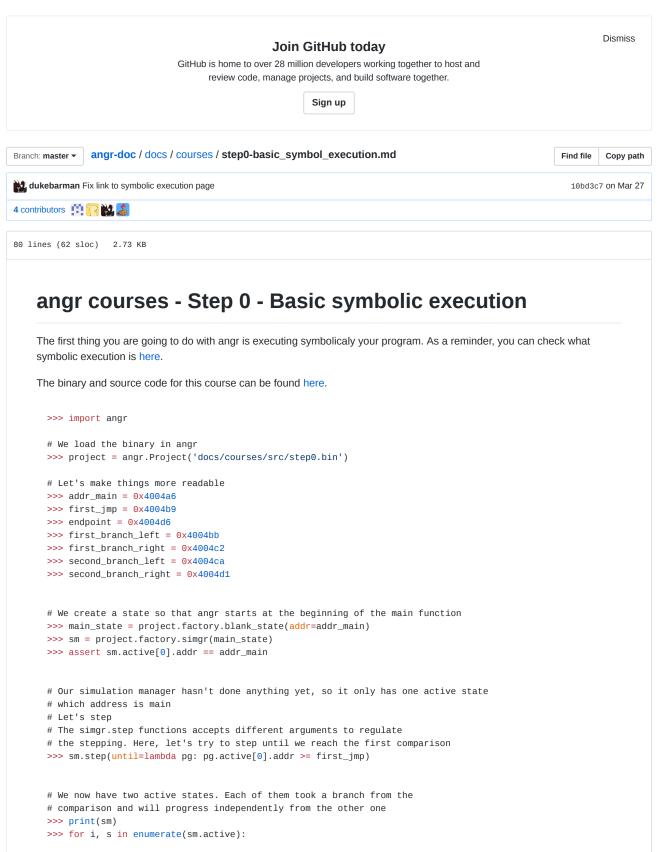
## angr / angr-doc



1 of 2 8/19/18, 5:27 PM

```
print 'Active state %d: %s' % (i, hex(s.addr))
>>> assert len(sm.active) == 2
>>> assert sm.active[0].addr == first_branch_left
>>> assert sm.active[1].addr == first_branch_right
# If we make the first step, it will continue until reaching the endpoint
# The other one, however, will reach another comparison and should
# split again
>>> sm.step()
>>> print(sm)
>>> for i, s in enumerate(sm.active):
... print 'Active state %d: %s' % (i, hex(s.addr))
>>> assert len(sm.active) == 3
>>> assert sm.active[0].addr == endpoint
>>> assert sm.active[1].addr == second_branch_left
>>> assert sm.active[2].addr == second_branch_right
# Good, we now have three states
\# - The two first states reached the endpoint, and became unconstrained, since
# we started executing directly at main function. We would have seen these 2 states
# if we had enabled save_unconstrained option of our SimulationManager.
# - The other one will have the same history thus stop stepping at the endpoint
>>> sm.step()
>>> print(sm)
>>> for i, s in enumerate(sm.active):
... print 'Active state %d: %s' % (i, hex(s.addr))
>>> assert len(sm.active) == 1
>>> assert sm.active[0].addr == endpoint
# The same effect can be done by using simgr.explore()
# The explorer will step every state until no more states are active
>>> sm = project.factory.simgr(main_state)
>>> sm.explore()
>>> assert len(sm.active) == 0
```

2 of 2 8/19/18, 5:27 PM