

# Advanced DTT

Brian O'Reilly  
*LIGO Livingston Observatory*

# DTT Command Line

- Allows non-GUI interaction with DTT.
- Many of you will be familiar with this feature because you have used it to clear testpoints “tp clear 0 \*”.
- I’m going to talk about how you can script repetitive or similar DTT tasks.
- The **AutoCalibrator** is an example of scripted DTT.

# Languages

- I'm going to talk about how to script using EXPECT.
- In principle any scripting language should work, some better than others.
- You could also (in theory?) use ROOT macros, or compiled C++.
- Even if you don't plan to write scripted DTT utilities it is a good idea to learn at least one scripting language.

# Expect

- <http://expect.nist.gov>
- Advantages:
  - Easy to get started, fairly intuitive commands
  - Useful for complex repeated tasks
  - Interfaces easily to Tcl/Tk
- Disadvantages?:
  - Poor regular expression interface.
  - Obscure errors.
  - Too complicated (depends on the application).
- There's even a book!

*A Tcl-based Toolkit for Automating Interactive Programs*



*Exploring*

Expect



*Don Libes*

# Basic Expect Commands

- set <variable> <value> : define variables.
- send <command> : send a command to a process.
- expect <string>: expect a response from a process.
- puts <string>: write output.
- set timeout <value>

# Basic DIAG Commands

- restore <filename>
- save <filename>
- read <filename>
- run
- Exit
- get \* or get Test.\* (thanks Daniel!)
- More info. in Appendix B of the Manual.

# A Simple Script

```
#!/usr/local/bin/expect
set timeout 5
while 1 {
    expect -nocase "Hello\n" {
        send "Hello World\n"
    } default {
        puts "Goodbye!"
        exit
    }
}
```

# Scripted DTT

- A more complicated example:
  - See `make_dtt` script in the tarball.
- Some fields names: **Test.**
  - StimulusChannel[#] StartFrequency  
StopFrequency Averages BW  
StimulusFrequency[#]  
StimulusFrequencyRange[#]  
StimulusAmplitude[#] StimulusFilter[#]}

# Simpler Approach?

- You can use the **read 'filename'** command from the DTT command line.

```
restore template.xml
```

```
set Test.MeasurementChannel[0] = "H1:LSC-AS_Q"
```

```
get Test.MeasurementChannel[0]
```

```
save test.xml
```

# Other Approaches

- You heard/saw many examples during this camp on how to interact with the data.
- You could even use C++.
- What is it all good for?

# A Suggested Change

- Currently much of the Science Run use of DTT involves:
  - template files (to perhaps take power spectra).
  - Compare these to a reference.
  - Hasn't always worked well.
- We could use a different approach.

# Charge?

- Develop smoother(!) interfaces that allow users to select GPS times, acquire and filter data, and produces at the end a set of plots.
- We should avoid information overload if possible.
- This camp is full of examples of tools which could be readily adapted to such an effort.
- An ideal task for people who want to contribute to on-site activities remotely.
- **BUT** Authors should look at the data.