Tawking AWK Extras

PRESENTED BY:
Kent Archie
kentarchie@gmail.com

Data Structure Examples

Some data on groceries as a CSV

Column Titles

```
item,store,price,date,categories
Milk,Family Foods,2.59,2014-04-07,"Dairy"
Chicken,Walmart,8.7,2014-04-12,"Meat,Chicken"
```

- Turn each line into a simple array of strings, hard code the columns
- Turn each line into an array using the column titles as indicies
- Turn it into an array of arrays to include all the data in the source
- Use nested arrays to view the data as a tree

Turn each line into a simple array of strings, hard code the columns(Slide41.awk)

```
#!/usr/bin/gawk -f
@include "csv.awk" # from http://lorance.freeshell.org/csv/
@include "utilities.awk"

BEGIN { #run once before processing lines
        FS=",";
} # BEGIN

FNR == 1 {next}   # skip first line

{
        if(NR % 100 == 0) printf("Lines so far (%d)\n", NR);

        num_fields = csv_parse($0, csv, ",", "\"", "\"", "\\n", 0)
        if (num_fields < 0) {
                printf("ERROR: %d (%s) -> %s\n", num_fields, csv_err(num_fields), $0);
                continue;
        }

        printf("Lines: store=:%s:, date=:%s:, item=:%s:, price=:%s:, label=:%s:\n",
                csv[1], csv[2], csv[3], csv[4], csv[5]);
} # for each line

END   { # run once after processing lines
                printf("END: processed %d data points\n",NR);
} # END
```

Slide41 output

```
...

Lines: store=:Family Foods:, date=:2014-05-19:, item=:Salt:,
price=:0.99:, label=:Salt:
Lines: store=:Family Foods:, date=:2014-05-19:, item=:Bread
Crumbs:, price=:2.69:, label=:Baking:
Lines: store=:Family Foods:, date=:2014-05-19:, item=:Garlic:,
price=:0.81:, label=:Spices:
Lines: store=:Family Foods:, date=:2014-05-19:, item=:Tax:,
price=:0.38:, label=:Tax:
Lines: store=:Family Foods:, date=:2014-05-19:, item=:Savings:,
price=:0.86:, label=:Savings:
END: processed 425 data points
```

Total spent at each store (Slide43.awk)

```
#!/usr/bin/gawk -f
@include "csv.awk" # from http://lorance.freeshell.org/csv/
@include "utilities.awk"

BEGIN { #run once before processing lines
        FS=",";
} # BEGIN

FNR == 1 {next} # skip first line

{
        if(NR % 100 == 0) printf("Lines so far (%d)\n", NR);

        num_fields = csv_parse($0, csv, ",", "\"", "\"", "\\n", 0)
        if (num_fields < 0) {
                printf("ERROR: %d (%s) -> %s\n", num_fields, csv_err(num_fields), $0);
                continue;
        }
        totals[csv[1]] += csv[4];

} # for each line

END { # run once after processing lines
                walk_array(totals, "totals", l);
                printf("END: processed %d data points\n",NR);
} # END
```

Slide43.awk output

```
Lines so far (100)
Lines so far (200)
Lines so far (300)
Lines so far (400)
totals[Target] = 306.74
totals[Walmart] = 625.06
totals[Family Foods] = 429.94
totals[Jewel] = 16.02
END: processed 425 data points
```

Turn each line into an array using the column titles as indicies (Slide45.awk) part 1

```
#!/usr/bin/gawk -f
@include "csv.awk" # from http://lorance.freeshell.org/csv/
@include "utilities.awk"

BEGIN { #run once before processing lines
   FS=",";
} # BEGIN

# first line are the titles
   FNR == 1 {
      num_titles = csv_parse($0, titles, ",", "\"", "\"", "\\n", 1)

      if (num_titles < 0) {
         printf("ERROR: %d (%s) -> %s\n", num_titles, csv_err(num_fields), $0);
         exit;
      }
   } # first line

```

Turn each line into an array using the column titles as indicies (Slide45.awk) part 2

```
FNR != 1 {
        if(NR % 100 == 0)
                printf("lines so far (%d)\n", NR);

        num_fields = csv_parse($0, csv, ",", "\"", "\"", "\\n", 0)
        if (num_fields < 0) {
                printf("ERROR: %d (%s) -> %s\n", num_fields, csv_err(num_fields), $0);
                continue;
        }

        for (t in titles)
                Data[titles[t]] = csv[t];
        printf("store=:%s:, date=:%s:, item=:%s:, price=:%s:, categories=:%s:\n",
                Data["store"], Data["date"], Data["item"], Data["price"], Data["categories"]);
}

END { # run once after processing lines
                printf("END: processed %d data points\n",NR);
}
```

Slide45.awk Output

```
…

store=:Family Foods:, date=:2014-05-19:, item=:Water:,
price=:0.49:, categories=:Water:
store=:Family Foods:, date=:2014-05-19:, item=:Water:,
price=:0.49:, categories=:Water:
store=:Family Foods:, date=:2014-05-19:, item=:Salt:,
price=:0.99:, categories=:Salt:
store=:Family Foods:, date=:2014-05-19:, item=:Bread
Crumbs:, price=:2.69:, categories=:Baking:
store=:Family Foods:, date=:2014-05-19:, item=:Garlic:,
price=:0.81:, categories=:Spices:
store=:Family Foods:, date=:2014-05-19:, item=:Tax:,
price=:0.38:, categories=:Tax:
store=:Family Foods:, date=:2014-05-19:, item=:Savings:,
price=:0.86:, categories=:Savings:
END: processed 425 data points
```

Add code to Slide45.awk to get totals

Around line 23, add this to sum the prices

```
totals[Data["store"]] += Data["price"];
```

And in the END section add

```
walk_array(totals,"totals", i);
```

Array of Arrays (Slide49.awk) part 1

```
#!/usr/bin/gawk -f
@include "csv.awk" # from http://lorance.freeshell.org/csv/
@include "utilities.awk"

BEGIN { #run once before processing lines
   FS=",";
   split("",Data); # weird idiom for empty array
   recordCount = 1;
} # BEGIN

   # first line are the titles
   FNR == 1 {
      num_titles = csv_parse($0, titles, ",", "\"", "\"", "\\n", 1)

      if (num_titles < 0) {
         printf("ERROR: %d (%s) -> %s\n", num_titles, csv_err(num_fields), $0);
         exit;
      }
   } # first line
```

Array of Arrays (Slide49.awk) part 2

```
FNR != 1 {
    if(NR % 100 == 0)
        printf("lines so far (%d)\n", NR);

    num_fields = csv_parse($0, csv, ",", "\"", "\"", "\\n", 0)
    if (num_fields < 0) {
        printf("ERROR: %d (%s) -> %s\n", num_fields, csv_err(num_fields), $0);
        continue;
    }

    for (t in titles) {
        Data[recordCount][titles[t]] = csv[t];
    }
    #printf("store=:%s:, date=:%s:, item=:%s:, price=:%s:, categories=:%s:\n",
    #  Data[recordCount]["store"], Data[recordCount]["date"], Data[recordCount]["item"], Data[recordCount]["price"], Data[recordCount]["categories"]);
    recordCount++;
  }
```

Array of Arrays (Slide49.awk) part 1

```
END { # run once after processing lines
        #walk_array(Data, "Data", i);
        for (r in Data) {
            #printf("index=:%d:, store=:%s:\n", r, Data[r]["store"]);
            totals[Data[r]["store"]] += Data[r]["price"];
        }
        walk_array(totals, "totals", i);
        printf("END: processed %d data points\n",NR);
    }
```

CSV to JSON Example

Translates CSV file into JSON format
For example:

```
2014-7-1,77,60,94,54,73,45,94,1948,2014,0.00,0.03,0.75
{
"date" : "2014-7-1"
,"actual_mean_temp" : "77"
,"actual_min_temp" : 60
,"actual_max_temp" : "94"
,"average_min_temp" : 54
,"average_max_temp" : "73"
,"record_min_temp" : 45
,"record_max_temp" : "94"
,"record_min_temp_year" : 1948
,"record_max_temp_year" : "2014"
,"actual_precipitation" : 0
,"average_precipitation" : "0.03"
,"record_precipitation" : 0
}
```

```
#!/usr/bin/gawk -f
@include "csv.awk" # from http://lorance.freeshell.org/csv/
# https://stackoverflow.com/questions/9985528/how-can-i-trim-white-
space-from-a-variable-in-awk
function ltrim(s) { sub(/^[ \t\r\n]+/, "", s); return s }
function rtrim(s) { sub(/[ \t\r\n]+$/, "", s); return s }
function trim(s) { return rtrim(ltrim(s)); }

BEGIN { #run once before processing lines
		#print "START"
		lines = 0;
		# formats is from the command line
		# ./csvToJson.awk -v
formats="ssssssssssssssssdsssssddddssssssssssssssss"
		# < ../data/MediumVoterData.csv
		nf = split(formats,formatList,"")
		supportedFormats = "sdf"; # to check for errors
		formatStrings["s"] = "\"%s\" : \"%s\"\n";
		formatStrings["d"] = "\"%s\" : %d\n";
		formatStrings["f"] = "\"%s\" : %f\n";
		print "["; # start of JSON array
}
```

```
{
      if(lines++ == 0) { # first line is titles
         num_titles = csv_parse($0, titles, ",", "\"",
"\"", "\\n", 1)
         next;
      }
      else {
         num_fields = csv_parse($0, csv, ",", "\"", "\"",
"\\n", 0)
         if (num_fields < 0) {
            printf("ERROR: %d (%s) -> %s\n", num_fields,
csv_err(num_fields), $0);
            next;
         }
         if(lines > 2) print(","); # row seperator
         printf "{"; # start of JSON object
         for (i=1; i <= length(titles); i++) {
            if(i > 1) printf(","); # field seperator
            format =
(index(supportedFormats,formatList[i]) != 0) ?
formatStrings[formatList[i]] : formatStrings["s"];
           gsub(/\"/,"",csv[i]); # remove quotes
            finalValue = trim(csv[i]); #remove spaces
            printf(format, titles[i], finalValue);
         }
         print "}"; # end of JSON object
   } # all other lines
}
```

```
END   { # run once after processing lines
     print "]"; # end of JSON array
     #printf("END: processed %d lines\n",lines-
1);
}
```

Questions?

CONTACT:
kentarchie@gmail.com