uTermCommand: Poking the MOOSDB with Pre-Set Values

June 2018

Michael Benjamin, mikerb@mit.edu
Department of Mechanical Engineering
MIT, Cambridge MA 02139
project-pavlab/appdocs/app_utermcommand

1 Overview

The uTermCommand application is a terminal based tool for poking the MOOS database with pre-defined variable-value pairs. This can be used for command and control for example by setting variables in the MOOSDB that affect the behavior conditions running in the helm. There are a few other ways of doing this:

- *pMarineViewer*: The action pull-down menu and on-screen buttons may be used for posting the MOOSDB.
- *uPokeDB*: A command-line tool for poking the MOOSDB. This may be used in conjunction with shell aliases or shell scripts for further convenience.
- *iRemote*: The custom keys feature may be used to bind variable-value pairs to the numeric keys. The primary drawback is the limitation to ten mappings.

2 Configuration Parameters for uTermCommand

The variable-value mappings are set in the uTermCommand configuration block of the MOOS file. Each mapping requires one line of the form:

cmd

```
cmd = cue --> variable --> value
```

The *cue* and *variable* fields are case sensitive, and the *value* field may also be case sensitive depending on how the subscribing MOOS process(es) handle the value. An example configuration is given in Listing 1.

Listing 2.1: An example uTermCommand configuration block.

```
//------------------------------------------------------------
// uTermCommand configuration block

ProcessConfig = uTermCommand
{
  cmd = override_true  -->  MOOS_MANUAL_OVERRIDE --> true
  cmd = override_false -->  MOOS_MANUAL_OVERRIDE --> false
  cmd = deploy_true    -->  DEPLOY       --> true
  cmd = deploy_false   -->  DEPLOY       --> false
  cmd = return_true    -->  RETURN       --> true
  cmd = return_false   -->  RETURN       --> false
}
```

Recall the type of a MOOS variable is either a string, double or binary data. If a variable has yet to be posted to the MOOSDB, it accepts whatever type is first written, otherwise postings of the wrong type are ignored. If quotes surround the entry in the value field, it is interpreted to be a string. If not, the value is inspected as to whether it represents a numerical value. If so, it is posted as a double. For example `true` and `"true"` are the same type (no such thing as a Boolean type), `25` is a double and `"25"` is a string.

2.1 Run Time Console Interaction

When uTermCommand is launched, a separate thread accepts user input at the console window. When first launched the entire list of cues and the associated variable-value pairs are listed. Listing 2 shows what the console output would look like given the configuration parameters of Listing 1. This configuration block is in the Alpha example mission. You can launch that mission and then launch uTermCommand:

```
$ cd moos-ivp/ivp/missions/s1_alpha
$ ./launch.sh
```

Then in a separate terminal:

```
$ cd moos-ivp/ivp/missions/s1_alpha
$ uTermCommand alpha.moos
```

The output in the terminal window should look similar to Listing 2. Note that even though quotes were not necessary in the configuration file to clarify that `true` was to be posted as a string, the quotes are always placed around string values in the terminal output.

Listing 2.2: Console output at start-up.

```
  Cue                     VarName                         VarValue
  -----------             ---------------                 --------------
  override_true           MOOS_MANUAL_OVERRIDE            "true"
  override_false          MOOS_MANUAL_OVERRIDE            "false"
  deploy_true             DEPLOY                          "true"
  deploy_false            DEPLOY                          "false"
  return_true             RETURN                          "true"
  return_false            RETURN                          "false"
```

```

>
```

In the Alpha mission, the `DEPLOY` button in the lower right corner of pMarineViewer is configured to post:

```
MOOS_MANUAL_OVERRIDE = false
DEPLOY = true
```

This will be handled instead by uTermCommand in our simple example. A prompt is shown on the last line where user key strokes will be displayed. As the user types characters, the list of choices is narrowed based on matches to the cue. After typing a single 'o' character, only the `override_true` and `override_false` cues match and the list of choices shown are reduced as shown in Listing 3. At this point, hitting the TAB key will complete the input field out to `override_`, much like tab-completion works at a Linux shell prompt.

Listing 2.3: Console output after typing a single character 'r'.

```
  Cue                     VarName                        VarValue
  -----------             ---------------                --------------
  override_true           MOOS_MANUAL_OVERRIDE           "true"
  override_false          MOOS_MANUAL_OVERRIDE           "false"

> o
```

When the user has typed out a valid cue that matches a single entry, only the one line is displayed, with the tag `<-- SELECT` at the end of the line, as shown in Listing 4.

Listing 2.4: Console output when a single command is identified.

```
  Cue                     VarName                       VarValue
  -----------             ---------------               --------------
  override_false          MOOS_MANUAL_OVERRIDE          "false" <-- SELECT

> override_false
```

At this point hitting the ENTER key will execute the posting of that variable-value pair to the MOOSDB, and the console output will return to its original start-up output. A local history is augmented after each entry is made, and the up- and down-arrow keys can be used to select and re-execute postings on subsequent iterations. To finish the launch in the Alpha mission, use uTermCommand to post `DEPLOY=true`.

3 Connecting uTermCommand to the MOOSDB Under an Alias

A convention of MOOS is that each application connected to the MOOSDB must register with a unique name. Typically the name used by a process to register with the MOOSDB is the process name, e.g., uTermCommand. One may want to run multiple instances of uTermCommand all connected to the same MOOSDB. To support this, an optional command line argument may be provided when launching uTermCommand:

```
$ uTermCommand file.moos --alias=uTermCommandAlpha
```

The command line argument may also be invoked from within pAntler to launch multiple uTermCommand instances simultaneously. The configuration block in the mission file needs to have the same name as the launch alias.

4 Publications and Subscriptions for uTermCommand

The only variables published by uTermCommand are those configured and selected by the user at run-time, and uTermCommand does not subscribe for any variables.