CS 208 (Software Design)	**Spring 2013**
Homework #1: RSS Reflection	*By: Holly French, Andrew Bacon, Veronica Lynn*

During our code review, there were three things we discussed, mostly pertaining to the notion of code readability, especially depending on who our audience is.

1. Long, convoluted conditionals: we had a number of long convoluted conditionals that ended up dealing with a lot of edge cases regarding date and time in our RSS parser. These conditionals helped us determine when to print, but ultimately, they checked a lot of edge cases. The conditionals were more than one long line long, so they ended up looking like brute-force solutions to our problem. Instead, we decided to move the conditionals to another function called *isPrintable* and try to clean them up there.

2. Use of the ternary operator: In our code review, we discussed how readable the ternary operator was. One thing we talked about was our audience and the role they play in reading our code and influencing our decisions to use certain Java features. The ternary isn't inherently readable, but it is much shorter and succinct, yet expressive. Ultimately we decided to keep it: we used it primarily in error handling (return null if the date can't be found).

3. Useless comments that described code that was already more-or-less self-documenting. When we were writing Java functions to write to and read from a file, it was useful to have comments (especially since some of us weren't as familiar with a lot of these Java Objects to interact with files). However, once the code was completed, the comments were basically redundant, beacuse our function and variable names basically expressed what the code was doing.

We also made a number of fixes that we did not discuss in the Code Review, but we thought that they would be useful to a better user experience, and they would help improve the program with more robust error handling.

1. Better, concise user feedback: We ran into this difficulty a few times: various feeds and entries would be missing specific attributes. For example, three of the feeds we tested did not have dates. We discussed how to present dateless feeds to the user, but we ended up ignoring date if there was none provided.

2. More robust error handling: What happens if the user types in command line arguments that are not valid? We decided to try to add in some error handling so that the program would not totally crash when there weren't valid command line arguments.

3. More beautiful output: We got rid of those ugly html tags!