Controlling Process Execution

Once the program stops at a breakpoint, it transfers control to the debugger. The developer can then interact with the debugger to step through instructions and either continue execution or quit.

Stepping In, Out, and Over Function Calls

Use the `thread step-over` (`next` or `n`) command to evaluate the current line and continue to the next line.

If the debugger is stopped at a function call, you can use the `thread step-in` (`step` or `s`) command to step inside the function and continue there. If the debugger is not stopped at a function call, the `thread step-in` command has the same effect as the `thread step-over` command.

The `thread step-out` (`finish`) command performs the inverse operation of the `thread step-in` command by continuing execution until the next function return statement is called, or the stack frame is popped, and stepping out of the current function to where it was called.

```
(lldb) thread step-over

Process 97209 stopped

* thread #1: tid = 0x1288be3, 0x00000001000019bd Greeter`main + 109 at
Greeter.swift:20, queue = 'com.apple.main-thread', stop reason = step over
    frame #0: 0x00000001000019bd Greeter`main + 109 at Greeter.swift:20

   17

   18      let greeter = Greeter()

   19

-> 20      greeter.greet(personNamed: "Anton")

   21      greeter.greet(personNamed: "Mei")

   22      greeter.greet(personNamed: "Anton")

(lldb) thread step-in

Process 97209 stopped

* thread #1: tid = 0x1288be3, 0x0000000100001bab Greeter`Greeter.greet(name="Anton",
self=0x0000000100606b10) -> () + 27 at Greeter.swift:9, queue = 'com.apple.main-
thread', stop reason = step in

    frame #0: 0x0000000100001bab Greeter`Greeter.greet(name="Anton",
self=0x0000000100606b10) -> () + 27 at Greeter.swift:9

   6           }

   7

   8           func greet(personNamed name: String) {

-> 9               if hasMet(personNamed: name) {

   10                  print("Hello again, \(name)!")

   11              } else {

   12                  acquaintances.insert(name)

(lldb) thread step-out

Process 97209 stopped

* thread #1: tid = 0x1288be3, 0x00000001000019bd Greeter`main + 109 at
Greeter.swift:20, queue = 'com.apple.main-thread', stop reason = step out
    frame #0: 0x00000001000019bd Greeter`main + 109 at Greeter.swift:20

   17

   18      let greeter = Greeter()
```

```
   19
-> 20      greeter.greet(personNamed: "Anton")
   21      greeter.greet(personNamed: "Mei")
   22      greeter.greet(personNamed: "Anton")
```

Continuing Execution

Once you are finished inspecting the program at the current breakpoint, you can resume execution of the program using the `process continue` (`continue` or `c`) command.

```
(lldb) process continue
Resuming thread 0x12f1d19 in process 97209
Process 97209 resuming
```