{:title “Clojure - Leiningen” :layout :post :date “2016-10-24” :tags [“Clojure” “Code” “Guide”]}

So if you been writing clojure code for a while you may have heard of Leiningen or probably already are using it. Leiningen is great for projects and sometimes it might be confusing at first with all of the options and freedom to customize your project layout. I kind of want to go through some of the options and how they work in your project.clj. Sort of showing you the bare minimum of what you need to know of Leiningen to get started. If you haven't yet installed Leiningen I suggest you do that first as it is currently the de facto standard in project management in clojure. There is a few other ones out there as well which you can look into if Leiningen doesn't find your tastes. In this post I won't really cover everything but if you want an indepth explanation on everything I think the best place would be the github repo of Leiningen. So usually you'll start a fresh project with something like this from the command line.

```
$ lein new myproject
```

Than you'll notice the Leiningen creates a directory called myproject in the directory that you ran the command from. You can now cd to that project and see you have a couple of things. Most importantly a src directory and also a project.clj file. Lets take a look at the project.clj and you'll notice something like this.

```
(defproject myproject "0.1.0-SNAPSHOT"
  :description "FIXME: write description"
  :url "http://example.com/FIXME"
  :license {:name "Eclipse Public License"
            :url "http://www.eclipse.org/legal/epl-v10.html"}
  :dependencies [[org.clojure/clojure "1.8.0"]])
```

The description and url is for when you deploy your project on clojars which helps people find it. The dependencies you may notice is the clojure language itself. So if you want to add dependencies you would do something like this.

```
:dependencies [[org.clojure/clojure "1.8.0"]
               [quil "2.4.0"]]
```

Go ahead and save that and now you can go back to the terminal and run lein deps which will install the dependencies for you which is nice. Even though we aren't going to be using quil in this post just showing you that Leiningen will go fetch it for you. Now lets go ahead and look at our src which you can do by cd. Once into the src directory you'll need to cd once more to find the core.clj which is where the heart of your project is located. This is what your project.clj should look like when you first open it.

```
(ns myproject.core)

(defn foo
  "I don't do a whole lot."
  [x]
  (println x "Hello, World!"))
```

So this is what our core.clj is doing which isn't too interesting but for the purposes of demonstrating how Leiningen runs the main clj file go ahead and change foo to -main which is important because when you run lein run it will search for a -main otherwise it will yell at you. So now it should look like this!

```
(ns myproject.core)

(defn -main
  "I don't do a whole lot."
  [x]
  (println x "Hello, World!"))
```

Now lets run this bad boy by going to the terminal and typing lein run.

```
$ lein run
```

Wait what happened? Its not enough to have a -main function in your clojure file you need to let Leiningen known where it is. So you have to specify that it is in the myproject.core namespace in the project.clj. So cd back to where you have your project.clj and add in the main like so.

```
(defproject myproject "0.1.0-SNAPSHOT"
  :description "FIXME: write description"
  :url "http://example.com/FIXME"
  :license {:name "Eclipse Public License"
            :url "http://www.eclipse.org/legal/epl-v10.html"}
  :dependencies [[org.clojure/clojure "1.8.0"]
                 [quil "2.4.0"]]
  :main myproject.core)
```

Now we can run lein run since we told Leiningen where our -main function is. So hopefully this time it'll work and Leiningen won't yell at us.

```
$ lein run
```

And the result which is what you expected.

```
Hello, World!
```

Thats about it if you want to know more about how you can make your project a standalone jar I have a seperate guide for that.