Add FileWave Custom Inventory fields remotely using a Fileset

Description

⚠ Although these forms of Custom Field may still be used, since FileWave 12.7 additional new methods of Custom Fields have been introduced. The newer form allow for backward compatibility to these older Custom Fields, but provide much greater flexibility and would be recommended. Details on these may be found here:

- 8.9. Custom Fields

Let's say you are using FileWave Integrated Inventory and you want to collect inventory data fields that FileWave does not report yet (you can check if you field is supported in the component list when you edit a query). The solution is to use the custom inventory fields introduced in FileWave version 7.0.0. There are 20 fields for each data type : String, Integer, Boolean, Date. Starting in version 8.5 the FileWave Client can accept input.

You can use this recipe to deploy a Fileset that is able to add the value you want based on a shell command. The script is already written for you and available in the Fileset, you only need to provide the command that outputs the value you want to add.

Usage
`$ fwcld -custom_write -key <key_name> [-value <value_to_save] [-silent]`

Ingredients

- FW Admin
- A command that outputs needed infrmation
- Basic understanding of scripting

Directions

Built-into the FileWave Client is the ability to receive the output of a command and save that to the inventory DB.

1. Determine what the output of your command is and the best place to save

String	Integer	Boolean	DateTime
`custom_string_01`	`custom_integer_01`	`custom_bool_01`	`custom_datetime_01`
`custom_string_02`	`custom_integer_02`	`custom_bool_02`	`custom_datetime_02`
`custom_string_03`	`custom_integer_03`	`custom_bool_03`	`custom_datetime_03`
`custom_string_04`	`custom_integer_04`	`custom_bool_04`	`custom_datetime_04`
`custom_string_05`	`custom_integer_05`	`custom_bool_05`	`custom_datetime_05`
`custom_string_06`	`custom_integer_06`	`custom_bool_06`	`custom_datetime_06`
`custom_string_07`	`custom_integer_07`	`custom_bool_07`	`custom_datetime_07`
`custom_string_08`	`custom_integer_08`	`custom_bool_08`	`custom_datetime_08`
`custom_string_09`	`custom_integer_09`	`custom_bool_09`	`custom_datetime_09`
`custom_string_10`	`custom_integer_10`	`custom_bool_10`	`custom_datetime_10`

custom_string_11	custom_integer_11	custom_bool_11	custom_datetime_11
custom_string_12	custom_integer_12	custom_bool_12	custom_datetime_12
custom_string_13	custom_integer_13	custom_bool_13	custom_datetime_13
custom_string_14	custom_integer_14	custom_bool_14	custom_datetime_14
custom_string_15	custom_integer_15	custom_bool_15	custom_datetime_15
custom_string_16	custom_integer_16	custom_bool_16	custom_datetime_16
custom_string_17	custom_integer_17	custom_bool_17	custom_datetime_17
custom_string_18	custom_integer_18	custom_bool_18	custom_datetime_18
custom_string_19	custom_integer_19	custom_bool_19	custom_datetime_19
custom_string_20	custom_integer_20	custom_bool_20	custom_datetime_20

2. Create a fileset with a script where it uses the clients

macOS Client

```
/usr/local/sbin/FileWave.app/Contents/MacOS/fwcld -custom_write -key FIELD_TO_SAVE_TO -value
INFORMATION_TO_SAVE
```

Windows

```
C:\Program Files (x86)\FileWave\fwcld -custom_write -key FIELD_TO_SAVE_TO -value INFORMATION_TO_SAVE
```

3. Associate this fileset

Examples

Setting "custom_bool_13" to a false:

> *$ fwcld -custom_write -key custom_bool_13 -value 0*
>
> *$ fwcld -custom_write -key custom_bool_13 -value false*

Setting "custom_bool_13" to true:

> *$ fwcld -custom_write -key custom_bool_13 -value 1*
>
> *$ fwcld -custom_write -key custom_bool_13 -value true*
>
> *$ fwcld -custom_write -key custom_bool_13 -value something*

Setting "custom_date_02" to a date:

> *$ fwcld -custom_write -key custom_date_02 -value 2014-02-20T15:22:43*

To remove any key value, just leave off the -value parameter - so to reset the "custom_date_02" value back to it's default.

> *$ fwcld -custom_write -key custom_date_02*

Example: Saving admins to string 01

```
#!/bin/sh
# This script is a verification sample
# benm @ fw

now=$(date +"%Y-%m-%d-%H-%M")
echo "$now -- Writing current admins to inventory"

#writes the current administrators to an inventory field
currentadmins=$(dscacheutil -q group -a name admin |grep users)

/usr/local/sbin/FileWave.app/Contents/MacOS/fwcld -custom_write -key custom_string_01 -value "$currentadmins"
```

Notes

If you set your script to run at the "verification" phase then it will continue to run (default every 24hrs), for more on scripts see: 5.11. Fileset Scripts.

Useful Commands

- The current logged in user: stat -f%Su /dev/console
- The Kernel version: uname -r
- Battery Condition: system_profiler SPPowerDataType | awk '/Condition/ {print $NF}'
- Current admins: dscl . read /Groups/admin GroupMembership | cut -d " " -f 2-