**Starting in Inspect Mode**

Consider a program named **app.js** containing the following code:

**function f (n = 99) {  
  if (n === 0) throw Error()  
  f(n - 1)  
}  
f()**

Node.js supports the Chrome Devtools remote debugging protocol. In order to use this debugging protocol a client that supports the protocol is required. In this example [Chrome browser](https://www.google.com/chrome/) will be used.

Inspect mode can be enabled with the **--inspect** flag:

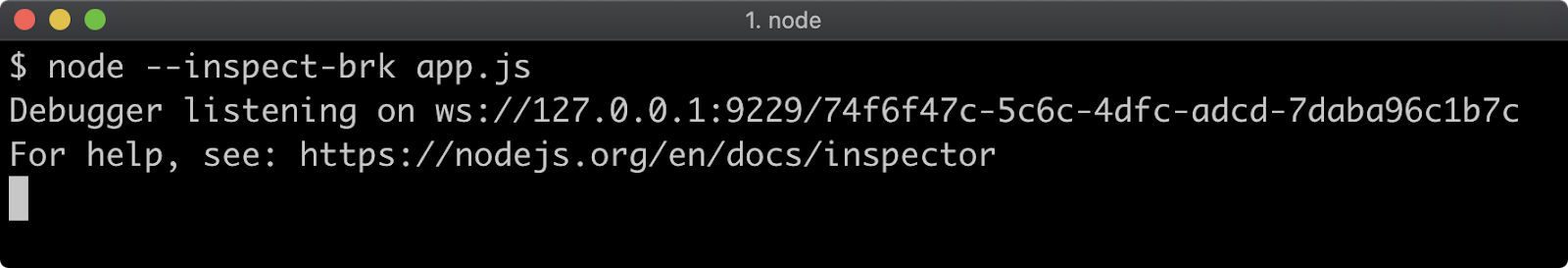
**node --inspect app.js**

For most cases however, it is better to cause the process to start with an active breakpoint at the very beginning of the program using the **--inspect-brk** flag:

**node --inspect-brk app.js**

Otherwise the application will have fully initialised and be performing asynchronous tasks before any breakpoints can be set.

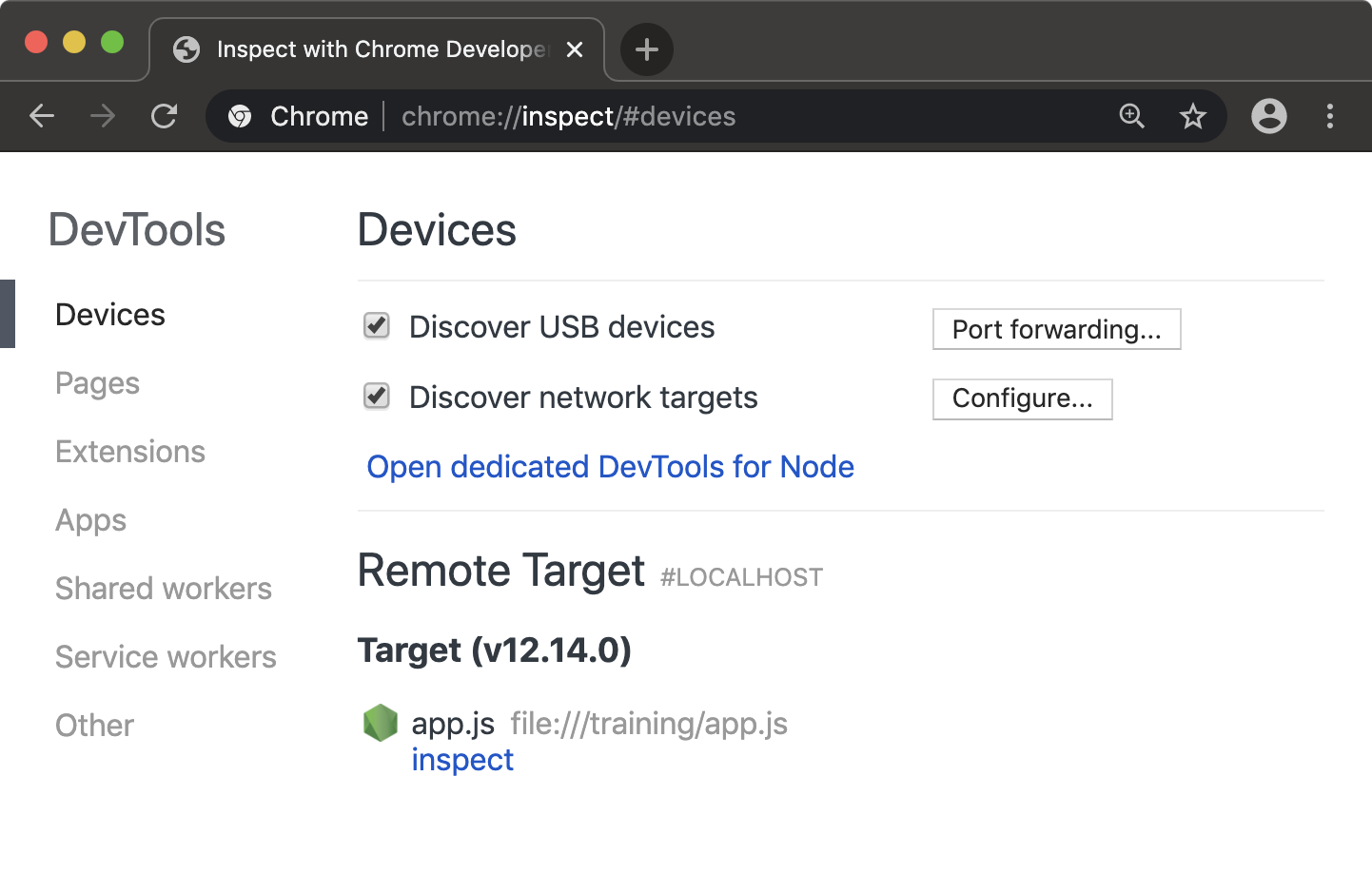
When using the **--inspect** or **--inspect-brk** flags Node will output some details to the terminal:



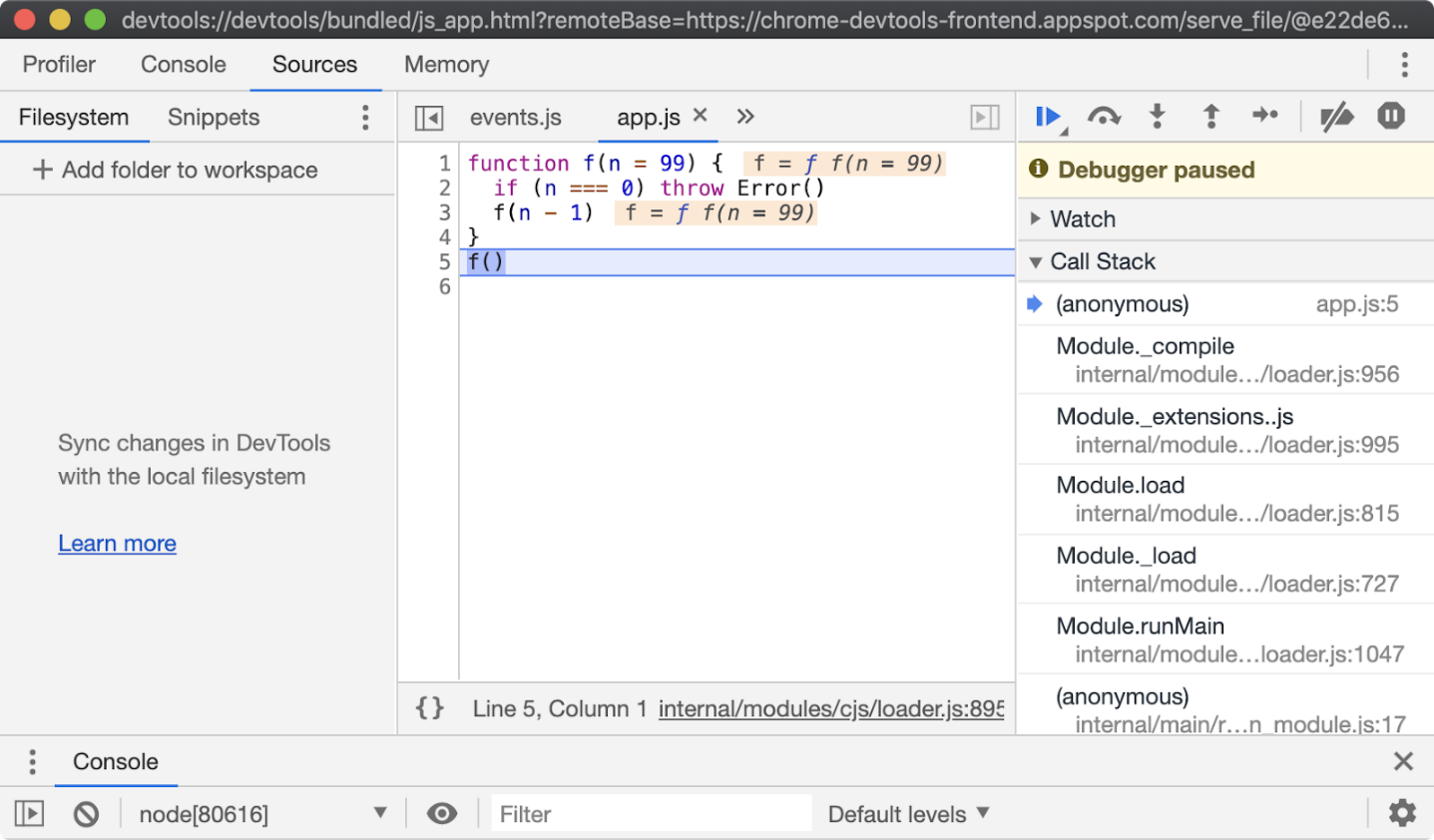
The remote debugging protocol uses WebSockets which is why a **ws://** protocol address is printed. There is no need to pay attention to this URI, as the Chrome browser will detect that the debugger is listening automatically.

In order to begin debugging the process, the next step is to set a Chrome browser tab's address bar to **chrome://inspect**.

This will load a page that looks like the following:



Under the "Remote Target" heading the program being inspected should be visible with an "inspect" link underneath it. Clicking the "inspect" link will open an instance of Chrome Devtools that is connected to the Node process.



Note that execution is paused is at the first line of executable code, in this case line 5, which is the first function call.

From here all the usual Chrome Devtools functionality can be used to debug the process. For more information on using Chrome Devtools, see [Google Developer's Documentation](https://developers.google.com/web/tools/chrome-devtools).

There are a range of other tools that can be used to debug a Node.js process using Chrome Devtools remote debugging protocol. To learn more, read [*"Debugging Guide"*](https://nodejs.org/en/docs/guides/debugging-getting-started/) by nodejs.org.