G51PRG Exercise Two: A very simple RSS Parser

Steven R. Bagley

Introduction

For this coursework, you are to implement a computer program which reads, or *parses*, an **RSS** document, looking for specific elements, and prints out only certain information (the title of articles) to the screen. We will use two BBC News RSS documents as an example.

This program is similar to the '*word counting*' program written in lectures eight and nine. That is, you will need to read characters from the input and process them differently depending on what **state** your program is in. However, **don't panic** as the approach is very similar to that program. You will also find it much easier if you break the problem down into smaller steps and build the program up step-by-step, some example steps are include in the *Hints* section below. Credit will be given for incomplete solutions, so please don't decide to not submit your coursework just because it is unfinished.

Your program does not need to work for all RSS files, just the two sample files provided. See the section below for details of how to get these files.

Parsing an RSS file

An RSS file is an XML file containing items of interest which are usually regularly updated, such as news items or blog entries. Like HTML, XML documents use tags contained between < and > symbols to mark-up the content, such as the following extract (from `rss-t.xml`):

```
<item>
  <title>Samsung revamps mobile line-up</title>
  <description>A Windows Phone 8 handset, Android "phablet" and…</description>
  <link>http://www.bbc.co.uk/news/technology-19416969#…</link>
  <guid isPermaLink="false">http://www.bbc.co.uk/news/technology-19416969</guid>
  <pubDate>Wed, 29 Aug 2012 19:54:12 GMT</pubDate>
  <media:thumbnail width="66" height="49" url="…"/>
  <media:thumbnail width="144" height="81" url="…"/>
</item>
```

Your program needs to extract the titles of each item, which is contained within the `<title>…</title>` tags. However, if you look at the sample files you will see that the `<title>` tag can appear elsewhere within the RSS file, so you need to find only the `<title>` tags that appear inside an `<item>…</item>` section.

The program requires you to read input from a file, and we will use the same method we did in lectures to write the word count program, i.e. repeatedly using `getchar()` to read each character from the standard input until the end-of-file (`-1`) is reached. You can pipe the contents of either RSS file into your program using the UNIX redirect operator '<' like this (assuming your compiled program is called '`readrss`', modify appropriately based on your program's name).

```
./readrss < rssfile.xml
```

You can either use `printf()` to write a character back to the output, but it is probably more efficient to use `putchar()` to print them out in this case, like so (assuming the character is in the variable `c`):

```
putchar(c);
```

Your program will be reading each character one-by-one, so you will need to implement this program by keeping state, like the word counting program. The state should change when you go in and out of relevant sections. For example, it should detect when it is reading a title, and then change state again when the title ends. Once it's in the '*found a title*' state, it should be printing the characters to the screen.

Hints

Do not under any circumstances attempt to write this program in one go … it won't work and will drive you insane! What follows is a list of hints and the steps I would take when implementing this coursework:

- ❏ Start off by making sure your program can read the whole file correctly. Test that by printing each character out, just as done in the lectures.
- ❏ There are three states this program needs to be in: *looking-for-item*, *Found-item-looking-for-title*, and *printing-title*.
- ❏ You are going to be looking for tags contained within '<' and '>'. A good first step would be to just print out the characters between the '<' and '>'. This will let you test your state switching code, but is also well on the way to the solution.
- ❏ Once that's working, you can look for the full `<title>` and `</title>` tags instead. Your program will then print out the text between the tags (i.e. the title).
- ❏ You can assume that the RSS files are '*well-formed*'. This means that you won't find an end-of-file character in the middle of a tag. This should make it easier to test for tags, once you see a '<' you can just use call `getchar()` again to see what the next character. If that is then a '`t`', you can see if the next one is an '`i`', etc.
- ❏ Once you've got it finding `<title>`s, the next step is to find the start of each `<item>` and use that third state.
- ❏ You only need to test enough characters of each tag so you know its definitely a `<title>` or `<item>`. There's no point testing more characters than you need to. You should find you probably only need to test the first three characters to be sure you've found the relevant tag (although you will need to skip over the remaining characters in the case of `<title>`).
- ❏ Did I mention that you'll go insane if you try and implement this all in one go? `:-)`

Remember, its fine while developing your solution to pop extra `printf()` statements in to see what your program is actually doing — you can always remove or comment them out when they are no longer necessary.

Sample Files

For this exercise, you should download the two example RSS files from the PRG webpages to test your program. The two examples were taken from the BBC news website on 30th August 2012.

You can either download and extract the samples via Windows and save it to your `H:\` drive in the correct place, or you can download it from the Linux command prompt by using the `curl` and `unzip` programs, like so:

```
curl -O http://g51prg.cs.nott.ac.uk/Distribution/Coursework/rss.zip
unzip rss.zip
```

Example Output

For file `rss-t.xml`.

```
Samsung revamps mobile line-up
US president appears on Reddit
Console games made 'free' online
Sony announces new touchscreens
Amazon launches daily deals in UK
Yahoo chief fired for storm gaffe
Baidu 'in search war' with rival
Motorola sets Intel phone date
Date set over Samsung 'phone ban'
UKNova made to halt torrent links
IBM debuts small, fast mainframe
World of Warcraft cut off in Iran
Log-in snag on Guild Wars 2 debut
Facebook ads: 100,000 bill for government
Emergency fish exit wins award
Underwater wheelchair put to test
VIDEO: Gadgets to make your business fly
VIDEO: Eco car Fisker's unique hybrid tech
VIDEO: Inside a digital car showroom
VIDEO: Are gamers 21st Century athletes?
VIDEO: Modified train for Paralympians
VIDEO: Samsung feels impact from court ruling
VIDEO: Billionaire unveils 'Titanic 2' plans
VIDEO: How to make a pop star 'hologram'
US builds virtual wall on border
Tech pushes athletes beyond human limits
Apple-Samsung: Expert opinions
How to hide data in plain sight
VIDEO: Robot puppetry makes 'alien art'
Is technology killing the lure of company heritage?
Fighting Fantasy is resurrected
```