Breakpoint Troubleshooting

Josh Paulson
June 18, 2015 21:52

Breakpoint Troubleshooting

Setting breakpoints

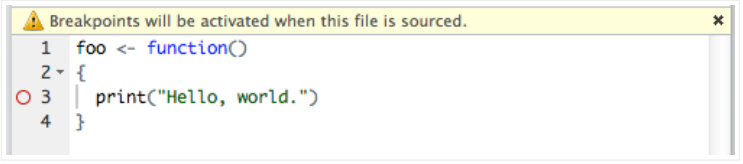
Nothing happens when clicking in the gutter

Breakpoints cannot be set anywhere; the most likely cause of this problem is that you're trying to set a breakpoint on a line that doesn't support breakpoints. Lines on which you can set breakpoints are those which:

- are in R code (.R file);
- contain an R statement (i.e. are not just comments or whitespace); and
- are a top-level expression or a simple, named function

A warning bar appears when setting a breakpoint

You may occasionally see an error bar when setting a breakpoint, indicating that the breakpoint hasn't been set.



The circle outline indicates that RStudio will remember the breakpoint, but the breakpoint has not yet been enabled. You need to take further action to enable it.

If this happens, the following should enable the breakpoint.

1. Save the file
2. Source the file (by pressing the Source button in the toolbar). If developing a package, rebuild and reload the package.

Hitting breakpoints

If the code on which you're setting a breakpoint is executing but the breakpoint is not being hit, it's possible that the type of code does not yet support breakpoints. Here are two common cases:

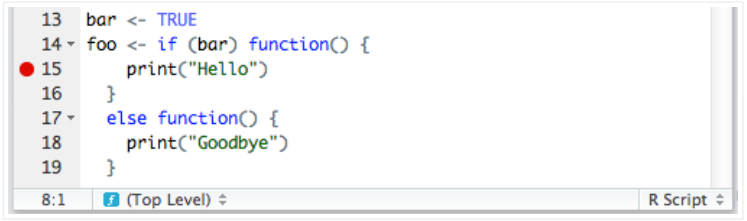
Complex function assignments

Recently viewed articles

Debugging with RStudio

Related articles

Debugging with RStudio
Keyboard Shortcuts
Developing Packages with RStudio
Using Rcpp with RStudio
Quick list of useful R packages



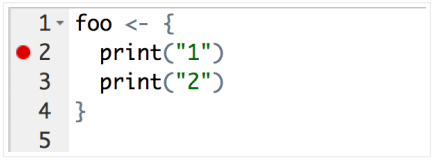
RStudio understands function assignments of the form:

```
name <- function(args, ...)
```

It does not understand more complex syntax.

In this case, the editor indicates that the current function is *(Top Level)* even when the cursor is inside **foo**. Because the editor does not know which function to place the breakpoint in, the breakpoint on line 15 in this example won't work.

Anonymous expressions



Code inside anonymous expressions is also immune to breakpoints, and for the same reason—the expression is neither a named function nor a top-level statement that can be stepped through as the file is sourced.

If you need a breakpoint in these cases, add the statement **`browser()`** to create one manually.

Other notes and things to try

- RStudio's breakpoint functionality is built on R's **`trace`** infrastructure. It is possible to disable breakpoints in a function by calling **`untrace`** on the function, and also possible to disable all breakpoints by disabling tracing. This is not recommended since the IDE does not check for these manipulations. Make sure **`tracingState()`** returns **`TRUE`** if you'd like RStudio breakpoints to work.
- In order to hit breakpoints during sourcing, you need to use the special debug sourcing command **`debugSource`** included in RStudio. If you are calling **`source`** manually on your file, breakpoints will still work, but will not be enabled until after all the code in the file has been executed.

Realated Topics

- Debugging with RStudio