

PyCharm

z

# Debugger





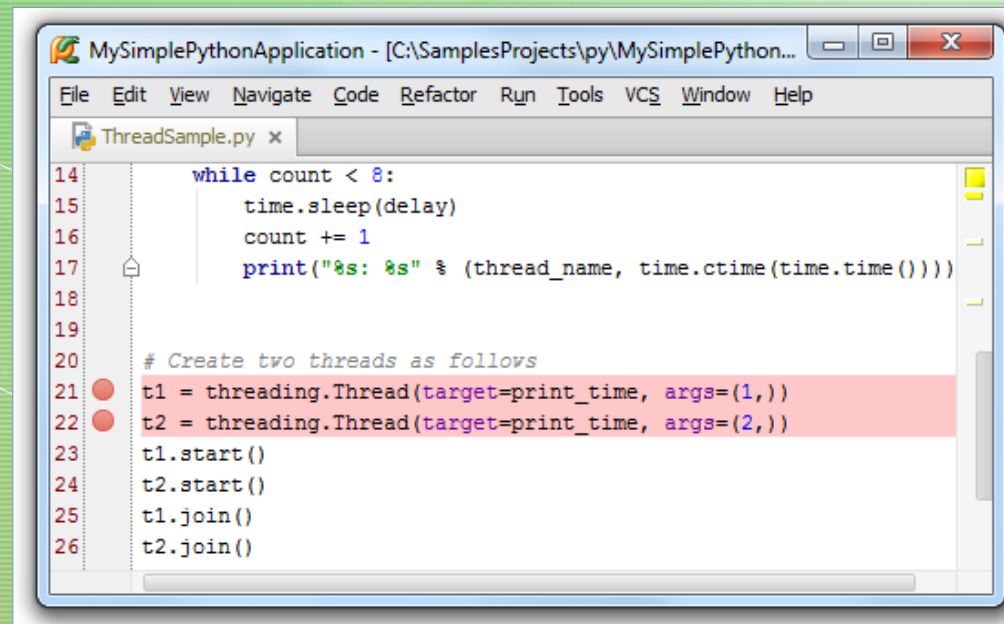






# z z set breakpoints

- Set breakpoints in the source code.
- left gutter on the lines you want PyCharm to pause at while debugging:



The screenshot shows a PyCharm IDE window titled "MySimplePythonApplication - [C:\SamplesProjects\py\MySimplePython...". The editor displays a Python file named "ThreadSample.py" with the following code:

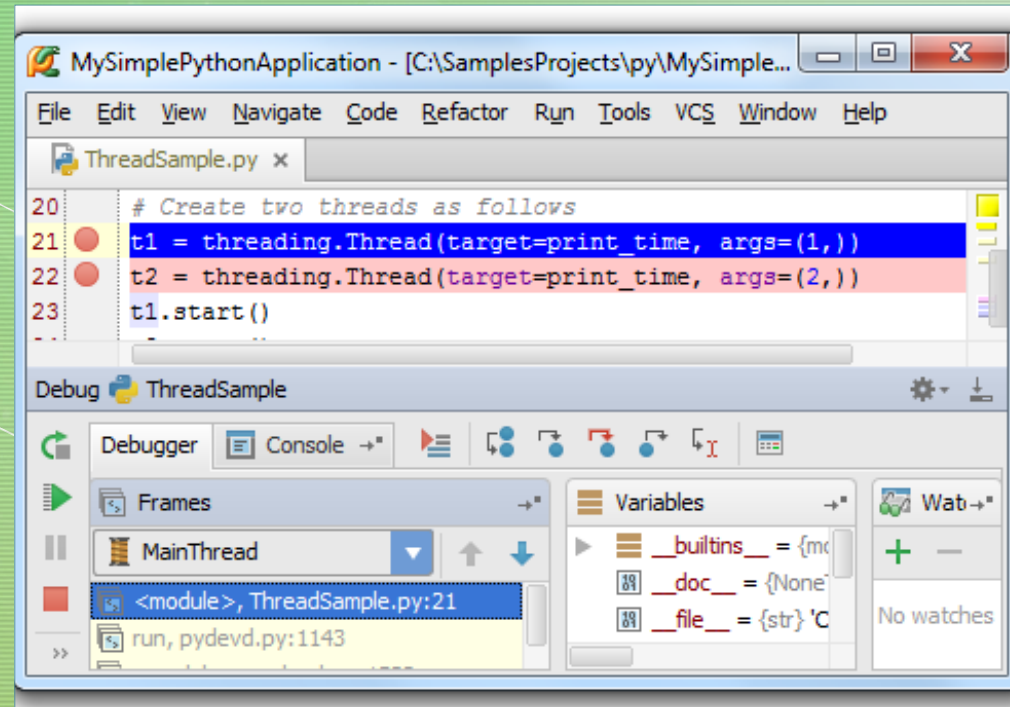
```
14     while count < 8:
15         time.sleep(delay)
16         count += 1
17         print("%s: %s" % (thread_name, time.ctime(time.time())))
18
19
20     # Create two threads as follows
21     t1 = threading.Thread(target=print_time, args=(1,))
22     t2 = threading.Thread(target=print_time, args=(2,))
23     t1.start()
24     t2.start()
25     t1.join()
26     t2.join()
```

Red circular markers in the left gutter indicate that breakpoints are set on lines 21 and 22. The code on these lines is highlighted in pink.

# z z Debugging session

- Select run/debug configuration "YourThread", and then press Shift+F9 (or click on the toolbar). The debugger will start, and pauses at the first breakpoint:

- The line will be marked blue. It means that PyCharm hit this line, but not yet executed it.





# Stepping through the code



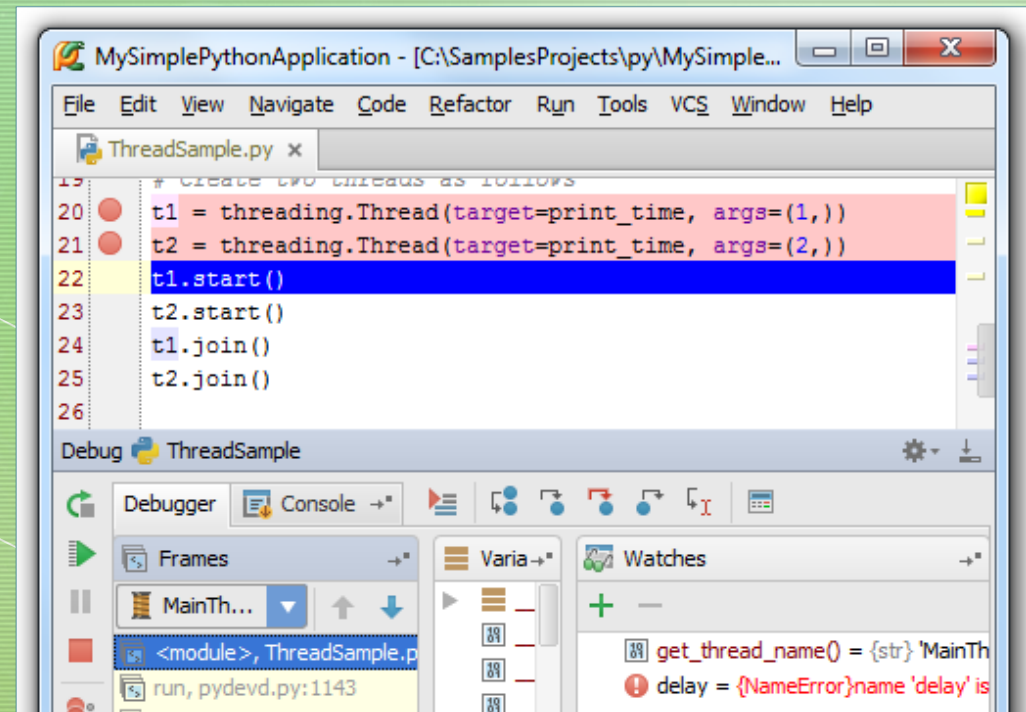
Step into the method called at the current execution point.



Step into only your method called at the current execution point.

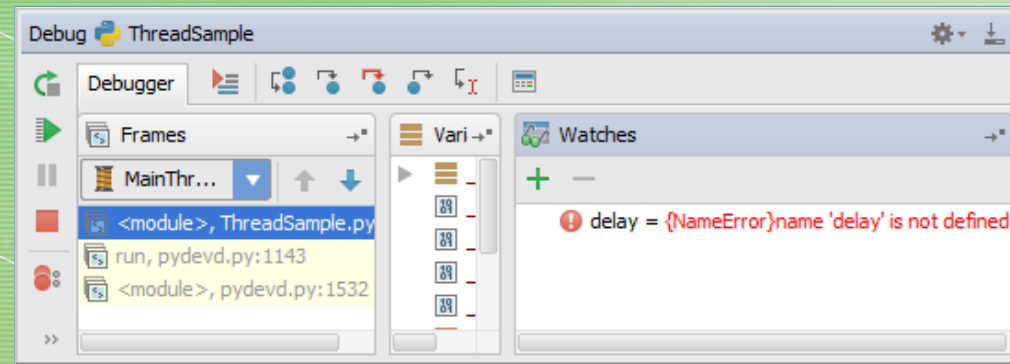


Click this button to execute the program until the next line in the current method or file, skipping the methods referenced at the current execution point



# z z Watches pane

- In the Watches pane you can evaluate any number of variables or expressions in the context of the current stack frame. The values are updated with each step through the application, and become visible every time the application is suspended.





# Variables pane

- The Variables pane enables you to examine the values stored in the objects of your application.

