

# FULL-TEXT RSS 2.8

FIVEFILTERS.ORG

## USER GUIDE

*Note: This guide is currently a work in progress. It is, at the moment, a somewhat expanded version of the information available at <http://fivefilters.org/content-only/faq.php>*

### TABLE OF CONTENTS

Introduction .....	2
What is a full-text feed?.....	2
What is Full-Text RSS (the software)?.....	2
Why is it useful? .....	3
How does it work? .....	3
A visual extraction example .....	3
System Requirements .....	4
Installation .....	4
Updating.....	5
Usage.....	5
Form Options .....	5
Extraction pattern .....	5
Max Items .....	6
Links .....	6
If extraction fails .....	6
Query string parameters.....	6
url.....	7
what .....	7
max.....	7
links .....	7
exc .....	7
html.....	7
format .....	8
Configuration .....	8
Create the custom config file.....	8
Enable caching .....	8
Site patterns.....	8
title: [XPath] .....	9

body: [XPath] .....	9
strip: [XPath] .....	9
strip_id_or_class: [string] .....	9
strip_image_src: [string] .....	9
tidy: [yes no] (default: yes) .....	10
prune: [yes no] (default: yes).....	10
autodetect_on_failure: [yes no] (default: yes) .....	10
single_page_link: [XPath].....	10
single_page_link_in_feed: [XPath] .....	10
test_url: [string] .....	11
# comments .....	11
Site Patterns by FiveFilters.org .....	11
Support .....	11

## INTRODUCTION

This document assumes that you are familiar with web feeds. If you're not, please first read the Wikipedia entry here: [http://en.wikipedia.org/wiki/Web\\_feed](http://en.wikipedia.org/wiki/Web_feed)

It's also important to point out that parts of this document are aimed at programmers and web developers. Full-Text RSS can be used with zero configuration and no programming knowledge, so please don't fret if you don't understand everything in this document. If you have any questions, email us at [fivefilters@fivefilters.org](mailto:fivefilters@fivefilters.org).

Now, before we get into the nitty-gritty, let's first distinguish between full-text feeds in general and the Full-Text RSS (FTR) software you're going to read about in this guide.

### WHAT IS A FULL-TEXT FEED?

A full-text feed is a standard web feed: essentially a list of the most recent articles published on a website. A feed can be as simple as a list of article links leading users to the full article, or it can contain the full article within the feed itself (letting users read the content without visiting the original site). When people refer to a full-text feed, they mean a feed which contains within it the full contents of the articles listed. A feed which does not include the full content is often called a partial feed (or truncated or abbreviated feed).

### WHAT IS FULL-TEXT RSS (THE SOFTWARE)?

Full-Text RSS (FTR from now on) is a free software (open source) tool which can convert partial feeds into full-text feeds. Alternatively, it can also process a single HTML page and return only its content as a single-item full-text feed.

At its heart, though, FTR is a content extraction tool. It uses web feeds as input and output because they are the most common feed format standards used on the web. So, applications which already support feeds can make use of FTR without change.

Developers can take advantage of FTR's content extraction by requesting JSON output or by using existing RSS libraries for their favourite programming language.

## WHY IS IT USEFUL?

In the simplest and most popular case, FTR is used to improve the use of news reading applications (e.g. Google Reader, RSS Owl). One common complaint among users of these applications is that many of the feeds they subscribe to contain only partial content – requiring them to read the full content outside of their news reading application (and thus defeating one of the major reasons for using the application in the first place).

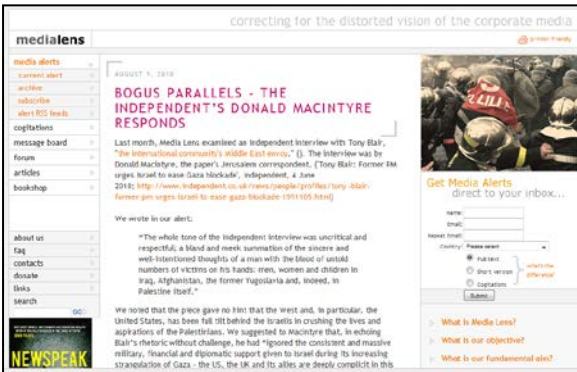
If you use a news reader, you can use FTR to convert these partial feeds into full-text feeds. FTR will generate a new URL which you can use with your news reader (replacing the original feed URL with the newly generated one).

## HOW DOES IT WORK?

FTR is designed to work with any web page carrying an article. It can accept as input a single web page (identified by its URL) or a list of web pages (identified by a feed URL). It then retrieves each page and uses a set of rules to identify the title and content block that's most likely to hold only the article's content (usually paragraphs of text). These blocks are then extracted and returned in the RSS feed format.

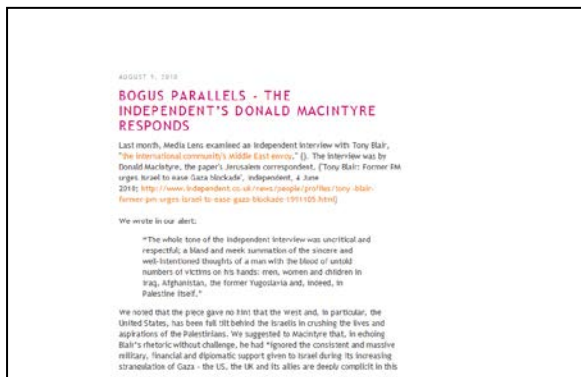
## A VISUAL EXTRACTION EXAMPLE

1. You give FTR the URL of the following page, or a feed URL which contains the following page (this is how the page looks in your browser):

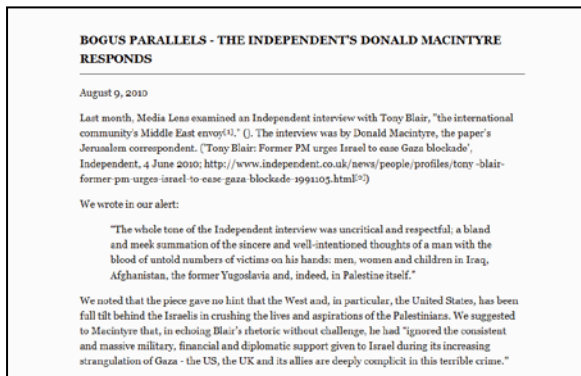


The screenshot shows the Media Lens website interface. At the top, it says "correcting for the distorted vision of the corporate media" and "Media Lens". The main content area features an article titled "BOGUS PARALLELS - THE INDEPENDENT'S DONALD MACINTYRE RESPONDS" dated August 9, 2010. The article text includes a quote from Donald MacIntyre, the paper's Jerusalem correspondent, and a link to a full-text version. A sidebar on the left contains navigation links like "media alerts", "message board", and "forum". On the right, there is a "Get Media Alerts" form with fields for name, email, and country, and a "Subscribe" button. Below the form are three bullet points: "What is Media Lens?", "What is our objective?", and "What is our fundamental aim?".

2. FTR retrieves the page, analyses it, and identifies the article block:



3. FTR then extracts this content block and returns it:



## SYSTEM REQUIREMENTS

PHP 5.2 or above is required.

FTR has been designed to run in most common PHP hosting environments. So, even space on a cheap, shared account will likely meet the system requirements.

To check if your server meets the requirements, we offer a simple compatibility test file. You can download it from <http://fivefilters.org/content-only/>. It's a single (zipped) PHP file you can upload to your server and access through your browser. It will tell you whether your server is capable of running FTR.

If you've installed FTR already, you'll find a link to run the test on the first page you see after installation.

## INSTALLATION

1. Extract the files in this ZIP archive to a folder on your computer
2. Using an FTP client, upload the files up to your server
3. Access index.php through your browser. E.g. <http://example.org/full-text-rss/index.php>
4. If you haven't done so already, click on the link to the compatibility test to see if your server meets the requirements

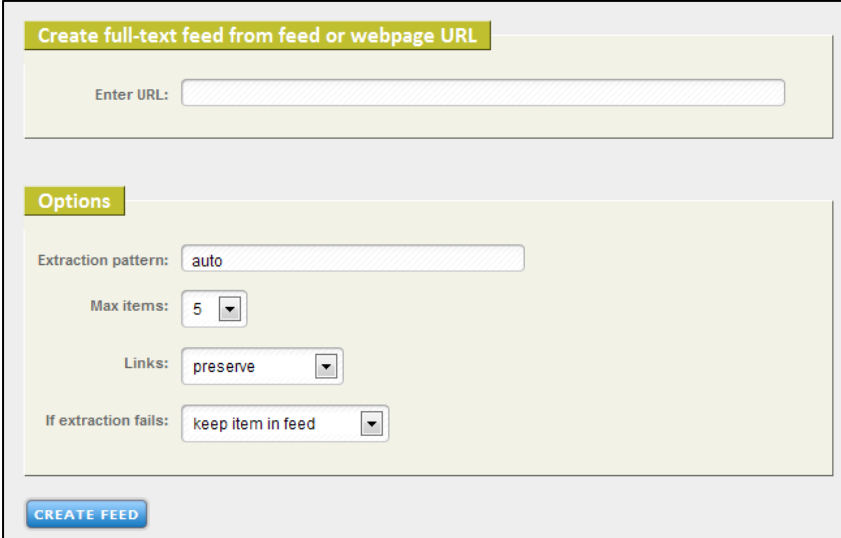
5. Enter a URL in the form field to test the code
6. If you get an RSS feed with full-text content, all is working well.

## UPDATING

Each new version of Full-Text RSS comes with an UPDATING.txt file with instructions. Please refer to that.

## USAGE

The simplest way to use FTR is to use the form provided.



The screenshot shows a web form titled "Create full-text feed from feed or webpage URL". It features a text input field labeled "Enter URL:". Below this is an "Options" section with four settings: "Extraction pattern:" set to "auto", "Max items:" set to "5", "Links:" set to "preserve", and "If extraction fails:" set to "keep item in feed". A blue "CREATE FEED" button is located at the bottom left of the form.

In the URL field, enter the URL of a partial feed (or if you're trying to extract from a single HTML page, enter its URL) and click 'CREATE FEED'. The resulting page should show you the new full-text feed and your browser will display the new URL in the address bar. You can now use this new URL in place of the original partial feed URL.

## FORM OPTIONS

In addition to the URL, you can also specify a number of other options in the form.

### Extraction pattern

By default FTR tries to figure out and extract the most likely content block. We suggest you leave this as 'auto', but if you'd like more control over what's extracted, you can override auto-detection and specify a CSS pattern here.

You can also combine auto-detection and a custom CSS pattern: to limit your CSS pattern to the detected content block, add 'auto ' at the beginning. Currently only a single element is extracted per item. Examples:

<code>auto</code>	Auto detected content block
<code>auto p</code>	First paragraph in content block
<code>auto img</code>	First image in content block
<code>div#content</code>	Don't detect, get everything in div#content

---

### Max Items

Set the maximum number of feed items we should process. The smaller the number, the faster the new feed is produced.

If your URL refers to a standard web page, this will have no effect: you will only get 1 item.

---

### Links

By default, links within the content are preserved. Change this field if you'd like links removed, or included as footnotes.

---

### If extraction fails

If the extraction pattern above fails to match, FTR can remove the item from the feed or keep it in.

Keeping the item will keep the title, URL and original description (if any) found in the feed. In addition, FTR inserts a message before the original description notifying you that extraction failed.

Note: This option has no effect on web page (non-feed) URLs: if extraction fails on such URLs, an error message is produced instead of a feed.

## QUERY STRING PARAMETERS

Using the form is the simplest way to create an FTR URL, but you can also construct one yourself. The form fields above are turned into query string parameters when you submit the form. Let's look at those parameters here, and a few more that are not represented on the form.

These parameters are to be appended on to the base URL. The base URL is where you installed FTR, e.g. `http://example.org/full-text-rss/makefulltextfeed.php`, because this will differ from installation to installation in this guide we'll simply use `makefulltextfeed.php` in examples.

Multiple parameters can be appended to the URL.

A note on encoding: if you're constructing URLs without using the form, make sure you URL encode the parameter values (anything after the '=' and before the '&'). In PHP the function to use is `urlencode()`. If you're doing it by hand, you can paste the parameter values into the form field at <http://meyerweb.com/eric/tools/dencoder/> and click 'Encode' to get the encoded the value.

---

## url

This is the only required parameter. It should be the URL to a partial feed or a standard HTML page.

Example: `makefulltextfeed.php?url= www.example.org%2Ffeed`

Note: %2F is the encoded value for '/'

---

## what

This is the extraction pattern (see section in form options above).

Example: `makefulltextfeed.php?url= www.example.org%2Ffeed&what= auto%20img`

---

## max

The maximum number of feed items to process (see section on max items in form options above).

Example: `makefulltextfeed.php?url= www.example.org%2Ffeed&max= 2`

---

## links

Link handling (see links section in form options above). Valid values are: preserve, footnotes, remove.

Example: `makefulltextfeed.php?url= www.example.org%2Ffeed&links= remove`

---

## exc

If extraction fails (see section in form options above). To remove items from feed if extraction fails, set this to 1. Exclude it from the URL (or set it to 0) if you'd like to preserve feed items.

Example: `makefulltextfeed.php?url= www.example.org%2Ffeed&exc= 1`

---

## html

If passed with the value of 1, FTR will not try to parse the response as a feed. This increases performance slightly and should be used if you know that the URL is not a feed. This option is not represented on the form.

Example: `makefulltextfeed.php?url= www.example.org%2Farticle.html&html= 1`

## format

The default FTR output is RSS. You can use this to specify JSON (currently the only other valid output format). Exclude it from the URL (or set it to 'rss') if you'd like RSS. This option is not represented on the form.

Example: `makefulltextfeed.php?url= www.example.org%2Ffeed&format= json`

## CONFIGURATION

FTR has a number of options which you might want to change in the configuration file. They're not all documented here, but the config file contains comments above each option which should give you an idea.

If you'd like to make any configuration changes, we suggest you follow these steps *after* you've made sure FTR is working as expected.

### CREATE THE CUSTOM CONFIG FILE

1. Save a copy of `config.php` as `custom_config.php`
2. Edit `custom_config.php`

### ENABLE CACHING

If you'd like to enable caching, we recommend you do it after you've configured everything else. Otherwise your changes to the configuration file may not immediately produce the result you want because of a previously cached copy.

1. Ensure the cache folder and its 2 sub folders are writable. (You might need to change the permissions of these folders to 777 through your FTP client.)
2. In `custom_config.php`, make sure you set caching to true:  

```
$options->caching = true;
```

### SITE PATTERNS

Site patterns can be used if FTR's automatic content extraction fails to pick out the correct content block for a particular site, or if additional fine tuning is required (e.g. to strip undesirable elements within the content block, or to include images which are not being included).

In the `site_config` folder you'll find two subfolders: 'standard' and 'custom'. FTR comes with a number of existing site patterns in the 'standard' folder. It's possible to change these, but we suggest users place their own site patterns in the custom folder to prevent future updates overwriting their site patterns.

---

title: [XPath]

The page title. XPaths evaluating to strings are also accepted.  
Multiple statements accepted. Will evaluate in order until a result is found.

Example:

```
title: //h1[@id='title']
```

---

body: [XPath]

The body-text container. Auto-detected by default.  
Multiple statements accepted.  
Will evaluate in order until a result is found.

Example:

```
body: //div[@id='body']  
# also possible to specify multiple  
# elements to be concatenated:  
body: //div[@class='byline'] | //div[@id='mainImage']
```

---

strip: [XPath]

Strip any matching element and its children.  
Multiple statements accepted.

Example:

```
strip: //div[@class='hidden']  
strip: //div[@id='content']//p[@class='promo']
```

---

strip\_id\_or\_class: [string]

Strip any element whose @id or @class contains this substring.  
Multiple statements accepted.

Example:

```
strip_id_or_class: hidden  
strip_id_or_class: navigation
```

---

strip\_image\_src: [string]

Strip any <img> whose @src contains this substring.  
Multiple statements accepted.

Example:

```
strip_image_src: /advert/  
strip_image_src: /tracker/
```

---

`tidy: [yes|no] (default: yes)`

Preprocess with Tidy. Tidy usually helps clean up the HTML for processing. It can, however, sometimes make matters worse. If it does, set this to no. (This setting may affect the final DOM tree produced, and with it affect your xpath expressions – so if your xpath is failing to match the desired elements, try setting this to ‘no’ to see if helps.)

Example:

```
tidy: no
```

---

`prune: [yes|no] (default: yes)`

Strip elements within body that do not resemble content elements. Sometimes this leads to elements which you’d like to keep from being stripped. If that happens, set this to no.

Example:

```
prune: no
```

---

`autodetect_on_failure: [yes|no] (default: yes)`

If set to no, we will not attempt to auto-detect the title or content block.

Example:

```
autodetect_on_failure: no
```

---

`single_page_link: [XPath]`

Identifies a link element or URL pointing to the page holding the entire article. This is useful for sites which split their articles across multiple pages. Links to such pages tend to display the first page with links to the other pages at the bottom. Often there is also a link to a page which displays the entire article on one page (e.g. ‘print view’). This should be an XPath expression identifying the link to that page. If present and we find a match, we will retrieve that page and the rest of the site config options will be applied to the new page.

Example:

```
single_page_link: //span[@class='singlePage']/a
```

---

`single_page_link_in_feed: [XPath]`

Same as above, but pattern applied to item description HTML taken from feed. Please be aware that a given item URL may appear in a variety of feeds which do not always contain the same item description. If both `single_page_link` and `single_page_link_in_feed` appear in the site config, `single_page_link_in_feed` will be ignored.

Example:

```
single_page_link_in_feed: //a
```

---

test\_url: [string]

A URL to use to test the pattern. In future, we'll have a tool which will use this to automatically test if the patterns in the file are still valid. Must be URL of an article from this site, not the site's front page. One or more.

Example:

```
test_url: http://www.example.org/2011/05/what-a-day/
```

---

# comments

Lines beginning with # are ignored.

Example:

```
# this is an advert
strip: //img[@class='ad']
```

## SITE PATTERNS BY FIVEFILTERS.ORG

If Full-Text RSS is not extracting what you want and you're uncomfortable creating your own site config file, you can ask us to create one for you. We currently charge a small fee to do this (unless you've purchased FTR and haven't requested your free site config yet, in which case we can do one for free – please email us with details). More information and purchase link at [http://fivefilters.org/content-only/custom\\_site\\_patterns.php](http://fivefilters.org/content-only/custom_site_patterns.php)

## SUPPORT

We have a public forum which anyone can use to discuss any issues, post questions and find answers: <https://member.fivefilters.org/f/> (it's free to join and post).

If you're purchased Full-Text RSS, you can also email us at [fivefilters@fivefilters.org](mailto:fivefilters@fivefilters.org) for support.