

Getting Set Up

Requirements

Everyone in the workshop should have a laptop. If it is running Windows, it should be running Windows 7 or above. If it is running OS X, it should be running 10.8 "Mountain Lion" or above. If it is running Linux, you are probably fine.

If anyone attending does not have access to a laptop running one of the above choices, let the instructors know. You can either pair with someone else, or we can provide a virtual machine that you can run if you have a laptop.

What we are installing

By the end of these instructions, you will have the following installed:

- Git, a program for managing your program's code
- Java, a "virtual machine" that Clojure runs atop of
- Leiningen, a tool for running Clojure programs
- Nightcode, an editor for Clojure and other programming languages
- Quil, a Clojure library for creating interactive drawings and animations

Instructions by operating system

Choose your operating system to get setup instructions:

- [OS X](#)
- [Ubuntu \(Linux\)](#)
- [Windows via Chocolatey package manager](#)
- [Windows](#)

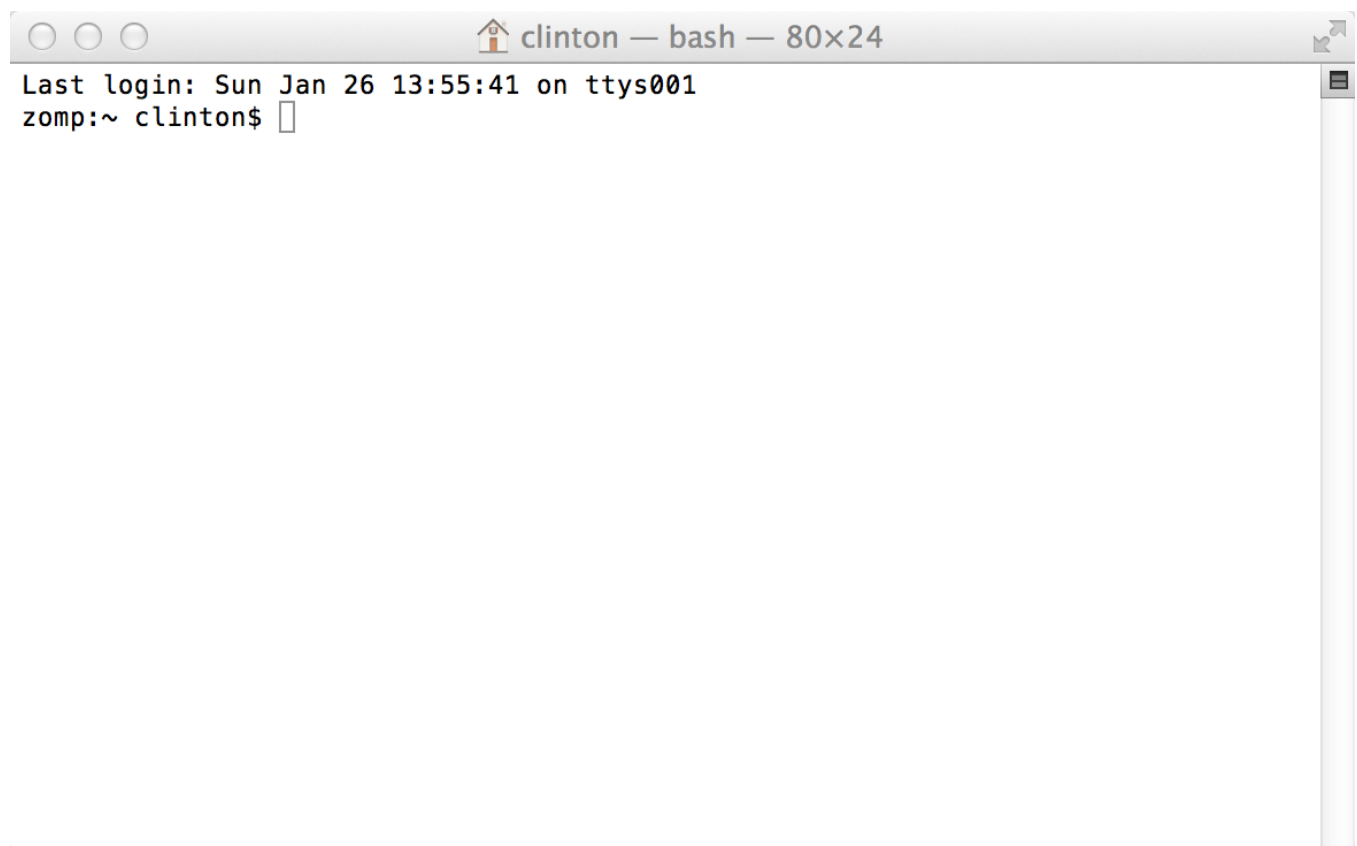
OS X Setup

- Start a terminal
- Install Git
- Configure Git
- Make sure Java is installed
- Install Leiningen
- Install Nightcode
- Test your setup

Starting a terminal

For these instructions, and for much of the class, you will need to have a terminal, or command line, open. This is a text-based interface to talk to your computer, and you can open it by running Terminal app, which is found under `/Applications/Utilities`. If you have never used the terminal before, you may want to spend some time [reading up on command-line basics](#).

Go ahead and open your terminal now. It should look something like this:



The prompt (where you will type your commands) may look different: it usually shows the computer name and user name, as well as the folder or directory you are currently in.

For the rest of this setup, I will tell you to run commands in your terminal. When I say that, I mean "type the command into the terminal and press the Return key."

Install Git

To see if you have git installed type in: `git --version` If you have `git version 1.9.3 (Apple Git-50)` or above you should be fine.

If not, visit git-scm.com. Click "Downloads for Mac". The Git installer may begin downloading automatically. If it does not, click the manual download link. Once the download has finished, open `~/Downloads` in Finder and double-click the downloaded file (named something like **git-2.0.1-intel-universal-snow-leopard.dmg**). This will mount the disk image and open a new Finder window. Double-click the installer package (named something like **git-2.0.1-intel-universal-snow-leopard.pkg**). You may be told that the installer can't be opened because it is from an unidentified developer. If so, click "OK", then right-click (or control-click) the file and select "Open" from the contextual menu. You may be warned again that the installer is from an unidentified developer, but this time you'll have the option to click "Open". Do so. This will launch the installer. Follow its directions, and enter your password when prompted to do so. Once you have finished this process it's safe to unmount the disk image (by clicking the eject button in the Finder sidebar) and delete the file from the Downloads folder.

Configure Git

If you've used Git before then you should already have `user.name` and `user.email` configured. Otherwise, type this in the terminal:

```
git config --global user.name "Your Actual Name"
git config --global user.email "Your Actual Email"
```

TIP: Use the same email address for git, github, and ssh.

Verify by typing this in the terminal:

```
git config --get user.name Expected result: your name
```


```
git config --get user.email Expected result: your email address
```

Making sure Java is installed

If you have OS X version 10.11 (El Capitan), you don't have Java installed. You need to install Java as well. Download Java from <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html> and follow the instruction. If you have a trouble, the article, <http://osxdaily.com/2015/10/17/how-to-install-java-in-os-x-el-capitan/>, may help.

Run `java -version` in your terminal. If you do not have Java installed, OS X will prompt you to install it. Follow all of the directions OS X gives you, then return to this part of the tutorial and run `java -version` again.

If Java is installed, you will see something like this in your terminal:



```
clinton — bash — 80x24
Last login: Sun Jan 26 13:55:41 on ttys001
zomp:~ clinton$ java -version
java version "1.7.0_45"
Java(TM) SE Runtime Environment (build 1.7.0_45-b18)
Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)
zomp:~ clinton$
```

Java's version details

The details of Java's version may differ from what you see above; that is perfectly fine.

Install Leiningen

Leiningen is a tool used on the command line to manage Clojure projects.

To install `lein`, execute the following commands in your terminal; you will be prompted to enter your password for at least the first command starting with `sudo` (The `%` character is a typical commandline prompt, don't type it):

```
% curl https://raw.githubusercontent.com/technomancy/leiningen/stable/bin/lein
% sudo mkdir -p /usr/local/bin/
% sudo mv lein /usr/local/bin/lein
% sudo chmod a+x /usr/local/bin/lein
```

Check that you can now see the command:

```
% which lein
/usr/local/bin/lein
```

If you don't see `/usr/local/bin/lein` as above, do this next:

```
% export PATH=$PATH:/usr/local/bin
```

Now run `which lein` and you should see the `lein` command.

After you set up Leiningen as above, run the `lein version` command. This should take a while to run, as it will download some resources it needs the first time. If it completes successfully, you are golden! If not, ask an instructor for help.

Install Nightcode

Download the latest version for Mac from the [Nightcode site](#) and open the .dmg file to install.

Test your setup

You have set up Java, Leiningen, Nightcode, and Git on your computer--all the tools you will need for this workshop. Before starting, we need to test them out.

Cloning our github repository

Go to your terminal and run the following command:

```
git clone https://github.com/ClojureBridge/welcometoclojurebridge
```

This will clone `welcometoclojurebridge` repository which includes sample Clojure apps. Your terminal should look similar to this picture:

A screenshot of a terminal window with a light gray title bar. The title bar contains three colored window control buttons (red, yellow, green) on the left, a home icon followed by the text 'fish /Users/yoko - fish - 86x24' in the center, and a close button icon on the right. The terminal content shows a user 'yoko@alcyone' running the command 'git clone https://github.com/ClojureBridge/welcometoclojurebridge.git'. The output shows the cloning progress: 'Cloning into 'welcometoclojurebridge'...', 'remote: Counting objects: 325, done.', 'remote: Compressing objects: 100% (18/18), done.', 'remote: Total 325 (delta 5), reused 0 (delta 0), pack-reused 307', 'Receiving objects: 100% (325/325), 949.32 KiB | 0 bytes/s, done.', 'Resolving deltas: 100% (130/130), done.', and 'Checking connectivity... done.'. The prompt 'yoko@alcyone ~>' is followed by a cursor.

```
yoko@alcyone ~> git clone https://github.com/ClojureBridge/welcometoclojurebridge.git
Cloning into 'welcometoclojurebridge'...
remote: Counting objects: 325, done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 325 (delta 5), reused 0 (delta 0), pack-reused 307
Receiving objects: 100% (325/325), 949.32 KiB | 0 bytes/s, done.
Resolving deltas: 100% (130/130), done.
Checking connectivity... done.
yoko@alcyone ~> █
```

Testing `lein repl`

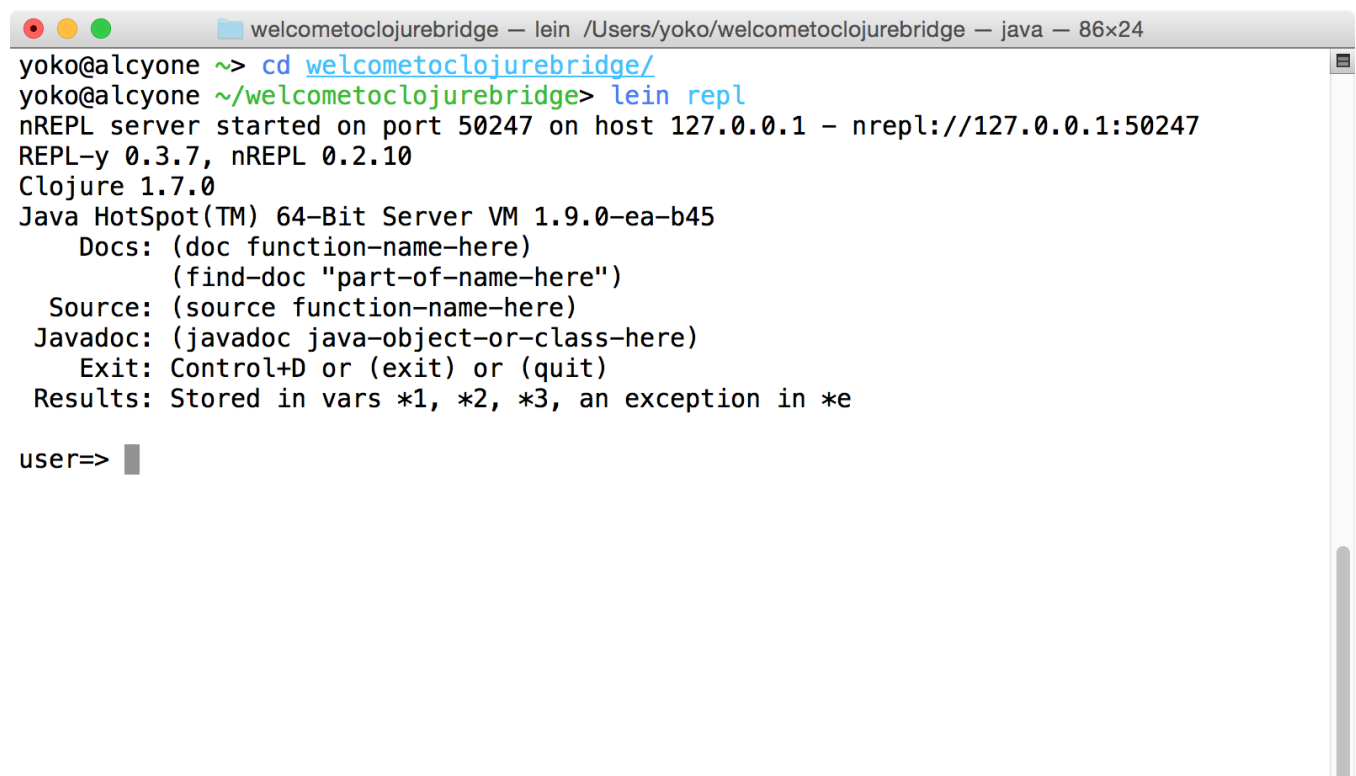
Then run the command:

```
cd welcometoclojurebridge
```

This will take you to the directory with the source code. After that completes, run:

```
lein repl
```

This could take a long time, and will download many other pieces of code apps rely on. You should see lines that start with `Retrieving ...` on your screen. When it finishes, your terminal should look like the following:

A terminal window titled 'welcometoclojurebridge — lein /Users/yoko/welcometoclojurebridge — java — 86x24'. The prompt is 'yoko@alcyone ~>'. The user enters 'cd welcometoclojurebridge/'. The prompt changes to 'yoko@alcyone ~/welcometoclojurebridge>'. The user enters 'lein repl'. The terminal shows the following output: 'nREPL server started on port 50247 on host 127.0.0.1 - nrepl://127.0.0.1:50247', 'REPL-y 0.3.7, nREPL 0.2.10', 'Clojure 1.7.0', 'Java HotSpot(TM) 64-Bit Server VM 1.9.0-ea-b45', 'Docs: (doc function-name-here)', '(find-doc "part-of-name-here")', 'Source: (source function-name-here)', 'Javadoc: (javadoc java-object-or-class-here)', 'Exit: Control+D or (exit) or (quit)', 'Results: Stored in vars *1, *2, *3, an exception in *e'. The prompt is now 'user=>' followed by a cursor.

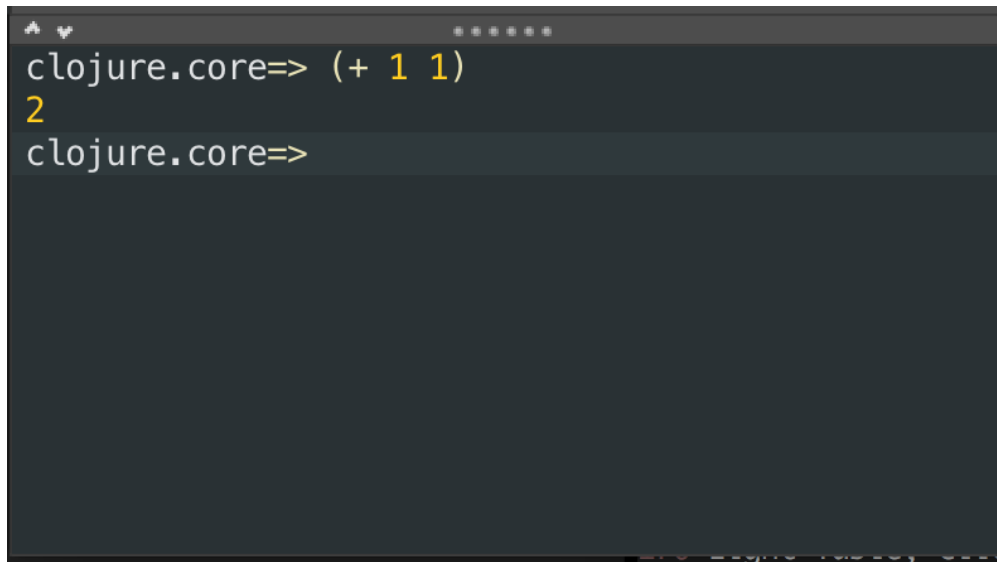
```
yoko@alcyone ~> cd welcometoclojurebridge/
yoko@alcyone ~/welcometoclojurebridge> lein repl
nREPL server started on port 50247 on host 127.0.0.1 - nrepl://127.0.0.1:50247
REPL-y 0.3.7, nREPL 0.2.10
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.9.0-ea-b45
Docs: (doc function-name-here)
      (find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> █
```

This is starting a REPL, which we will learn about soon. It's a special terminal for Clojure. At the REPL prompt, type `(+ 1 1)` and press Return. Did you get the answer `2` back? You will learn more about that in the course. For now, press the Control button and D button on your keyboard together (abbreviated as Ctrl+D). This should take you out of the Clojure REPL and back to your normal terminal prompt. Then, the terminal will show you the following message: `user=> Bye for now!`

Testing Nightcode

Open the Nightcode application. At the bottom left of the screen, type `(+ 1 1)` into the window.

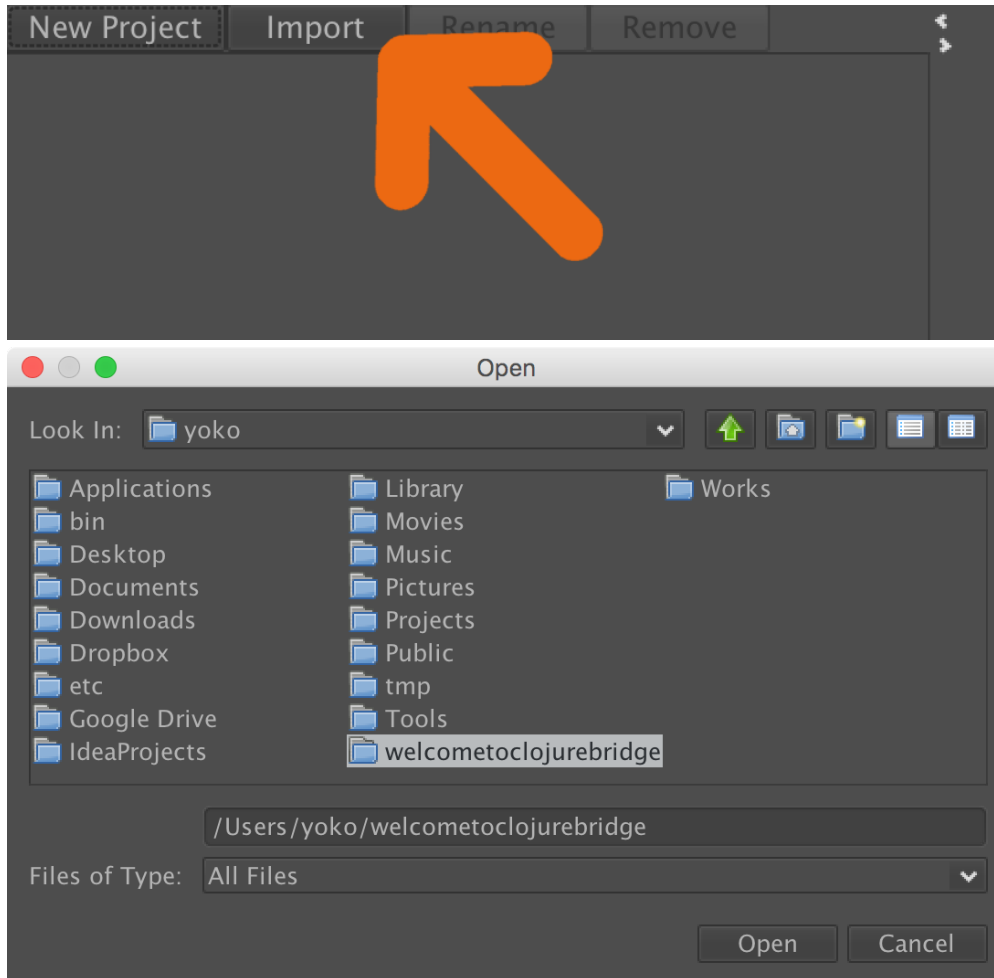
A screenshot of the Nightcode application's REPL window. The window has a dark background with light-colored text. The prompt `clojure.core=>` is followed by the expression `(+ 1 1)` in yellow. Below this, the result `2` is displayed in yellow. The prompt `clojure.core=>` appears again on the next line.

```
clojure.core=> (+ 1 1)
2
clojure.core=>
```

If you see the result, 2, that worked, great!

Testing apps

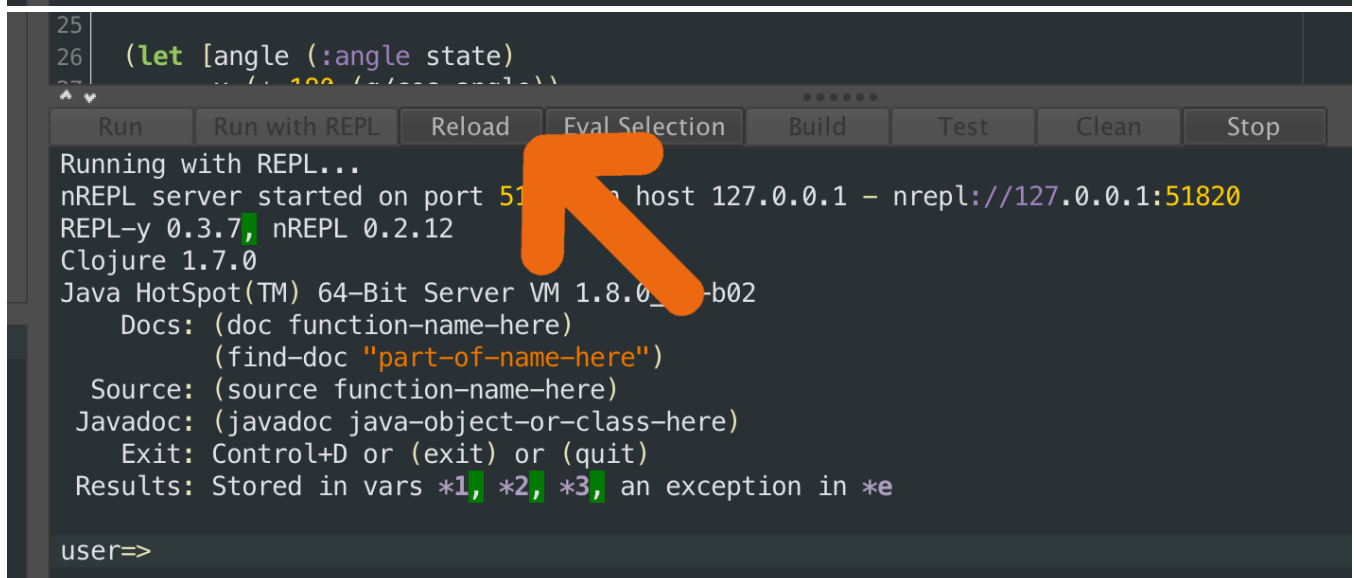
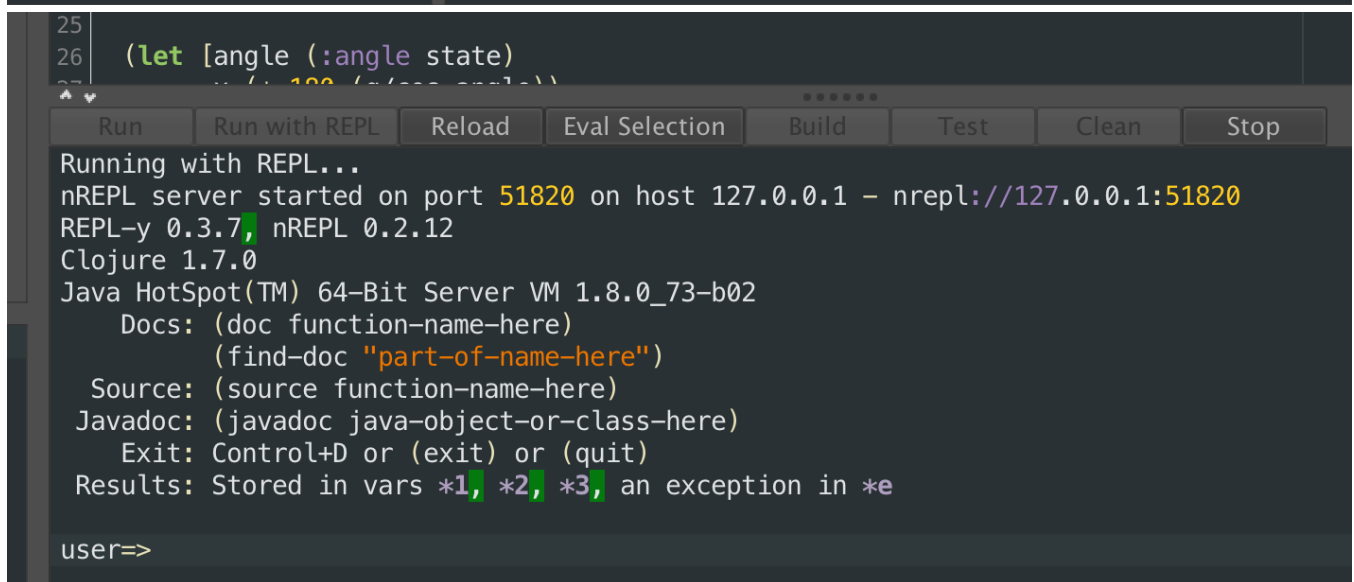
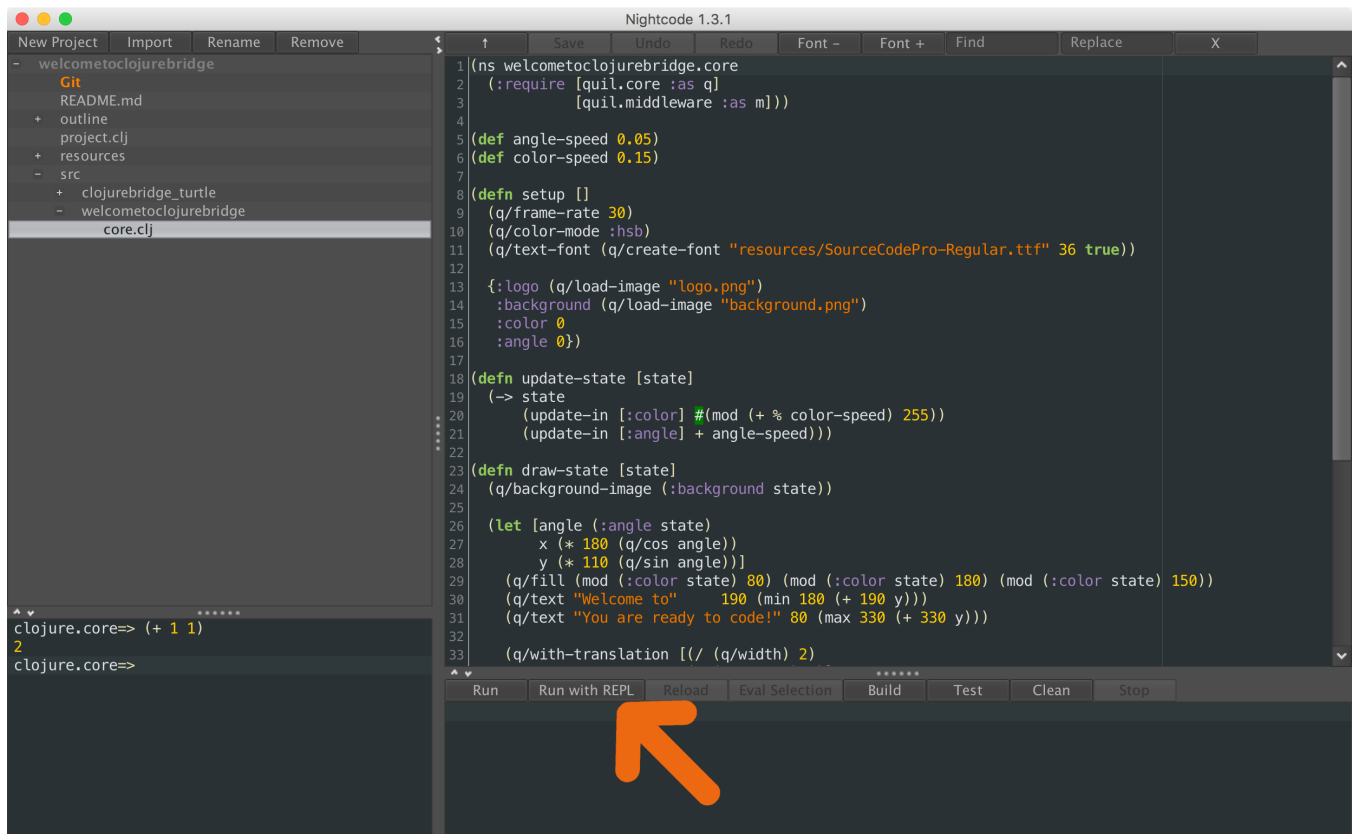
Now we will open and run the sample Clojure apps in Nightcode. On the top left corner, click "Import" then find the directory, `welcometoclojurebridge`, which was created when you ran `git clone` command. Click "Open." In the project directory tree on the left, click on `src` - `welcometoclojurebridge` - `core.clj`. The `core.clj` file will be opened on the right side. This is a Clojure program.



! [Testing apps - core.clj]

(img/nightcode-welcometoclojurebridge-core.png)

The next step is to run the code shown in the window. Click "Run with REPL" on the bottom of the right side. It may take a while. Eventually, repl will start and show a prompt on the bottom of the window. Once, you see the prompt, click "Reload" button.



You should see a fun welcome message.



Let's try one more sample. In the directory tree on the left, click on `welcometoclojurebridge` - `src` - `clojurebridge-turtle` - `walk.clj`. The `walk.clj` file will open on the right side. Like we did before, click "Reload" button.

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
;;
```

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

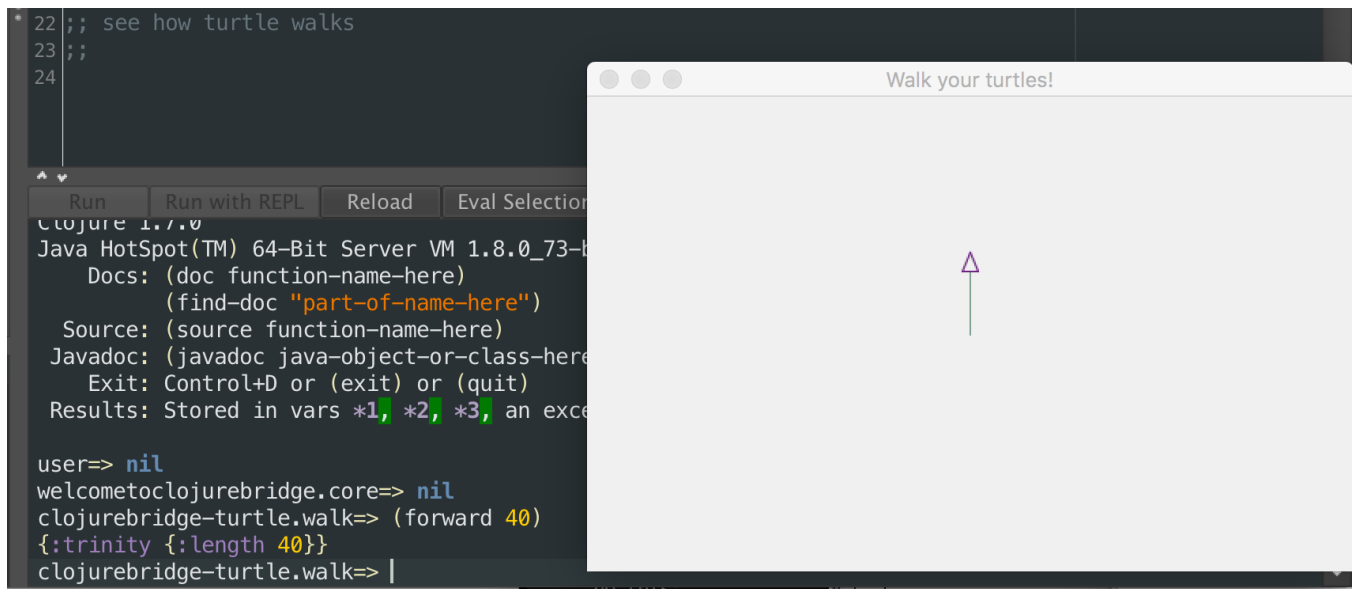
;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
;;
```

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

An initial image of the turtles app should pop up. A small triangle on the center is the *turtle*.

Type `(forward 40)` on the repl at the bottom of the window. You should see the turtle moved upward:

The image shows a Clojure REPL window on the left and a separate application window titled "Walk your turtles!" on the right. The REPL window displays the following code and output:

```
* 22 ;; see how turtle walks
23 ;;
24
```

```
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b
Docs: (doc function-name-here)
      (find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception object

user=> nil
welcometoclojurebridge.core=> nil
clojurebridge-turtle.walk=> (forward 40)
{:trinity {:length 40}}
clojurebridge-turtle.walk=> |
```

The "Walk your turtles!" window is a simple gray rectangle with a small purple triangle pointing upwards in the center, representing a turtle.

Success!

Congratulations! You have opened and run your first Clojure apps, and your install and setup are all completed!

If you want to know what the turtle (*a small triangle*) can do, see [Turtle App API](#) and [How to Walk Turtles](#) for more information.

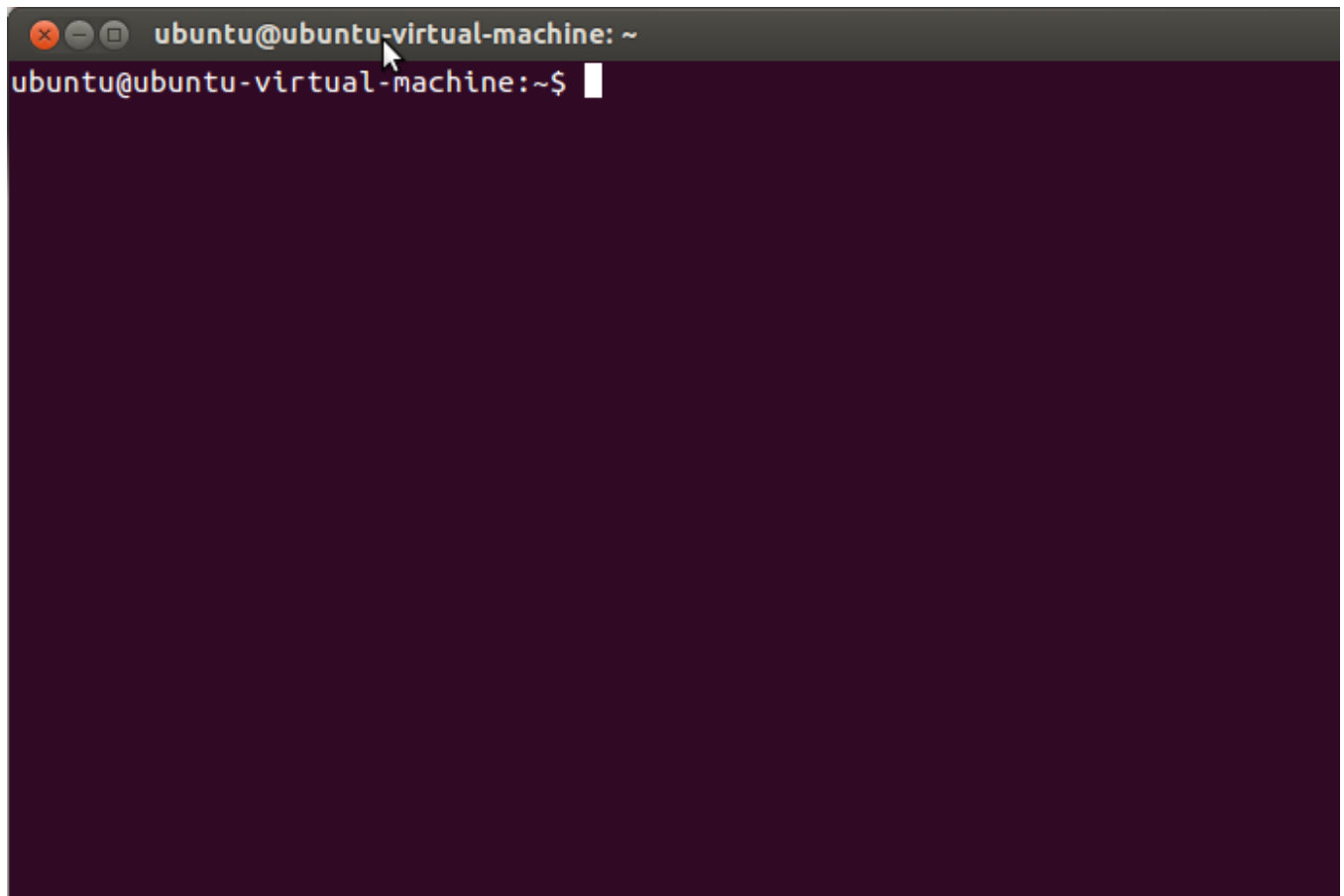
Ubuntu (Linux) Setup

- Start a terminal
- Install Git
- Configure Git
- Make sure Java is installed
- Install Leiningen
- Install Nightcode
- Test your setup

Starting a terminal

For these instructions, and for much of the class, you will need to have a terminal, or command line, open. This is a text-based interface to talk to your computer, and you can open it by clicking "Dash Home" and typing `Terminal`. You can also open a terminal at any time by pressing `CTRL-ALT-T`. If you have never used the terminal before, you may want to spend some time [reading up on command-line basics](#).

Go ahead and open your terminal now. It should look something like this:



The prompt (where you will type your commands) may look different: it usually shows the computer name and user name, as well as the folder or directory you are currently in.

For the rest of this setup, I will tell you to run commands in your terminal. When I say that, I mean "type the command into the terminal and press the Return key."

Installing Git

See if you already have Git installed with `git version`. If the `git` command is not found, install it with this command in the terminal:

```
sudo apt-get install git
```

Configure Git

If you've used Git before then you should already have `user.name` and `user.email` configured. Otherwise, type this in the terminal:

```
git config --global user.name "Your Actual Name"  
git config --global user.email "Your Actual Email"
```

TIP: Use the same email address for git, github, and ssh.

Verify by typing this in the terminal:

```
git config --get user.name Expected result: your name
```

```
git config --get user.email Expected result: your email address
```

Making sure Java is installed

Run `java -version` in your terminal. If you do not have Java installed, Ubuntu will prompt you to install it. It should look something like this:

```
ubuntu@ubuntu-virtual-machine: ~  
ubuntu@ubuntu-virtual-machine:~$ java -version  
The program 'java' can be found in the following packages:  
* default-jre  
* gcj-4.6-jre-headless  
* openjdk-6-jre-headless  
* gcj-4.5-jre-headless  
* openjdk-7-jre-headless  
Try: sudo apt-get install <selected package>  
ubuntu@ubuntu-virtual-machine:~$
```

Follow all of the directions Ubuntu gives you, selecting the package "openjdk-7-jre" then return to this part of the tutorial and run `java -version` again.

If Java is installed, you will see something like this in your terminal:

```
ubuntu@ubuntu-virtual-machine: ~  
java version "1.6.0_30"  
OpenJDK Runtime Environment (IcedTea6 1.13.1) (6b30-1.13.1-1ubuntu2~0.12.04.1)  
OpenJDK 64-Bit Server VM (build 23.25-b01, mixed mode)  
ubuntu@ubuntu-virtual-machine:~$
```

The details of Java's version may differ from what you see above; that is perfectly fine.

Install Leiningen

Leiningen is a tool used on the command line to manage Clojure projects.

Go to the [Leiningen website](#). You will see a link to the `lein` script under the "Install" heading. Right-click that link and choose "Save Link As...". Save it in your Downloads directory.



Leiningen

for automating Clojure projects without setting your hair on fire

[install](#) | [docs](#) | [community](#) | [source](#)

Install

1. Download the `lein script` (or on Windows `lein.bat`)
2. Place it on your \$PATH where your shell can find it (eg. `~/bin`)
3. Set it to be executable (`chmod a+x ~/bin/lein`)
4. Run it (`lein`) and it will download the self-install package

You can check your [package manager](#) as well, but be sure you get version 2.x. There's also an [installer for Windows users](#).

After that, run the following commands in your terminal. You will be prompted to enter your password.

```
sudo mkdir -p /usr/local/bin/  
sudo mv ~/Downloads/lein* /usr/local/bin/lein  
sudo chmod a+x /usr/local/bin/lein  
export PATH=$PATH:/usr/local/bin
```

After you run the above commands, run the `lein version` command. It should take a while to run, as it will download some resources it needs the first time. If it completes successfully, you are golden! If not, ask an instructor for help.

Install Nightcode

Download the latest version for Linux (deb) from the [Nightcode site](#).

```
sudo dpkg -i Nightcode-2.3.3.deb
```

Replace 2.3.3 with the version number downloaded.

If you are using Fedora, Red Hat or the like, download the Linux (rpm) version.

```
rpm -ivh Nightcode-2.3.3.rpm
```

If the above are not usable on your computer, download the Jar version and run using the `java` command:

```
cd ~/Downloads/  
java -jar Nightcode-2.3.3.jar
```

Testing your setup

You have set up Java, Leiningen, Nightcode, and Git on your computer--all the tools you will need for this course. Before starting, we need to test them out.

Cloning out github repository

Go to your terminal and run the following command:

```
git clone https://github.com/ClojureBridge/welcometoclojurebridge
```

This will clone `welcometoclojurebridge` repository which includes sample Clojure apps. Your terminal should look similar to this picture:

```
clojurebridge@Wolf: ~
clojurebridge@Wolf:~$ git clone https://github.com/ClojureBridge/welcometoclojurebridge
Cloning into 'welcometoclojurebridge'...
remote: Counting objects: 15, done.
remote: Total 15 (delta 0), reused 0 (delta 0), pack-reused 15
Unpacking objects: 100% (15/15), done.
Checking connectivity... done.
clojurebridge@Wolf:~$
```

Testing `lein repl`

Then run the command:

```
cd welcometoclojurebridge
```

This will take you to the directory with the source code. After that completes, run:

```
lein repl
```

This could take a long time, and will download many other pieces of code it relies on. You should see lines that start with `Retrieving ...` on your screen. When it finishes, your terminal should look like the following:

```
clojurebridge@Wolf: ~/welcometoclojurebridge
Retrieving bouncycastle/bcmail-jdk14/138/bcmail-jdk14-138.jar from central
Retrieving org/bouncycastle/bcmail-jdk14/1.38/bcmail-jdk14-1.38.jar from central
Retrieving org/clojure/tools.nrepl/0.2.6/tools.nrepl-0.2.6.jar from central
Retrieving quil/quil/2.2.4/quil-2.2.4.jar from clojars
Retrieving quil/processing-pdf/2.2.1/processing-pdf-2.2.1.jar from clojars
Retrieving quil/processing-dxf/2.2.1/processing-dxf-2.2.1.jar from clojars
Retrieving quil/processing-core/2.2.1/processing-core-2.2.1.jar from clojars
Retrieving quil/jogl-all-fat/2.1.5/jogl-all-fat-2.1.5.jar from clojars
Retrieving quil/gluegen-rt-fat/2.1.5/gluegen-rt-fat-2.1.5.jar from clojars
Retrieving clojure-complete/clojure-complete/0.2.3/clojure-complete-0.2.3.jar fr
om clojars
Retrieving quil/processing-js/1.4.8/processing-js-1.4.8.jar from clojars
nREPL server started on port 39473 on host 127.0.0.1 - nrepl://127.0.0.1:39473
REPL-y 0.3.5, nREPL 0.2.6
Clojure 1.6.0
OpenJDK Server VM 1.7.0_79-b14
  Docs: (doc function-name-here)
        (find-doc "part-of-name-here")
  Source: (source function-name-here)
  Javadoc: (javadoc java-object-or-class-here)
  Exit: Control+D or (exit) or (quit)
  Results: Stored in vars *1, *2, *3, an exception in *e
user=> 
```

This is starting a REPL, which we will learn about soon. It's a special terminal for Clojure. At the REPL prompt, type `(+ 1 1)` and press Return. Did you get the answer `2` back? You will learn more about that in the course. For now, press the Control button and D button on your keyboard together (abbreviated as Ctrl+D). This should take you out of the Clojure REPL and back to your normal terminal prompt. Then, the terminal will show you the following message: `user=> Bye for now!`

Testing Nightcode

If Nightcode isn't started yet or closed, open it by typing the command on terminal:

```
java -jar nightcode-1.3.2-standalone.jar
```

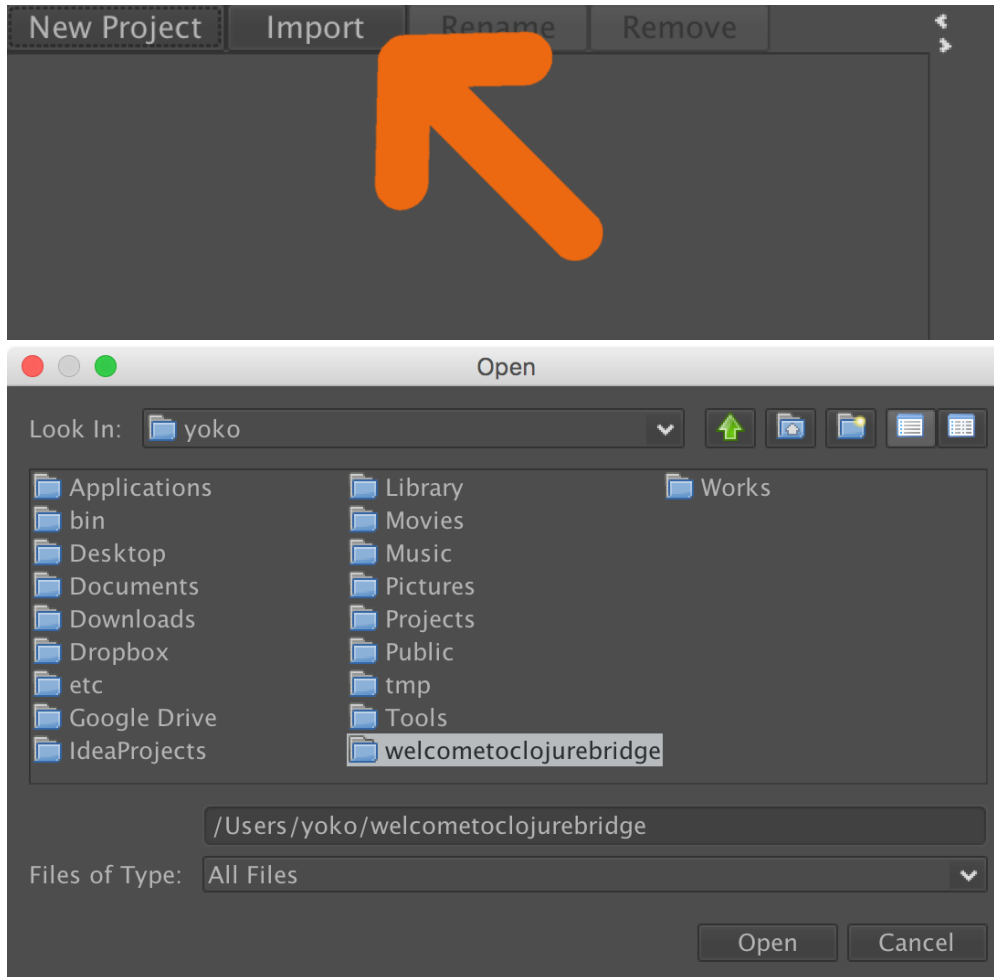
At the bottom left of the screen, type `(+ 1 1)` into the window. It should look like the following image:

```
clojure.core=> (+ 1 1)
2
clojure.core=>
```

If you see the result, 2, that worked, great!

Testing apps

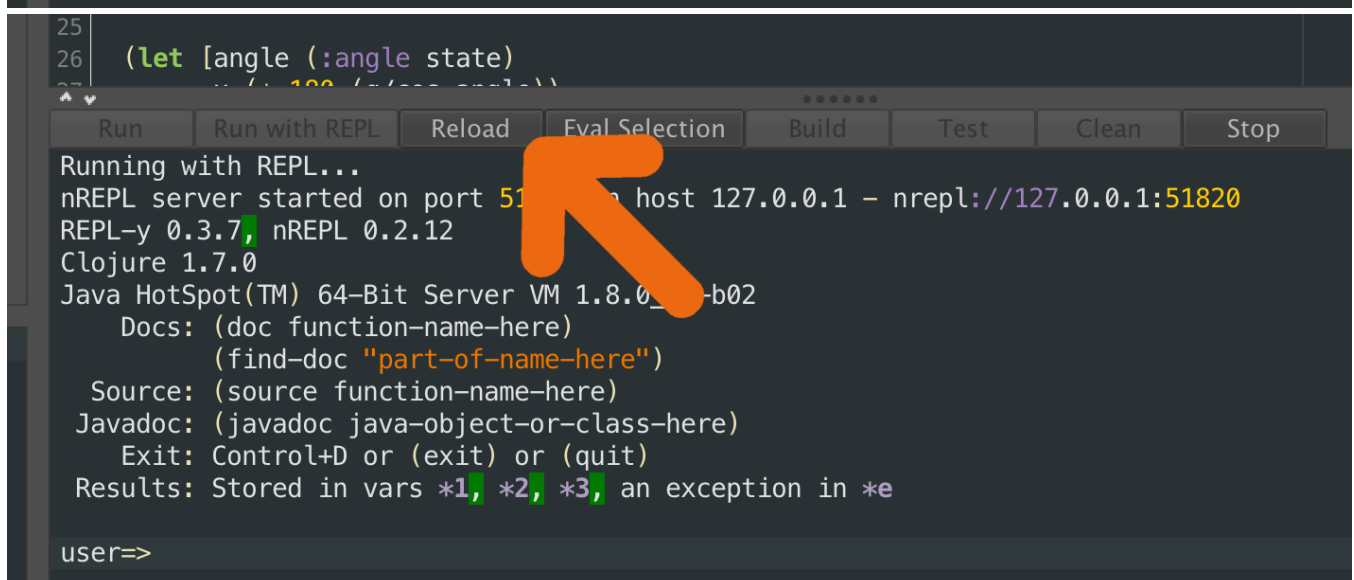
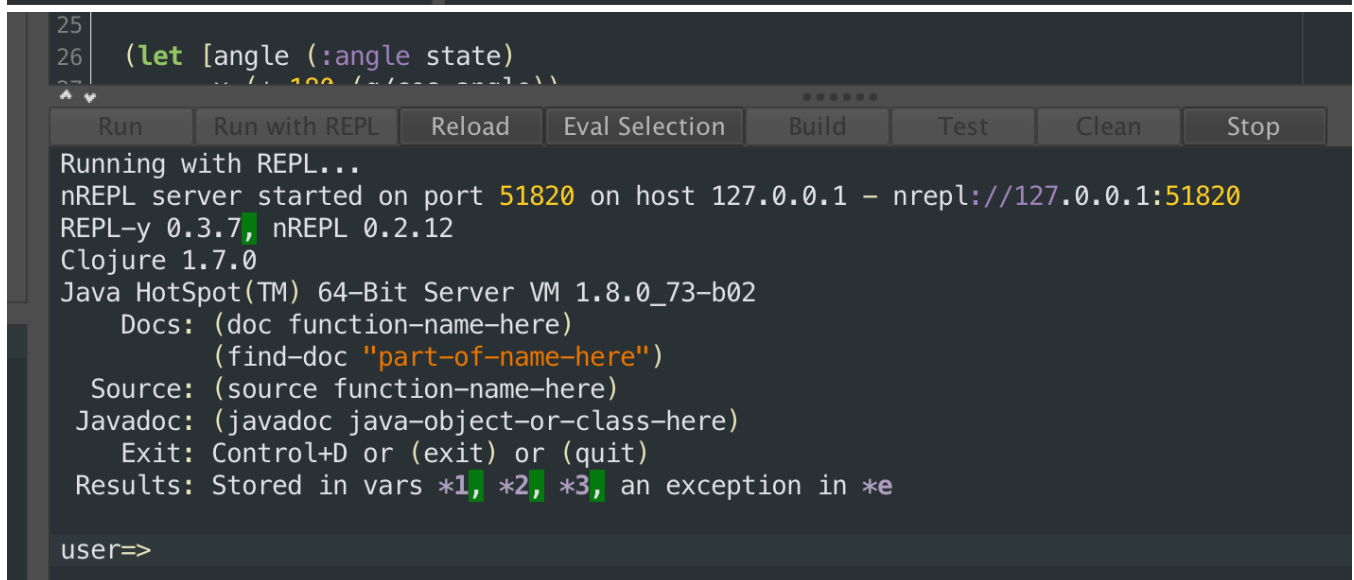
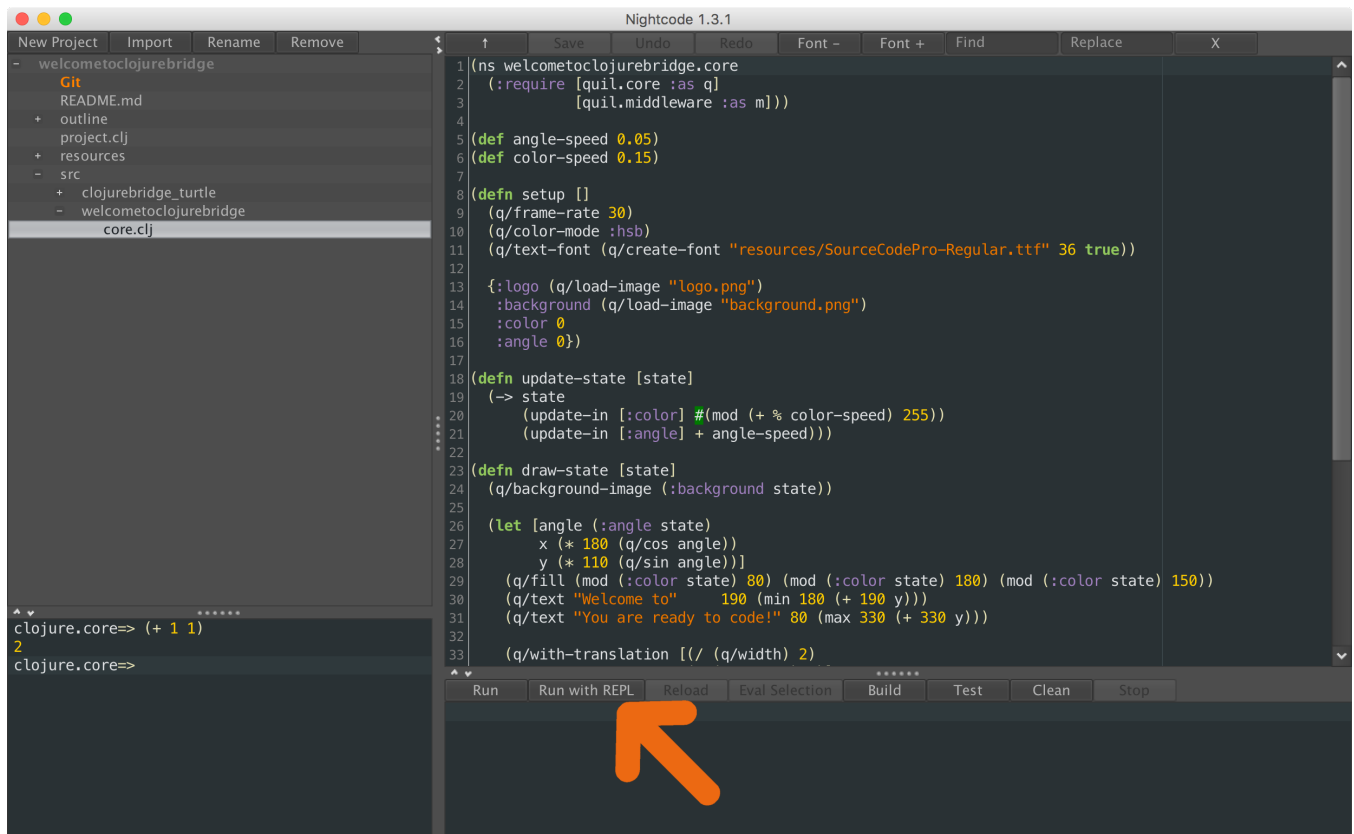
Now we will open and run the sample Clojure apps in Nightcode. On the top left corner, click "Import" then find the directory, `welcometoclojurebridge`, which was created when you ran `git clone` command. Click "Open." In the project directory tree on the left, click on `src` - `welcometoclojurebridge` - `core.clj`. The `core.clj` file will be opened on the right side. This is a Clojure program.



! [Testing apps - core.clj]

(img/nightcode-welcometoclojurebridge-core.png)

The next step is to run the code shown in the window. Click "Run with REPL" on the bottom of the right side. It may take a while. Eventually, repl will start and show a prompt on the bottom of the window. Once, you see the prompt, click "Reload" button.



You should see a fun welcome message.



Let's try one more sample. In the directory tree on the left, click on `welcometoclojurebridge` - `src` - `clojurebridge-turtle` - `walk.clj`. The `walk.clj` file will open on the right side. Like we did before, click "Reload" button.

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
;;
```

Run Run with REPL Reload Eval Selection Build Test Clean Stop

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
;;
```

Run Run with REPL Reload Eval Selection Build Test Clean Stop

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

An initial image of the turtles app should pop up. A small triangle on the center is the *turtle*.

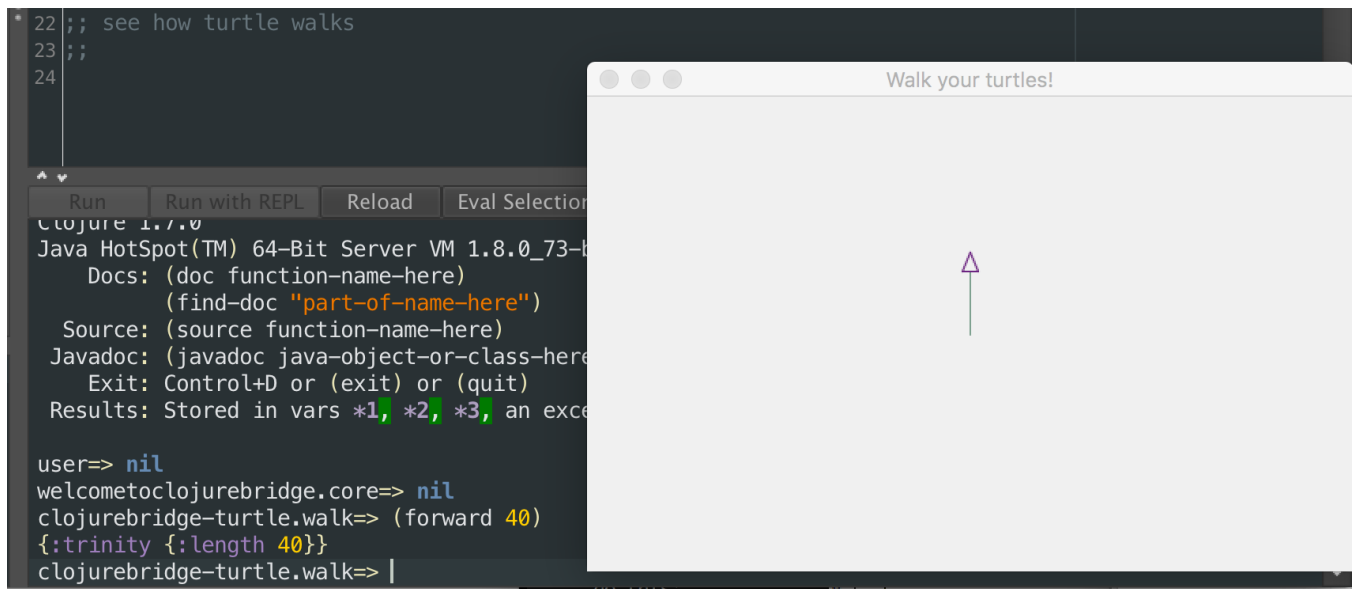
Type `(forward 40)` on the repl at the bottom of the window. You should see the turtle moved upward:

```
* 22 ;; see how turtle walks
23 ;;
24
```

Run Run with REPL Reload Eval Selection

Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b01
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception object

```
user=> nil
welcometoclojurebridge.core=> nil
clojurebridge-turtle.walk=> (forward 40)
{:trinity {:length 40}}
clojurebridge-turtle.walk=> |
```



Success!

Congratulations! You have opened and run your first Clojure apps, and your install and setup are all completed!

If you want to know what the turtle (*a small triangle*) can do, see [Turtle App API](#) and [How to Walk Turtles](#) for more information.

Windows via Chocolatey package manager

[Chocolatey](#) is a Windows package manager providing installation facilities similar to Mac and Linux. "The sane way to manage software on Windows"

- Install Chocolatey
- Start a command prompt
- Install Git
- Configure Git
- Install Java
- Install Leiningen
- Install Nightcode
- Test your setup
- Troubleshooting

Install Chocolatey

Follow the instructions on Chocolatey's [install page](#).

Make sure you are in an *administrative shell* as detailed in their first step. After installing Chocolatey, close and open a new administrative shell in order to use it.

Installing Git

See if you already have Git installed at the command prompt with the command `git --version`.

If you need to install it, open an administrative shell and type:

```
choco install git --params "/GitAndUnixToolsOnPath /NoAutoCrlf"
```

Close and open the administrative shell, so it can pick the updated PATH environment.

After installation, try the `git --version` command in a new command prompt window. If you see a version number, git was installed correctly.

If you've used Git before then you should already have user.name and user.email configured. Otherwise, type this in the command prompt:

Configure Git

```
git config --global user.name "Your Actual Name"
git config --global user.email "Your Actual Email"
```

TIP: Use the same email address for git, github, and ssh.

Verify by typing this in the command prompt:

```
git config --get user.name
```

 Expected result: `your name`

```
git config --get user.email
```

 Expected result: `your email address`

Install Java

To install, open an administrative shell and type:

```
choco install jdk8
```

After installation, try the `java -version` command in a new command prompt window. If you see a version number, java was installed correctly.

Install Leiningen

Leiningen is a tool used on the command line to manage Clojure projects.

Download and run the [installer](#) for Windows.

Install Nightcode

Go to the [Nightcode](#) site and get the latest version for Windows.

Test your setup

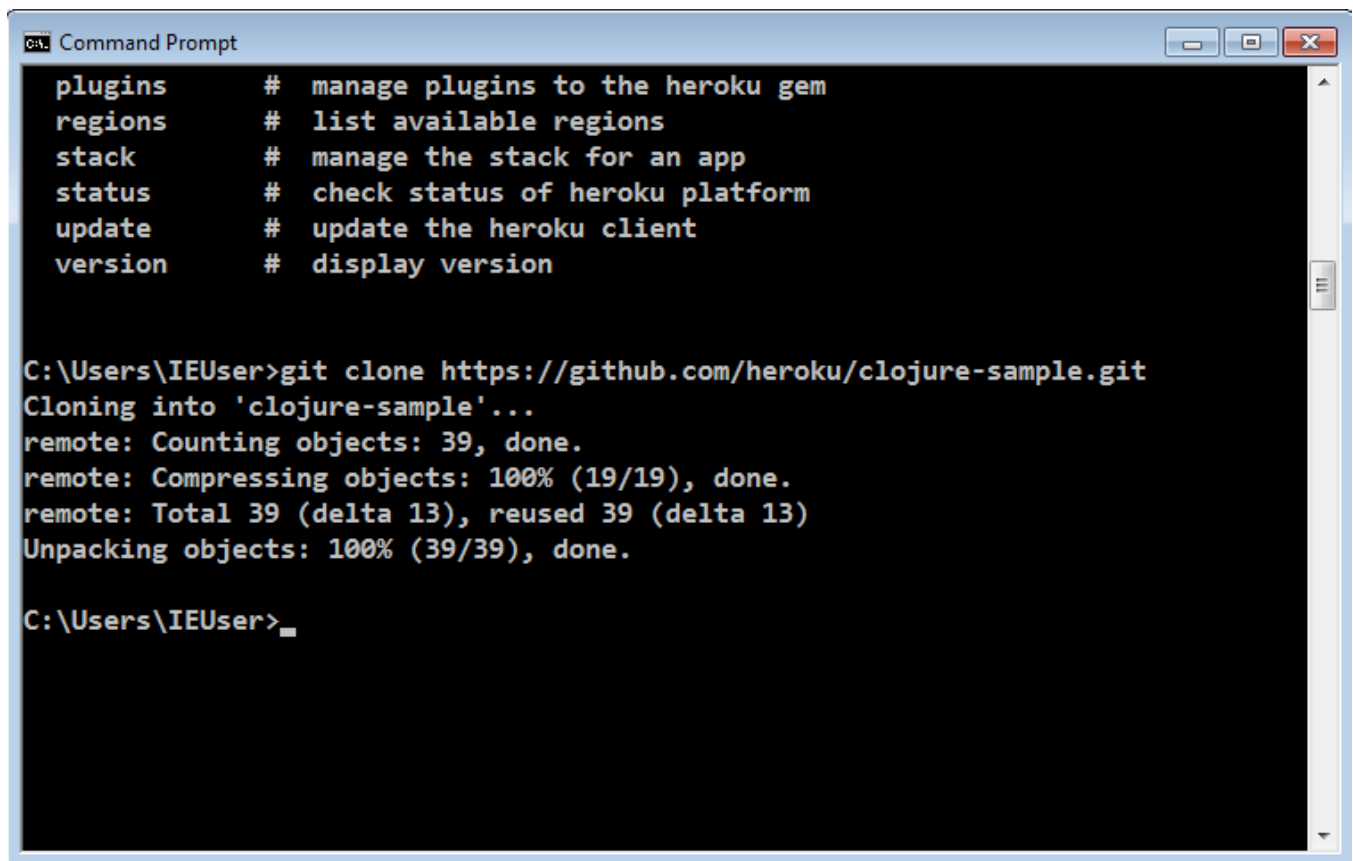
You have set up Java, Leiningen, Nightcode, and Git on your computer--all the tools you will need for this workshop. Before starting, we need to test them out.

Cloning our github repository

Go to your command prompt and run the following command:

```
git clone https://github.com/ClojureBridge/welcometoclojurebridge
```

This will clone `welcometoclojurebridge` repository which includes sample Clojure apps. Your command prompt should look similar to this picture:



```
CA. Command Prompt

plugins      # manage plugins to the heroku gem
regions      # list available regions
stack        # manage the stack for an app
status       # check status of heroku platform
update       # update the heroku client
version      # display version

C:\Users\IEUser>git clone https://github.com/heroku/clojure-sample.git
Cloning into 'clojure-sample'...
remote: Counting objects: 39, done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 39 (delta 13), reused 39 (delta 13)
Unpacking objects: 100% (39/39), done.

C:\Users\IEUser>
```

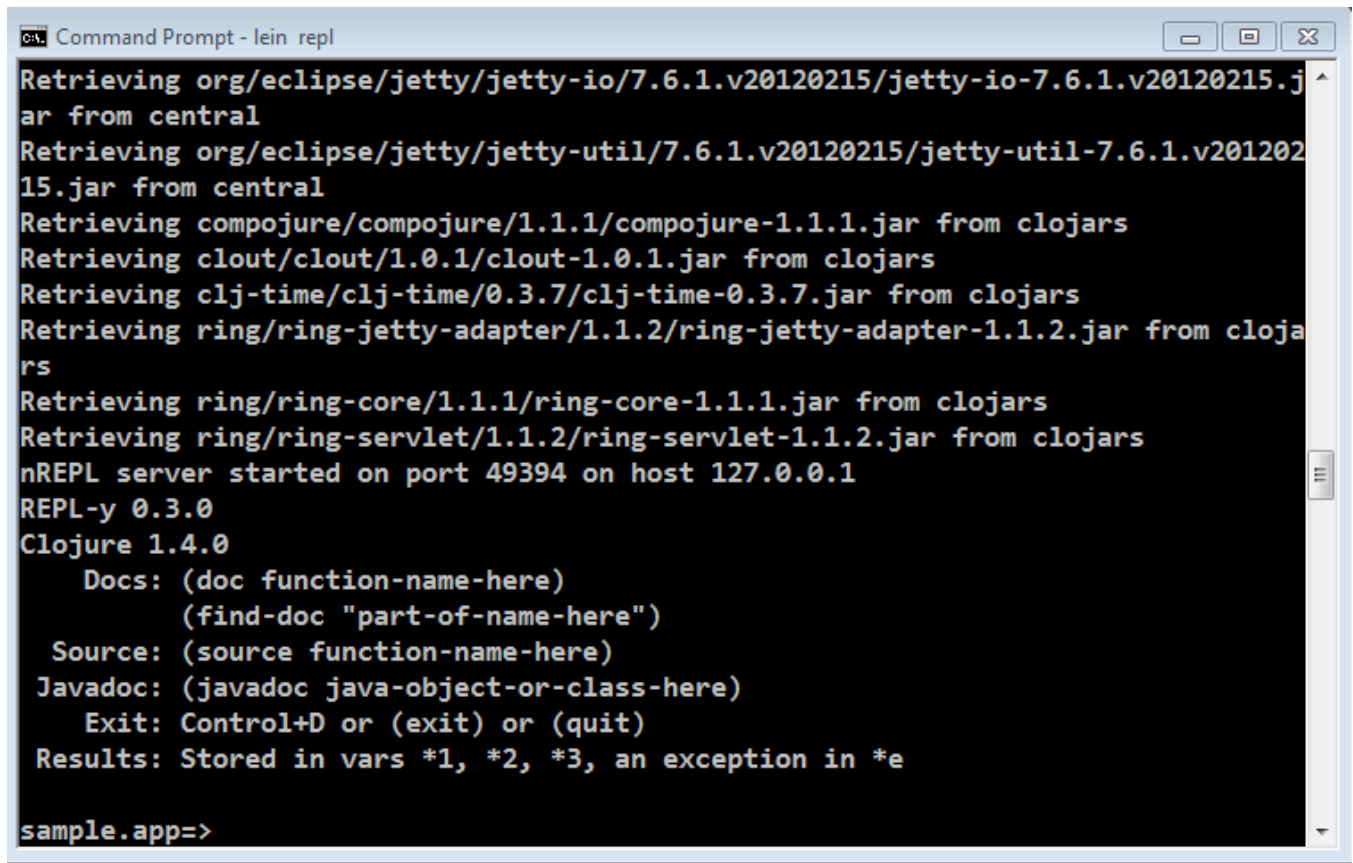
Then run the command:

```
cd welcometoclojurebridge
```

This will take you to the folder with the source code. After that completes, run:

```
lein repl
```

This could take a long time, and will download many other pieces of code it relies on. You should see lines that start with `Retrieving ...` on your screen. When it finishes, your command prompt should look like the following:



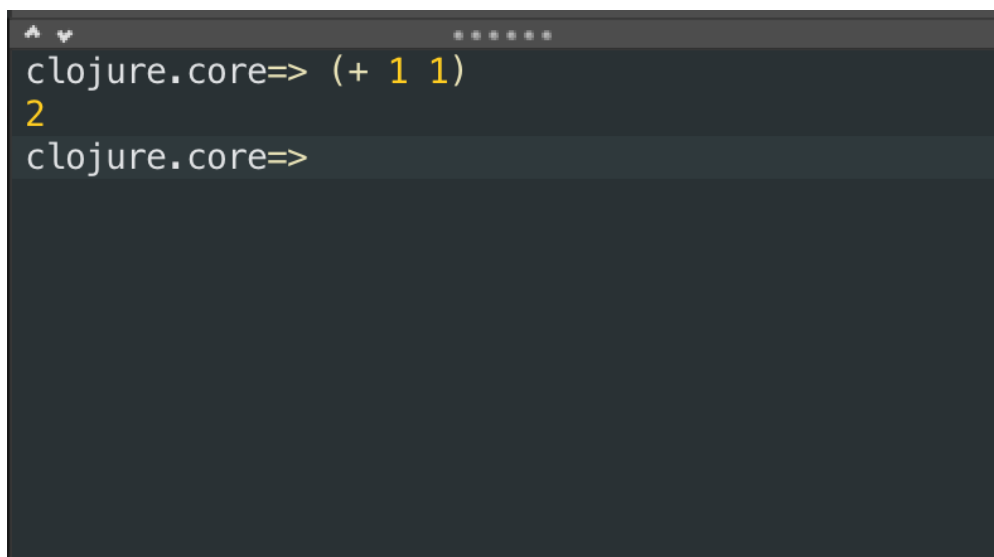
```
C:\> Command Prompt - lein repl
Retrieving org/eclipse/jetty/jetty-io/7.6.1.v20120215/jetty-io-7.6.1.v20120215.jar from central
Retrieving org/eclipse/jetty/jetty-util/7.6.1.v20120215/jetty-util-7.6.1.v20120215.jar from central
Retrieving compojure/compojure/1.1.1/compojure-1.1.1.jar from clojars
Retrieving clout/clout/1.0.1/clout-1.0.1.jar from clojars
Retrieving clj-time/clj-time/0.3.7/clj-time-0.3.7.jar from clojars
Retrieving ring/ring-jetty-adapter/1.1.2/ring-jetty-adapter-1.1.2.jar from clojars
Retrieving ring/ring-core/1.1.1/ring-core-1.1.1.jar from clojars
Retrieving ring/ring-servlet/1.1.2/ring-servlet-1.1.2.jar from clojars
nREPL server started on port 49394 on host 127.0.0.1
REPL-y 0.3.0
Clojure 1.4.0
  Docs: (doc function-name-here)
        (find-doc "part-of-name-here")
  Source: (source function-name-here)
  Javadoc: (javadoc java-object-or-class-here)
  Exit: Control+D or (exit) or (quit)
  Results: Stored in vars *1, *2, *3, an exception in *e
sample.app=>
```

This is starting a REPL, which we will learn about soon. It's a special command prompt for Clojure. At the REPL prompt, type `(+ 1 1)` and hit enter. Did you get the answer `2` back? You will learn more about that in the course. For now, press the Control button and D button on your keyboard together (abbreviated as Ctrl+D). This should take you out of the Clojure REPL and back to your normal command prompt. Then, the command prompt will show you the following message: `user=> Bye for now!`

Testing Nightcode

Open Nightcode

At the bottom left of the screen, type `(+ 1 1)` into the window. It should look like the following image:

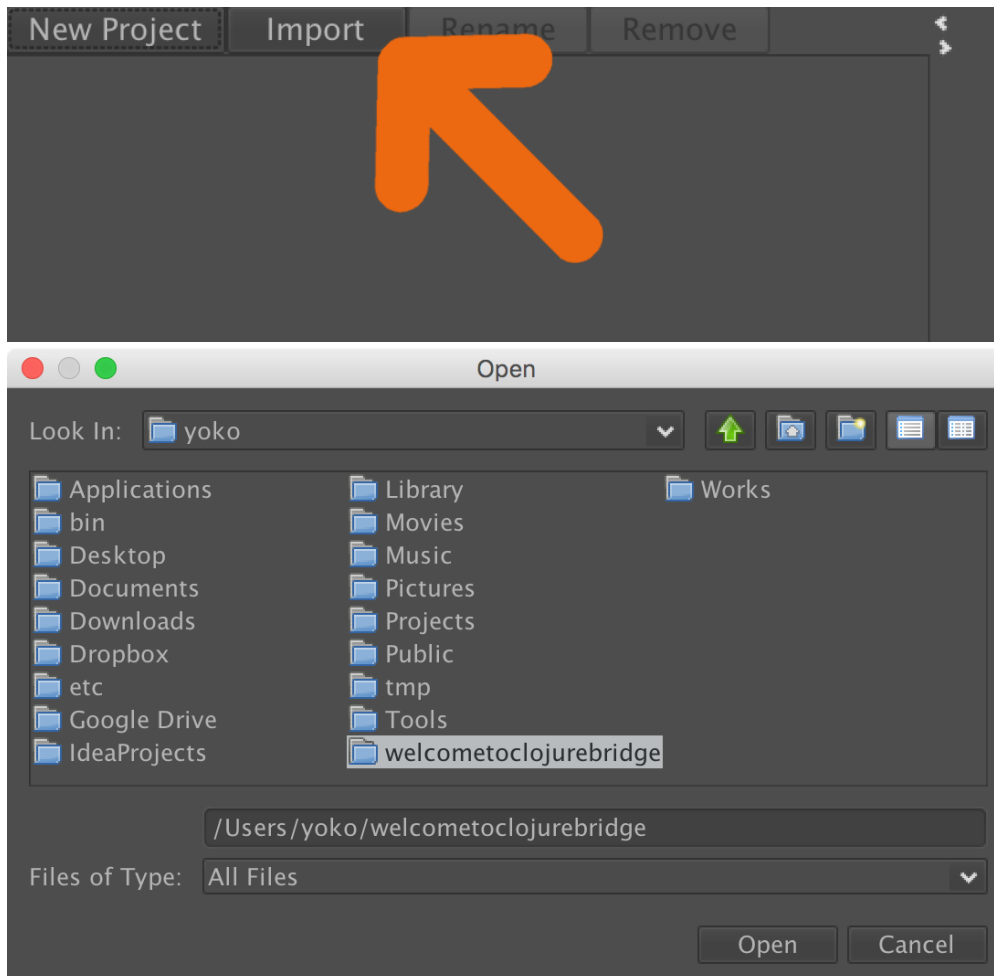


```
clojure.core=> (+ 1 1)
2
clojure.core=>
```

If you see the result, 2, that worked, great!

Testing apps

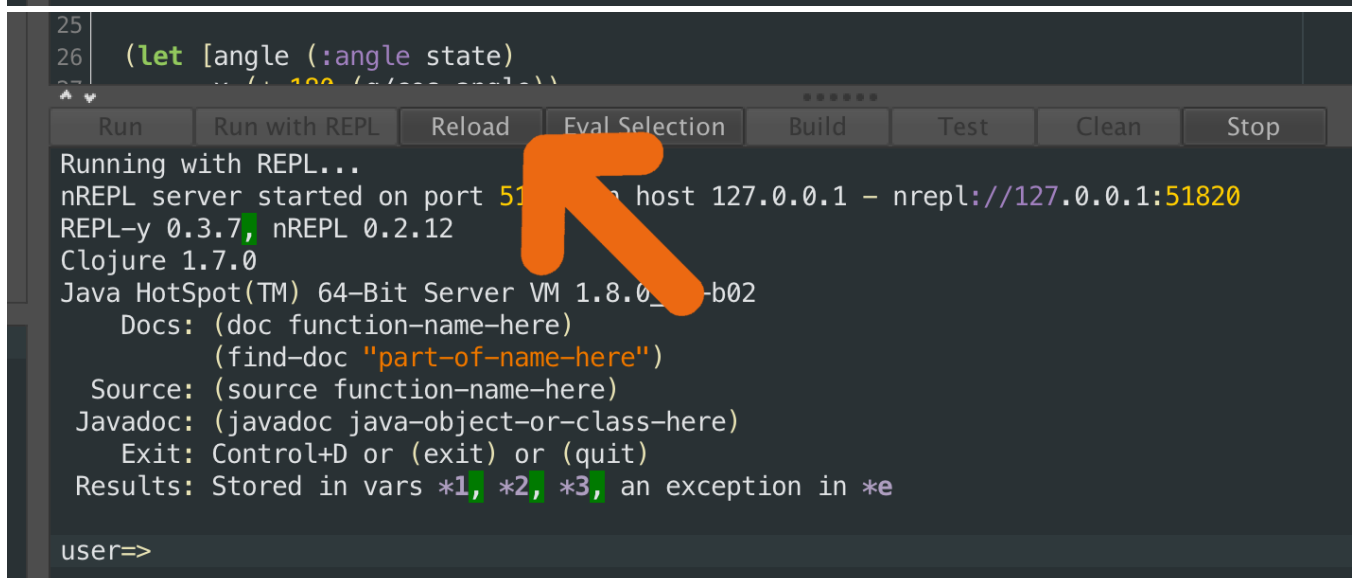
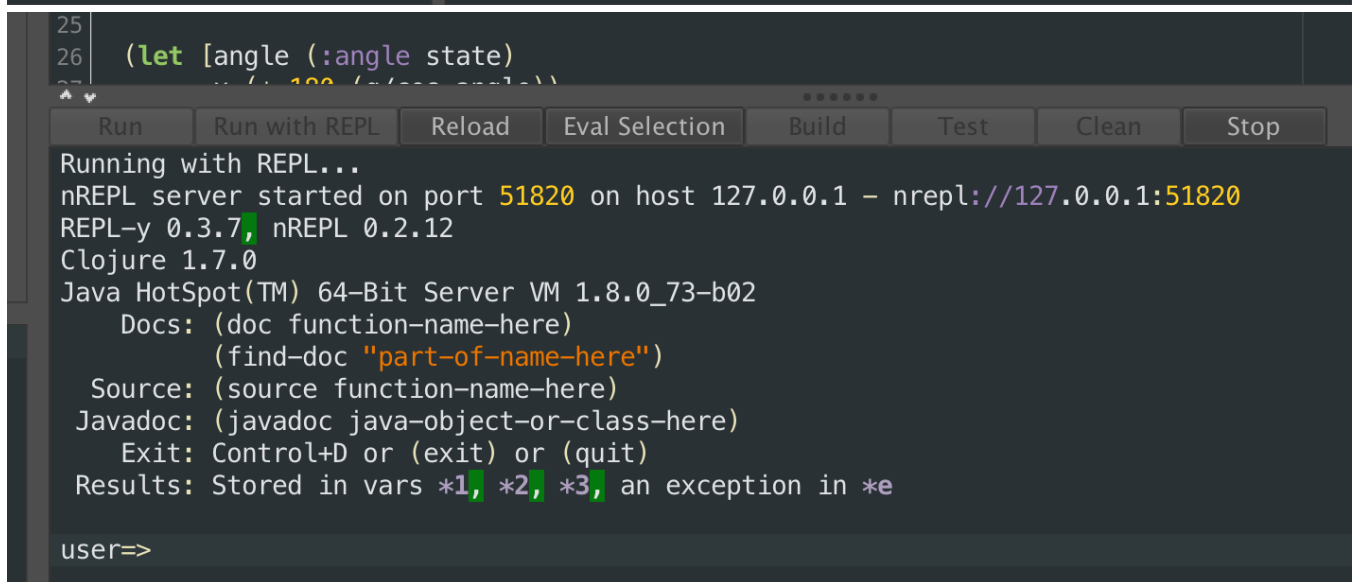
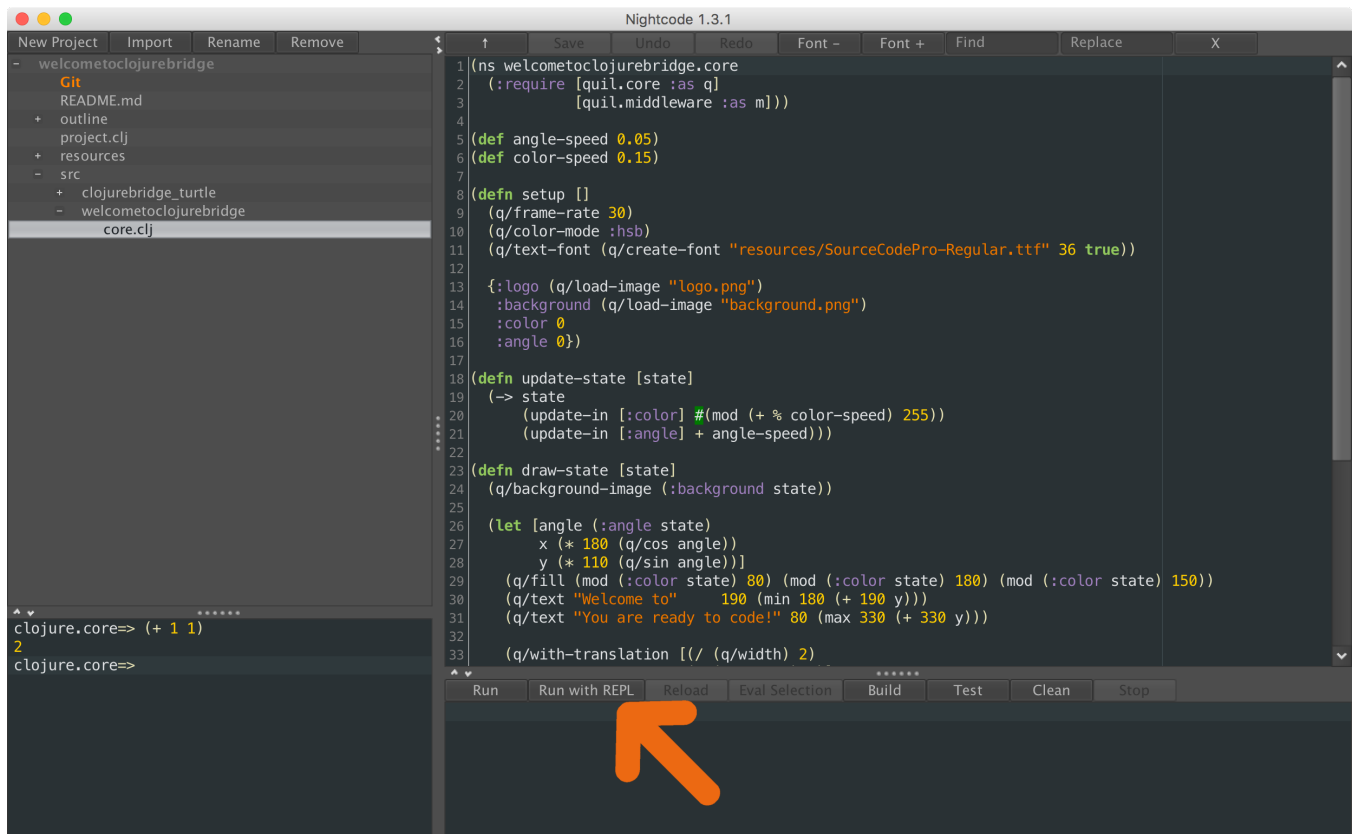
Now we will open and run the sample Clojure apps in Nightcode. On the top left corner, click "Import" then find the directory, `welcometoclojurebridge`, which was created when you ran `git clone` command. Click "Open." In the project directory tree on the left, click on `src` - `welcometoclojurebridge` - `core.clj`. The `core.clj` file will be opened on the right side. This is a Clojure program.



![Testing apps - core.clj]

(img/nightcode-welcometoclojurebridge-core.png)

The next step is to run the code shown in the window. Click "Run with REPL" on the bottom of the right side. It may take a while. Eventually, repl will start and show a prompt on the bottom of the window. Once, you see the prompt, click "Reload" button.



You should see a fun welcome message.



Let's try one more sample. In the directory tree on the left, click on `welcometoclojurebridge` - `src` - `clojurebridge-turtle` - `walk.clj`. The `walk.clj` file will open on the right side. Like we did before, click "Reload" button.

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
```

Run Run with REPL Reload Eval Selection Build Test Clean Stop

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
```

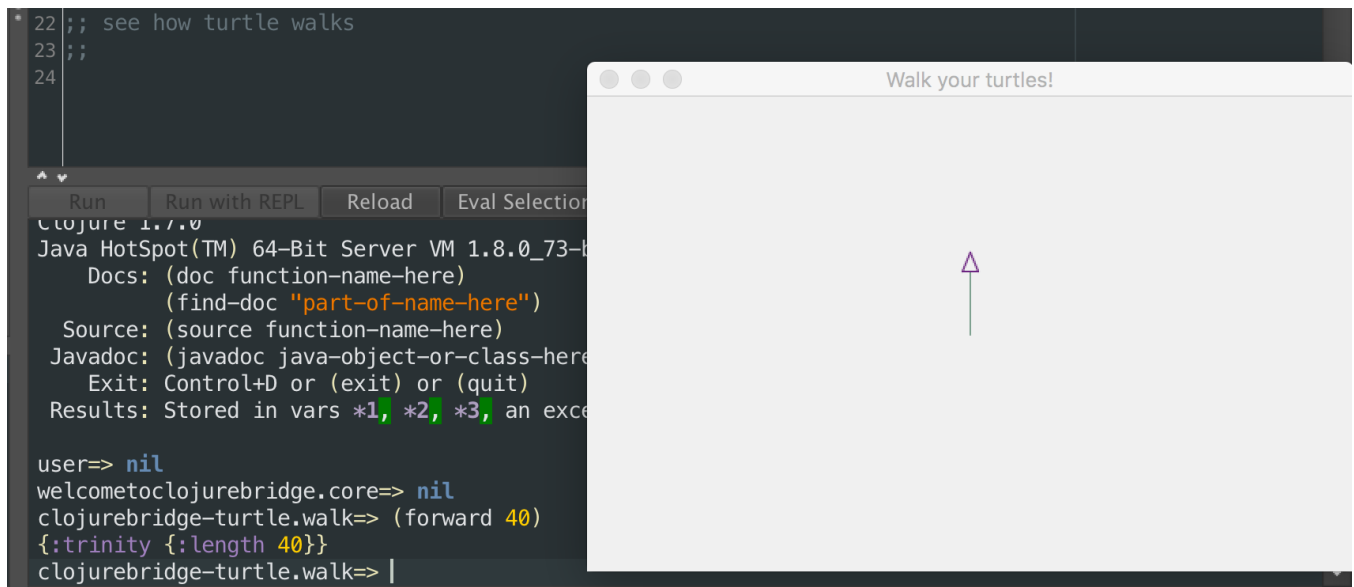
Run Run with REPL Reload Eval Selection Build Test Clean Stop

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

An initial image of the turtles app should pop up. A small triangle on the center is the *turtle*.

Type `(forward 40)` on the repl at the bottom of the window. You should see the turtle moved upward:



Success!

Congratulations! You have opened and run your first Clojure apps, and your install and setup are all completed!

If you want to know what the turtle (*a small triangle*) can do, see [Turtle App API](#) and [How to Walk Turtles](#) for more information.

Troubleshooting

- Leiningen Windows Installer has an issue that it doesn't install lein.bat correctly. This causes curl.exe to fail downloading files with the error below. Skip the Leiningen Windows Installer. Download lein.bat from leiningen.org and run self-installer.

| error:0307A071:bignum routines:BN_rand_range:too many iterations.

Windows Setup

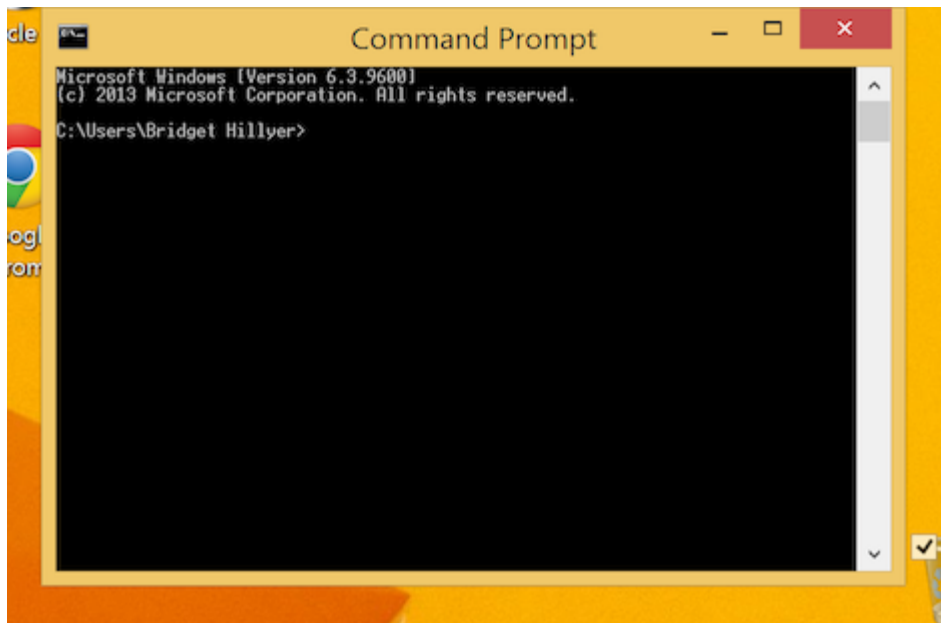
- Start a command prompt
- Install Git
- Configure Git
- Install Java
- Install Leiningen
- Install Nightcode
- Test your setup
- Troubleshooting

Starting a command prompt

For these instructions, and for much of the class, you will need to have a command prompt open. This is a text-based interface to talk to your computer. Go to the "Windows" screen (the "Start Screen") and type "command". Choose the "Command Prompt" program, like in this screenshot:



When you choose "Command Prompt," your screen should look similar to this:



If you have never used the command prompt before, you may want to spend some time [reading up on command prompt basics](#). For the rest of this setup, I will tell you to run commands in your command prompt. When I say that, I mean "type the command into the command prompt and press the Return key."

On other operating systems, the command prompt is called the terminal. We will use the terms terminal, command prompt, and command line interchangeably.

Installing Git

See if you already have Git installed at the command prompt with the command `git --version`. If not, download it from the [git-scm.com Windows download page](#) and run the executable to install.

After installation, try the `git --version` command in a new command prompt window. If you see a version number, git was installed correctly.

If you see a message that says, `'git' is not recognized as an internal or external command`, try these steps to update your PATH variable properly:

- Right-click "My Computer" and select "Properties".
- Click the "Advanced Tab" and then the "Environment Variables" button.
- Highlight the PATH entry and click "Edit".
- Scroll to the end of this value and check for a file path at the end that includes "...\\Git...".
- If that path existed:
 - Click "Okay" until the "My Computer" dialog box is closed.
 - Open a new command prompt window and try `git --version` again. If that does not succeed, restart your computer and try again.
- If that path did not exist:
 - If you did not change the install location of git during installation, add ";C:\\Program Files (x86)\\Git\\cmd" to the end of the line. Make sure you add the semi-colon between file paths and the line includes no spaces between paths.
 - Click "Okay" until the "My Computer" dialog box is closed.
 - Open a new command prompt window and try `git --version` again. If that does not succeed, restart your computer and try again.

If you've used Git before then you should already have user.name and user.email configured. Otherwise, type this in the command prompt:

Configure Git

```
git config --global user.name "Your Actual Name"
git config --global user.email "Your Actual Email"
```

TIP: Use the same email address for git, github, and ssh.

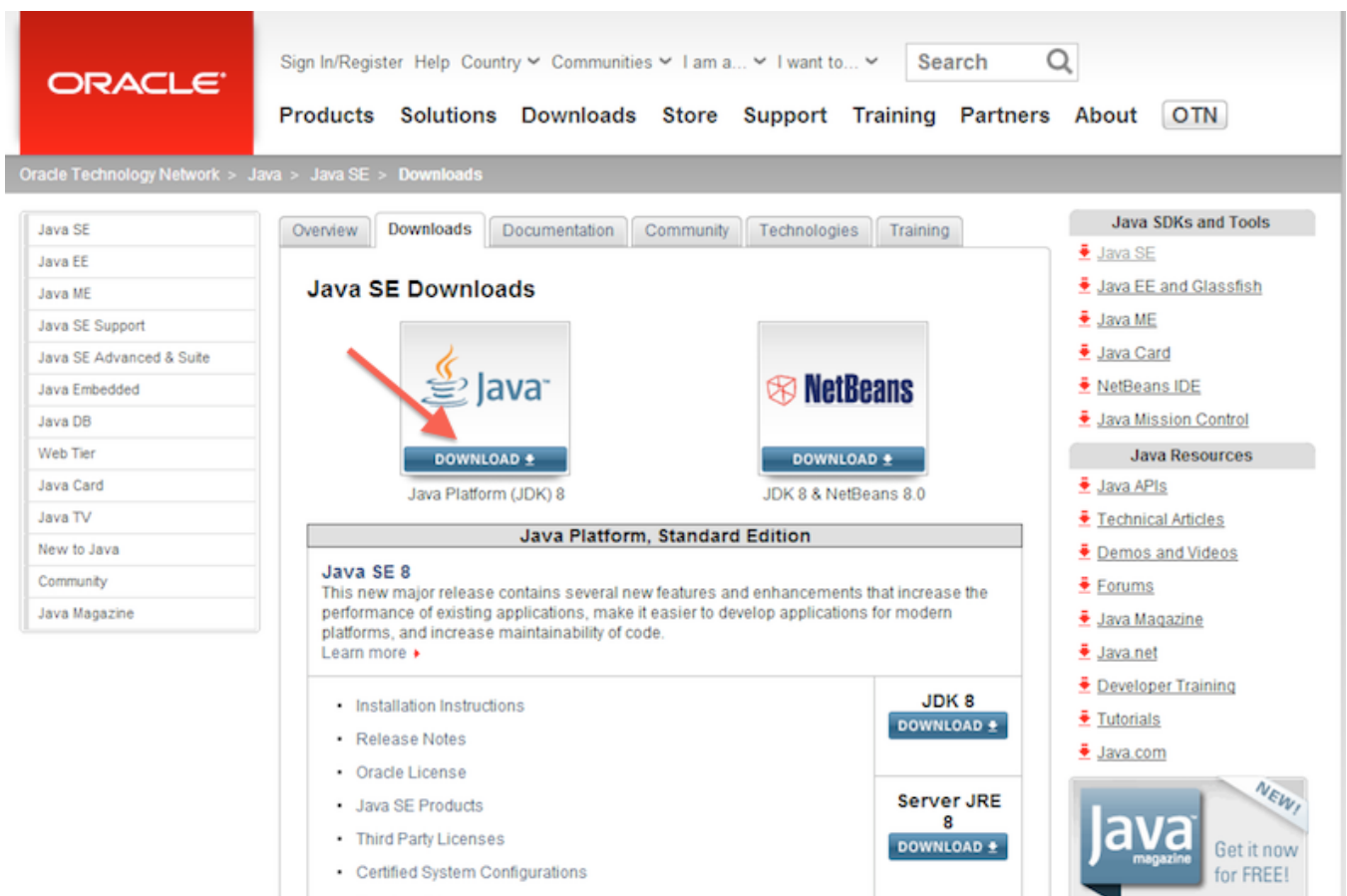
Verify by typing this in the command prompt:

```
git config --get user.name Expected result: your name
```

```
git config --get user.email Expected result: your email address
```

Install Java

Go to [the Leiningen Windows installer site](#). You should see two links, one for installing Java and another for "leiningen-win-installer." Click the Java link. Then, you should see a screen like the following:
















Click the button above "Java Platform (JDK)," as you can see in the above picture. Then you will come to a page that will have the following table on it:

Java SE Development Kit 8

You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

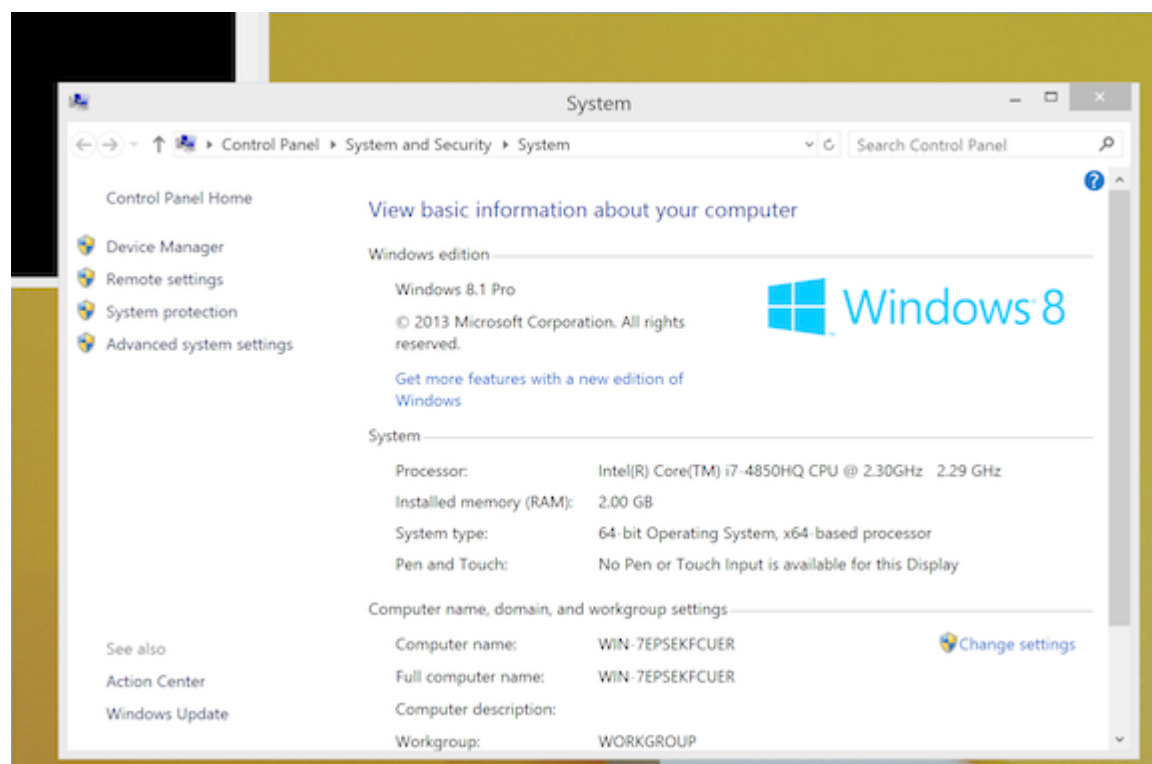
☐ Accept License Agreement
 ☒ Decline License Agreement

 **Click to Accept**

Product / File Description	File Size	Download
Linux ARM v6/v7 Hard Float ABI	83.51 MB	 jdk-8-linux-arm-vfp-hflt.tar.gz
Linux x86	133.57 MB	 jdk-8-linux-i586.rpm
Linux x86	152.47 MB	 jdk-8-linux-i586.tar.gz
Linux x64	133.85 MB	 jdk-8-linux-x64.rpm
Linux x64	151.61 MB	 jdk-8-linux-x64.tar.gz
Mac OS X x64	207.72 MB	 jdk-8-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	135.5 MB	 jdk-8-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	95.53 MB	 jdk-8-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	135.78 MB	 jdk-8-solaris-x64.tar.Z
Solaris x64	93.15 MB	 jdk-8-solaris-x64.tar.gz
Windows x86	151.68 MB	 jdk-8-windows-i586.exe
Windows x64	155.14 MB	 jdk-8-windows-x64.exe

Click the radio button to accept the license agreement, and then download one of the two Windows choices. If you are running 32-bit Windows, choose "Windows x86." If you are running 64-bit Windows, choose "Windows x64."

If you do not know if you are running 32-bit or 64-bit Windows, go to the "Windows" screen (the "Start Screen") and type "system." Choose "System." (If that does not work, type "Control Panel" and choose "System" from the Control Panel screen.) You should see a window like the following:



You should see if you are running 32- or 64-bit Windows beside "System Type."

Once you have downloaded the right Java version, run the executable you downloaded to install Java. Follow the installation wizard.

Install Leiningen

Leiningen is a tool used on the command line to manage Clojure projects.

| see troubleshooting for leiningen installation

Next, go back to [the Leiningen Windows installer site](#) and download the file linked as "leiningen-win-installer." Run this executable and follow the "Detailed installation" section at the Leiningen Windows Installer site. At the end of the installation, leave "Run a Clojure REPL" checked before you click "Finish." If a terminal window opens that looks like the one on the Leiningen Windows installer site, then you are good to go.

Install Nightcode

Go to the [Nightcode release site](#).

Note: Do NOT go to the main Nightcode page and download the most recent version. The most recent version is Nightcode 2.0.1 (released July 2016), but we will be using a slightly older one (Nightcode 1.3.2) because the new one just came out and might be a little unstable. Plus the user interfaces of the two versions are a bit different, and the current ClojureBridge instructional materials are written for the older one.

On the [Nightcode release site](#), you should see version numbers and links to download specific version of Nightcode.

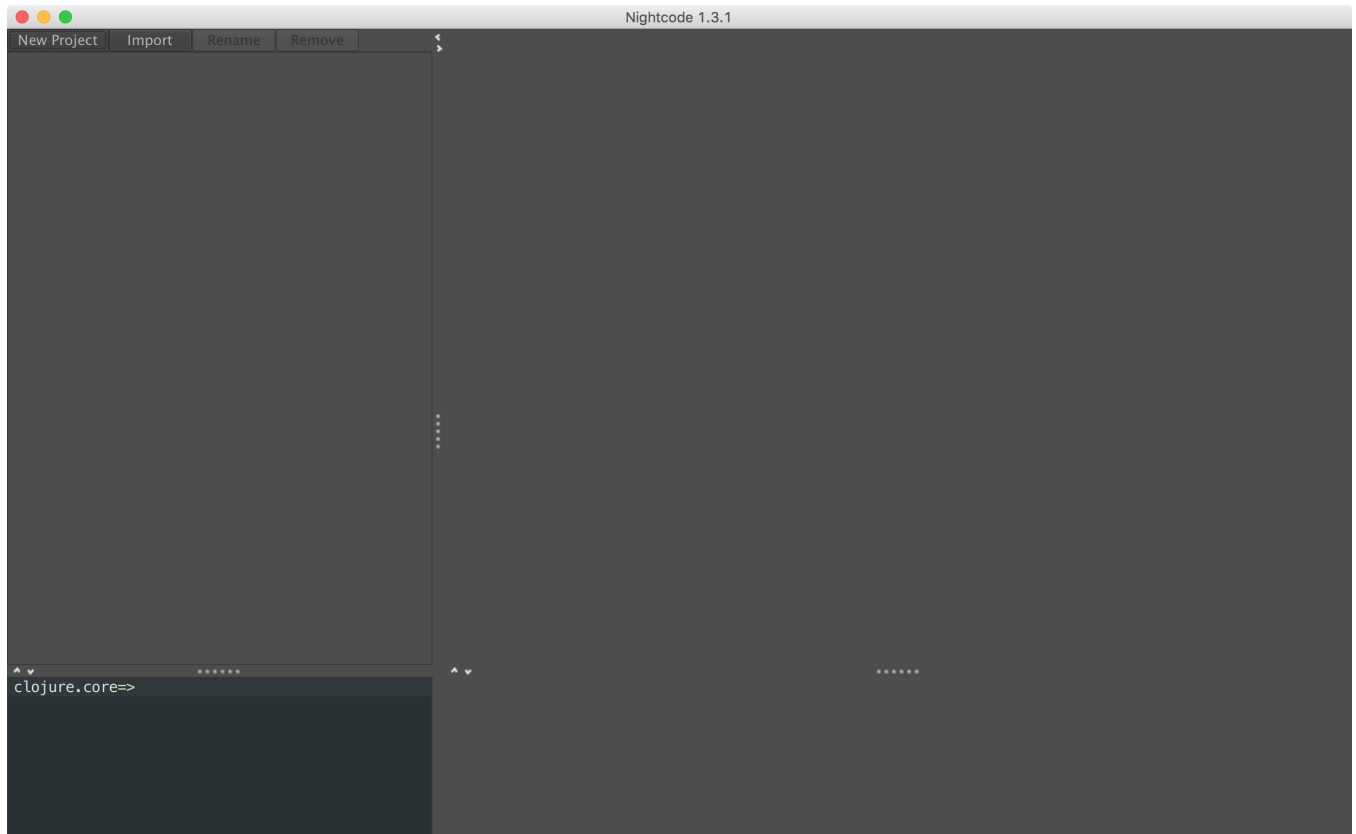
Scroll down to the heading "1.3.2", and click the download link labeled `nightcode-1.3.2-standalone.jar`.

Once the download is finished, we want to start the editor.

To start it up, go into your Downloads folder (or wherever you save files from your browser) and run the `nightcode-1.3.2-standalone.jar` file using the `java` command.

Open a command prompt and run the following commands:

```
cd ~/Downloads/  
java -jar nightcode-1.3.2-standalone.jar
```



Test your setup

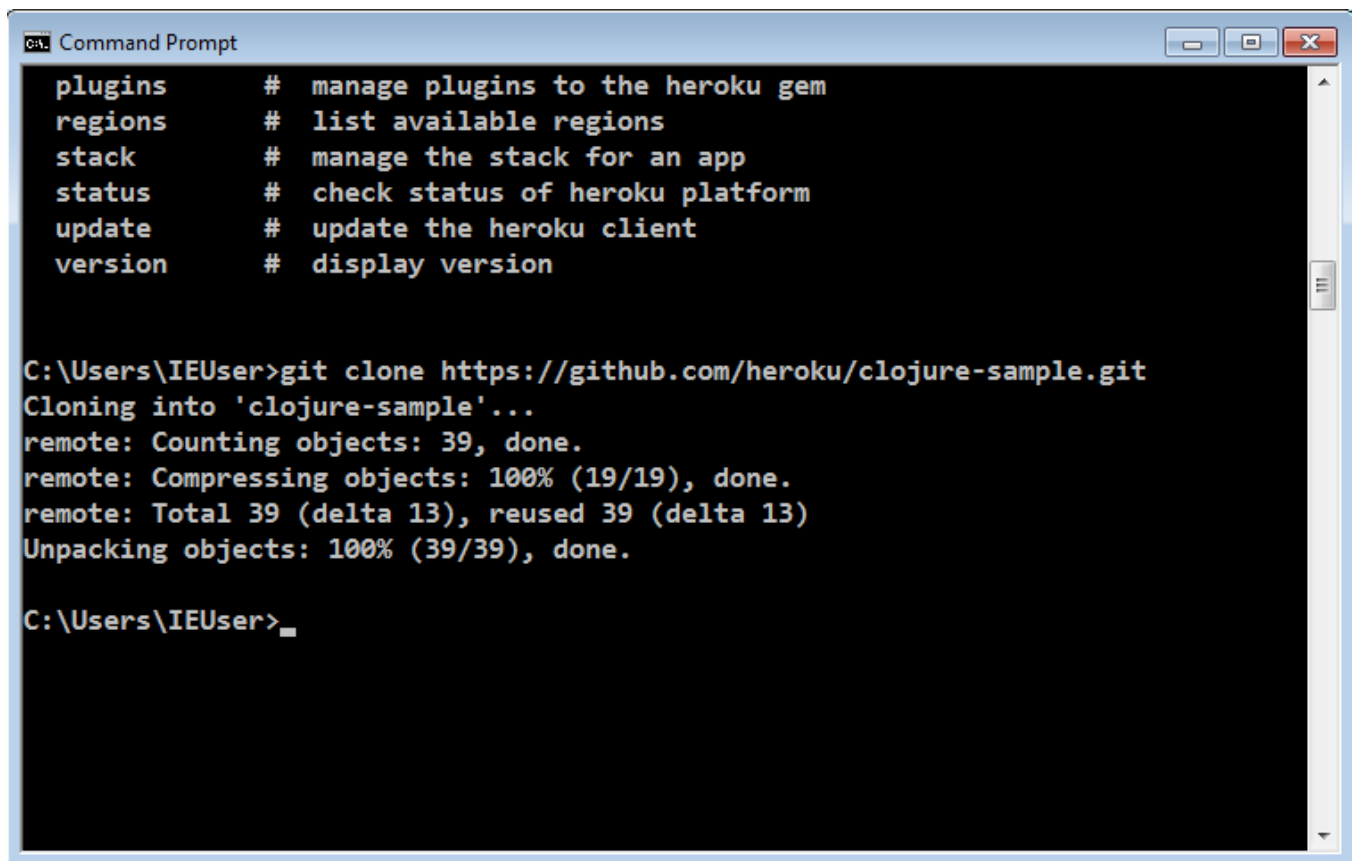
You have set up Java, Leiningen, Nightcode, and Git on your computer--all the tools you will need for this workshop. Before starting, we need to test them out.

Cloning out github repository

Go to your command prompt and run the following command:

```
git clone https://github.com/ClojureBridge/welcometoclojurebridge
```

This will clone `welcometoclojurebridge` repository which includes sample Clojure apps. Your command prompt should look similar to this picture:



```
CA. Command Prompt

plugins      # manage plugins to the heroku gem
regions      # list available regions
stack        # manage the stack for an app
status       # check status of heroku platform
update       # update the heroku client
version      # display version

C:\Users\IEUser>git clone https://github.com/heroku/clojure-sample.git
Cloning into 'clojure-sample'...
remote: Counting objects: 39, done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 39 (delta 13), reused 39 (delta 13)
Unpacking objects: 100% (39/39), done.

C:\Users\IEUser>
```

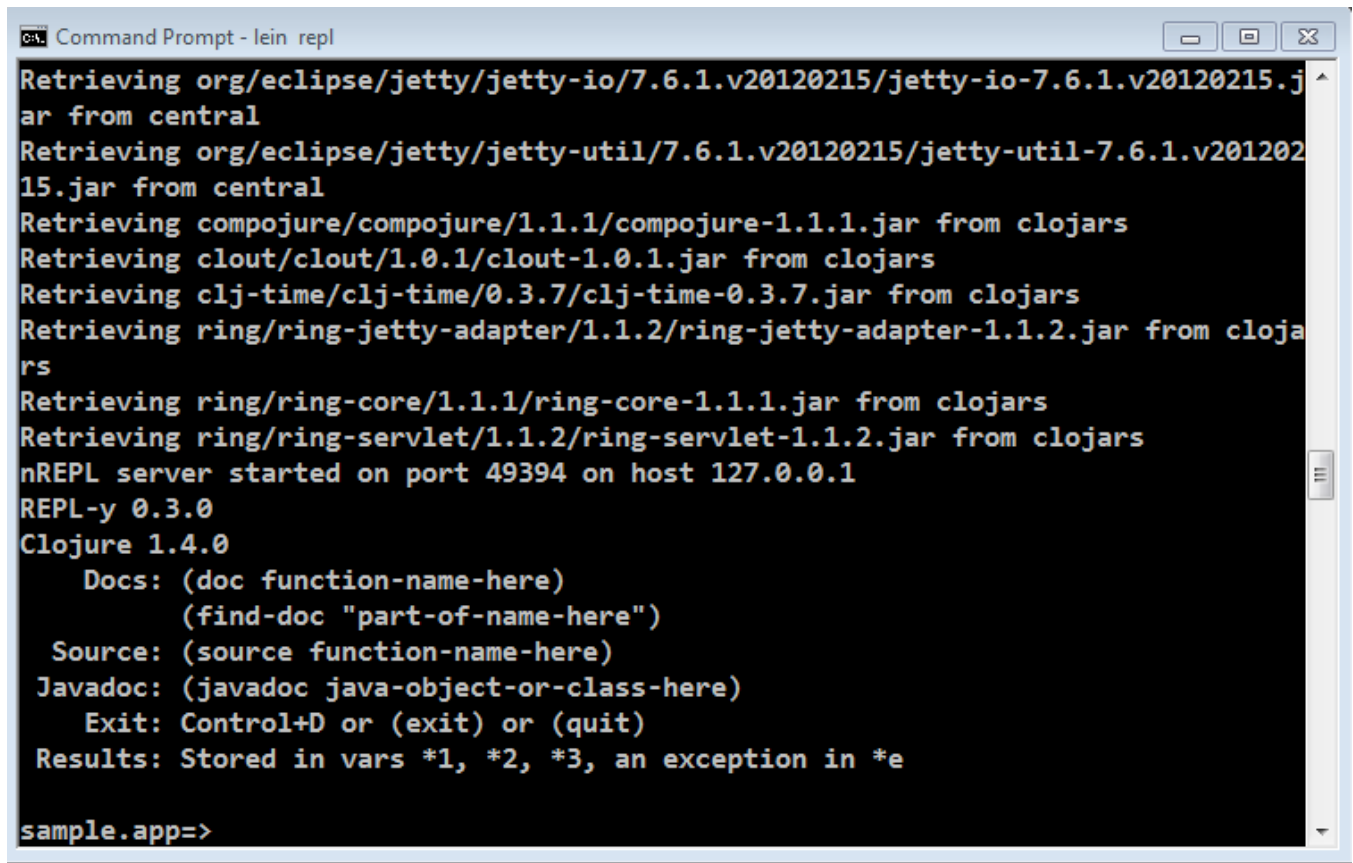
Then run the command:

```
cd welcometoclojurebridge
```

This will take you to the folder with the source code. After that completes, run:

```
lein repl
```

This could take a long time, and will download many other pieces of code it relies on. You should see lines that start with `Retrieving ...` on your screen. When it finishes, your command prompt should look like the following:



```
Command Prompt - lein repl
Retrieving org/eclipse/jetty/jetty-io/7.6.1.v20120215/jetty-io-7.6.1.v20120215.jar from central
Retrieving org/eclipse/jetty/jetty-util/7.6.1.v20120215/jetty-util-7.6.1.v20120215.jar from central
Retrieving compojure/compojure/1.1.1/compojure-1.1.1.jar from clojars
Retrieving clout/clout/1.0.1/clout-1.0.1.jar from clojars
Retrieving clj-time/clj-time/0.3.7/clj-time-0.3.7.jar from clojars
Retrieving ring/ring-jetty-adapter/1.1.2/ring-jetty-adapter-1.1.2.jar from clojars
Retrieving ring/ring-core/1.1.1/ring-core-1.1.1.jar from clojars
Retrieving ring/ring-servlet/1.1.2/ring-servlet-1.1.2.jar from clojars
nREPL server started on port 49394 on host 127.0.0.1
REPL-y 0.3.0
Clojure 1.4.0
  Docs: (doc function-name-here)
        (find-doc "part-of-name-here")
  Source: (source function-name-here)
  Javadoc: (javadoc java-object-or-class-here)
  Exit: Control+D or (exit) or (quit)
  Results: Stored in vars *1, *2, *3, an exception in *e
sample.app=>
```

This is starting a REPL, which we will learn about soon. It's a special command prompt for Clojure. At the REPL prompt, type `(+ 1 1)` and hit enter. Did you get the answer `2` back? You will learn more about that in the course. For now, press the Control button and D button on your keyboard together (abbreviated as Ctrl+D). This should take you out of the Clojure REPL and back to your normal command prompt. Then, the command prompt will show you the following message: `user=> Bye` for now!

Testing Nightcode

If Nightcode isn't started yet or closed, open it by typing the command on the command prompt:

```
java -jar nightcode-1.3.2-standalone.jar
```

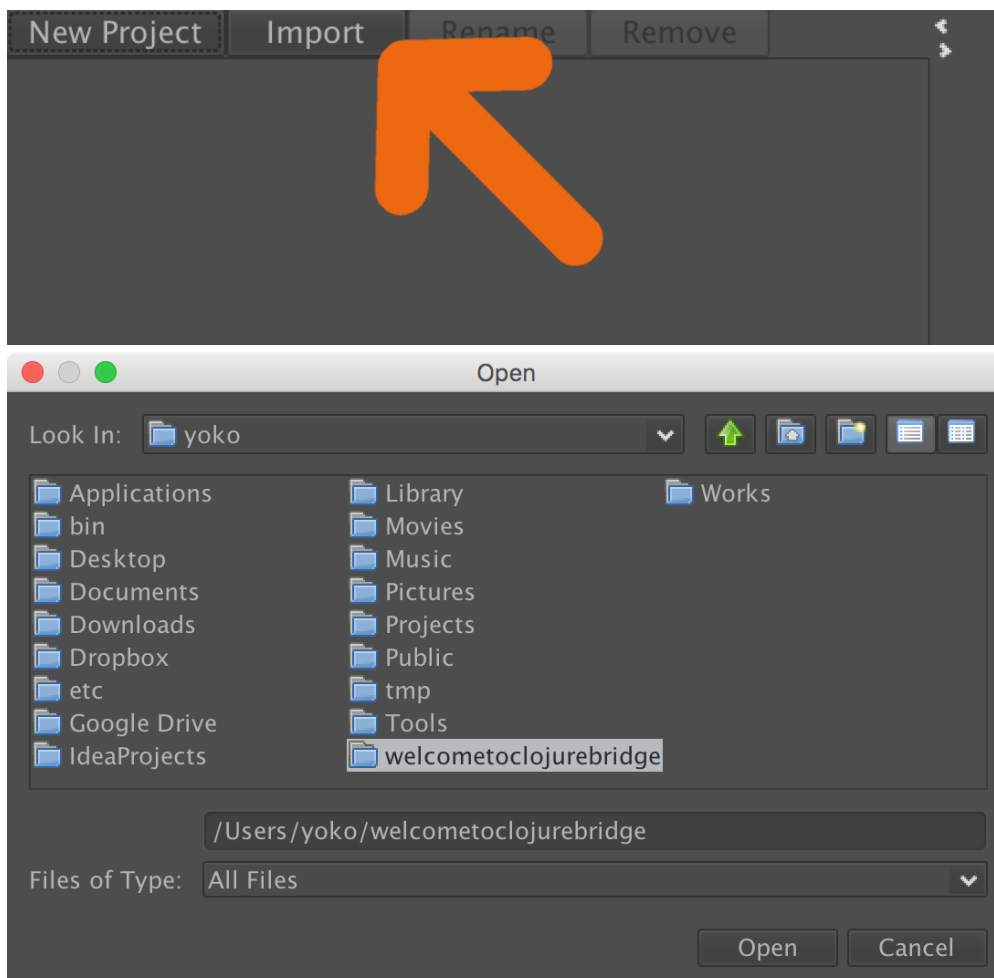
At the bottom left of the screen, type `(+ 1 1)` into the window. It should look like the following image:

```
clojure.core=> (+ 1 1)
2
clojure.core=>
```

If you see the result, 2, that worked, great!

Testing apps

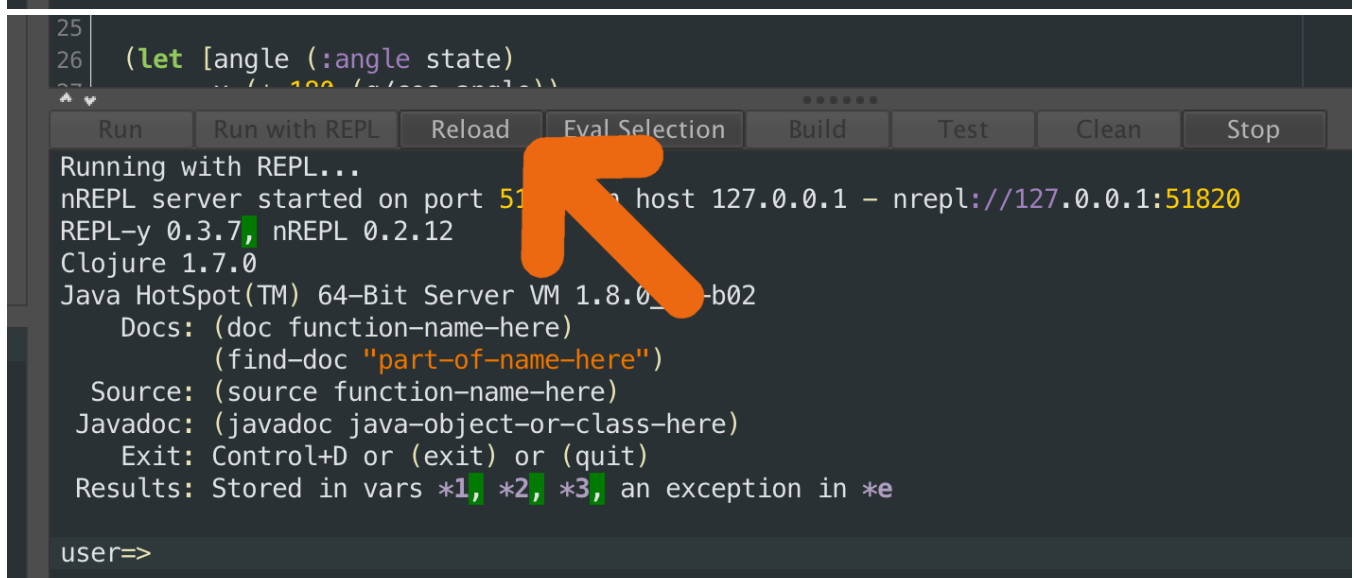
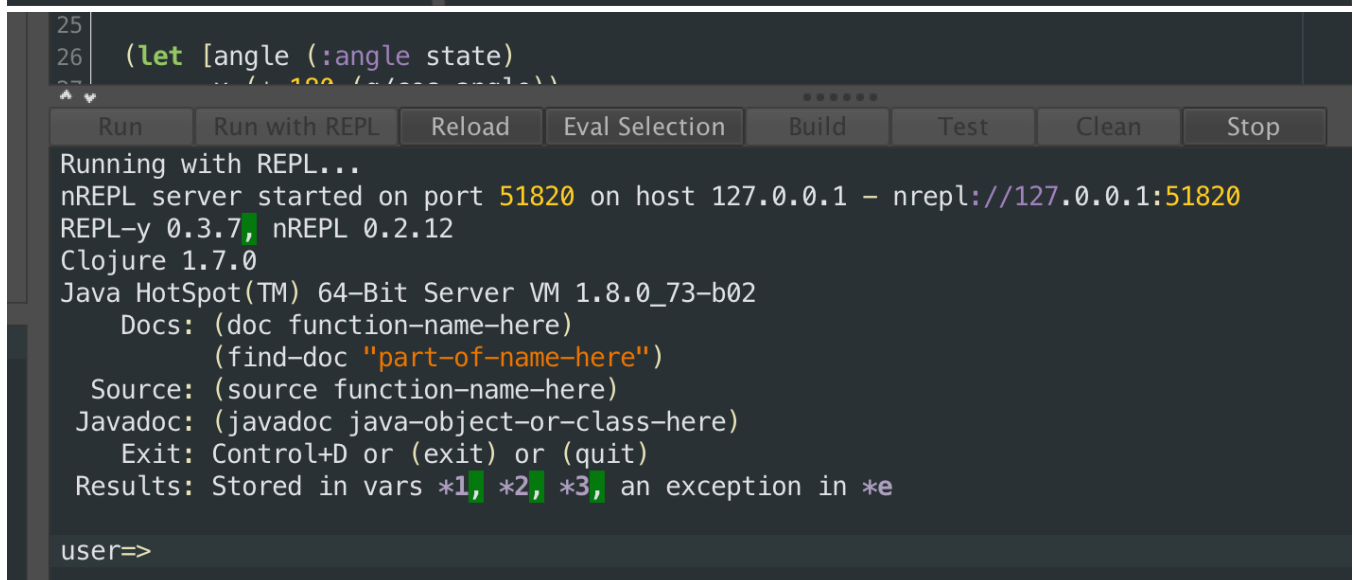
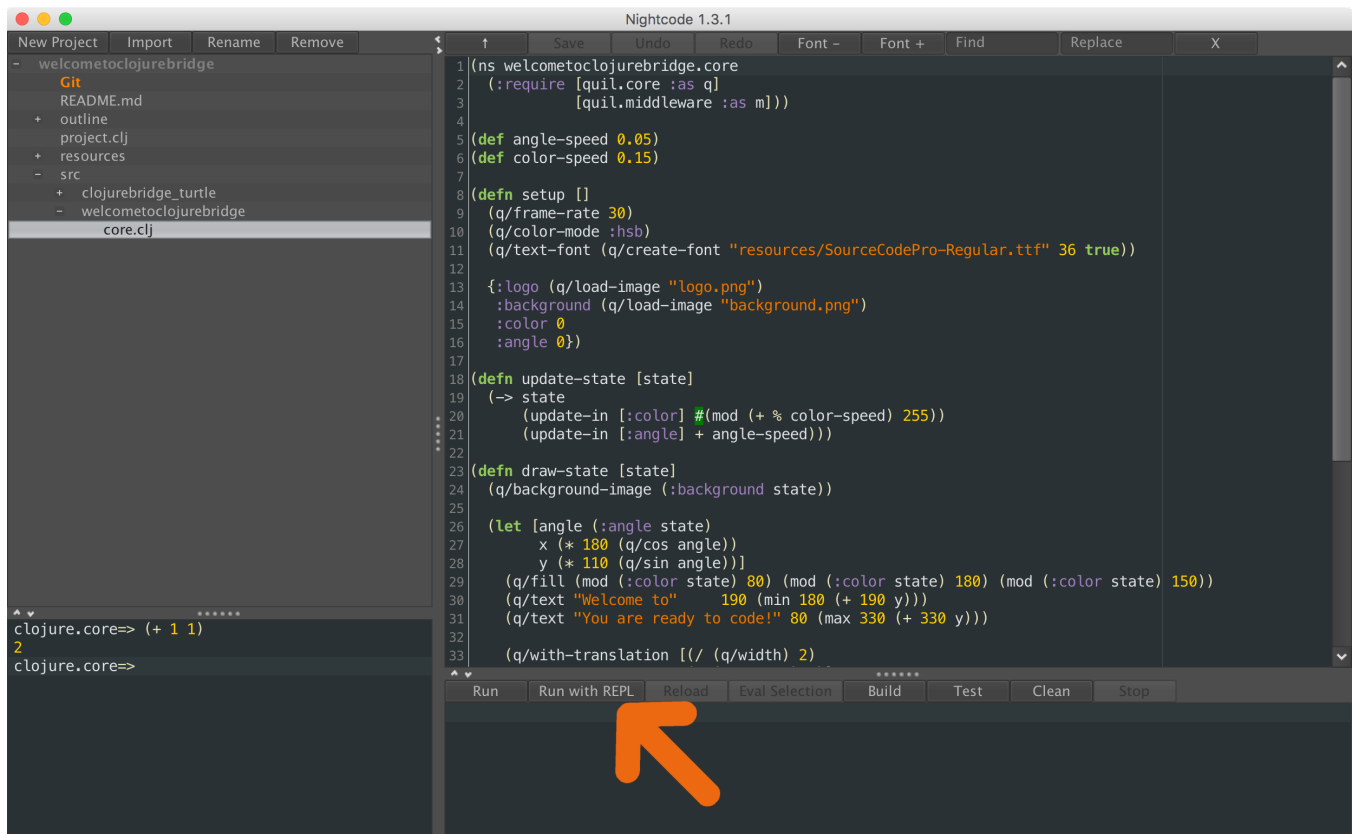
Now we will open and run the sample Clojure apps in Nightcode. On the top left corner, click "Import" then find the directory, `welcometoclojurebridge`, which was created when you ran `git clone` command. Click "Open." In the project directory tree on the left, click on `src` - `welcometoclojurebridge` - `core.clj`. The `core.clj` file will be opened on the right side. This is a Clojure program.



! [Testing apps - core.clj]

(img/nightcode-welcometoclojurebridge-core.png)

The next step is to run the code shown in the window. Click "Run with REPL" on the bottom of the right side. It may take a while. Eventually, repl will start and show a prompt on the bottom of the window. Once, you see the prompt, click "Reload" button.



You should see a fun welcome message.



Let's try one more sample. In the directory tree on the left, click on `welcometoclojurebridge` - `src` - `clojurebridge-turtle` - `walk.clj`. The `walk.clj` file will open on the right side. Like we did before, click "Reload" button.

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
```

Run Run with REPL Reload Eval Selection Build Test Clean Stop

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

```
(ns clojurebridge-turtle.walk
  (:use clojure.repl)
  (:use clojurebridge-turtle.core))
(init)

;; =====
;; evaluate whole file by hitting
;; ctrl + shift + enter, or
;; cmd + shift + enter
;;
;; add some functions under these comment lines
;; evaluate each form (line or function) by hitting
;; ctrl + enter, or
;; cmd + enter
;;
;; for example
;; (forward 30)
;; (right 90)
;; (forward 30)
;; (right 90)
;;
;; see how turtle walks
;;
```

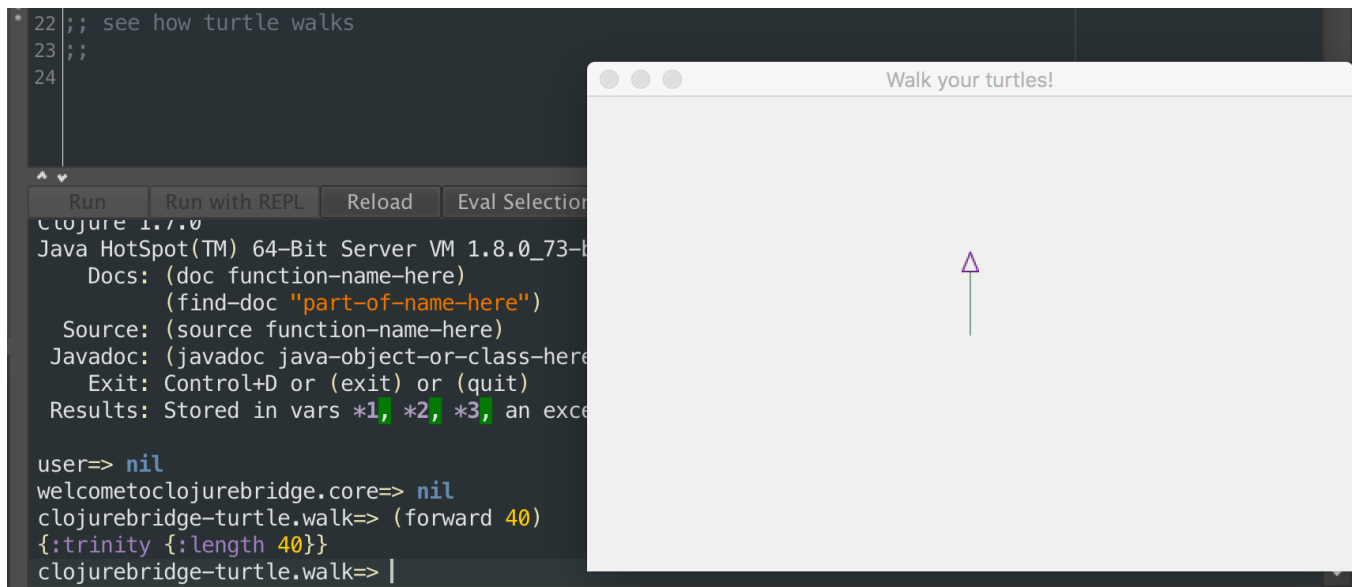
Run Run with REPL Reload Eval Selection Build Test Clean Stop

Running with REPL...
nREPL server started on port 51883 on host 127.0.0.1 - nrepl://127.0.0.1:51883
REPL-y 0.3.7, nREPL 0.2.12
Clojure 1.7.0
Java HotSpot(TM) 64-Bit Server VM 1.8.0_73-b02
Docs: (doc function-name-here)
(find-doc "part-of-name-here")
Source: (source function-name-here)
Javadoc: (javadoc java-object-or-class-here)
Exit: Control+D or (exit) or (quit)
Results: Stored in vars *1, *2, *3, an exception in *e

user=> nil
welcometoclojurebridge.core=>

An initial image of the turtles app should pop up. A small triangle on the center is the *turtle*.

Type `(forward 40)` on the repl at the bottom of the window. You should see the turtle moved upward:



Success!

Congratulations! You have opened and run your first Clojure apps, and your install and setup are all completed!

If you want to know what the turtle (*a small triangle*) can do, see [Turtle App API](#) and [How to Walk Turtles](#) for more information.

Troubleshooting

- Leiningen Windows Installer has an issue that it doesn't install lein.bat correctly. This causes curl.exe to fail downloading files with the error below. Skip the Leiningen Windows Installer. Download lein.bat from leiningen.org and run self-installer.

| error:0307A071:bignum routines:BN_rand_range:too many iterations.