## Introduction

Python is a computer language.

it is is human-friendly, readable, well-designed, and elegant. It is “pythonic”

Most every computer understands python, but web browsers do not.

Browsers understand javascript. Javascript is *not* pythonic.

There are technologies to translate python into javascript, but the tools for using them are far from pythonic.[[1]](#footnote-0)

For example, this is a complete python program.

* print(‘hello world!’)  
  When you run the program it prints

“hello world!”

There are similar commands in javascript, but it ain't pythonic

* console.log("hello world")   
  prints to "the console" (if you know where to find the console)
* alert("hello world")   
  triggers a popup window in the browser   
  (but it takes a few more lines of HTML code to make this work)
* document.body.innerHTML += "hello world"   
  makes it visible in the browser page  
  (but even more HTML is required).

**Lyte is for beginners, and for web programmers who seek pythonic simplicity.** With a few simple and consistent commands, you can write programs that will run on any computer *and* in any modern browser, even without a server.

With Lyte you can use the command

say('Hello World')

and easily have it show up in an a programming environment, in a command line program, and in the browser.

The point of Lyte is to make these scenarios as similar and pythonic as possible so you can focus on what what you want to do, and not have to learn all of the arcane details that make this possible.

## A programming environment.

A programming environment combines an *editor* for writing code with an engine for executing and debugging the code. The simplest powerful python programming environment I’ve found is Thonny.

### Meet Thonny

* Download it.
* Install It.
* Run it

### The Shell

Thonny gives you direct access to the Python engine via the Shell, the bottom window.

Here are some things you should try.

>>> 2+2

4

>>> 2\*2\*2\*2

16

>>> print('hello world')

hello world

>>> for name in "Tom Dick Harry".split():

print('hi', name)

hi Tom

hi Dick

hi Harry

Get it? You type commands, python executes them.

Hint: punctuation matters.

an open parenthesis demands a closed parenthesis

an opening single quote demands a closing single quote

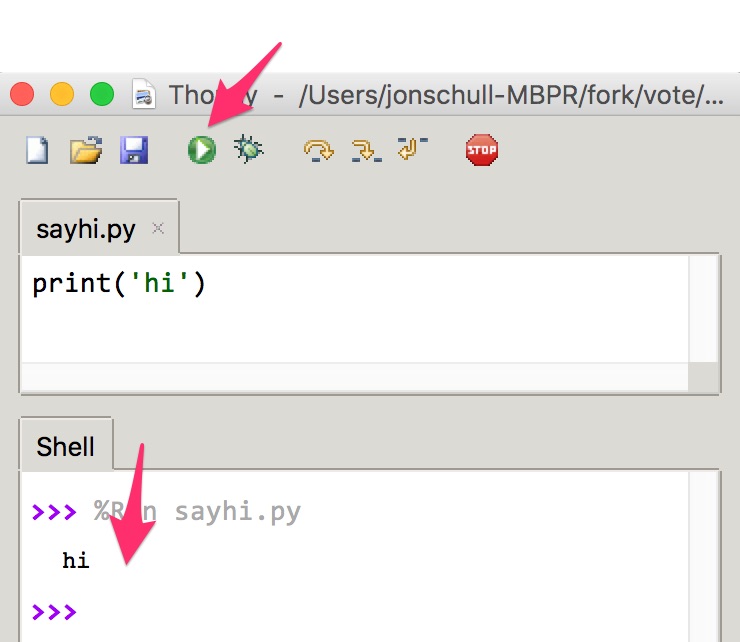
Thonny uses grey and green indicators to warn you if you forget.

[Here's](https://d2ffutrenqvap3.cloudfront.net/items/1v171R253O0G243p3L14/Screen%20Recording%202018-03-11%20at%2002.13%20PM.gif) an illustration and a recommendation:

type parens and quotes in pairs and *then* insert content.

### Write a program in the editor.

* Create a new file called sayhi.py. File:New, File:Save, "sayhi.py"
* Write your first program in the blank editor space  
    
  print( 'hi' )
  + Run it. (Click  in the menu bar.)
  + Look down at the Shell.

see?   


Now you can run it as many times as you want without retyping.

You can also save saveHi.py, turn off your computer, restart Thonny and run it again.

### 

### Debug our first program.

Newly-written code rarely works the first time. Especially in a new context.

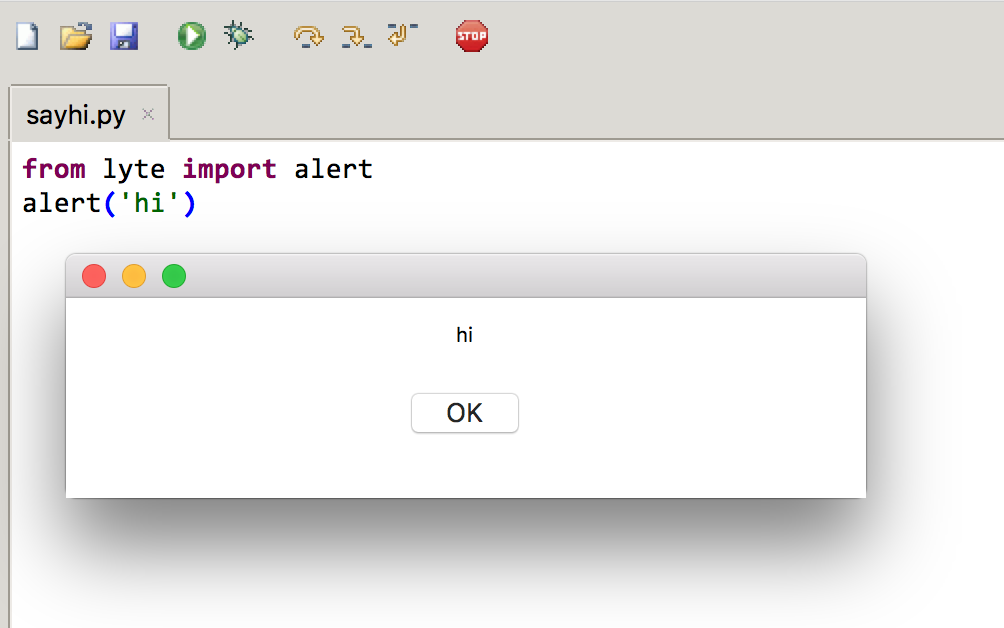
* Change **print** to **alert**, and Run the program again.  
    
  alert('hi')
* Thanks to Thonny, we get a helpful error message  
  We understand the problem: as mentioned previously, **alert** is not in python's vocabulary. (**alert** is a commonly used javascript command).

### 

### Expand our vocabulary

You can add commands by writing them from scratch, or by importing commands written by others.

* Add a line to your program so it reads like this:  
    
  from lyte import alert  
  alert('hi')
* Run the program (  or use the Thonny menu: Run/ Run Current Script) )
* Yay! You should see something like this



### 

### Under the hood

Let's peak under the hood to understand what's going on here.

There's a little program called lyte.py on your computer that makes **alert** work in python.

You can look at it, but you don't have to understand it. (That's why it's called Lyte (not Heavy or Dark).

The notes in red point out a few things that *are* worth noting: Lyte worries about language, and context so you don't have to.



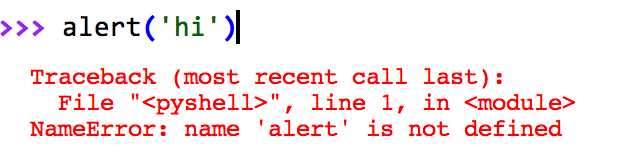
### 

### But you *should* understand context.

Let's review what we used, while learning about contexts.

* press Stop  to reset the python engine

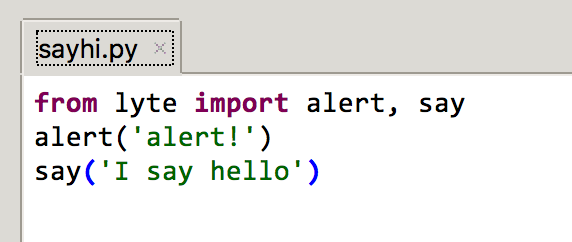
#### The Shell context:

* In the shell, type   
    
  alert('hi')   
    
  expect it to fail: 
* Make it succeed by importing the alert command and then try again. Type  
    
  from lyte import alert  
  alert('hi')
* let's import the say command too  
    
  from lyte import say  
  say('I say hello')

#### 

#### The Thonny Run Context

* let's revise **sayhi.py** to import the **say** command as well as the **alert** command and to test both. Make sayhi.py look like this:



* Run it. The alert should popup. Then, after you click 'OK' the shell should say hello.

#### The System Shell Context

* Save sayhi.py
* Let's get out to the system shell. Here's one way:
  + Thonny menu: Tools / Open System Shell...

You are now "under the hood" this is the engine of your operating system (windows, mac linux, etc.). This is a world of typed commands.

1. The problem may be documentation. It’s written by javascript jockeys for javascript jockeys. Here are some things that confused me.

   * The documentation assumes you know how to integrate put javascript into html(see simplestJS.html)
   * **There is no helloworld that runs in all environments**
   * The command line switches for the rapydscript compiler are hard to find. I found them here <https://www.npmjs.com/package/rapydscript#available-libraries>
   * The error messages are hard to read. It is sometimes easier to solve a problem using the python compiler.
   * The need for <meta charset=“UTF-8”>
   * Glowscript runs on Rapydscript but does not import rapydscript modules (unless they are converted into javascript and served from a server).

   [↑](#footnote-ref-0)