



SourceLair: The Cheat Sheet

SourceLair is an online editor / environment for developing software. While it might not be ideal for large-scale embedded projects, it's quite possibly the perfect way to learn TDD for embedded. We'll all be working on the same platform, and this should help us to focus on testing instead of porting and configuring and installing and setting up... at least for the beginning of the class. When we arrive at the full course project, we'll move on to setting up and using a proper local development environment.

Disclaimer: We have no relationship with SourceLair. We just tried to lower the getting-started hurdle as much as possible and arrived at a web-based solution in general and SourceLair specifically.

Start by signing up for a free [SourceLair account](#) (you might love it so much that you'll gladly shell out the money for a pro account). On the left, you will find a dropdown list that lets you add a project. Your free account allows you just one project... but that's enough for this class.

Click this button  to expand the project list and then click the button  to add a project (you'll find them in the upper left corner of the web application):

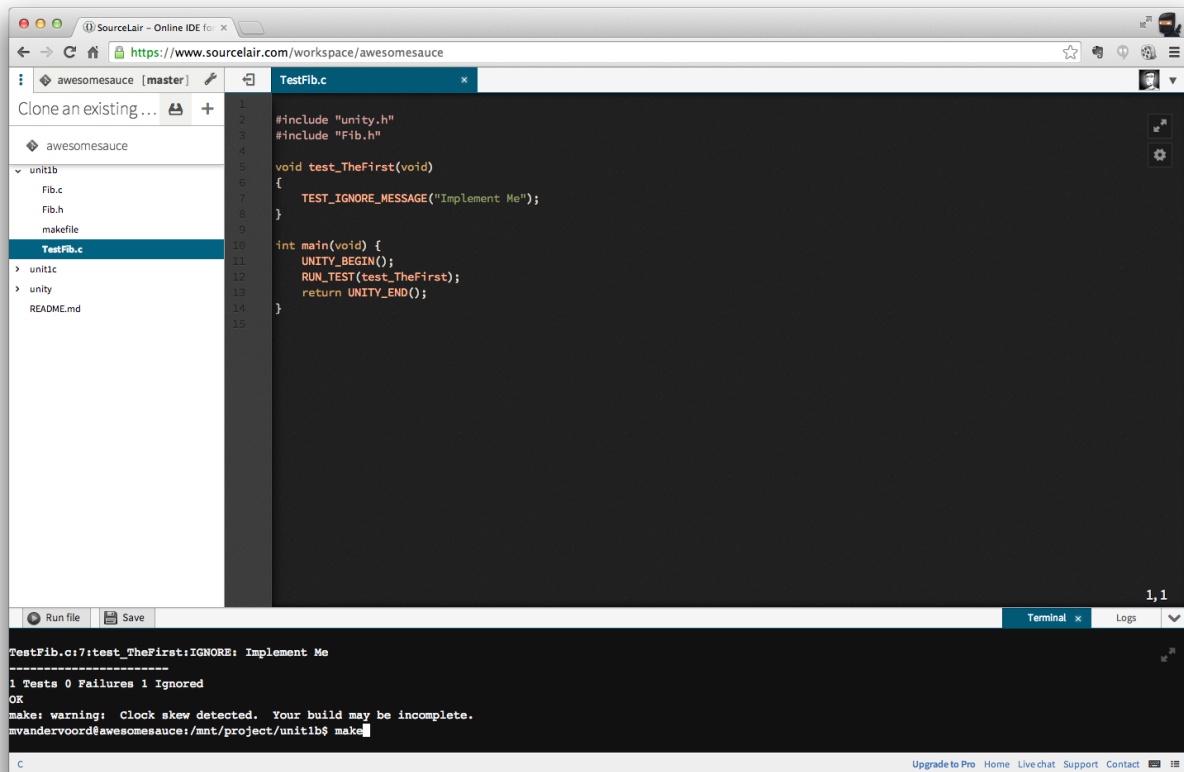


Adding a new project means working from a source code repository (see document *Introduction to Revision Control and Source Code Management Plus A Super Tiny Git Reference Guide* for much more on this topic). Type in (or better yet, copy / paste) the following URL in the add project pop-up window and then click the happy CLONE button:

<https://bitbucket.org/mvandervoord/awesomesauce.git>

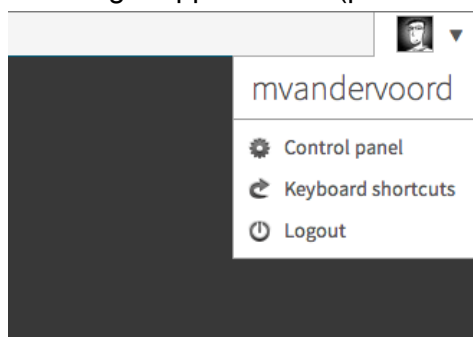
Even Better: If you happen to have a BitBucket account (or would like to sign up for one), there is an extra step you can optionally insert in the preceding. Before cloning your repository, log in to your BitBucket account, visit our [class repository](#), and click the “fork” button. This will create a complete working copy of the course repository in *your* BitBucket account. Now, clone *that* new repository from your account into SourceLair instead of ours. With this setup you have the ability to commit and version your work to your own repository as you go! If you don't choose this route, don't worry. SourceLair will save all your work just as you would expect with a basic text editor.

In either case, your screen will look very similar to the one below once you have cloned the repository:



Notice that on the left we have a tree of all our source code. We have a large editor space on the right. On the bottom you will see an area for messages. This area will present you Logs (the default) or host a Terminal. The Terminal may not be displayed when you launch SourceLair, but you're going to want it. Let's make that happen!

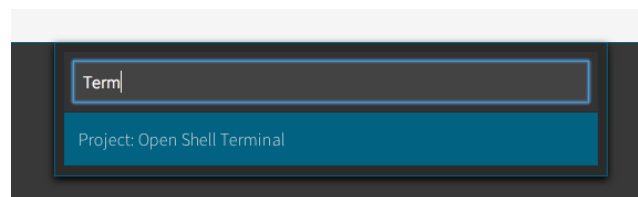
Check out the dropdown list on the right upper corner (pictured below).



See the item that says Keyboard shortcuts? These are going to be your friend. Select the keyboard shortcuts list entry, and you will see the beautiful list shown on the next page.



The keyboard shortcut we are interested in at this moment is “Show Command Palette.” This expands a dropdown of all the operations available to you. Press Command+Shift+P, and you will see a popup. Start to type “terminal”, and it will quickly narrow down your search to “Project: Open Shell Terminal”.



That’s the one you want. Press Enter (↵). This will give you the handy terminal (see below).

```
Hello mvandervoord and welcome to your fully-featured terminal.  
Have fun coding on the cloud.  
  
mvandervoord@awesomesauce:/mnt/project$
```

You can then type `cd unit1` followed by `make` to verify everything is set up properly. You should see this important line of output (pictured below):

```
1 Tests 0 Failures 1 Ignored
```

```
mvandervoord@awesomesauce:/mnt/project$ cd unit1
mvandervoord@awesomesauce:/mnt/project/unit1$ make
mkdir -p build/
gcc -c -I. -I../unity/ -DTEST testUnit1.c -o build/testUnit1.o
gcc -c -I. -I../unity/ -DTEST ../unity/unity.c -o build/unity.o
gcc -o build/test.out build/testUnit1.o build/unity.o
./build/test.out

testUnit1.c:6:test_TheFirst:IGNORE: Implement Me
-----
1 Tests 0 Failures 1 Ignored
OK
mvandervoord@awesomesauce:/mnt/project/unit1$
```

Sadly, if your experience is like ours, on occasion this handy terminal dies on you... usually after you haven't used it for a while (beware of your machine going to sleep). If this ever happens, close the terminal and launch it again. It should work just fine.

Now that you have cloned our class repo, you can make whatever changes you want, saving your work as you go. Your work will be stored in your SourceLair account and will be available next time you log into SourceLair.

If you forked the repo on BitBucket, you can also push your changes to this repo using the `git push` command (If you'd like a quick primer on using Git, again see the document *Introduction to Revision Control and Source Code Management Plus A Super Tiny Git Reference Guide*). As we mentioned before, this isn't a necessary step for this course, but it's good practice for real projects.

At this point, you should know your way around SourceLair well enough to continue this course. Good luck!