Article
Murray Oldfield · Apr 1, 2016 3m read

Extracting pButtons data to a csv file for easy charting

A short post for now to answer a question that came up. In post two of this series I included graphs of performance data extracted from pButtons. I was asked off-line if there is a quicker way than cut/paste to extract metrics for mgstat etc from a pButtons .html file for easy charting in Excel.

See: - Part 2 - Looking at the metrics we collected

pButtons compiles data it collects into a single html file to make it easier to send to WRC and review the collated data. However, especially for pButtons run over long collection times like 24 hours, some of the time based data like mgstat, vmstat etc is easier to review graphically to look for trends or patterns.

I know it sounds crazy to roll up pButtons data into an html file then spend time unpacking it… but remember that pButtons is tool for WRC to grab a view of many system metrics to help trouble-shoot performance problems. The system level metrics and Caché metrics can be run individually, but it is convenient for me to use pButtons in this series to capture and analyse performance metrics because I know that all Caché installations will have a copy - or can download a copy - and all the basic metrics are available for different operating systems in a single file. It is also convenient for you to be able to capture these metrics every day with one simple routine if you are not collecting the data any other way.

> ***Feb 2017. I have rewritten the scripts in this article in Python and added charting including interactive html.*** I think the Python utilities are much more useful . Please see Yape - Yet another pButtons extractor (and automatically create charts)

Extracting pButtons performance metrics to a csv file

Because I use an Apple laptop I have the Unix operating system, so its natural to write a quick shell script to extract the data to a csv file. The following script extracts mgstat, vmstat or Windows Performance Monitor data from a pButtons .html file. The example below uses Perl which is installed on most *nix systems, but there are endless possibilities using other scripting languages or powershell on Windows.

I will show you how I do the extraction so you have all the information you to do the same with your favourite tools. The key is that the html file has markers in the file to delimit the metrics. For example mgstat is bracketed by:

```
<!-- beg_mgstat -->
```

and

```
<!-- end_mgstat -->
```

In the mgstat section there is some other descriptor information, followed by the heading line of the mgstat output. There are similar markers for vmstat and winperfmon.

This simple script simply looks for the beginning marker then outputs everything from the header line to the line before the end marker.

```
#!/usr/bin/perl

# extract_pButtons.pl - Simple extractor for pButtons

# usage: ./extract_pButtons.pl <input pButtons> <search start> <search first line output>

# pButtons has the following markers in the html source
# Metrics                       Parameters to pass
# --------                      -------------------
# mgstat                        mgstat Date
# windows performance monitor   win_perfmon Time
# vmstat                        vmstat fre

# usage example - Search for mgstat and redirect to .csv file
# ./extract_pButtons.pl DB1_20160211_0001_24Hour_5Sec.html mgstat Date > myMgstatOutput.csv

# usage example - Process a set of html files
# for i in $(ls *.html); do ./extract_pButtons.pl ${i} vmstat fre > ${i}.vmstat.csv ; done

# usage example - Pipeline to add commas
# ./extract_pButtons.pl P570A_CACHE_20150418_0030_day.html vmstat fre | ./make_csv.pl >P570A_CACHE_20150418_0030_day.html.vmstat.csv

$filename=$ARGV[0];
$string=$ARGV[1];
$firstLine=$ARGV[2];

$searchBeg="beg_".$string;
$search2=$firstLine;
$foundEnd="end_".$string;

$foundString=0;
$printIt=0;
$break=0;

open HTMLFILEIN, "<".$filename or die "Bad input file";

while (<HTMLFILEIN>) {

    if (/$searchBeg/) {
            $foundString=1;
        }

    # Look for first actual line - use something on header line
    if (($foundString==1) && (/$search2/)) {
            $printIt=1;
        }

     # No more data
    if (/$foundEnd/) {
            $break=1;
        }

    if ($break==0) {
        if ($printIt==1) {
            print;
        }
    }

}

close HTMLFILEIN;
```

As shown in the # comments at the start of the script extractpButtons.pl can either output data to the screen or redirect output to a csv file or use pipelining in a longer workflow, for example to charting utilities. I use open source gnuplot, but excel is OK as well.

Add commas to space delimited text file

The following short perl script is handy to turn the output of vmstat or other text file to a comma delimited file for easier processing.

```
#!/usr/bin/perl

# Convert space delimited text file to csv

# Usage example 1:
# Will create backup file vmstat.csv.bak and original file called vmstat.csv will be
updated
# ./make_csv.pl freecnt.csv


# Usage example 2:
# No backup, original vmstat.txt file stays same, new output csv file
# ./make_csv.pl < vmstat.txt >freecnt.csv


use strict;

# create .bak backup file for each change
$^I = ".bak";


while (<>) {
    # remove leading blanks; substitute 1 or more blanks for a single comma
    s/^ +//;s/ +/,/g;
    print;
 }
```

Summary

I encourage you to look at the .html source of the pButtons file to get an idea of what it contains. There is more than just the system metrics. For example you will see a list of the commands run by pButtons and version information at the top of the .html file.

If you use a windows script for extracting or charting data or have a better or different workflow I encourage you to share with a post to the Developer Community.

#Best Practices #CSV #Performance #System Administration #InterSystems IRIS

Source URL:https://community.intersystems.com/post/extracting-pbuttons-data-csv-file-easy-charting