PART IV: USER-SPECIFIED TRIGGERS

Right now, your triggers are specified in your Python code, and to change them, you have to edit your program. This is very user-unfriendly. (Imagine if you had to edit the source code of your web browser every time you wanted to add a bookmark!)

Instead, we want you to read your trigger configuration from a `triggers.txt` file every time your application starts, and use the triggers specified there.

Consider the following example trigger configuration file:

```
# subject trigger named t1
t1 SUBJECT world

# title trigger named t2
t2 TITLE Intel

# phrase trigger named t3
t3 PHRASE New York City

# composite trigger named t4
t4 AND t2 t3

# the trigger set contains t1 and
t4
ADD t1 t4
```

The example file specifies that four triggers should be created, and that two of those triggers should be added to the trigger list:

- A trigger that fires when a subject contains the word 'world' (`t1`).
- A trigger that fires when the title contains the word 'Intel' and the news item contains the phrase 'New York City' somewhere (`t4`).

The two other triggers (`t2` and `t3`) are created but not added to the trigger set directly. They are used as arguments for the composite `AND` trigger's definition (`t4`).

Each line in this file does one of the following:

- is blank
- is a comment (begins with a `#`)
- defines a named trigger
- adds triggers to the trigger list.

Each type of line is described below.

- **Blank:** blank lines are ignored. A line that consists only of whitespace is a blank line.

- **Comments:** Any line that begins with a `#` character is ignored.
- **Trigger definitions:** Lines that do not begin with the keyword `ADD` define named triggers. The first element in a trigger definition is the name of the trigger. The name can be any combination of letters without spaces, except for "ADD". The second element of a trigger definition is a keyword (e.g., `TITLE`, `PHRASE`, etc.) that specifies the kind of trigger being defined. The remaining elements of the definition are the trigger arguments. What arguments are required depends on the trigger type:
  - `TITLE` : a single word.
  - `SUBJECT` : a single word.
  - `SUMMARY` : a single word.
  - `NOT` : the name of the trigger that will be NOT'd.
  - `AND` : the names of the two other triggers that will be AND'd.
  - `OR` : the names of the two other triggers that will be OR'd.
  - `PHRASE` : a phrase.
- **Trigger addition:** A trigger definition should create a trigger and associate it with a name but should not automatically add that trigger to the running trigger list. One or more `ADD` lines in the .txt file will specify which triggers should be in the trigger list. An addition line begins with the `ADD` keyword. Following `ADD` are the names of one or more previously defined triggers. These triggers will be added to the the trigger list.

PROBLEM 11

We have implemented the function `readTriggerConfig(filename)` for you. We've written code to open the file and throw away all the lines that don't begin with instructions (e.g. comments and blank spaces), and then reads in the code that defines triggers and instantiates the triggers by making a call to the helper function `makeTrigger`. The function returns a list of triggers specified in the configuration file

First, read through the definition of `readTriggerConfig`. You should be able to understand everything this function is doing at this point in the course.

Next, implement the function `makeTrigger(triggerMap, triggerType, params)`. This helper function should build and return a trigger depending on its type. It also keeps track of triggers and names in a map. We have defined for you the specifications for this function to make it easier for you to write.

Once that's done, modify the code within the function `main_thread` to use the trigger list specified in your configuration file, instead of the one we hard-coded for you:

```
        # TODO: Problem 11
        # After implementing makeTrigger, uncomment the line below:
        # triggerlist = readTriggerConfig("triggers.txt")
```

After completing Problem 11, you can try running `ps6.py`, and depending on how your `triggers.txt` file is defined, various RSS news items should pop up for easy reading. The code runs an infinite loop, checking the RSS feed for new stories every 60 seconds.

Hint: If no stories are popping up, open up `triggers.txt` and change the triggers to ones that reflect current events (if you don't keep up, just pick a trigger that would fire on one of the current Google news (http://news.google.com/) stories).

GRADER IS CURRENTLY DOWN; IT WILL BE AVAILABLE SOON!

In the meantime, you can work on implementing this problem on your own machine, and verify your implementation with the provided test suite file.

Check

Show Discussion

New Post

© 2012 edX, some rights reserved.

terms of service (/tos) privacy policy (/privacy) honor code (/honor) help (/help)