HOUSEBLEND

Home / Blog

How to use Atom with NetSuite

September 17, 2024

Introduction

When starting to work with NetSuite, inevitably there will come a day where you will need to create a script based on the SuiteCloud framework. There are a few IDE's (Integrated Development Environments) that have compatibility with the SuiteCloud framework and allow for fast and efficient coding.

Why is this important? Because you will want to be properly equipped for NetSuite SuiteCloud development. As a whole, if not properly equipped, it can be quite cumbersome to create scripts for NetSuite.

f you are reading this, the assumption is that you already are using NetSuite in some capacity (either as a customer or as a solution provider) and have started the journey to your first customization. Or, maybe you have been using another IDE and would like to switch to Atom.

In any case, extending the performance and capabilities of NetSuite using SuiteCloud can make a huge difference when well executed.

In this post, we'll cover how to use Atom with NetSuite and how to configure it to use GitHub and an autocomplete plugin to help you with your scripting saga.

What is Atom, and Why Does it Matter?

Atom is a free and open-source text editor for macOS, Linux and Windows. It is preferred by many developers for its great design, clean interface, and the excellent ecosystem of community-built packages.

Most of the extending packages are free and are community-built. Atom is based on Electron, a framework that enables cross-platform desktop applications using Chromium and Node.js. It is written in CoffeeScript and Less.

Atom is a great option as it contains a built-in connector to GitHub using Git as well as some community packages for NetSuite Autocomplete. Those two factors alone make it a viable candidate for any organization to adopt and use as their development IDE.

How to Setup Atom for NetSuite

Setting up Atom to use with NetSuite is fairly straightforward.

Download Atom

The first step is to download Atom. Navigate to this website and download the most recent version of Atom.

Installing Atom on Windows

Atom is available with Windows installers that can be downloaded here or from the Atom releases page. Use **AtomSetup.exe** for 32-bit systems and **AtomSetup-x64.exe** for 64-bit systems. This setup program will install Atom, add the **atom** and

apm commands to your **path**, and create shortcuts on the desktop and in the start menu.

The context menu **open with Atom** in File Explorer, and the option to make Atom available for file association using **Open with...**, is controlled by the System Settings panel as seen above.

With Atom open, click on **File -> Settings**, and then the **system** tab on the left. Check the boxes next to **show in file context menus**, as well as **show in folder context menus**. And you're all set.

Installing Atom on Mac

Atom follows the standard Mac zip installation process. You can either press the download button from the site or you can go to the Atom releases page to download the **atom-mac.zip** file explicitly. Once you have that file, you can click on it to extract the application and then drag the new **Atom** application into your "Applications" folder.

When you first open Atom, it will try to install the **atom** and **apm** commands for use in the terminal. In some cases, Atom might not be able to install these commands because it needs an administrator password. To check if Atom was able to install the **atom** command, for example, open a terminal window and type **which atom**. If the **atom** command has been installed, you'll see something like this:

If the **atom** command wasn't installed, the **which** command won't return anything:

Updating Atom

Automatically update is enabled by default in Core Settings of the Settings View, which will allow Atom to check for updates automatically.

To perform a manual update:

- Click on the **help -> check for update** menu item in the menu bar.
- Search for **Application: About** in the Command Palette (ctrl-shift-p) and click on the **Check Now** button.

Install the autocomplete-netsuite Package

This package is an autocomplete package for atom that autocompletes functions from the NetSuite Suitescript api.

You can install it by searching for **autocomplete-netsuite** in the install tab in the settings view. More details on the settings view can be read here. Or, you can install via the command use with: **apm install autocomplete-netsuite**.

Some of the features included are:

- SuiteScript 1.0 support
  - All nlapi* functions
  - All nlobj* constructors and member functions
- SuiteScript 2.0 support
  - All member functions
  - All log and util global objects
  - All define and require global functions
  - All **record.type** enum values (ex: **record.type.employee**)

It also comes with a nifty NetSuite File Header comment snippet. If you type **suiteletcomment** and hit tab, it will insert a template for file header. You can then use tab to cycle through the template options.

Note: this autocomplete only activates on files with the extension type ".js"

Connect Atom with your GitHub Repository

Once you have Atom up and running with autocomplete-netsuite, the next step will be to connect it to your GitHub repository.

If you do not already have a GitHub repository, you will need to create one before moving to the next step.

Once you have your GitHub repository setup, you will see a link like this:

You will need to copy the small link ending in **".git"**. Once you have that copied, navigate back to your Atom and type **"ctrl-shift-p"** and then **"clone"** and you should see:

Select the option **"GitHub: Clone"** and paste the URL you had copied from GitHub.

Once your repository has been cloned in Atom, the next step will be to authorize GitHub to be used with Atom. You will do this by navigating to

github.atom.io/login.

Make sure to copy your token.

Once you have your token, navigate back to Atom and click on bottom right where it says **“GitHub”**, enter your token and click on **“Login”**.

Once you have completed that step, your GitHub tab should look like this:

And there you have it, you now have a completely functional Atom IDE, complete with a NetSuite autocomplete and connected to your GitHub Repository.

2 Tips and Reminders for using Atom with NetSuite

1. Make sure to use the “fetch” and “push” the changes to the script your are making to ensure you are using the most up to date and accurate version.
2. Another package that is quite useful is linter-jshint. This package can be used to validate your javascript files in real time.

Conclusion

If you’ve been using Eclipse and are fed up, this may be a good alternative for you to use. Alternatively, if you’re just starting to script in NetSuite and are looking for a starter package, this will definitely help you out.

Give it a try and let me know what you think!

Are you using another IDE with great results? If you are, let us know below!