channel, you may want to test it, either to see that it does not work yet, or just to have opportunity to look at the actual document's source code.

Running Novobot from NovEdit

At any time you can run your channel definition with Novobot by clicking on the Run button. NovEdit will run Novobot and tell it to open the channel currently being edited. This is very useful for debugging your channel.

When Novobot loads the channel that is not installed yet, as in the case of test run from NovEdit, it displays the dialog to ask you what to do with that channel (Figure 2).



Install ParseDef

C:\glebd\NewsDemon\temp\The Register.nov
The Register
[Cyberculture]
Version 1

This ParseDef file is not installed. Do you want to install it now?

- [✓] Install
- [✓] Add to queue
- [✓] Open

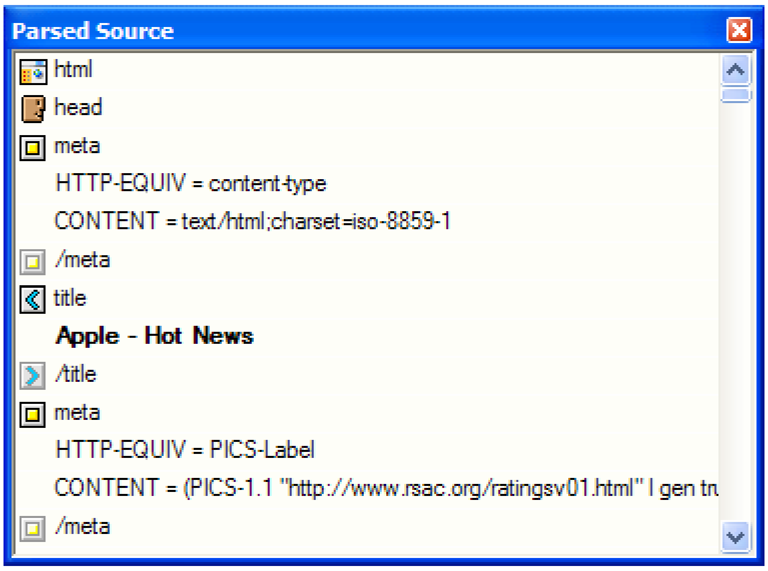
Figure 2: Novobot Install Channel Dialog

Deselect the two first choices (Install and Add to queue) while you are developing your channel. This was Novobot will just open it, but won't copy the incomplete channel to its *parsedef* directory, nor will it add the channel to the queue, and that's what you need at the moment.

If after clicking Run in NovEdit you get Novobot's message that the channel file (or *ParseDef*) cannot be loaded, most probably you have an error in your XML channel definition. Check compliance with XML syntax rules and try again.

Looking at the Channel Source Document

When you see your channel does not work as expected (and THIS is unfortunately very much expected situation, so don't worry), it often helps to study the actual source code of the channel document received over the internet. To look at what Novobot received using your channel instructions, activate Novobot and select View | Source from the menu. You will see something like the following picture:



```
html
head
meta
    HTTP-EQUIV = content-type
    CONTENT = text/html;charset=iso-8859-1
/meta
title
    Apple - Hot News
/title
meta
    HTTP-EQUIV = PICS-Label
    CONTENT = (PICS-1.1 "http://www.rsac.org/ratingsv01.html" l gen tr
/meta
```

Figure 3: Novobot Parsed Source Window

The parsed source document view is useful for pinpointing any problems with your channel definition. The explanations for every concrete type of channel are below.

Editing a Channel From Within Novobot

Starting with Novobot version 2.1, it is possible to open the corresponding channel file directly from Novobot. This may be useful to debug your own channels, or to try fixing a channel that stopped working because the format of the feed had changed.

To open a channel file, select a headline or an error message produced by that channel and choose Headline | Open channel file from the menu. Novobot will open the corresponding channel file for the headline in NovEdit.

After you edit the opened channel file in NovEdit and save it, you can switch back to Novobot and try running the channel again to see whether you changes have been successful or not. Repeat editing as necessary until the channel produces correct headlines in Novobot.

Creating a RSS channel

RSS is the most prominent syndication format currently in use. Many news sites offer RSS feeds of their headlines for free. I suggest you start developing new Novobot channels from a RSS source document.

The Template

After you finish with the New Channel dialog, the template text in NovEdit looks like the following:

```
<?xml version="1.0"?>
<PARSEDEF
    name="My Channel"
    url="http://www.server.com/feed.rss"
    logo="logo.bmp"
    category="General\News"
    history="7"
```

```
08      version="1">
09      <SECTION>
10              <START><item/></START>
11              <HEADLINE><title/></HEADLINE>
12              <SUBHEAD><description/></SUBHEAD>
13              <LINK><link/></LINK>
14      </SECTION>
15 </PARSEDEF>
```

Listing 1: The initial RSS (RDF) channel definition

Let's look at the channel code line by line. This will help you understand what parts of the channel definition perform which functions when processed by Novobot.

XML Explained

A XML document starts with the header, which must be of the predefined format (line 01). If this line is not present, many XML parsers[1] will not be able to process the document.

XML documents consist of tags. The general format of a tag is:

```
<TAG attribute=”value”>text</TAG>
```

For our purposes you have to remember several things about tags:

- Every tag has to be closed. If a tag does not contain any text, you can use `<TAG/>` syntax to open and close it in one statement.
- Every attribute value must be enclosed within double or single quotes.
- Tag and value names are not case-sensitive.
- Special symbols have to be replaces by the appropriate symbol names (ampersand by `&amp;` greater symbol by `&gt;` less symbol by `&lt;` be sure not to omit semicolons in these names).
- There must be exactly one top-level tag in every XML document (in our case this is the `PARSEDEF` tag).
- You can insert comments or temporarily comment out your tags to disable them. Comments are everything between the following symbols: **`<!--`** *`comment`* **`-->`**
- If Novobot tells you the channel definition file cannot be loaded, most probably there is an error in your XML document.

Parts of a RSS Channel Definition

Let's look at the tags that comprise a RSS channel definition, one by one.

PARSEDEF

The `PARSEDEF` tag (lines 02 to 15) is the main tag in a channel definition file. It has several attributes with obvious meaning (just remember what you have entered into the New Channel dialog box), so we won't stop here.

SECTION

Every `PARSEDEF` tag contains one or more `SECTION` tags. A section defines how a distinct part of the channel source is processed. For RSS channel source there is only

[1] A XML parser is a piece of computer code that processes (parses) XML documents and converts them into a format that is better suited to the needs of a particular program. Novobot contains a XML parser.

one section (lines 09 to 14 in our case), but for complex HTML source web pages there can be more than one.

START

For each section, the processing of headlines must start somewhere. This place in the XML source is identified by the `START` tag (line 10). Novobot will start looking for headlines once it reaches anything that looks like the contents of the `START` tag. In our RSS case, the processing starts after Novobot reaches the first `ITEM` tag in the channel source document. Since every RSS newsfeed document contains at least one `ITEM` tag, you don't need to change anything here.

HEADLINE

Novobot will treat as headline anything within every part of the source document that looks like the contents of the `HEADLINE` tag (line 11). In our case, every `TITLE` tag in the source document produces one headline, with the text of the tag being the actual headline text.

For example, the following fragment of a source document:

```
<TITLE>London bridge is falling down.</TITLE>
```

produces the following headline:

```
London bridge is falling down.
```

SUBHEAD

A headline looks better in Novobot if it has some explanatory text below. This way users often don't even need to go to the source web site to read more, since they already get an idea what the headline is about from its sub-headline, or subhead. This tag tells Novobot how to find subheads in the source document, if it has any (line 12).

Not all RSS documents offer extended information for the headlines. This is where your action may be needed. In case channel definition specifies a subhead, but there aren't any subheads in the source document, Novobot will not be able to complete headlines from that source, and no headlines will be generated.

If you discover that your RSS channel definition does not work, check carefully if the source document contains `DESCRIPTION` tags. If it does not, you must remove the `SUBHEAD` tag from the channel definition. This will tell Novobot there are no subheads in the source document, and it is not necessary to look for them in order to complete headlines.

LINK

To enable Novobot to follow headlines to their originating web sites, you must specify how links are represented in the source document. In a RSS document a link for each headline is represented with `LINK` tag, as follows:

```
<link>http://www.somenews.com/story01.html</link>
```

To have Novobot extract the actual URL from this kind of XML code, you use the following statement (line 13):

```
<LINK><link/></LINK>
```

If, for example, a link in your source code is represented with a tag other than `LINK`, say, `URL`, you could modify your channel definition as follows:

```
<LINK><url/></LINK>
```

Final Testing of your RSS channel

Now that your channel is completed, you should test it with Novobot by clicking on the Run button in NovEdit. If you see headlines from the site in the Novobot main window, then the channel is ready to be installed and added to the Novobot queue.

Creating a XML channel

This is much the same as with RSS channel, after all, RSS is a kind of XML, but with generalized XML format we get unpredictable diversity of tags and sub-formats. Still, to make a channel definition for a XML-based channel is much, much easier than for a HTML-based one.

The Template

After you finish with the New Channel dialog, the template text in NovEdit looks like the following:

```
01 <?xml version="1.0"?>
02 <PARSEDEF
03     name="My Channel"
04     url="http://www.server.com/feed.xml"
05     logo="logo.bmp"
06     category="General\News"
07     history="7"
08     version="1">
09     <SECTION>
10            <START><moreovernews/></START>
11            <HEADLINE><headline_text/></HEADLINE>
12            <SUBHEAD><source/></SUBHEAD>
13            <LINK><url/></LINK>
14     </SECTION>
15 </PARSEDEF>
```

Listing 2: The initial XML (Moreover) channel definition

The next step is to run the channel in Novobot and see if any headlines appear. In case of a Moreover channel it should work right from the start. With other kind of XML-based channels you most probably will have to modify your channel definition.

Writing the Channel Definition

When Novobot finishes downloading the source document, open Parsed Source window by selecting View | Source from Novobot menu. Take a look at the parsed source code.

START

When looking at the parsed source code, look for a tag that appears just before all the headlines, or, alternatively, the first tag of the first headline. Use that tag in your `START` statement (line 10).

HEADLINE

Look for the tag that makes the actual headline text. In case of a Moreover channel, this tag is `HEADLINE_TEXT` (line 11). Modify the definition in your channel accordingly.

SUBHEAD

If every headline in the source code of your channel has some explanatory text marked by a tag that is consistent from headline to headline, you can use it in your `SUBHEAD` statement (line 12). You can also try running your channel without subheads and see if it works, then add the statement for processing subheads.

Sub-heading does not have to be an explanatory text. It can be any additional information that is present for every headline. For example, in Moreover channels Novobot uses the source of a headline as its sub-heading.

LINK

Look for the tag containing all the links for the headlines in the source document. Use that tag in your `LINK` statement (line 13).

Final Testing of Your XML channel

After you have edited all the tags in the definition accordingly to the parsed source document, try running the channel and see if it produces any headlines. If not, you are set for trial and error method. Be sure your XML code is compliant, and Novobot can load it. Try commenting out `SUBHEAD` tag and see if that helps. Try using other tags from the source document. With some creativity and patience, you will have your XML channel running in Novobot before long.

Creating a HTML channel

Getting Novobot to scrape raw HTML code is possible in many cases, but it can be difficult and sometimes frustrating. I have to admit (and you probably already know it) that Novobot is far from being perfect, and it does not contain some features that could help the scraping process. But even at this stage of its evolution Novobot is capable of processing fairly complex HTML pages and extracting headlines from them for your enjoyment. So if you are not a faint of the heart, read on.

The Template

After you finish with the New Channel dialog, the template text in NovEdit looks like the following:

```
01 <?xml version="1.0"?>
02 <PARSEDEF
03     name="My Channel"
04     url="http://www.server.com/"
05     logo="logo.bmp"
06     category="General\News"
07     history="7"
08     version="1">
09     <SECTION>
10            <START></START>
11            <HEADLINE></HEADLINE>
12            <SUBHEAD></SUBHEAD>
13            <LINK></LINK>
14            <SKIPTO></SKIPTO>
15            <END></END>
16     </SECTION>
17 </PARSEDEF>
```

Listing 3: The initial HTML channel definition

Not much of a help really. The most important part, though, is already there, and it‘s the URL of the site. This means that you can run the channel with Novobot and see the parsed code.

Advice	Keep your browser open with the page you are about to process visible to be able to compare parsed source, the actual page, and your channel.

Setting the Tags

You probably already noticed that there are several new tags in the definition. We will come to them shortly, but let us first start with the sections in HTML.

SECTION

As I said before, a channel definition for a HTML-based channel can have several sections. This is because in a HTML file (or on a web page) there can be several parts that you would like to extract headlines from. For example, one part of a page may contain information about new software releases, and another part of the same page may contain some general news headlines. Using separate section definitions in your channel, you can gather all the headlines from that page in one go.

When looking for the place where headlines start, look for the first headline on the page displayed in your browser, then look for this text in the parsed source window. After you find that place, look for any tags it is surrounded with, or for any attributes associated with the headline text, or any distinctive tag or attribute preceding the first headline. If you don‘t find one, you can use the construct used for the headlines, because Novobot will start looking for headlines as soon as it encounters the contents of the `START` tag.

Example: Let‘s use The Register (http://www.theregister.co.uk) to find out how to program our channel. The Register‘s headlines look as follows:

```
<DIV CLASS="indexheadlink"><A
HREF="/content/4/24450.html"><STRONG>Blender surrenders to slender
sales</STRONG></A></DIV> <DIV CLASS="indexintro">Numbers don't stack up
for Not a Number</DIV> <DIV CLASS="indexposted">16 March 2002
2:15pm</DIV> <BR></DIV>
```

Listing 4: The Register headline source code

Since tag `<DIV CLASS=”indexheadlink”>` does not appear in the page source code until the actual headlines start, you can use it in your `START` tag.

HEADLINE

Look at the source code in Listing 4. Every headline on The Reg’s page starts with `<DIV CLASS=”indexheadlink”>` tag. The actual headline text is inside this `DIV` tag. Because of that we can use the `DIV` tag as out `HEADLINE` tag contents in the channel definition. Novobot will pick up any text inside the tag and make it a headline.

The general rule is you should find something–a tag, an attribute, or a combination of those–that every headline is enclosed with. Other things can be enclosed in that tag together with the headline, like any sub-headings, pictures, and any other text. But it is important that a link tag `A HREF` is together, and at the same level or below, the headline tag. Novobot will pick up the link automatically.

SUBHEAD

Not all sites have sub-headlines for their news headlines. The Register has those, so we’ll look at how to process sub-headlines using The Reg as an example. Look again at Listing 4. Locate the tag `<DIV CLASS=”indexintro”>`. It contains the text displayed below each headline, which comments on the contents of the news article. As you can see, the text itself is enclosed by the `DIV` tag. This is what we need. If you place the tag inside your `SUBHEAD` tag in the channel definition, Novobot will take the text in that tag and make it into a sub-heading for the appropriate headline.

If you cannot find anything that looks like a sub-heading in your source, don’t worry–your channel may just well work without sub-headings.

Another case is when a headline-containing tag is followed by a text that appears to be a sub-headline, but is not enclosed in any tags. To catch the sub-headline text in this case, use the following in your channel definition:

```
<SUBHEAD>*</SUBHEAD>
```

Novobot then will treat any free-standing text after a headline tag and up to the next tag as a sub-heading.

LINK

Since in HTML links are always represented by `A HREF` tag, there is no need to provide a separate `LINK` definition in your channel code. Novobot will pick up any `A HREF` links within headlines in your channel source document and use them as links for its headlines. That's why it is very important to find an enclosing tag for your headline text that also contains `A HREF` link.

SKIPTO

This is a tag that is generally used with HTML source documents that contain several sections (see SECTION chapter). Use it if there is too much HTML code between the two sections that your channel is processing, or to skip over to some part of the source document after processing a section. The `SECTION` tag can contain any combination of tags and attributes.

If your channel contains only one section, there is no need to use this tag.

END

The `END` tag is useful if you don't want Novobot to process headlines after some point in the source document. For example, there may be a situation when at the start of the web site a distinct tag is used for headlines, but later in the web site the same tag is used for some other purpose. Choose a tag after the headlines end, and use it in the `END` tag to tell Novobot not to process anything after that.

The Register Channel Sample

The finished channel is shown in Listing 5.

```
<?xml version="1.0"?>
<PARSEDEF
       name="The Register"
       url="http://www.theregister.co.uk/"
       logo="theregister.bmp"
       category="Cyberculture"
       version="1">
       <SECTION>
              <START><div class="indexheadlink"/></START>
              <HEADLINE><div class="indexheadlink"/></HEADLINE>
              <SUBHEAD><div class="indexintro"/></SUBHEAD>
       </SECTION>
</PARSEDEF>
```

Listing 5: The Register channel definition.

Note several interesting things about this channel:

- The contents of `START` and `HEADLINE` tags are the same. Novobot uses it to start looking for headlines (this is determined by the `START` tag), and immediately to look for every headline, as determined by the `HEADLINE` tag.

- `DIV CLASS` tags are used to distinguish between headlines, sub-headings, and other parts of the source document. The use of distinct styles and CSS[1] to mark headlines makes your work of designing a channel much easier.
- There is no `LINK` tag because Novobot uses `A HREF` tag nested in the headlines.

Section Options

Novobot has advanced capability of finding a specific text and matching channel definition instructions to the actual source document. This is accomplished by using pattern matching.

Pattern Matching

To use pattern matching, you need to turn it on for a specific section. The syntax is as follows:

```
<SECTION match="substr">
<SECTION match="regexp">
```

The first matching option makes Novobot look for any source text that has the exact specified piece of text in it. With substring matching, if you specify the following as your link definition:

```
<LINK><a href=".php"/></LINK>
```

—then any links containing text `.php` will be picked up by Novobot.

With regular expression, or pattern matching, if you specify your link definition as the following:

```
<LINK><a href="*.php"/></LINK>
```

—then Novobot will match any of strings that end with `.php`.

Note	Novobot uses case-insensitive string comparison when performing pattern matching.

You can use the following syntax in your patterns:

Pattern	**Matches**	**Sample**
*	Any sequence of characters (zero or more).	`*/News/*` matches `http://www.site.com/News/`, `/News/page.html`, and `/News/`.
?	Matches any single symbol.	`page?.html` matches `page0.html`, `page1.html`, but not `page10.html`.
\	Turns off pattern matching for the next character.	`\?param=value` matches `?param=value`.
[SET]	Matches any character in the set enclosed between square brackets.	`[123]` matches `1, 2, 3,` but not `4.`
[!SET] or [^SET]	Matches any character not in the set.	`[!123]` matches `4, 5, 6,` but not `1.`
SET	Several characters or range of characters	`A,B,C-Ea-z1-8,0`

Link Search Mode

In the situation when a headline-enclosing tag contains more than one A HREF link, and you need only the first one, use the following syntax to tell Novobot to ignore all subsequent links it encounters after the first one when processing a headline:

[1] Cascading Style Sheets (CSS) are used for advanced formatting of web pages, and to make web site consistent by using styles that define how various web page elements look.

```
<SECTION linksearch=”first”>
```

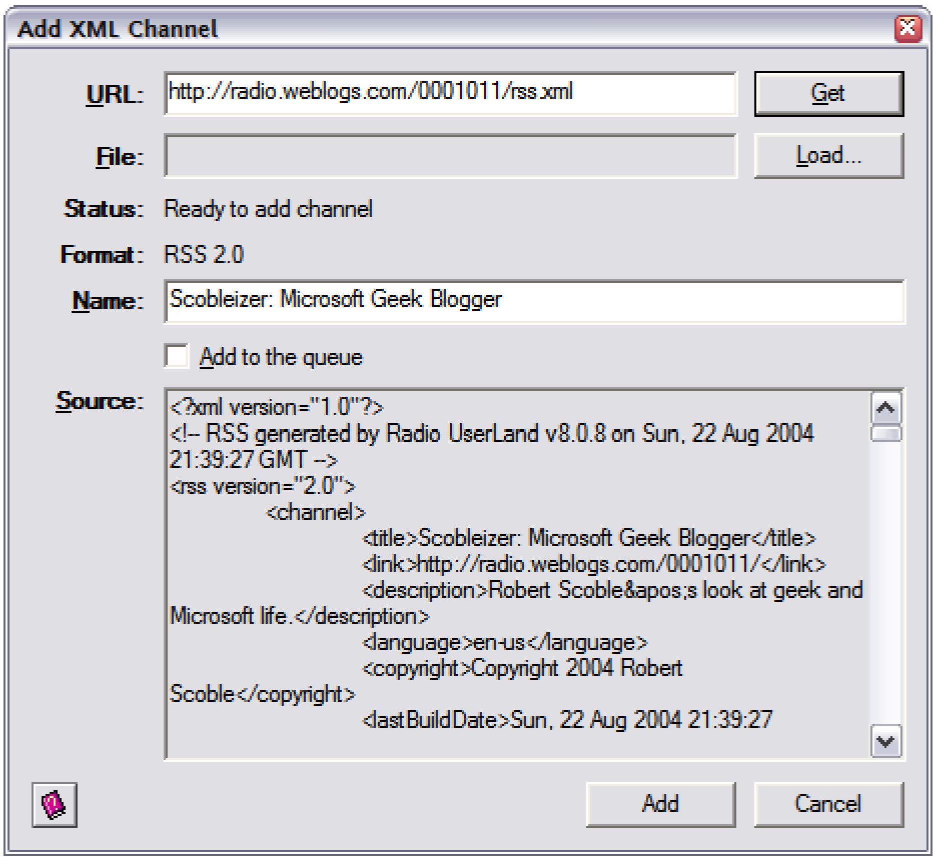
No Guarantees

With HTML-based channels, there are no guarantees that your created channel will work the next month, or even the next day. If the webmaster changes the design of the site you are scraping, the channel you created stops working. So the best choice is to create RSS or XML-based channels because they are more or less guaranteed to stay in a working state even after the web design of their sites changes.

Creating New Channels Automatically

Starting with version 2.1 Novobot can create channels automatically from feed URLs. It can figure out feed format if it is RDF or RSS v0.9/2.0, and add the newly created channels to the channel queue.

To create a channel automatically, choose File | New channel from the menu, or press Ctrl+N. Novobot will display New Channel dialog (Figure 4).



URL: http://radio.weblogs.com/0001011/rss.xml

File:

Status: Ready to add channel

Format: RSS 2.0

Name: Scobleizer: Microsoft Geek Blogger

☐ Add to the queue

Source:

```
<?xml version="1.0"?>
<!-- RSS generated by Radio UserLand v8.0.8 on Sun, 22 Aug 2004
21:39:27 GMT -->
<rss version="2.0">
        <channel>
                <title>Scobleizer: Microsoft Geek Blogger</title>
                <link>http://radio.weblogs.com/0001011/</link>
                <description>Robert Scoble&apos;s look at geek and
Microsoft life.</description>
                <language>en-us</language>
                <copyright>Copyright 2004 Robert
Scoble</copyright>
                <lastBuildDate>Sun, 22 Aug 2004 21:39:27
```

Figure 4: Add Channel Dialog

This dialog allows importing a XML-based feed given just its URL and creating a channel for it automatically. You can also add the new channel to the site queue immediately. For more information about this dialog please refer to the online help.

Conclusion

As you see, making Novobot channels is no magic, although sometimes it can get pretty close. Don‘t panic if you don‘t succeed right from the start. Take your time, study the predefined channels, and you will succeed. Well, if not, just drop me a message, and I‘ll try to help.

Remember, Novobot needs your support. If you have already purchased a license, then let me thank you for that. If not, I sincerely hope you will do that in the nearest future. In any case, tell your friends about Novobot, make channels for your sites and share them with other Novobot users. The more money Proggle makes from Novobot, the better the program will become, and the less time this will take. Right now I have probably spent more money on electricity while typing this text than I have made by selling Novobot. Nonetheless, I still hope Novobot will make a splash on the market one day, and that day is near. This document is one proof of my belief.

Glossary

HTML	Hypertext Markup Language, a tag-based language with fixed set of tags which is used to create web pages.
Parsedef	Parse Definition file, another name for a channel definition file
Parsedef root	The parsedef directory under the directory where Novobot is installed, which serves as the root of all installed channels.
Scrape	To process HTML source code of a web page in order to extract some useful information, like headlines.
Syndicate	To provide information for others in a predefined format. Web sites often syndicate news headlines for others to use in RSS or other formats.
URL	Uniform Resource Locator, another name for web link.
XML	eXtensible Markup Language, a tag-based description language that allows to define new tags.