ESP Threads

5/20/22 Brent Roman   brent@mbari.org

Copyright MBARI 2022

What is an ESP Thread

- A series of computational steps
- Enhancement of Ruby’s built-in Threads
  - Adds Naming of threads
    - may be :symbol, “string” or number
  - tracking of parents and child threads
  - time of birth
- Many ESP protocols run as multiple threads
  - to perform steps in parallel

```
-> threads    #lists the state of all active threads
```

Copyright MBARI 2022

Thread Basics

- Threads may be referenced by their name

```
-> Thread["mary"]           #shows the thread's state
-> Thread["mary"].finish    #wait for "mary" to exit
-> Thread["mary"].abort     #abort "mary" with error
-> Thread["mary"].exit      #exit "mary" with no error
```

- Parent Threads are notified of Error if a child thread dies
  - Detaching a thread orphans it from all parent thread(s)
- Exiting MainThread causes ESP server to exit

```
-> ESP.main.exit    #or MainThread.exit
```

- Each espclient *name*
  - creates a corresponding Thread[*name*]
  - Exiting that Thread exits the client

Copyright MBARI 2022

Starting new Threads

```
-> Thread(name){code}
```

- Run *code* in a new thread called *name*

```
-> Thread(:myDA){shortDA}
```

- Run shortDA in a new thread called :myDA
  - with the client thread as its controlling parent!
  - so, exiting the client will likely cause shortDA to fail

```
-> Thread[:myDA].finish    #wait for shortDA to finish

-> Thread[:myDA].abort     #abort shortDA
```

- note that “myDA” != :myDA

```
-> threads    #lists names of all active threads
```

Copyright MBARI 2022

Starting Detached Threads

- Detached threads
  - Have no parent threads
  - Continue running when client exits for any reason
  - Make little sense unless started from espclient
- To start a block of code in a detached thread

```
-> start(:myShortDA) {shortDA}
```

- You may substitute any code for `shortDA` above

```
-> Thread[:myShortDA].abort   #aborts myShortDA
```

- To start a mission script (safely detacted) within a client

```
-> runMission "script" #searches $ESPpath
```

  - Mission thread will be named "*script_mission*"

Copyright MBARI 2022

Threads in Simulated Time

- All synchronized threads advance Thread.time
  - before simulated time can advance

```
-> Thread.unsync    #allows time to flow by this thread
-> Thread.resync    #forces time to wait for this thread
```

- You must unsync espclient threads
  - to allow other threads to run free in simulated time

```
    -> Thread.unsync
    -> start(:myShortDA) {shortDA}
```

- New threads (or clients) start ‘synchronized’

```
-> Thread.unsync    #until you unsync them
```

Copyright MBARI 2022