Debugging CoffeeScript

CoffeeScript code is not processed by browsers that work with JavaScript code. Therefore to be executed, CoffeeScript code has to be translated into JavaScript. This operation is referred to as *transpilation* and the tools that perform it are called *transpilers*.

To debug CoffeeScript in IntelliJ IDEA, you need *source maps* generated in addition to the JavaScript code. Source maps set correspondence between lines in your CoffeeScript code and in the generated JavaScript code, otherwise your breakpoints will not be recognised and processed correctly. JavaScript and source maps are generated by transpiling the CoffeeScript code manually using the File Watcher of the type *CoffeeScript Source Map*. After that you can debug the output JavaScript code as if it were a Node.js application.

For more details about *transpilation* in IntelliJ IDEA, see the section Using File Watchers.

Debugging CoffeeScript is supported only in the *local* mode. This means that IntelliJ IDEA itself starts the Node.js engine and the target application according to a run configuration and gets full control over the session.

For more details about debugging Node.js applications, see Running and Debugging Node.js.

Transpiling CoffeeScript manually and debugging the output as a Node.js application

1. Set the breakpoints in the CoffeeScript code, where necessary.
2. Transpile the CoffeeScript code into Javascript using the File Watcher of the type *CoffeeScript Source Map*.
3. Start creating a Node.js run configuration with the following mandatory settings:
    1. The Node.js engine to use. By default, the field shows the path to the interpreter specified on the Node.js page during Node.js configuration.
    2. In the **Working directory** field, specify the location of the files referenced from the starting CoffeeScript file to run, for example, *includes*. If this file does not reference any other files, just leave the field empty.
    3. In the **Path to Node App JS File** text box, specify the full path to the JavaScript file that was generated from the original CoffeeScript file during the transpilation.
4. Save the configuration and click 🐞 on the toolbar.
5. Proceed as while debugging a Node.js application locally.

See Also

Concepts:

- Running, Debugging and Testing

Procedures:

- Running and Debugging Node.js
- Previewing Pages with Web Contents in a Browser
- CoffeeScript Support

Reference:

- Run/Debug Configuration: Node JS
- Web Browsers

Web Resources:

- Developer Community