DebugApp: Debugging an Application with Project Builder

This tutorial shows how to debug a project with Project Builder. In it, you'll build a debuggable project, set a breakpoint, step through your code, and examine your data. You'll also learn how to enter commands directly to GDB, the command-line tool Project Builder uses as its debugger.

This tutorial assumes that you are already familiar with another debugger and with Mac OS X programming.

Build a Debuggable Application

Here, you'll create a new project and turn on the option that creates debugging information.

1. Create a new project.

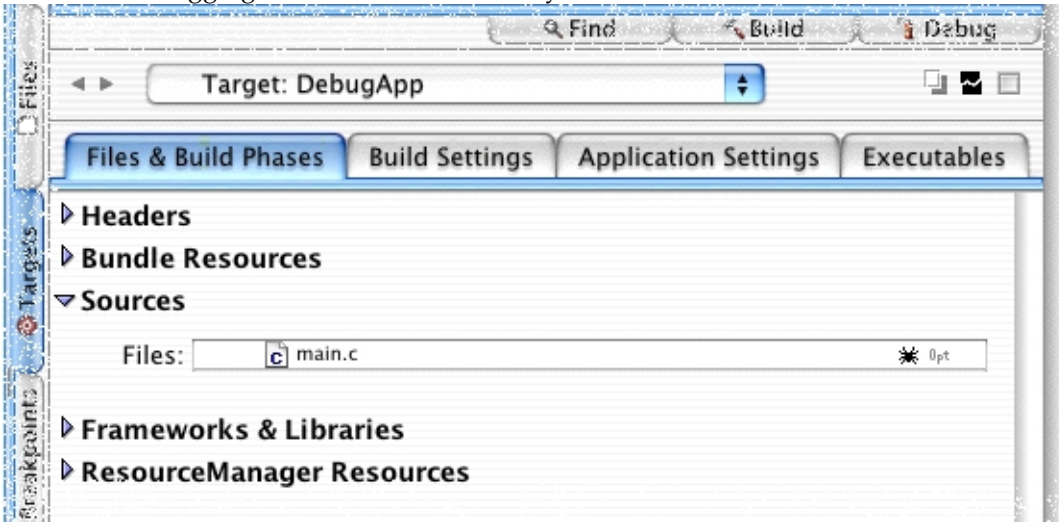
Draft. Confidential. © Apple Computer, Inc. 4/24/00

Choose File > New Project. Select Carbon Application, and click Next. Then enter DebugApp as the project name, choose a location, and click Finish. Project Builder creates a new project and opens its project window. The project contains some sample files that you can compile and run without change.

2. Click the Targets tab, select DebugApp, and click the Files & Build Phases tab in the target editor.
3. Turn on the Debug option for `main.c`.

   If you can't see `main.c`, click the triangle beside Sources.

   Find the bug icon that's to the right of `main.c` and click it so it turns black (instead of gray). Now, when you build the application, Project Builder will include debugging information in the binary.



4. Click the Build button at the top of the project window.

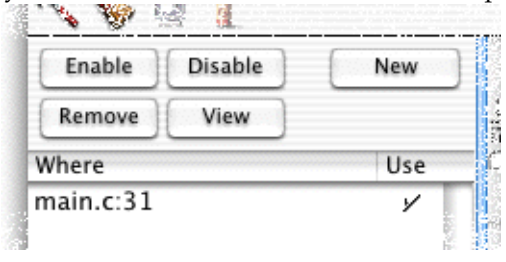
Draft. Confidential. © Apple Computer, Inc. 4/24/00

Set a Breakpoint

In Project Builder, you can set a breakpoint even before you start the debugger. Project Builder saves your breakpoints when you close your project, so they're still there the next time you open it.

Click the Files tab, open `main.c` and scroll down until you see the `main` function. Click in the margin beside the `Initialize` statement. An arrow appears beside it.

If you want, look at the breakpoint list by clicking the Breakpoints tab. From here, you can view the source code for a breakpoint, and enable, disable, or remove it.



Draft. Confidential. © Apple Computer, Inc. 4/24/00

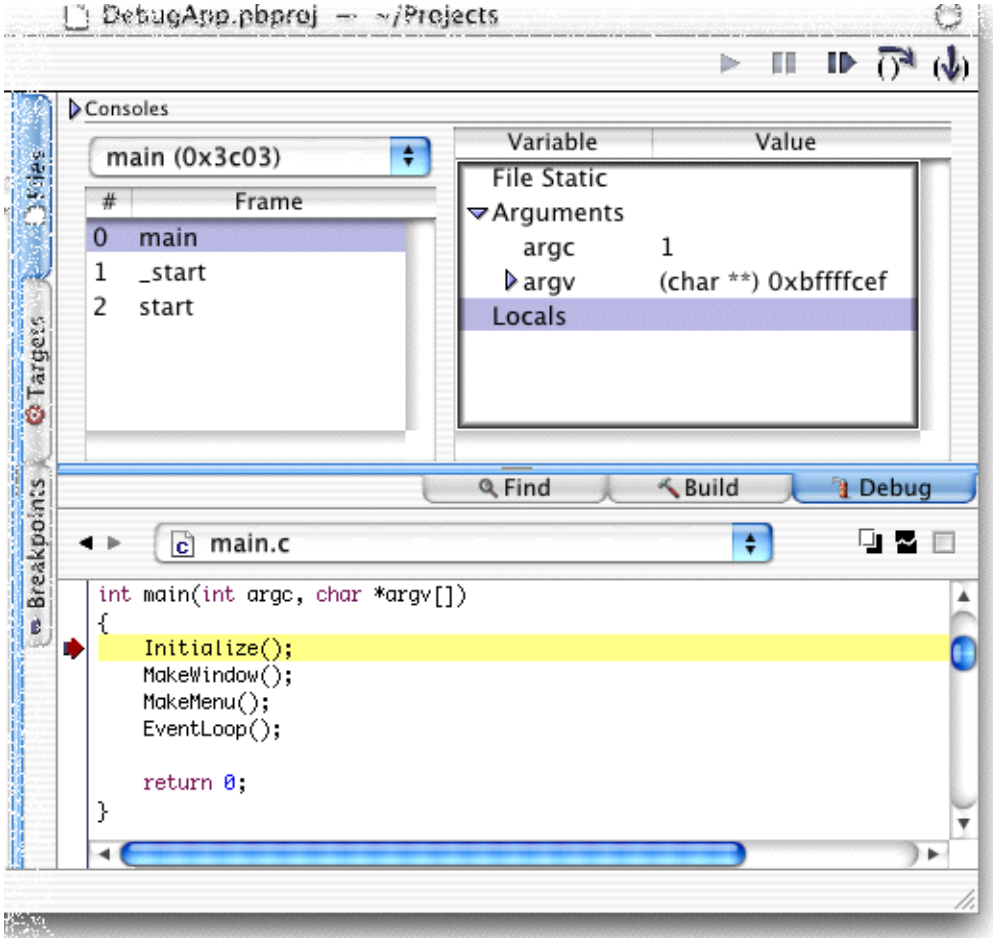
Start Debugging the Application

Click the Debug button, which looks like a spray can, at the top of the project window.

The debug panel drops down and a Debug tab appears alongside the Find and Build tabs. Project Builder starts running the application under the debugger and

Draft. Confidential. © Apple Computer, Inc. 4/24/00

stops at the breakpoint. The currently executing statement is displayed in the code editor, highlighted with a red arrow pointing at it.



In the right side of the toolbar are buttons that control the debugger's operations. You'll use them in "Step Through the Code" (page 20).

- Run starts running application if it's not already running.
- Pause pauses the application and displays the currently executing statement in the editor.
- Continue continues running a paused application.

Draft. Confidential. © Apple Computer, Inc. 4/24/00

- Step Over executes the next statement, staying within the same function.
- Step Into executes the next statement, jumping to the first line in the next function if its source code is available
- Stop force quits the running application and quits the debugger Then it closes the Debug panel and removes the Debug tab.

At the top of the debug panel is a disclosure triangle that displays two consoles that let you control GDB, the GNU command-line debugger. You can handle most debugger tasks through Project Builder's interface, but you can access GDB directly for more advanced tasks. You'll use these panes in "Use the GDB Command-Line" (page 25).

At the bottom of the debug panel are a two lists filled with information about the current function in the current thread. You'll use these lists in "Examine Your Data" (page 23).

- The Threads pop-up menu displays all the threads for your application. In this tutorial, one thread runs your project's source code, and the rest are system threads that handle inter-application communication and the debugger. Selecting a thread displays its call chain in the frames list.
- The Frames list displays the call chain for the selected thread. Clicking a frame displays its variables in the variables list and displays its currently executing statement in the code editor with a red arrow pointing at it.
- The Variables list displays all the local and global variables visible in the selected frame.

Step Through the Code

Now, you'll use the Step Into and Step Over buttons to watch your application execute.

1. Step over the `Initialize` function.

   Click the Step Over button.

Draft. Confidential. © Apple Computer, Inc. 4/24/00

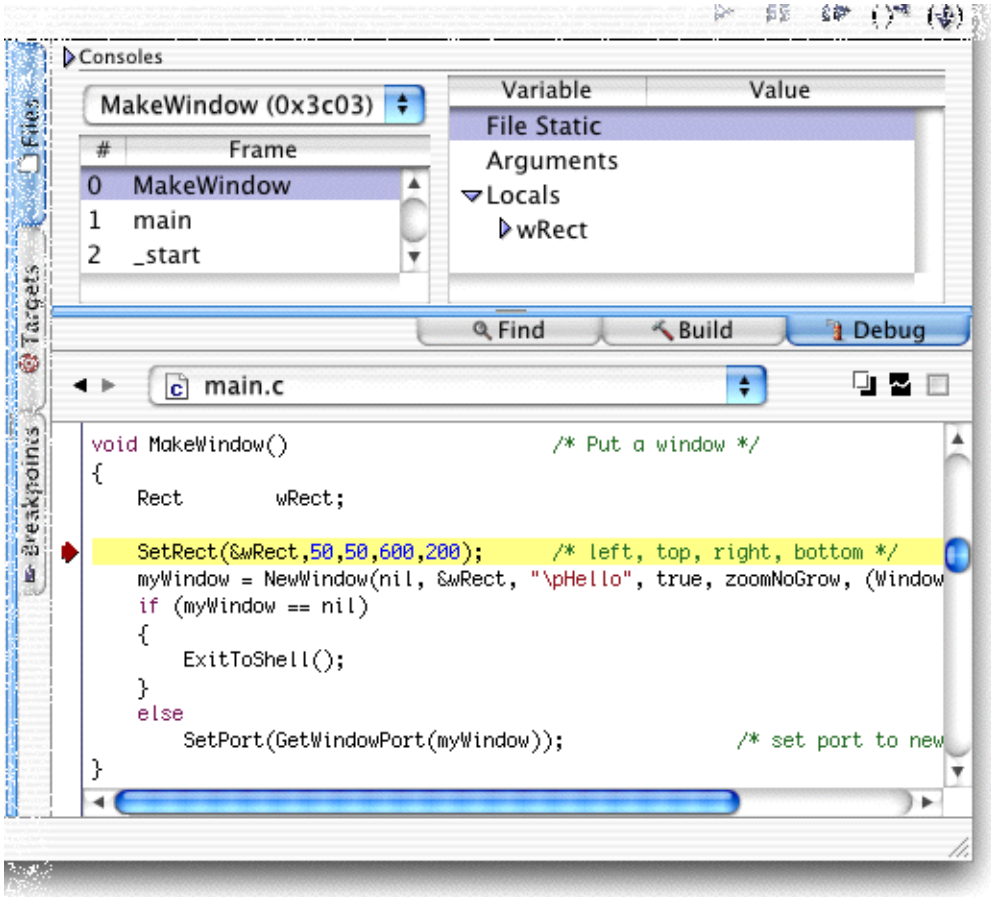
Project Builder runs the Initialize function and stops at the next statement `MakeWindow`.

2. Step into the `MakeWindow` function.

   Click the Step Into button.

   Project Builder displays `MakeWindow's` first statement in the code editor and its variables in the variables list.

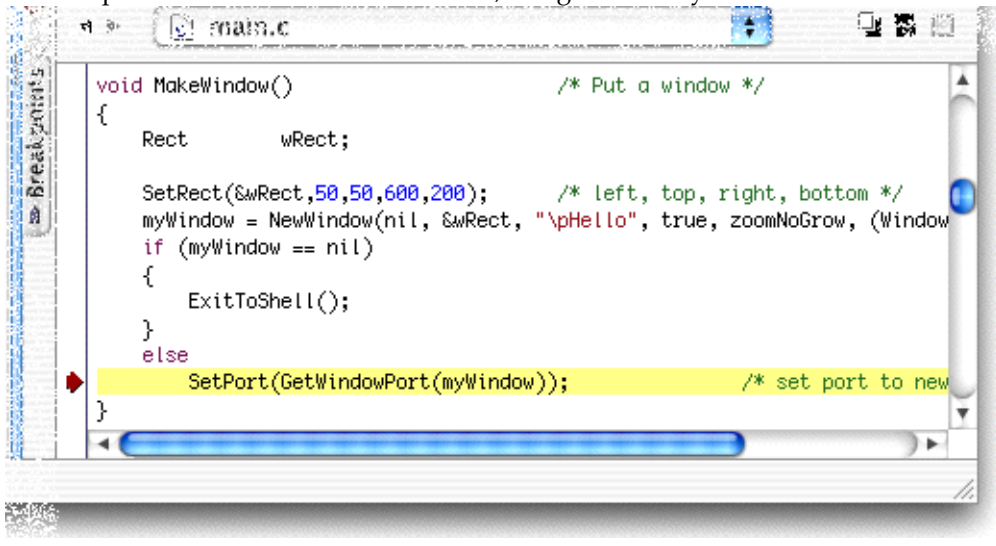


3. Watch it skip over part of the `if` statement.

Draft. Confidential. © Apple Computer, Inc. 4/24/00

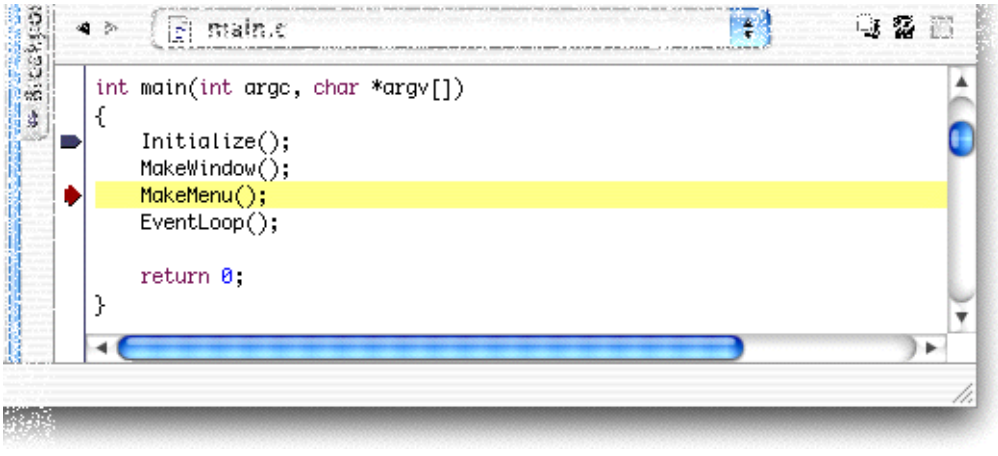
Press either the Step Into or Step Over button a couple times, until the arrow is beside the `if` statement. Notice that when you're executing a function whose source code you don't have (such as a Mac OS Toolbox function), the Step Into and Step Over buttons do the same thing.

Press Step Into or Step Over again. Project Builder evaluates the `if` statement, skips over the `ExitToShell` statement, and goes directly to the `SetPort` statement.



4. Watch it skip back to the main function.

   Press Step Into or Step Over a couple more times, until the arrow goes back to the `main` function, beside `MakeMenu`.



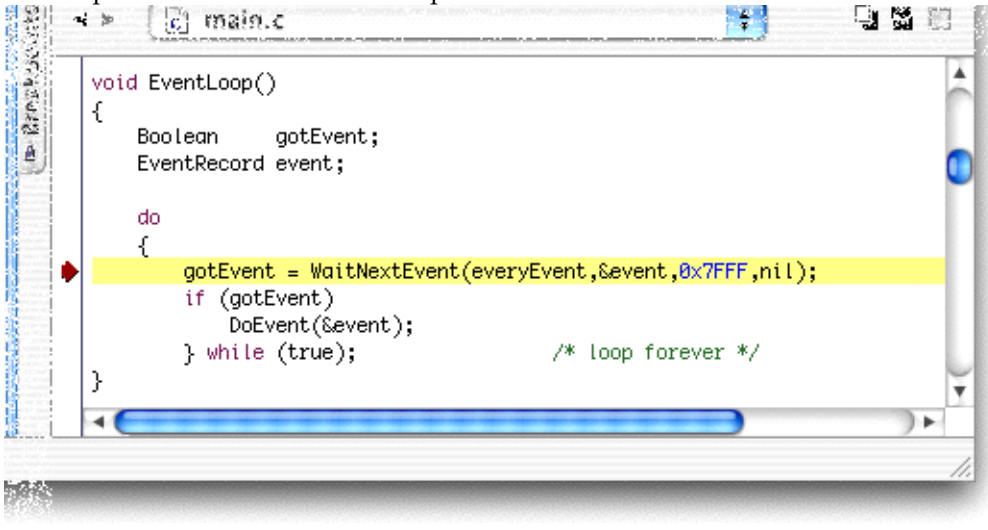
Draft. Confidential. © Apple Computer, Inc. 4/24/00

Examine Your Data

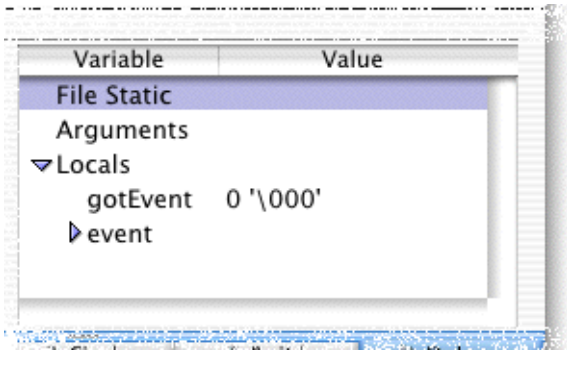
Now you'll examine the applications data with Project Builder.

1. Step into `DoEvent`.

   Step over `MakeMenu` and then Step into `DoEvent`. It should look like this:



   Now look at the variable list:



Draft. Confidential. © Apple Computer, Inc. 4/24/00

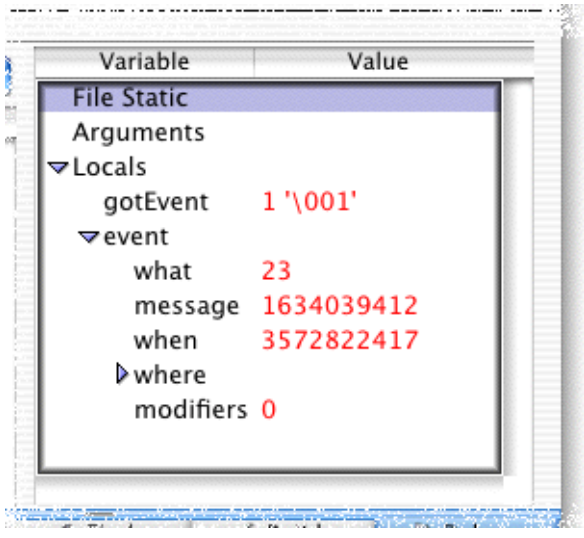
It has three sections:

- File Static lists static variables declared at a file's top level.
- Arguments lists the arguments to the currently executing function.
- Locals lists the local variables declared in the currently executing function.

In this function, only the Locals section has variables: the boolean `gotEvent`, and the structure `event`.

2. Step over the `WaitNextEvent` statement, and examine the record `event`.

   Step over `WaitNextEvent`. Then click the triangle beside `event` to view its contents:

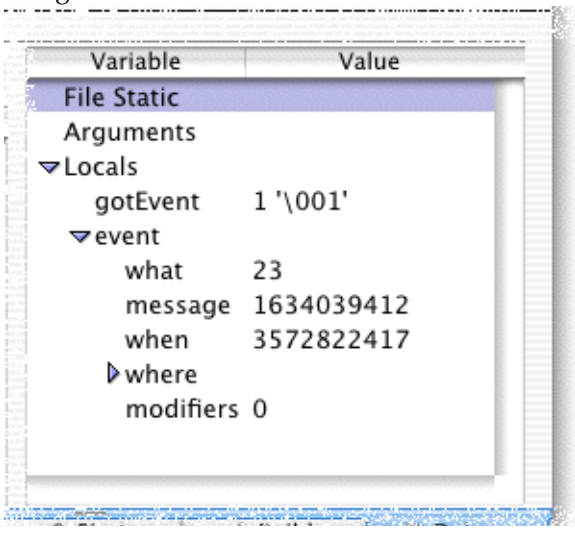


   The values are all in red to show they've changed since the last statement ran. If you want, you can also expand the `where` structure as well.

3. Step over the `if` statement.

Draft. Confidential. © Apple Computer, Inc. 4/24/00

Notice that the variables are in black because this time, the values haven't changed:



4. Continue the application.

   Press Continue.

Use the GDB Command-Line

Project Builder debugs your programmer by sending commands to GDB, the GNU command-line debugger. Project Builder provides you with a UI that handles the basic features. For more advanced features, you can type commands to GDB directly.

In this section, you'll use the GDB command-line to set a conditional breakpoint, a breakpoint that Project Builder stops at only when a specified condition is true.

Note that with it when you close the project, Project Builder saves your breakpoints but not any conditions associated with them. Project Builder does not save anything entered at the GDB command-line.

Draft. Confidential. © Apple Computer, Inc. 4/24/00

1. Pause the program.

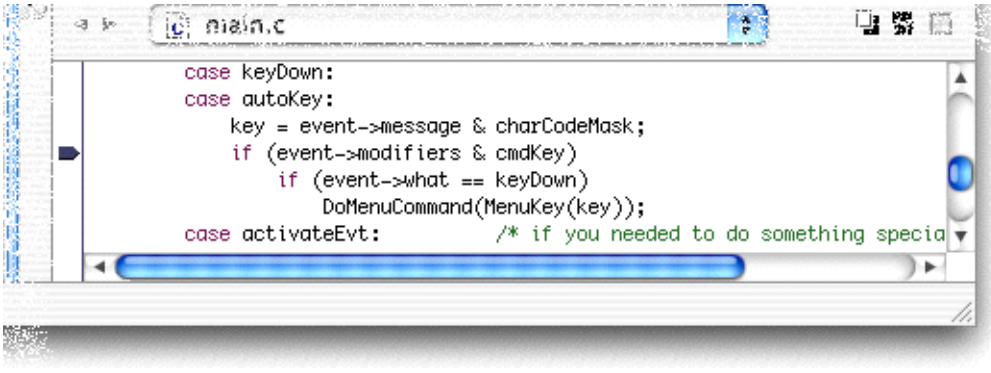
   Press the Pause button.

   Project Builder pauses the program at whatever statement is currently executing. That's usually the `WaitNextEvent` statement in `EventLoop`, but don't worry if it's not.

2. Create a breakpoint, so the program pauses when you press a key.

   Scroll down to the `DoEvent` function, and find the `case keyDown` statement. That's where it handles a key press. After that is a statement that sets the variable `key`. Because you might want to see what key was pressed, place the breakpoint right after that statement, at the statement `if (event->modifiers & cmdKey)`.



3. Continue the program.

   Press the Continue button.

4. In your application, press a letter.

   Click DebugApp's icon in the dock, and the application displays its Hello window. Press any letter.

   Project Builder comes to the front, pauses your program, and displays the statement you breakpointed.

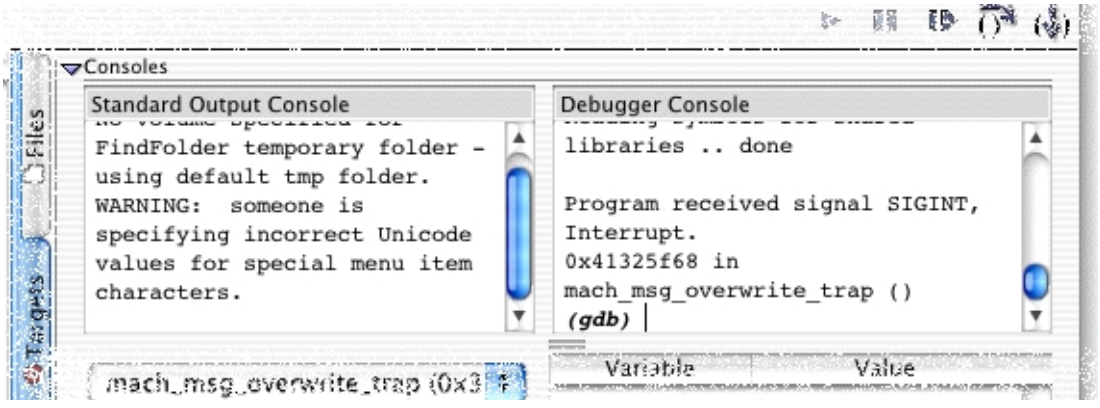
5. Find the breakpoint's number.

   At the top of the Debugger panel, click the triangle besides Consoles. Two panes are displayed:

Draft. Confidential. © Apple Computer, Inc. 4/24/00

- Standout Output Console displays messages sent to standard output and error. Also, if your application uses standard input, you'll enter it in this pane.
- Debugger Console lets you enter commands directly to GDB.

It should look like this:



In the Debugger Console, enter `info br` to displays information about the breakpoints in your program. You may need to adjust the divider bar between the two consoles to see all the output. It should look like this:

```
(gdb) info br
Num Type           Disp Enb Address    What
1   breakpoint     keep y   0x000034b8 in main at main.c:31
    breakpoint already hit 1 time
2   breakpoint     keep y   0x0000392c in DoEvent at main.c:139
    breakpoint already hit 1 time
```

Note the number of the breakpoint in `DoEvent`, which in this example is number `2`.

6. Create a conditional breakpoint.

   Enter `cond <num> (key=='a')`, where `<num>` is the breakpoint's number. For example:

   ```
   (gdb) cond 2 (key=='a')
   (gdb)
   ```

   Now, Project Builder will pause at this breakpoint only if you press `a`. If you want, you can enter `info br` again, to make sure. For example:

Draft. Confidential. © Apple Computer, Inc. 4/24/00

```
(gdb) info br
Num Type           Disp Enb Address    What
1   breakpoint     keep y   0x000034b8 in main at main.c:31
    breakpoint already hit 1 time
2   breakpoint     keep y   0x0000392c in DoEvent at main.c:139
    stop only if key == 97 'a'
    breakpoint already hit 1 time
```

7. Continue the program.

   Press the Continue button, and click DebugApp's icon in the dock.

8. Press `q`.

   The application doesn't pause.

9. Press `a`.

   Project Builder comes to the front, pauses your program, and displays the statement you breakpointed.

Stop the Debugger

Because Mac OS X has protected memory, you don't need to worry about quitting the program directly from the debugger. Just press the stop button at the top right corner of the Debug panel. Project Builder force quits the application, closes the Debug panel and removes the Debug tab.

Draft. Confidential. © Apple Computer, Inc. 4/24/00