Installing and Using Gourmet Recipe Manger

This document explains at a high level some of the quirks about Gourmet, an open source recipe manager. This document is currently a stub with very basic information primarily focused on a script to reformat the HTML output into a more sane format.

1. Installing Gourmet

Under Ubuntu, simply run **sudo apt-get install gourmet**. You can also download the latest (non-repository) version or versions for other operating systems here (http://sourceforge.net/projects/grecipe-manager/files/).

2. Exporting to HTML

Gourmet has a very nice method of exporting the entire cookbook to HTML format. It generates individual files and a master recipe list. It would be nice if it broke the list into sections or organized things better, but honestly there are much worse decisions that could have been made.

What I found difficult, however, is that the recipe filenames are essentially randomly generated. They contain spaces (yuck!) and are appended with what appears to be a recipe number. The problem is this number can change, so when you post a link to the world's best haggis recipe and then make a change to said recipe, you need to do work to make sure the link is updated.

To handle this, I've created a script which should run under most BASH-oriented operating systems (it's designed for Ubuntu Linux, but may work under OSX or under Windows with Cygwin (http://www.cygwin.com/) installed). It does two primary things. First, it strips out any whitespace in the name and removes the sequence number. I also add some META tags, this helps search engines know what they're looking at. I also do things like delete any old renamed recipes first (to start with a clean slate), skip processing the `index.htm` file (which is the list of recipes), and skip processing anything that's not an HTML file (like the CSS definition). I then go to the `index.htm` file and replace the old URL with the new one. Below is a listing of the code, including my comments, but you can also download the file here (recipe-process.sh). Simply go to wherever you exported the files to and run **recipe-process.sh** and it will automatically update things.

Example 1. recipe-process.sh

```
#!/bin/bash

# Make a backup of index.htm
echo Backing up index.htm
cp index.htm index_$(date +"%Y%m%d-%H%M%S").htm
echo Removing old processed recipes
rm recipe_*

for filename in *; do
  if [ -f "$filename" ]; then
    if [ "$filename" = "index.htm" ]; then
      echo Index file found
    else
      if [ $(echo $filename | sed 's/.*\(...\)$/\1/') = "htm" ]; then
        # First strip out the numbers
echo HTML $filename
        newfilename=$(echo $filename | sed 's/[0-9]//g')
        # Strip the extension
        newfilename=$(echo $newfilename | sed 's/....$//')
        # Create metatag data
        meta=\<meta\ name\=\"keywords\"\ content=\"$(echo $newfilename),$(echo $newfilename
        # Add +1 link
        plusone='<table width="100%"> <tr> <td><iframe frameborder="0" width="100%" height='

        # Finally strip out the spaces and add back the .htm
        newfilename=recipe_$(echo $newfilename | sed 's/\ //g').htm

        # Modify index.htm with the new reference
        sed -i "s/${filename}/${newfilename}/g" index.htm

        # Convert spaces to something Bash likes better...
        filename=$(echo "$filename" | sed 's/[ ]/\ /g')

         # Insert Meta tags into the header.
        head -n5 "$filename" > $newfilename
        echo $meta >> $newfilename
        echo $plusone >> $newfilename
        tail -n+6 "$filename" >> $newfilename
      else
        echo Skipping, $filename not an HTML file
      fi
    fi
  fi
```