Assignment 2, Part A: News Feed Aggregator

(21%, due 11:59pm Sunday, October 20th, end of Week 12)

Overview

This is the first part of a two-part assignment. This part is worth 21% of your final grade for IFB104. Part B will be worth a further 4%. Part B is intended as a last-minute extension to the assignment, thereby testing the maintainability of your solution to Part A and your ability to work under time pressure. The instructions for completing Part B will not be released until Week 12. Whether or not you complete Part B you will submit only one solution, and receive only one mark, for the whole 25% assignment.

This a complex and challenging assignment. If you are unable to solve the whole problem, submit whichever parts you can get working. You will receive partial marks for incomplete solutions.

Motivation

The way we consume news has changed dramatically in recent years. The days of morning and afternoon home newspaper deliveries are long gone. Where readers were once restricted to a handful of local news sources, we now have a bewildering range of online options from around the world. Most newspapers, radio and television stations now make their news services available online, in addition to new purely online news services. Making sense of this cacophony is a challenge.

One response is news aggregation services. These allow readers to create their own news channels by mixing their preferred news sources together into a single stream. In this assignment you will create your own news aggregation application in Python. Your program will have a Graphical User Interface that allows its user to select how many stories they want to see from each source and then export an HTML document containing the selected stories. This document can be examined in a standard web browser or printed as a hardcopy.

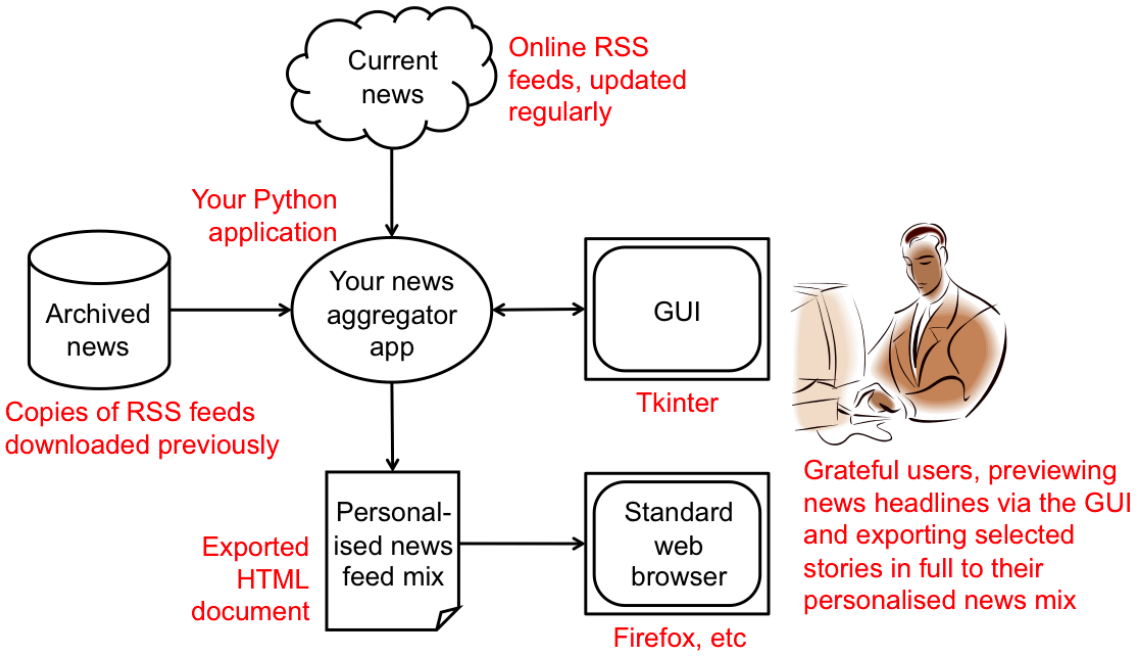
This "capstone" assignment is designed to incorporate all of the concepts taught in IFB104. To complete it you will need to: (a) use Tkinter to create an interactive Graphical User Interface; (b) download web documents using a Python script and use pattern matching to extract specific elements from them; and (c) generate an HTML document that integrates the extracted elements, presenting them in an attractive, easy-to-read format.

Goal

Your aim in this assignment is to develop an interactive "app" which allows its users to select how many news stories they want to see from several different news sources. There must be at least four different sources, two of them "live" news feeds and two "archives" of previously-downloaded news items. Most importantly, the two online web documents from which you get your "live" news must be ones that are updated on a continuous basis (at least daily but preferably much more often) so your program needs to be resilient to changes in the source documents. These two news sources must also come from different web sites (i.e., different web servers), to allow for one of the sites being temporarily offline.

For the purposes of this assignment you have a free choice of which news sources to use, provided there are always at least ten stories in each one, the stories are updated frequently, and the information available for each story includes a headline, the date/time of publication, a photo, and a short textual description. Appendix A below lists many "RSS Feed" web sites which should be suitable for this assignment, but you are encouraged to find your own of personal interest.

Using these news sources you are required to build an IT system with the following general architecture.



Your application will be a Python program with a Graphical User Interface. Under the user's control, it allows news feeds to be previewed in the GUI, from both online and archived news sources. When the user is happy with their selections they can then export the selected stories as an HTML document. This document will contain full detail of each story and can be studied by the user in any standard web browser.

This is a large and complex project, so its design allows it to be completed in distinct stages. You should aim to build the system incrementally, rather than trying to solve the whole problem at once. A suggested development sequence is:

1. Develop code that allows the static, archived news stories to be previewed in the GUI.
2. Extend your solution so that it allows "live" news stories to be previewed in the GUI.
3. Extend your solution further so that the user's selected stories can be exported as an HTML document.

If you can't complete the whole assignment submit whatever parts you can get working. You will get **partial marks for incomplete solutions** (see the marking guide below).

Illustrative example

To demonstrate the idea, below we describe our own news aggregation application, which uses information extracted from four different news sites. Our demonstration solution allows

users to select stories from two archived news feeds, The *Queensland Times* as it appeared on September 8th 2019 and the *Crikey* satirical news service from August 29th. It also allows two "live" news sources to be seen, the *Canberra Times* and *FOX News Entertainment*. Both of these sources are directly accessed by our Python application, so the news displayed is always the latest available. The application allows users to select how many stories they wish to see from each source. The program then accesses all the necessary information from the online and archived web sites and displays a preview of the top stories from each in the GUI. The user can then export a personalised news feed which includes stories from all selected sources as an HTML document which can be viewed in any standard web browser.

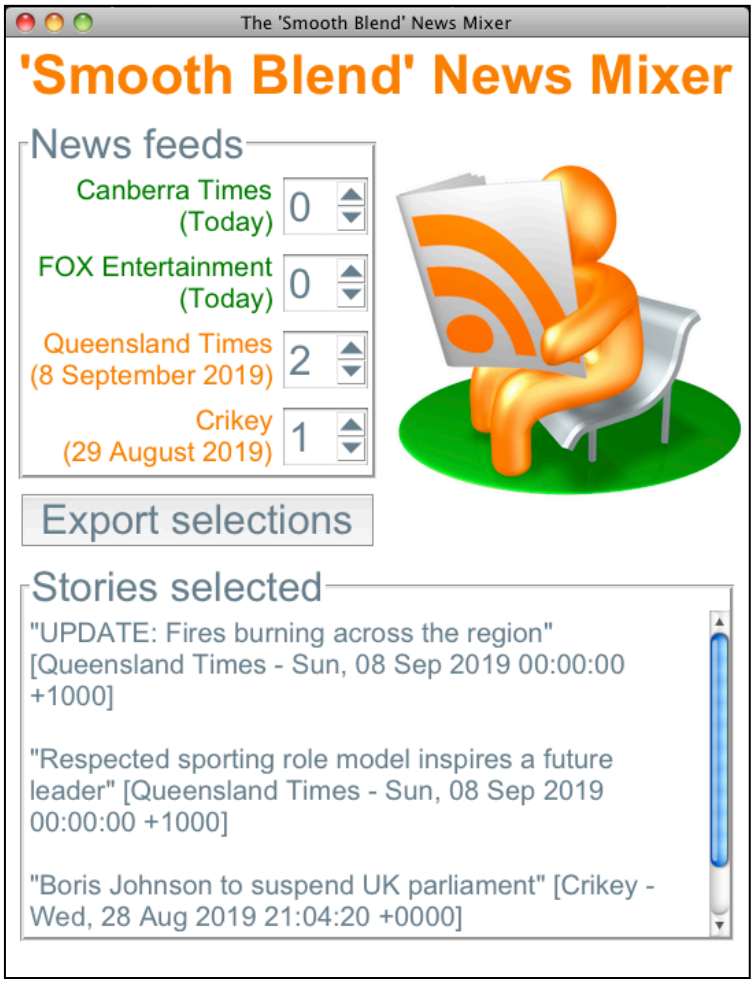
The screenshot below shows our example solution's GUI when first started. We've called it *The 'Smooth Blend' News Mixer* and have included a suitably evocative image of someone reading news from an RSS feed, but you should choose your own name and GUI design.

The GUI has four 'spin box' widgets allowing the user to select how many stories they want to see from each source, a scrollable text area for displaying previews of the selected stories, and a push button for exporting the selections. You do not need to copy our example and are encouraged to design your own GUI with equivalent functionality. For instance, pull-down menus or text entry boxes could be used for making the selections rather than spin boxes.

Selecting archived stories

When the user chooses a number of stories from the two archived sources, the application extracts headlines and publication dates for each story from local files, previously down-

loaded from the web. For instance, in the screenshot below the user has chosen to see two stories from our archived copy of the *Queensland Times* and one from the *Crikey* archive file.

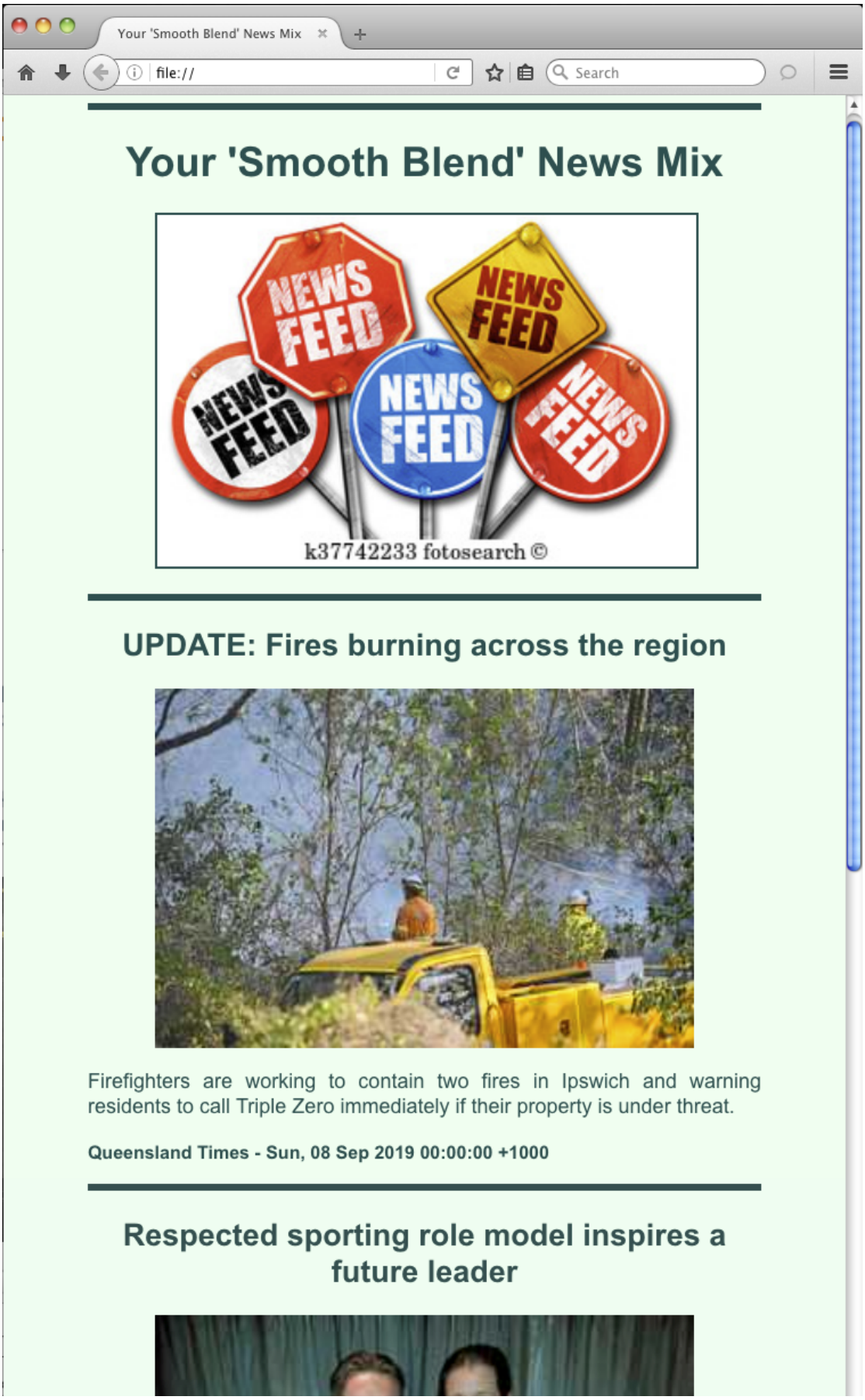


Accordingly, our application extracts the top two stories from the *Canberra Times* file and the top story from *Crikey*. It displays each story's headline, source and publication date in the preview pane. (We downloaded a copy of the *Crikey* site at 8:46am on August 29th but, as can be seen above, the most recent story on the site at that time was from the previous evening, with an announcement about Boris Johnson's latest Brexit move.)

Exporting the selected stories

Happy with their selections, the user then presses the "Export" button. This causes our application to generate an HTML file called `news.html` (in the same folder as the Python program). This document contains copies of the same stories previewed in the GUI, plus additional detail including a photo and a short story summary.

When opened in a standard web browser this file appears as shown overleaf. It includes a heading identifying the document and a "splash" image. Following this are the selected stories, each consisting of a headline, photo, short description, identification of the story's source, and the publication date. At the end of the file are four hyperlinks to the original web sites from which both the "live" and "archived" data is/was sourced.


Your 'Smooth Blend' News Mix
file://
Search
Your 'Smooth Blend' News Mix
NEWS FEED
NEWS FEED
NEWS FEED
NEWS FEED
NEWS FEED
k37742233 fotosearch ©
UPDATE: Fires burning across the region
Firefighters are working to contain two fires in Ipswich and warning residents to call Triple Zero immediately if their property is under threat.
Queensland Times - Sun, 08 Sep 2019 00:00:00 +1000
Respected sporting role model inspires a future leader

Both are Ipswich-bred Hancock Brothers A-Grade players who have excelled this season on and off the hockey field.

Queensland Times - Sun, 08 Sep 2019 00:00:00 +1000

Boris Johnson to suspend UK parliament

Good morning, early birds. Boris Johnson will suspend parliament in a bid to halt Brexit resistance, and NSW Labor has suspended general secretary Kaila Murnain amid fallout from the party's donations scandal. It's the news you need to know, with **Chris Woods**.

Crikey - Wed, 28 Aug 2019 21:04:20 +0000

Sources

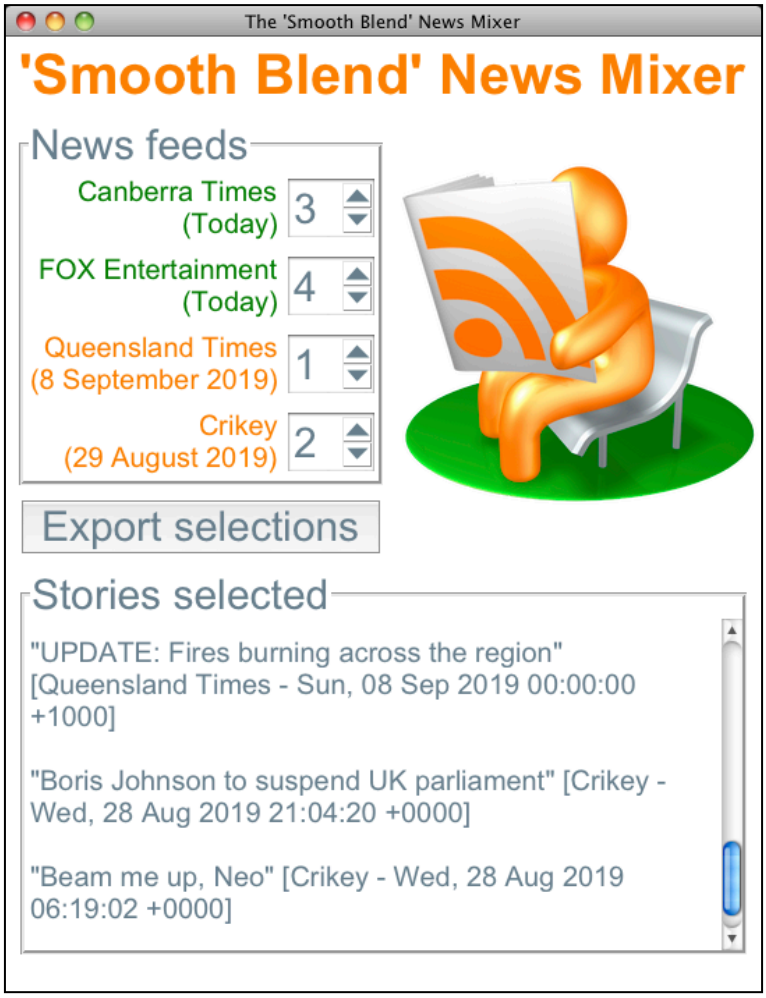
- Canberra Times: http://www.canberratimes.com.au/rss.xml
- FOX Entertainment: http://feeds.foxnews.com/foxnews/entertainment
- Queensland Times: https://www.qt.com.au/feeds/rss/homepage
- Crikey: https://www.crikey.com.au/feed

The document is well presented and the various elements of each story all match one another. Importantly, all the images in the exported document, including the "splash" image up the top and the story photos, are online images, not ones stored on our local computer. To ensure that the exported web document is portable and can be viewed on any computer, the photos are all links to online images using appropriate HTML "`img`" tags and URLs.

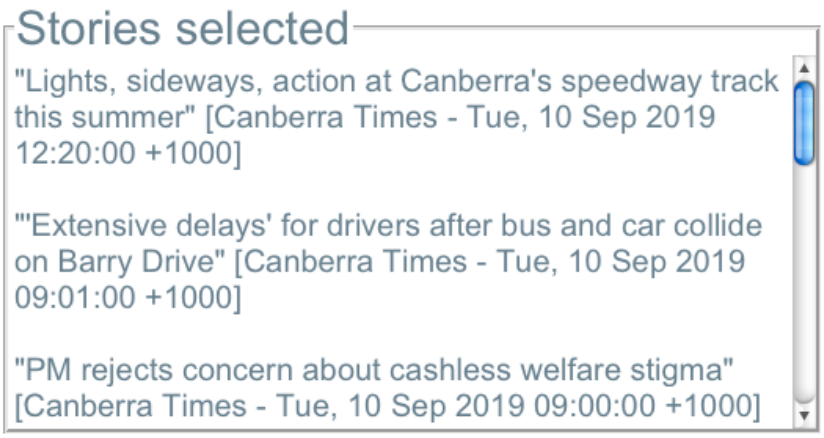
You are *not* required to follow the details of our demonstration GUI or exported HTML document. You are strongly encouraged to use your own skills and initiative to design your own solution, provided it has all the functionality and features described herein.

Selecting current stories

Tiring of reading "old news", our user next selects some "live" stories in the GUI as shown below, as well as changing the choice of archived stories in the mix.



As can be seen in the preview pane, our program updates the archived stories selected. More importantly, however, scrolling through the headlines reveals current stories downloaded "live" from the Internet. To do so our program downloaded copies of the source web pages and extracted appropriate elements from them. When we ran the program on September 10th the top three stories in the *Canberra Times* were as shown below.



Our user also selected four stories from *FOX News Entertainment*. Scrolling down in the preview pane reveals these stories as well.

Notice in the screenshots above that the publication dates for the stories from different sources are in different formats. This is how the dates were represented in the source web documents. There is no need to try to normalise or unify the date/time formats; they should simply be reproduced exactly as they appeared in the original documents.

Exporting the selected stories (again)

At this point our user again presses the "Export" button, causing all ten selected stories to be written to the `news.html` file. When opened in a web browser the chosen news feed mix appears as shown in the following extracts in this case. Here we have a unique mixture of both old and current news, and Australian and overseas news, but all of the stories are clearly labelled with their source and publication date, so there is no confusion. As usual, all the images in the document are links to online files. (And, no, we don't understand the second *Crikey* story either, but that's what they had on their web site at the time we downloaded their news feed!)


Your 'Smooth Blend' News Mix
file://
Search
Your 'Smooth Blend' News Mix
NEWS FEED
NEWS FEED
NEWS FEED
NEWS FEED
NEWS FEED
k37742233 fotosearch ©
Lights, sideways, action at Canberra's speedway track this summer
Persistence has paid off for Canberra's speedway club, with approval granted for three meetings this summer
Canberra Times - Tue, 10 Sep 2019 12:20:00 +1000
'Extensive delays' for drivers after bus and car collide on Barry Drive
Police
The collision happened before 9am at the intersection of Barry Drive and Dryandra Drive in O'Connor.
Canberra Times - Tue, 10 Sep 2019 09:01:00 +1000

Canberra Times - Tue, 10 Sep 2019 09:01:00 +1000

PM rejects concern about cashless welfare stigma

Prime Minister Scott Morrison has dismissed concerns that pushing people onto cashless welfare cards if they fail drug tests would further stigmatise them.

Canberra Times - Tue, 10 Sep 2019 09:00:00 +1000

Remodeled 'Brady Bunch' house set to be revealed by HGTV: 'We could be sitting on a gold mine,' says exec

The moment that many "Brady Bunch" fans have been waiting for is almost here!

FOX Entertainment - Tue, 10 Sep 2019 00:38:26 GMT

Sharon Osbourne reveals new facelift on season 10 premiere of 'The Talk'

by “misadventure,” and was not intentional, according to a new report.

FOX Entertainment - Mon, 09 Sep 2019 22:42:48 GMT

'The View' blasts Felicity Huffman over plea for no jail time in college admissions scandal

Days after Felicity Huffman pleaded with a Boston federal judge to spare her jail time for her connection to the college admissions bribery scandal, "The View" hosts blasted the actress on Monday's show, saying the "Desperate Housewives" alum "deserves to go to jail."

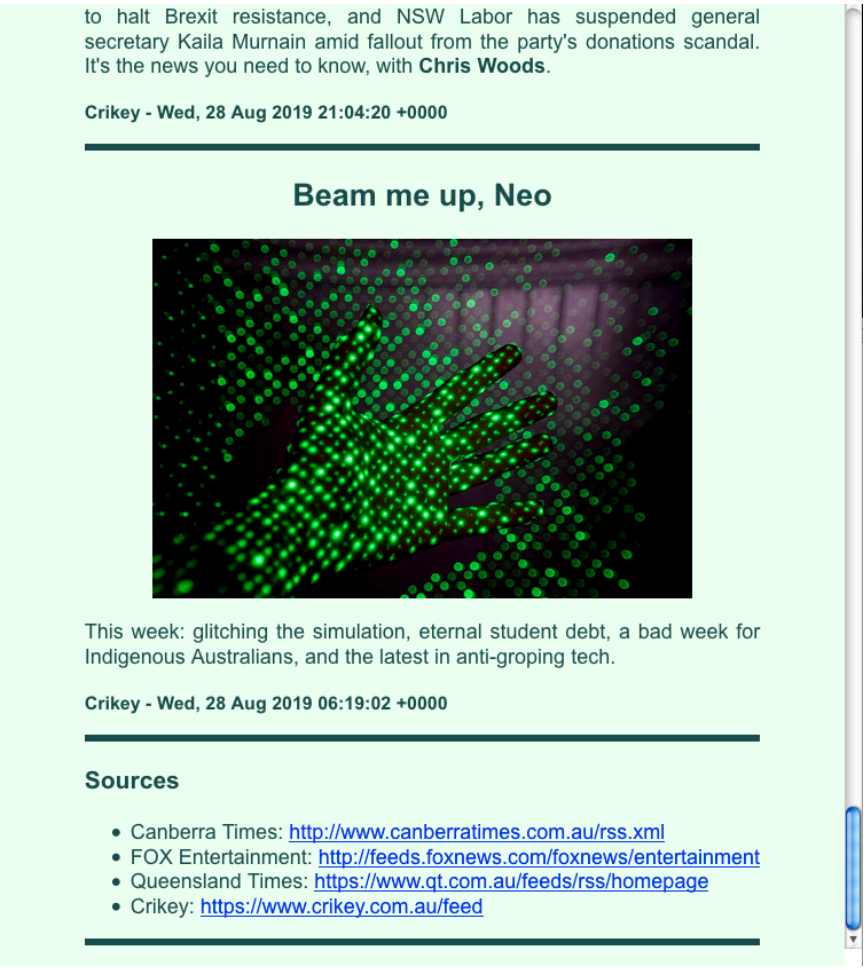
FOX Entertainment - Mon, 09 Sep 2019 21:47:11 GMT

UPDATE: Fires burning across the region

Firefighters are working to contain two fires in Ipswich and warning residents to call Triple Zero immediately if their property is under threat.

Queensland Times - Sun, 08 Sep 2019 00:00:00 +1000

Boris Johnson to suspend UK parliament



Extracting the HTML elements

To produce the news story details for displaying in the GUI and exporting as part of our HTML document, our application used regular expressions to extract elements from the relevant source web documents, whether they were stored in the static archive or are downloaded from the Internet whenever the program runs.

A significant challenge for this assignment is that web servers deliver different HTML/XML documents to different web browsers or other software clients. This means the web document you see in a browser may be different from the web page downloaded by your Python application. For this reason, to create your "archived" files you should download the web documents using our `downloader` program (see below), or a similar application. This will ensure that the "live" and "archived" documents have similar formats, thus making your pattern matching task easier.

To produce our demonstration solution, we first downloaded copies of the web pages and then studied their source code to identify text patterns that would help us find the specific elements we wanted. For instance, the XML source code for the *FOX Entertainment* news feed was as follows on September 10th.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" media="screen" href="/~d/styles/rss2full.xsl"?><?xm
  <channel>
                    <title>FOX News</title>
    <link>http://www.foxnews.com/</link>
    <description><![CDATA[FOXNews.com - Breaking news and video. Latest Current News
        <image>
        <url>http://tools.foxnews.com/sites/tools.foxnews.com/files/images/fox-news-
        <title>FOX News</title>
        <link>http://www.foxnews.com/</link>
    </image>
          <atom10:link xmlns:atom10="http://www.w3.org/2005/Atom" rel="self" type=
                <guid isPermaLink="false">https://www.foxnews.com/entertainment/re
      <link>http://feeds.foxnews.com/~r/foxnews/entertainment/~3/YjoBuUKB5W4/remodel
                <media:group>
      <media:content url="https://static.foxnews.com/foxnews.com/content/uploads/2
      <media:content url="http://a57.foxnews.com60/60/The-Brady-Bunch.jpg" medium=
    </media:group>
    <media:thumbnail url="http://a57.foxnews.com60/60/The-Brady-Bunch.jpg" width="
          <category domain="foxnews.com/metadata/dc.identifier">7457bd8c-f4a7-5ac5
                <category domain="foxnews.com/taxonomy">fox-news/entertainment/tv<
                      <category domain="foxnews.com/metadata/prism.channel">fnc</c
                      <category domain="foxnews.com/section-path">fnc/entertainmen
                <category domain="foxnews.com/content-type">article</category>
    <title>Remodeled 'Brady Bunch' house set to be revealed by HGTV: 'We could be
    <description>The moment that many "Brady Bunch" fans have been waiting for is
                <dc:creator>Mariah Haas</dc:creator>
                       <pubDate>Tue, 10 Sep 2019 00:38:26 GMT</pubDate>
                      <category domain="foxnews.com/metadata/dc.source">Fox News</
                                <category domain="foxnews.com/metadata/dc.creato
                           <feedburner:origLink>https://www.foxnews.com/entertainme
      <item>
```

Looking closely at this code we can, for instance, see the various elements of the top story, concerning the Brady Bunch's house renovation, including the headline (in `<title>` tags), the story summary (in `<description>` tags), the publication date (in `<pubDate>` tags), and the photo (as a JPG image URL, one of several alternatives). This knowledge was enough to allow us to create regular expressions which extract all the necessary elements for all strories in the document using Python's `findall` function.

Sometimes it's easier to use other Python features as well as, or instead of, regular expressions to help extract the data. For instance, we found that our regular expression for extracting headlines from the *FOX Entertainment* web site also matched the two "FOX News" titles at the top of the web page in addition to the headlines we wanted. Rather than complicating our regular expression, we therefore simply deleted the first two items returned by `findall` each time we extracted headlines from this page. (We also found that the URLs for the photos had inconsistent formats in this site, making them difficult to extract, so we don't recommend using this site in your own solution.) Most importantly, you must extract elements in a general way that will still work when the contents of the source web page are updated.

Obviously working with such complex code is challenging. You should begin with your static, "archived" documents to get some practice at pattern matching before trying the dynamically changeable web documents downloaded from online.

Care was also taken to ensure that no HTML/XML tags or other HTML entities appeared in the extracted text when displayed in either the GUI or the exported HTML document. In some cases it was necessary to delete or replace such mark-ups in the text after it was extracted from the original web document. The information seen by the user must not contain any extraneous tags or unusual characters that would interfere with the appearance of the news stories either in the GUI or the exported document.

Exporting the HTML document

Our program creates the exported HTML document by writing code into a text file, integrating the various elements extracted from the news feeds. Two segments of the HTML code generated by our Python program are shown below. Although not intended for human consumption, the HTML code is nonetheless laid out neatly, and with comments indicating the purpose of each part. Your HTML code must also be well presented to facilitate future maintenance of your application.

```
<!DOCTYPE html>
<html>

  <head>

    <!-- Tell browser to expect Unicode chars -->
    <meta charset="UTF-8">

    <!-- Title for browser window or tab -->
    <title>
      Your 'Smooth Blend' News Mix
    </title>

    <!-- Establish the overall document style -->
    <style>
      body {color:DarkSlateGrey; background-color:Hc
```

```
  <body>

    <!-- Horizontal lines are used as story separat
    <hr>

    <!-- Main heading -->
    <h1 align = "center">
      Your 'Smooth Blend' News Mix
    </h1>

    <!-- Splash image -->
    <p align = "center">
      <img src = "http://cdn.grid.fotosearch.com/CSI
           style="border:2px solid DarkSlateGrey">
    </p>

    <!-- Separator -->
    <hr>

    <!-- A news article -->
```

Robustness

Another important aspect of your solutin is that it must be resilient to error. The biggest risk with this kind of program is problems accessing the source web sites. We have attempted to make our `download` function as robust as possible. In particular, if it detects an error while downloading a web document it returns the special value `None` instead of a character string, so your program should allow for this. (We don't claim that the `download` function is infallible, however, because the results it produces are dependent on the behaviour of your specific Internet connection. For instance, some systems will generate a default web document when an online site can't be reached, in which case the `download` function will be unaware that a failure has occurred and won't return `None`.)

For instance, in our demonstration solution the GUI alerts the user to a failure to download a web site as follows.

Therefore, as insurance against the risk of a web site failing completely, your program's two "live" web sources must come from different web servers. One way of achieving this is to ensure that the part of the address at the beginning of each site'sURL is entirely distinct. For example, our sample solution used two totally different sources for the "live" news feeds, the *Canberra Times* and *FOX Entertainment*. These two sites have the following URLs and clearly come from different web servers.

(Since they never change, there is no need to use distinct servers for the two "archived" documents. Nonetheless, we did so in our sample solution to make the program more interesting.)

Specific requirements and marking guide

To complete this part of the assignment you are required to produce an application in Python 3 with features equivalent to those above, using the provided `news_aggregator.py` template file as your starting point. In addition you must provide the two (or more) previously-downloaded web documents that serve as your archive of "old news" and one or more image files needed to support your GUI. (However, all of the images in the exported HTML file must be online images and must *not* be included in your submission.)

Your complete solution must support *at least* the following features.

- **An intuitive Graphical User Interface (4%).** Your application must provide an attractive, easy-to-use GUI which has all the features needed for the user to choose how many news stories they want from each of four news feeds (two "archived" and two "live"), preview the headlines for their selections, and export the complete stories as a web document. You have a free choice of which Tkinter widgets to use to do the job, as long as they are effective and clear for the user. This interface must have the following features:
    - An image which acts as a "logo" to identify your application. The image file should be included in the same folder as your Python application.
    - Your GUI must name your application in both the Tkinter window's title and as a large heading in the displayed interface. Inside the window the name may appear as an integrated part of the logo, or as a separate textual label (as in our demonstration solution).
    - One or more widgets that allow the user to select how many stories they want to see from each of four news feeds (two "archived" and two "live").
    - One or more widgets that allow the user to see details of the stories selected (headlines, sources and publication dates).
    - One or more widgets that allow the user to choose whether or not to export their story selections as an HTML document.

  Note that this criterion concerns the front-end user interface only, not the back-end functionality. Functionality is assessed in the following criteria.
- **Previewing archived news stories in the GUI (4%).** Your GUI must be capable of displaying the top stories, in the quantities selected by the user, from each of two distinct sources of "archived" news, allowing selection of up to ten stories per source. For each story the GUI must display
    - the headline,
    - the news source (usually the name of a newspaper, magazine, TV or radio station), and
    - the publication date/time for the story.

  The necessary elements must be extracted from HTML/XML files previously downloaded and stored along with your Python program. The documents must be stored in exactly the form they were downloaded from the web server; they cannot be edited or modified in any way. Pattern matching must be used to extract the relevant elements from the documents so that the code would still work if the archived documents were replaced with others in the same format. To keep the size of your solution manageable only single HTML/XML source files can be stored. No image or style files may be stored in your "archive".
- **Previewing "live"news stories in the GUI (4%).** Your GUI must be capable of displaying the top stories, in the quantities selected by the user, from each of two distinct sources of "live" news, allowing selection of up to ten stories per source. For each story the GUI must display
    - the headline,

- the news source (usually the name of a newspaper, magazine, TV or radio station), and
- the publication date/time for the story.

The necessary elements must be extracted from HTML/XML files directly downloaded from the web while your Python program is running. Pattern matching must be used to extract the relevant elements from the documents so that the code still works even after the online documents are updated. The chosen source web sites must be ones that are updated on a regular basis, at least daily and preferably hourly. The two source web sites must come from different web servers (as insurance against one of the web sites being offline when your assignment is assessed).

- **Exporting selected news stories as an HTML document (5%).** Your program must be able to generate an HTML document containing full details of the top stories from each of the four news sources, live and/or archived, in the quantities selected by the user. The resulting "mixed news feed" must be written as an HTML document in the same folder as your Python program and must be easy to identify through an appropriate choice of file name, "`news.html`". The generated file must contain HTML markups that make its contents easily readable in any standard web browser, and it must be self-contained (i.e., not dependent on any other local files), although it may reference online images and style files. When viewed in a browser, the displayed document must be neat and well-presented and must contain at least the following features:
  - A heading identifying your application.
  - A "splash" image characterising your application, downloaded from online when the generated HTML document is viewed (i.e., not from a local file on the host computer).
  - Details of each of the news stories selected by the user in the GUI. For each story at least the following information must be displayed:
    - The headline.
    - A photograph or image illustrating the story.
    - A short story summary or description.
    - The identity of the original news feed (typically a newspaper, magazine, TV or radio station).
    - The date/time at which the story was published. (There is no need to standardise the format of this timestamp. It can appear in exactly the same format as the source web document. There is also no need to sort the stories from different sources into chronological order, although this would be a helpful feature.)

  All of this information must be extracted via pattern matching from HTML documents downloaded from the web. Most importantly, each of these sets of items *must all belong together*, e.g., you can't have the headline of one story paired with a photo from another story. Each of the elements must be extracted from the original document(s) separately and used to construct your own HTML document.

- Hyperlinks to the original four web sites from which the information was extracted, both live and archived. (For the live feeds this will help the markers compare the current web pages with your extracted information).

When viewed in a web browser the exported document must be neatly laid out and appear well-presented regardless of the browser window's dimensions. The textual parts extracted from the original documents must not contain any visible HTML tags or entities or any other spurious characters. The images must all be links to images found online, not in local files, must be of a size compatible with the rest of the document, and their original aspect ratio must be preserved (i.e., they should not be stretched in just one direction).

- **Good Python and HTML code quality and presentation (4%).** Both your Python program code and the generated HTML code must be *presented in a professional manner*. See the coding guidelines in the *IFB104 Code Presentation Guide* (on Blackboard under *Assessment*) for suggestions on how to achieve this for Python. In particular, each significant Python or HTML code segment must be *clearly commented* to say what it does, e.g., "Extract the link to the photo", "Display the story's publication date", etc.
- ***Extra feature (4%).*** *Part B of this assignment will require you to make a 'last-minute extension' to your solution. The instructions for Part B will not be released until just before the final deadline for Assignment 2.*

You can add other features if you wish, as long as you meet these basic requirements. You must complete the task using only basic Python 3 features and the modules already imported into the provided template. **You may not use any Python modules that need to be downloaded and installed separately, such as "Beautiful Soup" or "Pillow". Only modules that are part of a standard Python 3 installation may be used.**

However, your solution is *not* required to follow precisely our example shown above. Instead you are strongly encouraged to *be creative* in your choices of web sites, the design of your Graphical User Interface, and the design of your generated HTML document.

Support tools

To get started on this task you need to download various web documents of your choice and work out how to extract the necessary elements for displaying data in the GUI and generating the HTML output file. You also need to allow for the fact that the contents of the web documents from which you get your data will change regularly, so you cannot hardwire the locations of the elements into your program. Instead you must use Python's string `find` method and/or regular expression `findall` function to extract the necessary elements, no matter where they appear in the HTML/XML source code.

To help you develop your solution, we have included two small Python programs with these instructions.

1. `downloader` is a Python program containing a function called `download` that downloads and saves the source code of a web document as a text file, as well as returning the document's contents to the caller as a character string. A copy of this function also appears in the provided program template. You can use it both to save copies of your chosen web documents for storage in your "archive", as well as to

download "live" web documents in your Python application at run time. Although recommended, you are not required to use this function in your solution, if you prefer to write your own "downloading" code to do the job.

2. `regex_tester` is an interactive program introduced in the lectures and workshops which makes it easy to experiment with different regular expressions on small text segments. You can use this together with downloaded text from the web to help perfect your regular expressions. (There are also many online tools that do the same job you can use instead.)

Portability

An important aspect of software development is to ensure that your solution will work correctly on all computing platforms (or at least as many as possible). For this reason you must complete the assignment using standard Python 3 functions and modules only. You may not import any additional modules or files into your program other than those already imported by the given template file. In particular, **you may not use any Python modules that need to be downloaded and installed separately, such as "Beautiful Soup" or "Pillow". Only modules that are part of a standard Python 3 installation may be used.**

Security warning and plagiarism notice

This is an individual assessment item. All files submitted will be subjected to software plagiarism analysis using the MoSS system (http://theory.stanford.edu/~aiken/moss/). Serious violations of the university's policies regarding plagiarism will be forwarded to the Science and Engineering Faculty's Academic Misconduct Committee for formal prosecution.

As per QUT rules, you are not permitted to copy or *share* solutions to individual assessment items. In serious plagiarism cases SEF's Academic Misconduct Committee prosecutes both the copier and the original author equally. It is your responsibility to keep your solution secure. In particular, **you must not make your solution visible online via cloud-based code development platforms such as GitHub.** Note that free accounts for such platforms are usually public. If you wish to use such a resource, do so only if you are certain you have a private repository that cannot be seen by anyone else. For instance, university students can apply for a free private repository in GitHub to keep their assignments secure (https://education.github.com/pack). However, we recommend that the best way to avoid being prosecuted for plagiarism is to keep your work well away from the Internet!

Internet ethics: Responsible scraping

The process of automatically extracting data from web documents is sometimes called "scraping". However, in order to protect their intellectual property, and their computational resources, owners of some web sites may not want their data exploited in this way. They will therefore deny access to their web documents by anything other than recognised web browsers such as Firefox, Internet Explorer, etc. Typically in this situation the web server will return a short "Access Denied" document to your Python script instead of the expected web document (Appendix B).

In this situation it's possible to trick the web server into delivering you the desired document by having your Python script impersonate a standard web browser. To do this you need to

change the "user agent" identity enclosed in the request sent to the web server. The provided `download` function has an option that disguises its true identitity. We leave it to your own conscience whether or not you wish to activate this feature, but note that this assignment can be completed successfully without resorting to such subterfuge.

Deliverables

You should develop your solution by completing and submitting the provided Python template file `news_aggregator.py`. Submit this in a "zip" archive containing all the files needed to support your application as follows:

1. Your `news_aggregator.py` solution. Make sure you have completed the statement at the beginning of the Python file to confirm that this is your own individual work by inserting your name and student number in the places indicated. **Submissions without a completed statement will be assumed not to be your own work.**
2. One or more *small* image files needed to support your GUI interface, but **no other image files**.
3. The previously-downloaded web documents used as your static "archive" of old news stories. Only HTML/XML source code files may be included. **No image or style files associated with the web documents may be included.** All images or styles needed to support your exported HTML document must be sourced from online when it is viewed in a web browser.

Once you have completed your solution and have zipped up these items submit them to Blackboard as a single file. **Submit your solution compressed as a "zip" archive. Do not use other compression formats such as "rar" or "7z".**

Apart from working correctly your Python and HTML code must be well-presented and easy to understand, thanks to (sparse) commenting that explains the *purpose* of significant elements and *helpful* choices of variable, parameter and function names. **Professional presentation** of your code will be taken into account when marking this assignment.

If you are unable to solve the whole problem, submit whatever parts you can get working. You will receive **partial marks for incomplete solutions**.

How to submit your solution

A link is available on Blackboard under Assessment for uploading your solution before the deadline (11:59pm Sunday, October 20th, end of Week 12). Note that you can submit as many drafts of your solution as you like. You are strongly encouraged to *submit draft solutions* before the deadline as insurance against computer and network failures. If you are unsure whether or not you have successfully uploaded your file, upload it again!

Students who encounter problems uploading their files to Blackboard should contact the IT Helpdesk (ithelpdesk@qut.edu.au; 3138 4000) for assistance and advice. Teaching staff will not answer email queries on the weekend the assignment is due, so ensure that you have successfully uploaded at least one solution by close-of-business on Friday, October 18th.

Appendix A: Some RSS feeds that may prove helpful

For this assignment you need to find sources of regularly-updated news or current affairs stories, containing a headline, a publication date, a link to a photo, and a short summary of the story. You can choose any web sites that have these features, but to simplify your task you should seek ones that have a simple source-code format (when downloaded by a Python script). This appendix suggests some such sites, but you are strongly encouraged to find your own of personal interest.

The following links point to *Rich Site Summary*, a.k.a. *Really Simple Syndication*, web feed documents. RSS documents are written in XML and are used for publishing information that is updated frequently in a format that can be displayed by RSS reader software. Such documents have a simple standardised format, so we can rely on them always formatting their contents in the same way, making it relatively easy to extract specific elements from the document's source code via pattern matching.

However, a disadvantage of using RSS feeds is that they can be hard to find! Often you can discover them only by looking for the RSS symbol at the bottom of web pages or by trial-and-error to create the corresponding URL. Because RSS feeds are not intended for human consumption, they don't usually feature prominently in the results of web searches using standard search engines such as Google, DuckDuckGo, Bing, etc. You will need to do some exploration online to find suitable news feeds for your solution.

Note that you are *not* limited to using RSS sites for this assignment, but you may find other, more complex, web documents harder to work with. Most importantly, you are *not* required to use *any* of the sources below for this task. You are strongly encouraged to find online documents of your own, that contain material of personal interest.

The following web sites are RSS feeds which we found could be viewed in a standard web browser. We have *not* confirmed that these are all ideally suited to the assignment. You will need to work that out for yourself. In particular, we have *not* checked to see if their servers deny access to programs other than web browsers, nor have we checked to see what data they deliver to Python programs.

The following RSS news feeds appear to have headlines, publication dates, story summaries and links to photos all on the same page. (We used some of these for our demonstration solution.)

- http://www.sbs.com.au/news/rss/Section/Top+Stories — SBS News: Top stories
- https://www.9news.com.au/rss — Channel 9 (Australia) news
- https://www.qt.com.au/feeds/rss/homepage — Queensland Times
- http://www.canberratimes.com.au/rss.xml — Canberra Times: Local news
- https://www.whitsundaytimes.com.au/feeds/rss/homepage — Whitsunday Times
- https://www.crikey.com.au/feed/ — Crikey: Australian news and politics
- http://www.goulburnpost.com.au/rss.xml — Goulburn Post: Local news
- https://www.darkreading.com/rss_simple.asp — Dark Reading: Security news
- https://www.abc.net.au/radionational/feed/2890360/podcast.xml — ABC Radio National Breakfast news (includes audio links)
- http://www.perthnow.com.au/feed — Perth Now: Breaking news
- https://www.cnet.com/rss/news/ — CNet technology news
- https://www.news-mail.com.au/feeds/rss/homepage — Bundaberg News-Mail
- http://rss.upi.com/news/news.rss — UPI news

- https://www.coffscoastadvocate.com.au/feeds/rss/homepage — Coffs Coast Advocate
- https://www.dailyexaminer.com.au/feeds/rss/homepage — Grafton Daily Examiner
- https://www.sunshinecoastdaily.com.au/feeds/rss/homepage — Sunshine Coast Daily
- https://www.dailymail.co.uk/articles.rss — Daily Mail (UK): Latest news
- https://www.dailymail.co.uk/auhome/index.rss — Daily Mail: Australian news
- https://www.dailymail.co.uk/tvshowbiz/index.rss — Daily Mail: TV and Showbiz
- https://www.dailymail.co.uk/sciencetech/index.rss — Daily Mail: Science news
- https://www.blogger.com/feeds/2886832199291333748/posts/default — Vivian Violine's blog: Entertainment and business news
- http://rss.cnn.com/rss/cnn_topstories.rss — CNN News: Top Stories
- https://www.themorningbulletin.com.au/feeds/rss/homepage — Rockhampton Morning Bulletin
- http://feeds.nbcnews.com/feeds/topstories — NBC News
- https://rss.nytimes.com/services/xml/rss/nyt/HomePage.xml — New York Times: Top Stories
- http://www.nytimes.com/services/xml/rss/nyt/Business.xml — New York Times: Business News
- https://rss.nytimes.com/services/xml/rss/nyt/Technology.xml — New York Times: Tech News
- http://rssfeeds.usatoday.com/usatoday-NewsTopStories — USA Today: Top stories
- http://rssfeeds.usatoday.com/usatoday-LifeTopStories — USA Today: Lifestyle and entertainment
- http://rssfeeds.usatoday.com/usatoday-TechTopStories — USA Today: Tech news (not updated as often as other USA Today pages)
- https://threatpost.com/feed/ — Threatpost: Security news
- https://www.infoworld.com/index.rss — InfoWorld
- https://www.thechronicle.com.au/feeds/rss/homepage — Toowoomba Chronicle
- https://www.indiatvnews.com/rssnews/topstory.xml — India TV News: Top stories
- https://www.indiatvnews.com/rssnews/topstory-world.xml — India TV News: World news
- https://www.indiatvnews.com/rssnews/topstory-entertainment.xml — India TV News: Entertainment
- http://feeds.foxnews.com/foxnews/world — Fox News: World news
- http://feeds.foxnews.com/foxnews/entertainment — Fox News: Entertainment news
- http://feeds.foxnews.com/foxnews/science — Fox News: Science news
- http://feeds.foxnews.com/foxnews/tech — Fox News: Technology news
- https://nypost.com/feed/ — New York Post: All stories
- https://nypost.com/news/feed/ — New York Post: Breaking news
- https://nypost.com/entertainment/feed/ — New York Post: Entertainment news
- https://www.engadget.com/rss.xml — Technology and gadget news
- https://www.wired.com/feed — Wired magazine: Technology news
- https://abcnews.go.com/abcnews/internationalheadlines — ABC (USA) international news
- http://www.lemonde.fr/rss/une.xml — Le Monde: News from France (in French)
- https://feeds.gizmodo.com.au/gizmodoaustralia — Gizmodo's Australian feed: News for geeks
- http://syndication.eonline.com/syndication/feeds/rssfeeds/topstories.xml — E! Entertainment news: Top stories
- http://syndication.eonline.com/syndication/feeds/rssfeeds/tvnews.xml — E! Entertainment news: TV news
- http://feeds2.feedburner.com/TheNextWeb — The Next Web's news for geeks
- https://www.thelocal.es/feeds/rss.php — Local Spain: International news
- http://feeds.searchengineland.com/searchengineland — News about Internet search engines
- https://www.buzzfeed.com/world.xml — Buzzfeed's world news
- https://feeds.lifehacker.com.au/lifehackeraustralia — LifeHacker's Australian feed: Lifestyle for the Internet generation
- http://www.cbc.ca/cmlink/rss-world — CBC (Canada) world news
- http://www.globalissues.org/news/feed — Global issues news
- http://feeds.washingtonpost.com/rss/world — The Washington Post newspaper
- http://timesofindia.indiatimes.com/rssfeeds/296589292.cms — The Times of India world news
- https://www.rt.com/rss/news/ — RT world news
- http://feeds.feedburner.com/time/world — Time magazine world news
- https://www.northernstar.com.au/feeds/rss/homepage — Northern Star newspaper: Headline items
- https://sputniknews.com/export/rss2/world/index.xml — Sputnik international multimedia news
- http://www.independent.co.uk/news/world/rss — The Independent newspaper (UK) world news
- http://feeds.feedburner.com/daily-express-world-news — Daily Express (UK) world news
- https://www.mirror.co.uk/news/world-news/?service=rss — Daily Mirror (UK) world news

- http://www.latimes.com/world/rss2.0.xml — The Los Angeles Times world news
- http://feeds.skynews.com/feeds/rss/world.xml — Sky News: World news
- https://feeds.skynews.com/feeds/rss/entertainment.xml — Sky News: Entertainment
- http://en.rfi.fr/general/rss — Radio France International news
- http://feeds.news24.com/articles/news24/World/rss — News24 (South Africa) world news
- http://www.rawstory.com/category/world/feed — Raw Story (USA) world news
- http://globalnews.ca/world/feed — Global News: World news
- http://www.ctvnews.ca/rss/world/ctvnews-ca-world-public-rss-1.822289 — CTV (Canada) world news
- http://www.france24.com/en/top-stories/rss — France 24: Top stories
- http://www.seattletimes.com/nation-world/world/feed — Seattle Times: World news
- http://www.channelnewsasia.com/rssfeeds/8395884 — Channel News Asia: World news
- https://www.pri.org/stories/feed/everything — Public Radio International news (includes audio)
- https://www.neweurope.eu/category/world/feed — New Europe Newspaper: World news

The Australian Broadcasting Corporation has a wide range of RSS Feeds, but no longer appears to promote their use. The ABC's RSS Feed web site is now offline. Nonetheless, these sites have all the elements needed for the assignment (while they last!).

- ABC News: Just In — http://www.abc.net.au/news/feed/51120/rss.xml
- ABC News: Top Stories — http://www.abc.net.au/news/feed/45910/rss.xml
- ABC News: Australia — http://www.abc.net.au/news/feed/46182/rss.xml
- ABC News: Business — http://www.abc.net.au/news/feed/51892/rss.xml
- ABC News: Sport — http://www.abc.net.au/news/feed/45924/rss.xml
- ABC News: New South Wales — http://www.abc.net.au/news/feed/52498/rss.xml
- ABC News: Victoria — http://www.abc.net.au/news/feed/54242/rss.xml
- ABC News: Queensland — http://www.abc.net.au/news/feed/50990/rss.xml
- ABC News: Western Australia — http://www.abc.net.au/news/feed/52764/rss.xml
- ABC News: South Australia — http://www.abc.net.au/news/feed/54702/rss.xml
- ABC News: Tasmania — http://www.abc.net.au/news/feed/50042/rss.xml
- ABC News: ACT — http://www.abc.net.au/news/feed/48320/rss.xml
- ABC News: Northern Territory — http://www.abc.net.au/news/feed/53408/rss.xml

The following sites contain all the information needed for the assignment but are more complex than the ones above or have some other issue that may limit their suitability. Use them only if you are confident in your abilities.

- https://www.autoexpress.co.uk/car-news/feed — Auto Express Car News (slightly complicated site)
- http://rss.tvguide.com/breakingnews — TV Guide breaking news (complex site)
- https://www.autocar.co.uk/rss — Autocar motoring news (complex site)
- https://www.theverge.com/rss/frontpage — The Verge: Tech news (complex site)
- https://nakedsecurity.sophos.com/feed/ — Cyber security news (inconsistent article style)
- https://www.theguardian.com/world/rss — The Guardian newspaper (Beware: Only a few stories were listed when we checked!)
- https://feeds.kotaku.com.au/kotakuaustralia — Gaming news (complex page)
- https://world.wng.org/taxonomy/term/72/feed — World News Group: International news (complex page)
- https://www.androidpolice.com/feed/ — Android operating system news (complex page)
- http://feeds.mashable.com/Mashable — News for the connected generation (complex page)
- https://www.theverge.com/rss/index.xml — Technology and gadget news (complex page)
- http://www.polygon.com/rss/index.xml — Entertainment news (story summaries are very brief)
- https://www.prnewswire.com/rss/all-news-releases-from-PR-newswire-news.rss — PR Newswire (not all stories have photos on first page)
- https://www.vox.com/rss/index.xml — News in detail (complex site, full stories instead of summaries)
- http://feeds.feedburner.com/WarNewsUpdates — Warfare news (complex site)
- http://feeds.macrumors.com/MacRumors-All — News about Apple (very complex site)
- https://www.yahoo.com/news/rss/world — Yahoo! news (not all stories have an image)
- http://feeds.mashable.com/Mashable — Mashable geek news (complex page)

The following sites appear to have headlines, datelines and story summaries on the main page, but you may need to follow each story's link to get a photo. This is possible, but makes using them harder than the ones above.

- http://www.thedenverchannel.com/rss/ — Denver Channel 7 news
- https://www.theregister.co.uk/security/headlines.atom — The Register: Security news
- https://www.tripwire.com/state-of-security/feed/ — Tripwire: Security news
- http://feeds.feedburner.com/Securityweek?format=xml — Security Week
- http://feeds.arstechnica.com/arstechnica/index — Ars Technica: All news
- http://feeds.arstechnica.com/arstechnica/technology-lab — Ars Technica: IT news
- http://rss.slashdot.org/Slashdot/slashdot — Slashdot news for nerds
- http://www.couriermail.com.au/feed — Courier Mail newspaper: Breaking news
- http://feeds.bbci.co.uk/news/technology/rss.xml — BBC News: Technology
- http://www.themercury.com.au/feed — The Mercury: Latest news
- http://www.goldcoastbulletin.com.au/feed — Gold Coast Bulletin: Breaking news
- http://www.businessnews.com.au/rssfeed/latest.rss — Business News: Latest news
- http://feeds.watoday.com.au/rssheadlines/top.xml — Western Australia Today: Top news
- http://feeds.brisbanetimes.com.au/rssheadlines/top.xml — Brisbane Times: Latest news
- http://www.theaustralian.com.au/feed/ — The Australian newspaper: Latest news
- http://feeds.theage.com.au/rssheadlines/top.xml — The Age newspaper: Latest news
- https://feeds.a.dj.com/rss/RSSWorldNews.xml — Wall Street Journal: World news
- https://feeds.a.dj.com/rss/RSSWSJD.xml — Wall Street Journal: Technology news
- https://feeds.a.dj.com/rss/RSSLifestyle.xml — Wall Street Journal: Lifestyle
- http://feeds.sydneysun.com/rss/ae0def0d9b645403 — Sydney Sun newspaper: Headlines
- https://www.aceshowbiz.com/rss/asb_news.xml — Ace Showbiz News
- http://www.news.com.au/feed — News Corp (Australia) news
- https://feeds.feedburner.com/techcrunch — TechCrunch web technology news
- http://ifpnews.com/feed — Iran Front Page news
- https://www.usnews.com/rss/news — U.S. News: Breaking news
- https://www.espn.com/espn/rss/news — ESPN News: Latest news
- https://zeenews.india.com/rss/india-national-news.xml — ZeeNews (India): World news
- https://zeenews.india.com/rss/entertainment-news.xml — ZeeNews (India): Entertainment news
- http://www1.cbn.com/cbnnews/world/feed — CBN News: World news
- http://www.washingtontimes.com/rss/headlines/news/world — Washington Times: World news
- https://www.smh.com.au/rss/world.xml — Sydney Morning Herald: World news
- https://www.cbsnews.com/latest/rss/world — CBS (USA) world news
- https://www.thesun.co.uk/news/worldnews/feed — The Sun (UK) world news
- https://www.cnbc.com/id/100727362/device/rss/rss.html — CNBC (USA) world news
- http://feeds.nature.com/nature/rss/current — Nature, the world's leading scientific journal
- https://www.aljazeera.com/xml/rss/all.xml — Al Jazeera's English news service
- http://feeds.feedburner.com/ndtvnews-world-news — NDTV (India) world news
- http://feeds.bbci.co.uk/news/world/rss.xml — BBC News: World
- http://feeds.bbci.co.uk/news/rss.xml — BBC News: Home
- https://newslanes.com/feed — Newslanes: World news
- http://rss.slashdot.org/Slashdot/slashdot — News for nerds
- http://feeds.reuters.com/reuters/topNews — Reuters: Top News
- http://feeds.reuters.com/Reuters/worldNews — Reuters: World News
- http://feeds.reuters.com/reuters/technologyNews — Reuters: Technology News
- http://feeds.reuters.com/reuters/businessNews — Reuters: Business News
- http://feeds.arstechnica.com/arstechnica/index/ — IT news
- http://feeds.feedburner.com/Techcrunch — Web news
- https://www.npr.org/rss/rss.php — National Public Radio (USA) news
- https://www.reddit.com/r/technology/.rss — Reddit's tech news
- https://defence-blog.com/feed — Defence news

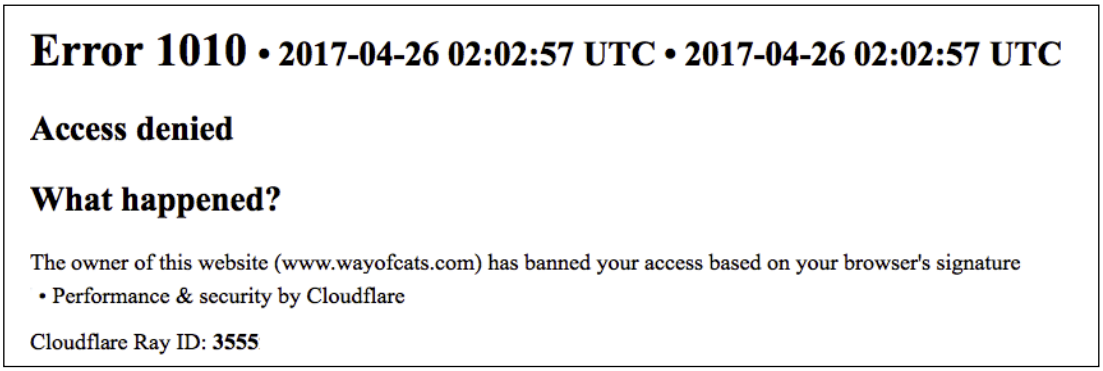
Finally, the following sites appear to have only headlines and datelines, or only headlines and stories, on the main page. Again, it may be necessary to follow the links for each story to get the rest of the information needed, making them hard to use.

- https://www.computerweekly.com/rss/All-Computer-Weekly-content.xml — Computer Weekly
- https://www.thestar.com/feeds.articles.news.world.rss — Toronto Star world news (some articles have stories and photos but not all)
- https://feeds.howtogeek.com/HowToGeek — How-To Geek's news (links take you to different source web sites, so hard to use for the assignment)
- https://news.ycombinator.com/rss — News for code hackers (links take you to different source web sites, so very hard to use for the assignment)
- https://www.reddit.com/r/worldnews/top/.rss — Reddit's world news
- http://indaily.com.au/feed — InDaily newspaper
- https://zeenews.india.com/rss/india-national-news.xml — ZeeNews: Indian national news
- http://www.dailytelegraph.com.au/news/world/rss — Daily Telegraph: World news
- http://www.heraldsun.com.au/rss — Herald Sun newspaper: Breaking news
- http://www.ntnews.com.au/news/rss — Northern Territory newspaper (not clear how often it is up-dated)
- http://www.townsvillebulletin.com.au/news/rss — Townsville Bulletin newspaper (not clear how often it is updated)

Appendix B: Web sites that block access to Python scripts

As noted above, some web servers will block access to web documents by Python programs in the belief that they may be malware attempting a "denial of service" attack or in order to protect the web site owner's intellectual property. In this situation they usually return a short HTML document containing an "Access Denied" message instead of the document requested. This can be very confusing because you can usually view the document without any problems using a standard web browser even though your Python program is delivered something entirely different.

If you suspect that your Python program isn't being allowed to access your chosen web page, use the `downloader` program to check whether or not Python programs are being sent an access denied message. When viewed in a web browser, such messages typically look something like the following example. In this case blog `www.wayofcats.com` has used anti-malware application *Cloudflare* to block access to the blog's contents by our Python program.



In this situation you are encouraged to choose another source of data. Although it's possible to trick some web sites into delivering blocked pages to a Python script by changing the "user agent" signature sent to the server in the request we *don't* recommend doing so, partly because this solution is not reliable and partly because it could be considered unethical to deliberately override the web site owner's wishes.