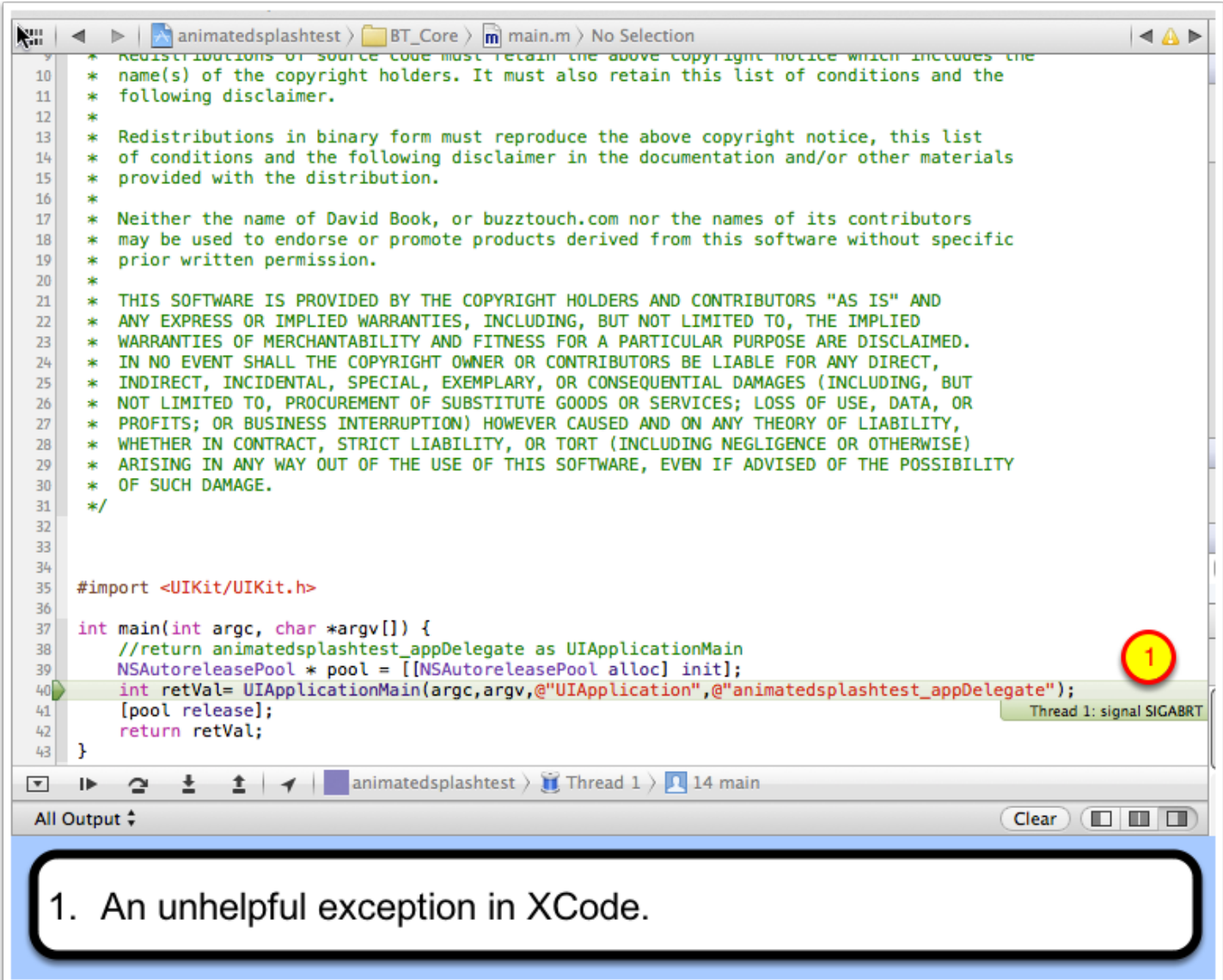


Debugging Strange Exceptions in XCode

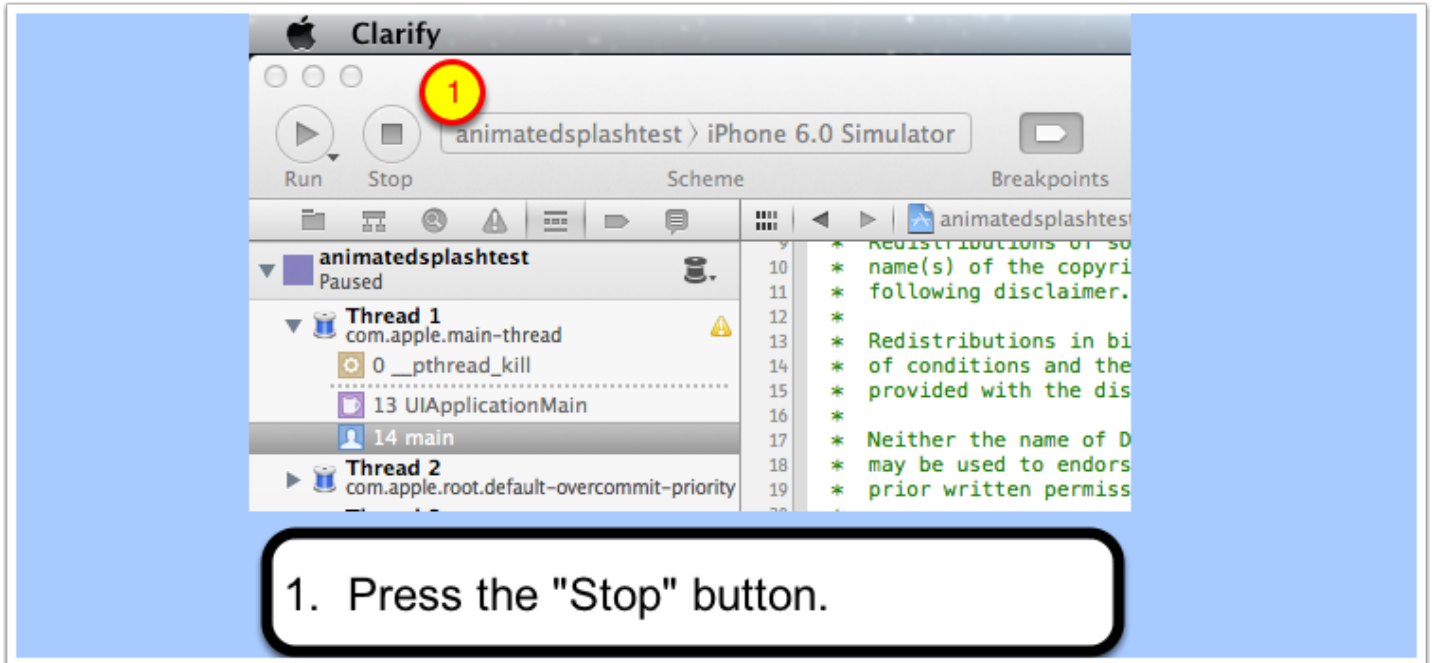
It will inevitably happen. You will be testing an iOS app out and XCode will crash with a strange, cryptic error. You will see a series of memory addresses, code that doesn't make a lick of sense, and a green line with a very unhelpful comment. In the screenshot below, it says "SIGABRT". Technically speaking, this is called an "exception". This tutorial will show you how to identify what is going on and how to easily get help on the forums.



The screenshot shows the XCode IDE with a source file named `main.m` open. The code contains a copyright notice and a `main` function. The `main` function calls `UIApplicationMain` with the following arguments: `argc`, `argv`, `@UIApplication`, and `@animatedsplashtest_appDelegate`. A red circle with the number '1' highlights the `@animatedsplashtest_appDelegate` argument. The console output shows the error: `Thread 1: signal SIGABRT`. Below the code editor, the 'All Output' pane is visible, containing the text: `1. An unhelpful exception in XCode.`

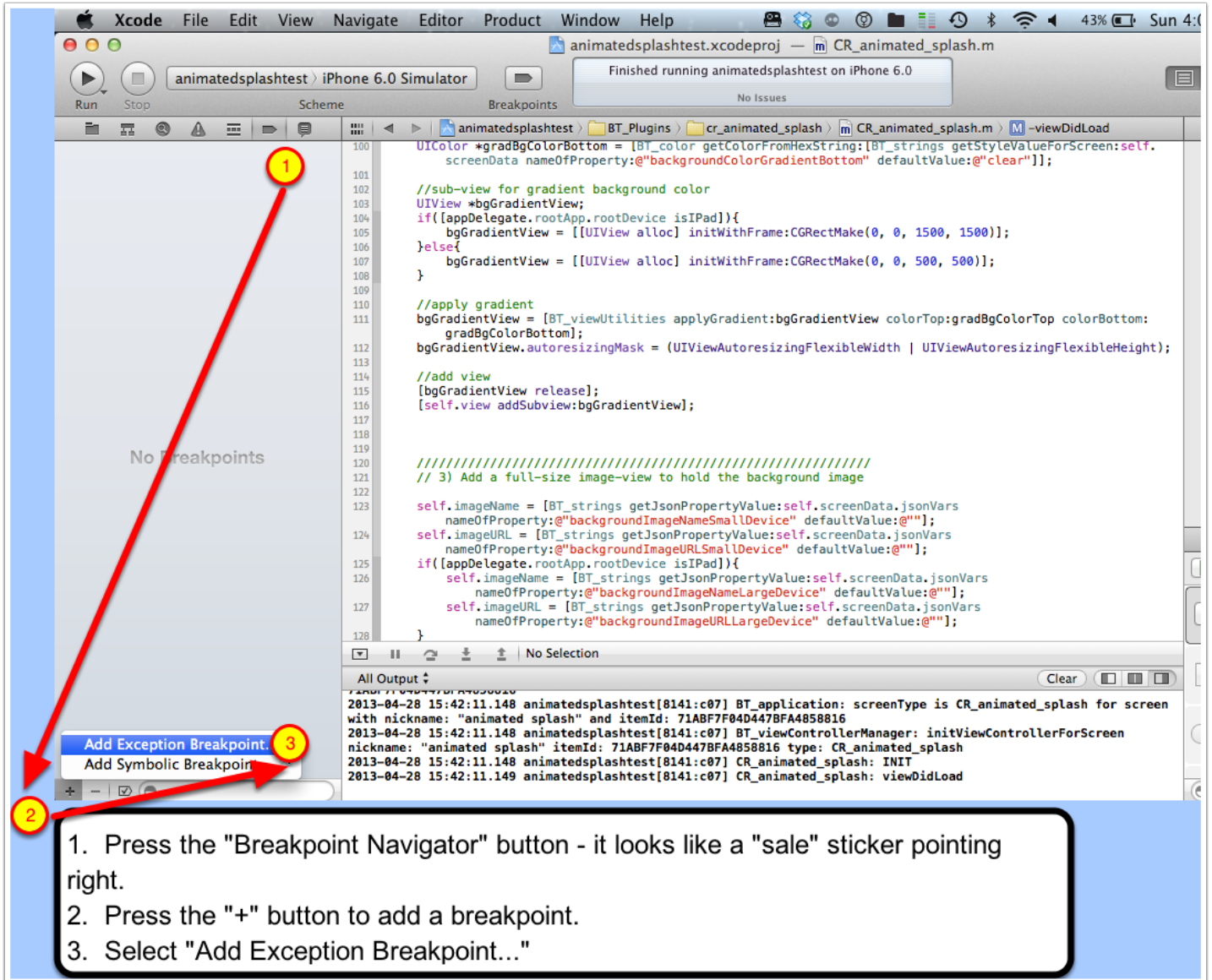
Preparing XCode

The first thing to do when you receive an exception like this is to make the sure the app is not running. Press the "Stop" button.



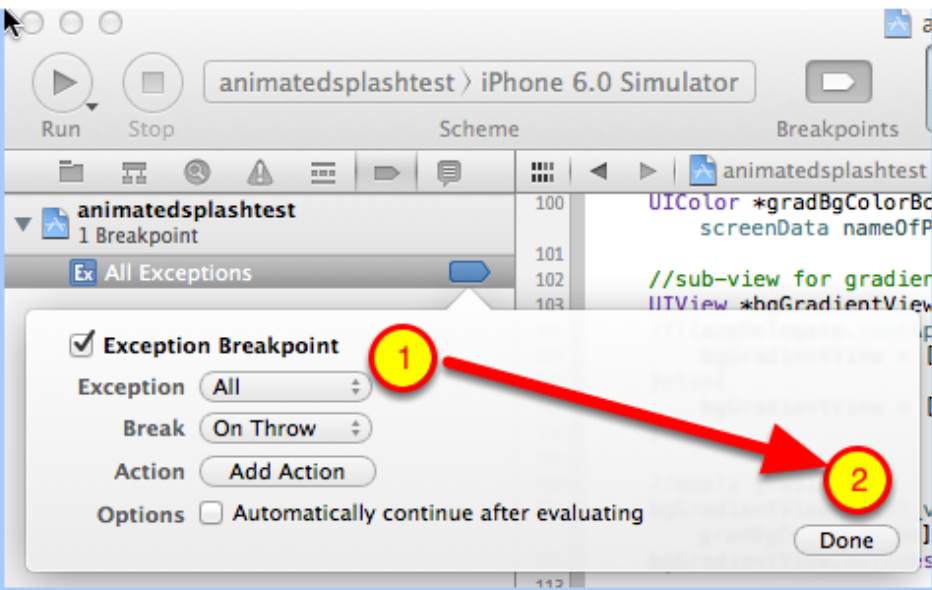
Setting up a breakpoint exception

Next, we will set up a breakpoint exception. To do so, we will open up the breakpoint navigator and click the 'add breakpoint' button:



Creating the breakpoint

Create the exception breakpoint as follows:

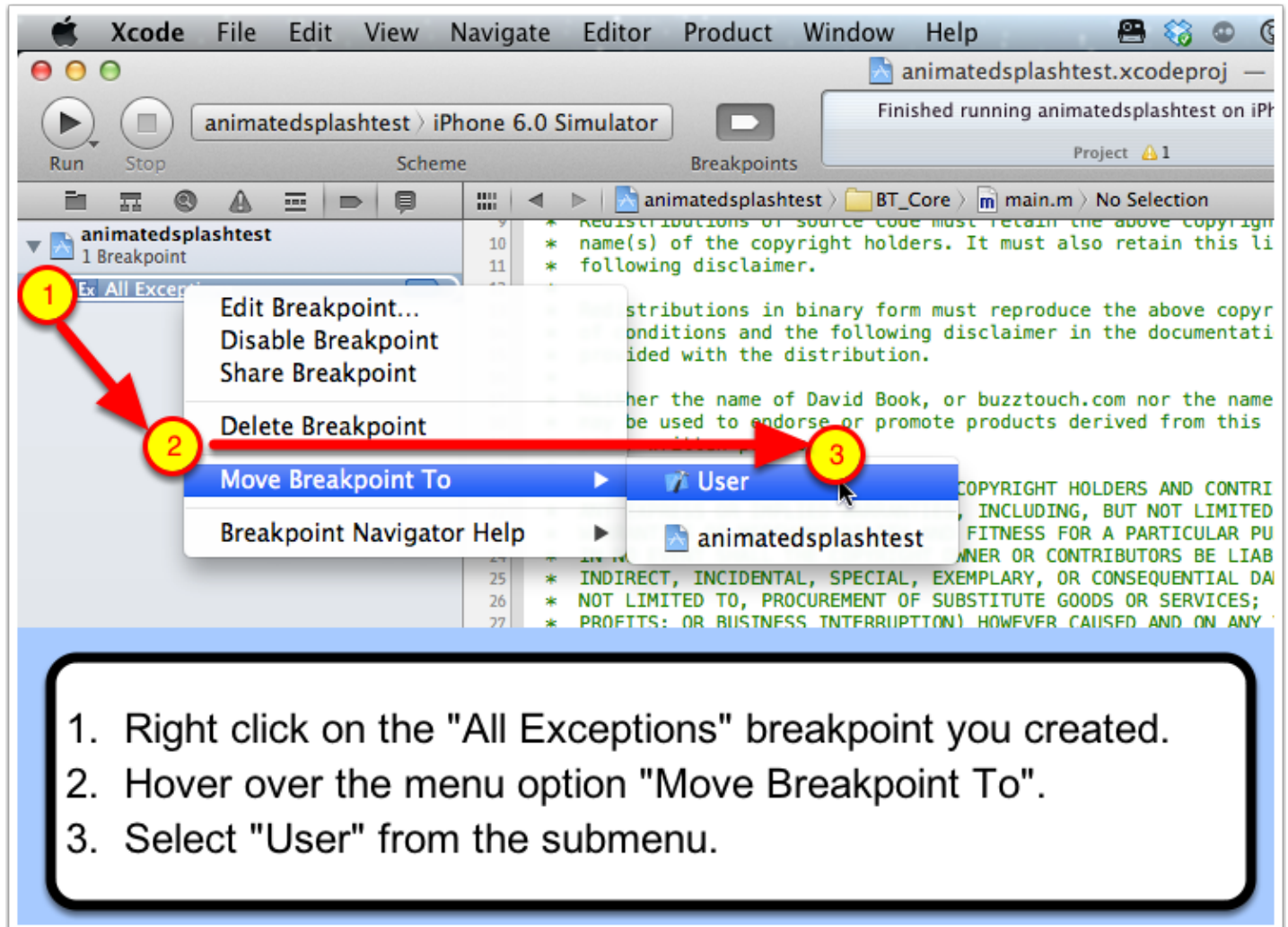


1. Add the exception breakpoint, making sure that the "Exception" drop-down is set to "All".

2. Press "Done".

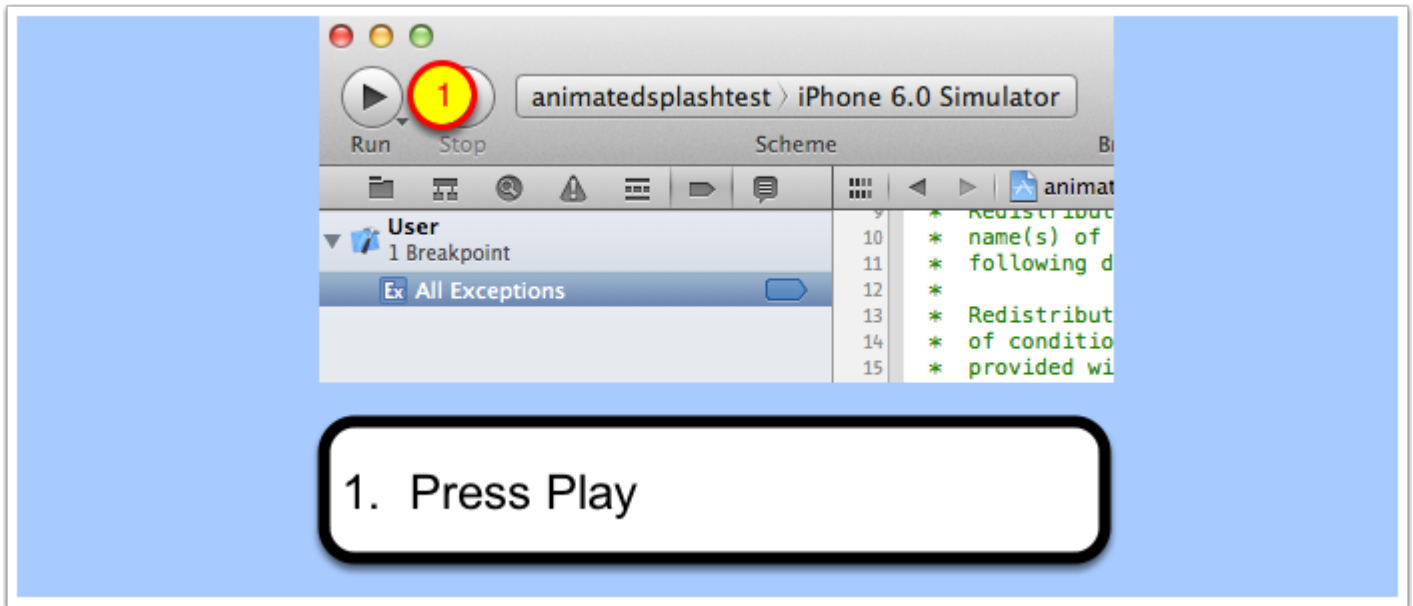
Apply Breakpoint to All Projects

Optionally, you can apply the breakpoint you created to all XCode projects. This way, you won't have to repeat the steps in this tutorial for your other apps.



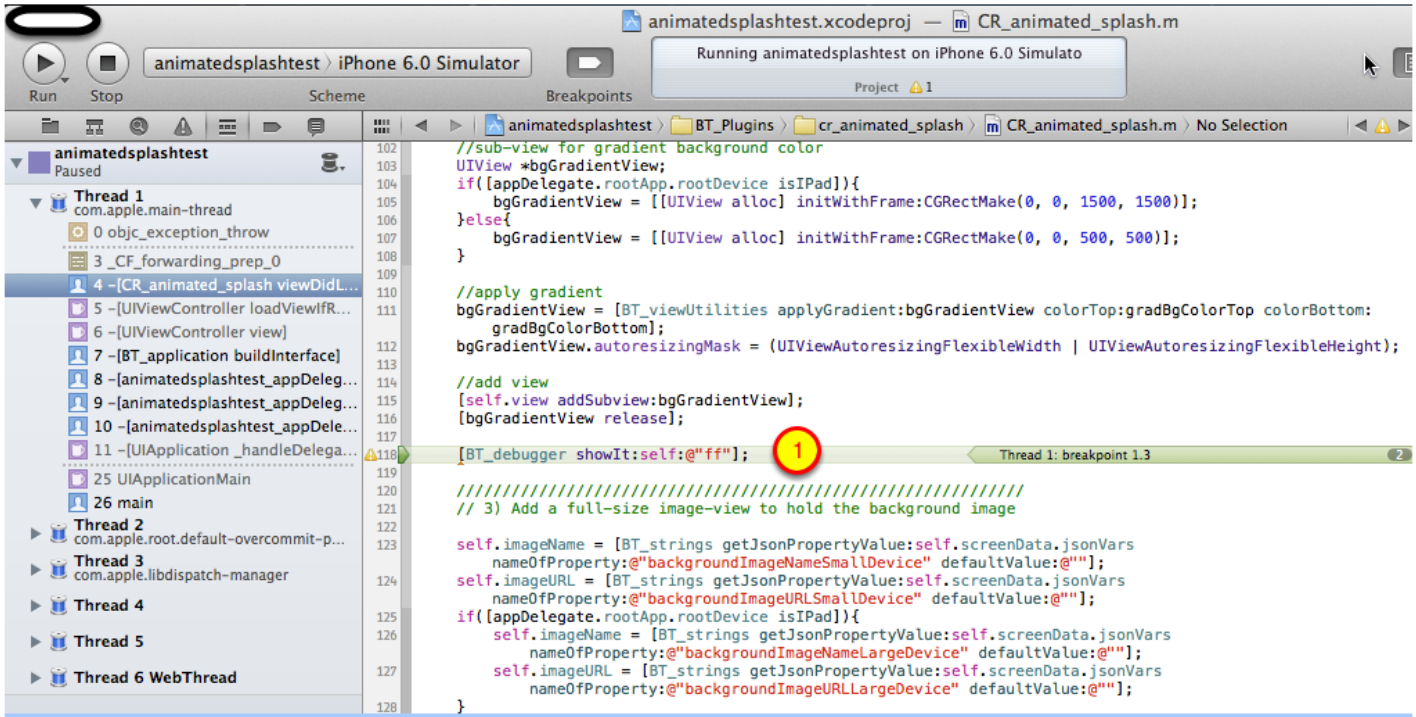
Run the app again

Run the app again to get your improved debugging results



Identify source of error

When you run the app again, it will still crash. However, this time it will (hopefully) present a different error message in XCode. In this case, we see that XCode identified the exception occurred on line 118 of the file "CR_animated_splash.m". The error message is "Thread 1: breakpoint 1.3", which is not helpful by itself. But the statement that caused the exception is very helpful. It should be enough to diagnose the problem by others on the forum, if you can't diagnose it yourself.



The screenshot shows the Xcode IDE with the following details:

- Scheme:** animatesplashtest | iPhone 6.0 Simulator
- Breakpoints:** Thread 1: breakpoint 1.3 (set at line 118)
- Call Stack (Thread 1):**
 - 118 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 117 [bgGradientView release];
 - 116 [self.view addSubview:bgGradientView];
 - 115 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 114 [bgGradientView autorelease];
 - 113 [bgGradientView removeFromSuperview];
 - 112 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 111 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 110 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 109 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 108 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 107 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 106 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 105 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 104 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 103 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
 - 102 [BT_debugger showIt:self:@\"ff\"]; (highlighted with a red circle)
- Code Snippet:**

```

102 //sub-view for gradient background color
103 UIView *bgGradientView;
104 if([appDelegate.rootApp.rootDevice isIPad]){
105     bgGradientView = [[UIView alloc] initWithFrame:CGRectMake(0, 0, 1500, 1500)];
106 }else{
107     bgGradientView = [[UIView alloc] initWithFrame:CGRectMake(0, 0, 500, 500)];
108 }
109
110 //apply gradient
111 bgGradientView = [BT_viewUtilities applyGradient:bgGradientView colorTop:gradBgColorTop colorBottom:
112     gradBgColorBottom];
113 bgGradientView.autoresizingMask = (UIViewAutoresizingFlexibleWidth | UIViewAutoresizingFlexibleHeight);
114
115 //add view
116 [self.view addSubview:bgGradientView];
117 [bgGradientView release];
118 [BT_debugger showIt:self:@\"ff\"];
119
120 ////////////////////////////////////////////////////
121 // 3) Add a full-size image-view to hold the background image
122
123 self.imageName = [BT_strings getJsonValue:self.screenData.jsonVars
124     nameOfProperty:@\"backgroundImageNameSmallDevice\" defaultValue:@""];
125 self.imageURL = [BT_strings getJsonValue:self.screenData.jsonVars
126     nameOfProperty:@\"backgroundImageURLSmallDevice\" defaultValue:@""];
127 if([appDelegate.rootApp.rootDevice isIPad]){
128     self.imageName = [BT_strings getJsonValue:self.screenData.jsonVars
129         nameOfProperty:@\"backgroundImageNameLargeDevice\" defaultValue:@""];
130     self.imageURL = [BT_strings getJsonValue:self.screenData.jsonVars
131         nameOfProperty:@\"backgroundImageURLLargeDevice\" defaultValue:@""];
132 }

```

Callout Box:

1. A more detailed error causing your app to crash. In this case, the statement causing the problems is: [BT_debugger showIt:self:@\"ff\"];

This error was caused by an improperly formatted line of code. The code should read: [BT_debugger showIt:self theMessage:@\"ff\"];

Exceptions not covered by breakpoint

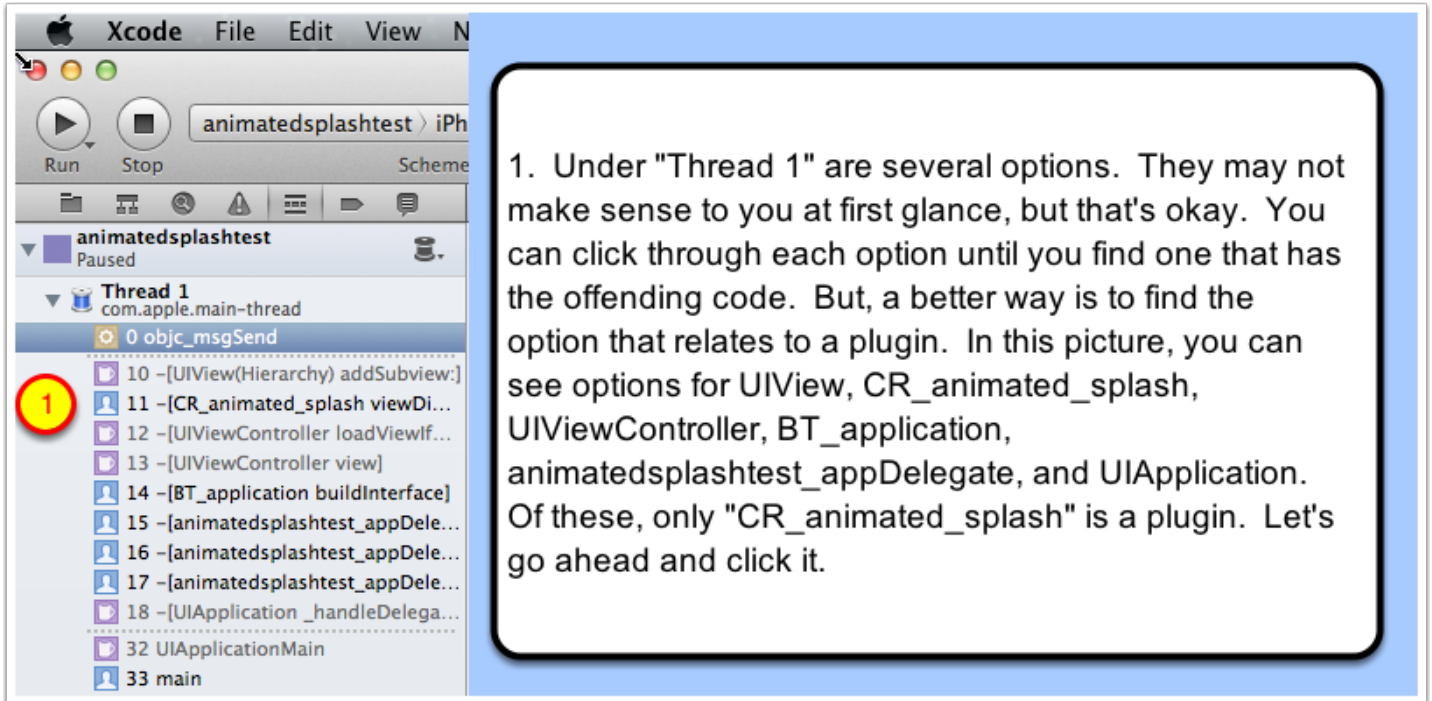
Occasionally, you may still get a crash that doesn't go to the line of code with the problem. In the example below, we see an exception of "EXC_BAD_ACCESS", even though we previously set up the breakpoint.

The screenshot shows the Xcode debugger interface for the 'animatedsplashtest' project on an iPhone 6.0 Simulator. The main window displays assembly code for the 'objc_msgSend' function. A breakpoint is set at line 11, which is highlighted in green. However, the crash (EXC_BAD_ACCESS) occurs at line 10, which is also highlighted in green. A red circle with the number '1' is placed over the crash point. The crash message in the status bar reads: 'Thread 1: EXC_BAD_ACCESS (code=1, address=0x616c4372)'. The assembly code includes instructions like 'movl (%edi), %esi' at line 10 and 'movl %ecx, %edx' at line 11.

1. Exception not caught by breakpoint.

Other Ways to Find Error

Fret not, our cause is not lost. XCode still gives us the tools to find the line of code causing the error. On the left hand side of XCode, you will see a series of options under "Thread 1"



Finding the exception

Success! Clicking on the plugin gives us a screen that shows the offending line of code. In this case, the exception was in the "cr_animated_splash" plugin.

The screenshot shows the Xcode IDE with the following details:

- Simulator:** iPhone 6.0 Simulator, running 'animatedsplashtest'.
- Call Stack (Left):** Thread 1 (Paused) at line 11: `11 -[CR_animated_splash viewDi...]`. A yellow circle '1' is next to this line, with a red arrow pointing to the source code.
- Source Code (Right):** Lines 101-130 show the initialization of `bgGradientView`. Line 115 shows `[self.view addSubview:bgGradientView];`, which is highlighted with a yellow circle '2'.
- Error Message (Bottom):** `Thread 1: EXC_BAD_ACCESS (code=1, address=0x616c4372)`.

1. We selected "-[CR_animated_splash viewDi..." to reveal the code causing the exception.
2. The line of code causing the issue is: `[self.view addSubview:bgGradientView];`. Sending this as a screen shot should be enough for someone to diagnose the problem. In this case, we have released "bgGradientView" before attempting to add it to the self.view.

Get help for your error

Once you've identified the code causing your app to crash, it's time to ask for help on the Buzztouch.com forums. Ideally, you should take a screenshot of XCode and save it on an online storage site (e.g., Dropbox), and link to the image in your forum post. On a Mac, you can hold CMD+Shift and press "3" to take a screenshot. Or you can hold CMD+Shift and press "4" to select just a section of the screen for the screenshot.