

HomeWork #5

Bots, Spiders, and Emotional Agents

Fall 2001

Homework 5: InfoDroppings - Desktop Folder for Information Processing

Estimated Time: 4 hours (Basic), 3+ hours (extras)

Goal: Create a desktop folder for “on the fly” drag-and-drops of URLs, email address, text clippings, etc. for background processing by a Rebol Bot Family (coupled with other scripting applications, too).

Due: Thursday, December 13, 2001

(submit via email to BotClass <degroot@liacs.nl>)

Description

Problem:

In this project you will create a repository for various types of information pointers that you would like to handle at some time, but not right when you first come across them. For example, most people are plagued by the problem of distraction when they are searching for something on the Internet. For example, when using their favorite search engine, they often come across very interesting pointers that they would like to pursue but which they currently do not have time for (or should not take the time for!). Current Browser usage paradigms do not adequately support offline, delayed browsing and processing, so many people find that they are driven to explore these tangential pages anyway --- and right in the midst of their research.

One potential solution is to add these interesting links to your Favorites menu, under a “To Do” category. The problem with this approach is that most people find that they seem to rarely possess the time to actually go back and pursue all those “To Dos” that they so carefully stored away.

There are lots of other examples, clearly, most of which you might already be aware of or even currently suffering from. In this homework, you will explore one solution class that addresses some (but clearly not all) problems you might frequently encounter.

The approach taken involved the following:

1. a desktop folder named “InfoDroppings” (or whatever you would like to name it)
2. a customizable set of folders that you create within the InfoDroppings folder to be used for holding your information droppings
3. a Rebol Bot that is responsible for the top-level scanning of the folder
4. a family of Rebol Bots that handle specific types of processing requests

Approach:

Fortunately, much of the code has already been written for you (by yours truly) so your main tasks lie within the Design, Test, and Explore stages of this project. A suggested approach follows:

Step 1:

Create a folder called InfoDroppings somewhere high up in your folder/disk hierarchy, e.g.,

C:\InfoDroppings

or wherever you want it. (We experienced problems when we placed it directly on the desktop, so be careful). Place an alias to this folder in the lower, right-hand corner of your desktop. The result should look vaguely similar to the illustration shown in Figure 1.

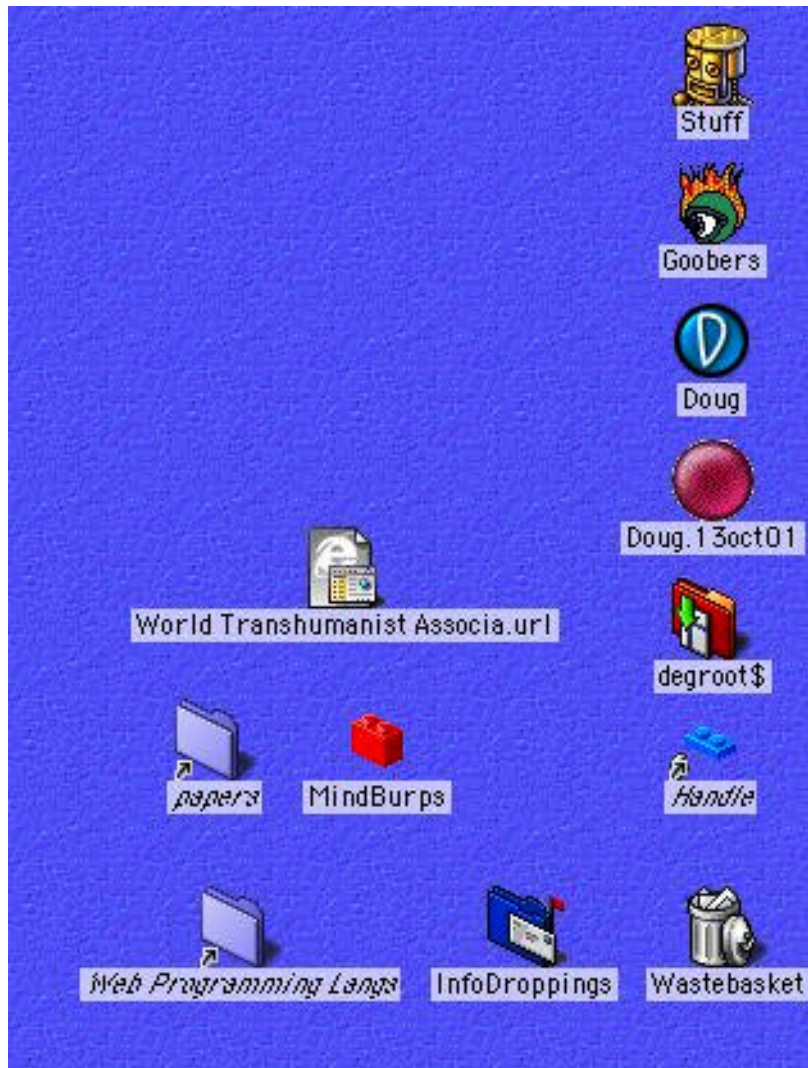


Figure 1. Your Desktop Appearance

Step 2:

Within the InfoDroppings folder, create the following sub-folders:

1. GrabThePics
2. SendToBCchat
3. SendToDeGroot
4. SendToMe
5. SendToDad
6. HIDE ME FOR NOW

You can add others, if you want, as you will see later. Set the “view type” of the InfoDroppings folder to “list” (under the View menu option). Figure 2 shows an example of how your InfoDroppings folder should look when opened.

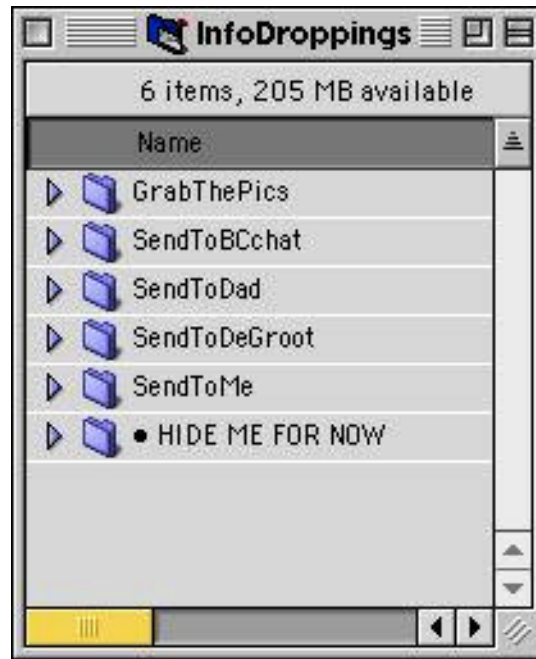


Figure 2. A Sample InfoDroppings Organization

Your folder will likely look somewhat different than the one shown in Figure 2, but the figure will give you an idea of how yours should look. Once the InfoDroppings folder is open, position it too near the bottom right-hand corner of your desktop. Henceforth, when you double-click the alias, the original folder should open up and be positioned at this same place (rebooting may destroy this setting, but you should have to reset it at most once per session).

Step 3:

Now go back and open the InfoDroppings folder. For each folder within InfoDroppings, open the folder and create another folder within it named Archives.

Step 4:

Surf to <http://nbc.msnbc.com> (or some other interesting site) using your Internet browser. In the Address slot of your browser, you will see the resulting URL of the displayed page. You should see something like this (Figure 3).



Figure 3: After surfing to www.nbc.msnbc.com

At this point, pretend like this is one of the most interesting pages you have ever seen in your entire adult-surfing-life and that you think Prof. DeGroot would get a kick out of this page, too. So grab the URL icon in the Address slot, and drag-and-drop it into the SendToDeGroot folder inside your InfoDroppings folder. (If you are using Netscape, then the icon is the one preceding the word "Location" near the Location Bar."

OK, now surf somewhere else – perhaps to a site that is related to bots, spiders, the Turing Test, the Xnet, Digital Immortality, or whatever. Once you find a site you really like, drag its URL to the SendToBCchat folder inside the InfoDroppings folder. At this point, you should have two little Internet Shortcut files – one inside the SendToDeGroot folder, and one inside the SendToBCchat folder.

Step 5:

Examine one or both of these files in a text-editor. You will (should?) find that it looks somewhat like the illustration in Figure 4.

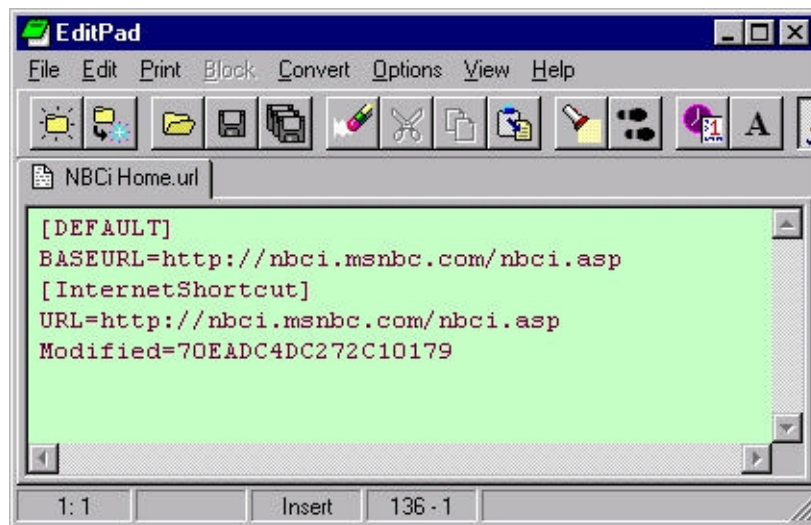


Figure 4: The Internals of an Explorer "Internet Shortcut" File

Of course, if you are using Netscape, the file will appear differently, but it should still exhibit basically the same content and similar format.

Step 6:

You now need to develop a bot that will, when invoked, scan your InfoDroppings folder, see which folders are inside it, and process these folders one-by-one. What this consists of is simply scanning each of these folders to see what is inside, if anything, and for each file within the folder, processing it according to rules that are defined for that specific folder.

For example, the bot will find within the SendToDeGroot folder the Internet Shortcut file for the NBCi site. It does not really matter what the file name is, as the bot will simply read it and discover the URL information within the file. It is that URL that the bot will send to Prof. DeGroot via an email message, and not the whole file. Of course, the name of the file can/will be used as the Subject of the email, but that is not as critical as getting the URL correctly of the page you want to share with the person to whom you are sending the URL.

Actually, you would benefit considerably from thinking this problem through in detail and writing your own bot for this exercise, but due to time constraints and the like, I am providing an example solution for your use in this homework. You will need to extend it in ways described below.

The files you will need are named:

- absolutizeURLs.r (a modified version of the original absolute-urls.r, by Graham Chiu)
- grabThePics.r (will grab all pictures, both thumbnails and the linked-to pictures)
- urlDroppings.r (the top-level Bot Runner and dispatcher)
- emailUrlDroppings.r (will send a dropped-URL message to someone)
- myUtils.r (the same as the utility file you have worked with before)
- myMoUtils.r (contains some new I/O routines and a few other "jewels")

They are available on the HotLine server under "Class Rez/Nov 22 Class". There are several changes you need to make to the source files before the system will work properly. The required changes *should* all be documented in the code (however, I always reserve the right to make innumerable mistakes, so don't count on it!).

At a minimum, you should change the following elements:

bcUrlDroppings.r

1. dropTo (data structure) – Change every FIXME@liacs.nl to your own email address. Change the greetings (the string elements) anyway you want.

```
dropTo: [
  Dad: ["Hi, Dad. (BTW, I could use more money.)" [FIXME@liacs.nl] ]
  DeGroot ["Hello, most esteemed Prof. DeGroot:" [degroot@liacs.nl] ]
  Deutz ["Hello, most esteemed Prof. Deutz:" [deutz@liacs.nl] ]
  Me ["Hey, YOU!" [FIXME@liacs.nl] ]
  BCchat ["Hi, everyone." [bcchat@email.com] ]
  Testing ["Hi ya, Tester." [FIXME@liacs.nl] ]
  ;add the rest here
]
```

2. If you really want to use your Dad as guinea pig, add his email address.

bcEmailUrlDroppings.r

1. Edit the URLdroppingsMsg email template to your suiting. Be aware that the first line of the template will be used as the Subject of your email. The \$GREETING\$ setting comes from the dropTo data structure, while the \$URL\$ is read from the URL dropping file itself.

```
URLdroppi ngsMsg: {
  BotURL: $URLS ← this will become the Subject line of your email

  --- start ---
  $GREETINGS

  This is a "URL Droppings Bot" message from <your name here>.
  He thought you might find the following URL to be of interest/value:

      $URLS

  Goodbye,
  Your friendly URLdroppi ngsBot
  --- end ---
}
```

2. Edit the TXTdroppingsMSG email template to your suiting in a similar manner.
3. In particular, in both of these templates, replace “<put your name here>” with your own Name.
4. Change the next-to-last line of send-one-dropping by using your own email address. You should leave this line uncommented while testing, but once you are confident that things are working fairly well, comment out this line and uncomment the line above it. The bot will now send out actual emails to real people, so be careful and have fun.

bcGrabThePics.r

1. Create some high-level folder with a suitable name for your pictures grabbed off the Internet (e.g., C:GrabbedPictures). Change the global variable picsDest (near the beginning of the file) to the path to that folder.

All of them

1. You will notice that I have commented out all dprint and vprint calls. This is only for those of you who do not wish to use these debugging routines. I would encourage you to uncomment them, though. The relevant routines are in myUtils.r.

Step 7:

At this point, launch Rebol and execute (do) all the relevant files (the ones listed above). To launch the top level Bot Dispatcher, simply type (within Rebol, of course):

```
process-ud-folders
```

Hopefully, everything will work properly and you will see some email messages being sent. After a reasonable period of time, you should receive emails in your email inbox at LIACS (or wherever you specified). Practice dragging-and-dropping URLs from your browser into the various “SendToX” folders, running the bot, and watching how she works. As you become familiar with the process, create more SendTo-type folders for anyone you want and modify the code to handle their droppings as well.

NOTE: While you are testing and debugging the process, I suggest you move all folders but one (the one you want to work on) into the HIDE ME FOR NOW folder. This way, you can limit your exposure to one process at a time. When you have everything fairly well debugged, you can move them

Step 8:

OK, now move all your SendToX folders into the HIDE ME FOR NOW folder (unless you really don't want to). Retrieve the GrabThePics folder (if you have hidden it) and make sure it is in the top level of the InfoDroppings folder. Using your browser, surf around and find some sites that contain graphic images you think are interesting, valuable, or cool. When you find such a site, drag the URL into your GrabThePics folder inside InfoDropping.

After you have collected 2 or 3 or more URL shortcuts, run the bot again and debug any remaining problems (hopefully, there will be none, but who knows?). You just *might* have problems with the parsing routine in GrabThePic.r. It may on occasion try to read a URL that is not a legal URL. In such cases, the code will usually break and throw you back into the top-level listener. Don't worry about these. We will provide solutions (or band-aids) at a later time.

Step 9:

Once everything is working, pull all your hidden folders back to the top level within InfoDroppings. Make sure you have commented out the “testing” send statement within send-one-dropping and have uncommented the “real” send statement (send emAddr theMsg). You can now begin to actually use this new capability to boost your productivity (or whatever else you like to boost). Grab some URLs and let your bot send them to bcChat to let us all know you are now up and running.

Create as many more SendToX folders as you want, but be sure to add the support data to the dropTo data structure in bcUrlDroppings.r, one entry per folder name and person.

Step 10:

Now the fun really begins! Think about 1) what other activities you frequently find yourself spending a lot of time on, and 2) what other activities that you wish you could do but that you don't simply because they would take too much time. Regardless of whether you can implement a bot to handle the process for you, go ahead and create a folder in the InfoDropping folder for each such activity you come up with. You can always come back and implement the bots later, but in the meantime, you will have a collection of URLs to both remind you and to provide value to your research.

Examples include:

- GrabTheMP3s – about as similar to GrabThePics as you could get! (so it's *trivial*)
- SnarfTheSite – for use with a WebWhacker/WebBuddy bot service
- AddToPhotoAlbum – self-explanatory
- SMSme – send the “cleaned up” info dropping to your mobile via SMS
- PrintTheSite – snarf the site and then print it (the whole thing)
- PrintFirstPages – print the first 2 pages of each web page specified
- Googlefy – read the specified Google page, parse the URL results, read each resulting web page, and print out the first page(s) of the site
- BookMarkEm – add the URLs on the page to your bookmark manager
- PDFem – read the web page and print as an Adobe Acrobat file
- etc., etc. – there are many more examples possible

Step 11:

Implement one or more additions to your bot suite (similar to the examples above). Submit the final code, including all changes to the provided code modules, along with a written description of the bot, it's purpose, instructions for activating it, URLs you tried it on (if it works with URLs), and one or more log files exhibiting its workings.

The basic grade will be awarded for achieving correct, basic functionality within the allocated time frame. Extra grade points will be awarded for the level of difficulty or sophistication of the extra bots implemented as part of Step 11.

Questions:

If you have questions, please consider sending them to bcChat@email.com, so that others in the class can see every one else's questions and answers, just in case multiple people have the same problem. If you don't feel comfortable with this though, then please do feel free to send a question directly to either me or Andre anyway.

-- end --