Fiber

Listing 1: aufgaben.rb

```
einmaleins = Fiber.new do
  for x in 1..10 do
    for y in 1..10 do
      frage = "#{x} x #{y} = "
      Fiber.yield frage,x*y
    end
  end
end

20.times {
  f,a = einmaleins.resume
  print f
  answer = gets
  if answer.to_i == a
    puts "OK"
  else
    puts "falsch"
```

Fiber

Listing 2: Ergebnis

```
$ ruby aufgaben.rb
1 x 1 = 1
OK
1 x 2 = 2
OK
1 x 3 = 3
OK
1 x 4 = 4
OK
1 x 5 = 5
OK
1 x 6 = 6
OK
1 x 7 = 7
OK
1 x 8 = 8
OK
```

Fiber

Fiber

- ähnlich wie Threads
- aber Programmautor legt fest, wann und wo Rückkehr ins aufrufende Programm erfolgt
- auch Semi-Coroutinen genannt
- mittels `require 'fiber'` auch Transfer zu anderen Fiber möglich

Aufruf externer Programme

Aufruf externer Programme

- `puts ‘tar archiv.tar /home/tp/sourcecode‘`
- oder `system(’tar archiv.tar /home/tp/sourcecode’)`
- Fehlercode des Programms in globaler Variable `$?`

IO.popen

IO.popen

Listing 3: popen-beispiel.rb

```
programm = IO.popen("cat", "w+")
programm.puts "Programm-Input"
programm.close_write
puts programm.gets
```

unabhängige Subprozesse

unabhängige Subprozesse

- exec
- fork