ScopingExperiment

September 19, 2016

1 Scoping Experiment

This experiment is designed to show a pecularity in Julia'd scoping. Julia uses a layered scoping where the scope of the inner function has access to the values of the outer function. For example:

```
In [1]: x=5; y=7; #Defined globally
        function scopeTest(z)
          x += z #Changes global value
          y = Vector{Float64}(1) #Declares a variable, local scope
          y[1] = 2
          return x + y + z
        end
```

```
Out[1]: scopeTest (generic function with 1 method)
```

However, what is happening here, and why?

```
In [37]: addprocs(1)
         function f1()
           @parallel for i = 1:100
             x = 10
             if x < 100
               x = x + 1
             end
           end
           x = x + 100 + 10
           return x
         end
         f1()
```

```
WARNING: Method definition f1() in module Main at In[36]:3 overwritten at In[37]:3.
```

```
        LoadError: UndefVarError: x not defined
    while loading In[37], in expression starting on line 12
```

```
in f1() at ./In[37]:3

in execute_request(::ZMQ.Socket, ::IJulia.Msg) at /home/crackauc/.julia/v0

in eventloop(::ZMQ.Socket) at /home/crackauc/.julia/v0.5/IJulia/src/eventl

in (::IJulia.##9#15)() at ./task.jl:360
```

In [38]:
```
function f2()
  @parallel for i = 1:100
    x = 10
    if x < 100
      x = x + 1
    end
  end
  return x
end
f2()
```

```
WARNING: Method definition f2() in module Main at In[35]:2 overwritten at In[38]:2.
```

Out[38]:
```
5
```