

Introduction to Python

DMS 102: Programming Digital Media

Lecture 4

JES: Jython Environment for Students

- IDE (Integrated Development Environment)
- No byte code; no executables
- Learn: how to program
- Learn: the *true* nature of digital media
- Install JES
 - On your own laptop
 - Installation files:** <https://github.com/gatech-csl/jes/releases>
 - Or use the lab computers
- JES Help
 - <https://b.gatech.edu/2zVME3h>
 - Also available in JES | HELP menu

Python syntax

(compared to JavaScript)

- White space sensitive!
 - statements: instead of semi-colons, EOL
 - blocks: instead of curly braces, indentations
- Comments:
`# ...single line comments`
- Variables
 - no `var` keyword, just create a *name*
- Operators - all same (including the assignment operator) except...
 - `and` (instead of `&&`)
 - `or` (instead of `||`)

Python syntax, continued

(compared to JavaScript)


- Functions

- Use `def` keyword
- Use colon at the end
- whitespace sensitive!
EOL and indentations

```
def helloWorld():  
    → # Python statements go here...  
    → print "Hello World!"  
  
# "call" the function...  
helloWorld()
```

- Conditionals

- If, Else, Elif
- ~~Switch~~ (none)



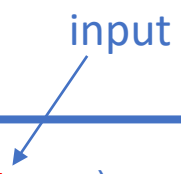
```
if temperature < 55:  
    print "It's cold outside today!"  
elif (temperature >= 55) and (temperature <= 75):  
    print "The weather is fine today."  
else:  
    print "It's a scorcher!"
```

JES Built-in Functions

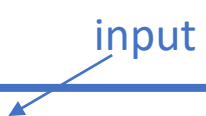
- A bunch of functions are pre-defined in JES for sound and picture manipulations
 - pickAFile()
 - makePicture()
 - makeSound()
 - show()
 - play()
 - explore()
- Note: some of these functions require *input* values (arguments) and return objects

Functions that takes input: parameter(s)

- Functions: like a box (with a name)
- The box has a "hole" (with a name): pass objects in
- The *named input* can only be used within the function ("scope")
- "parameters" ...when you write a function
"arguments" ...when you use a function



```
def playFile(myFileName):  
    mySound = makeSound(myFileName)  
    play(mySound)
```



```
def showFile(myFileName):  
    myPicture = makePicture(myFileName)  
    show(myPicture)
```

*Python examples
Similar to JavaScript*

Variable Scope

- Variables created INSIDE → available inside only
- "Local variable"
- Variables created OUTSIDE → available anywhere
- "Global variable"
- Must *not* use the **var** keyword

```
alert(x); //undefined

function mySimpleFunction() {
    var x = 500;
    alert(x);
}
mySimpleFunction(); //dialog box that says: 500

alert(x); //undefined still
```

```
var x;

function mySimpleFunction() {
    x = 500; //must NOT use "var" keyword
    alert(x);
}

mySimpleFunction(); //dialog box that says: 500

alert(x); //still says 500
```

Functions with Returns

- Instead of acting on Global variables (less secure) → use a **return**
- Define what comes out of a function, explicitly
- "return" command, must be last
- E.g.
 - `alert()` ...has no return
 - `prompt()` ...returns a character string
 - `confirm()` ...returns a boolean

These are examples using build-in functions from JavaScript

For next time...

Back to Python...

- Chapter 3 - p44-57 (stop before section 3.2.3)

Try:

- programs 9 - 23 (a lot but short and informative)

Learn:

- quotes
- concatenation (mad libs)
- functions with parameters (mad libs 2)
- multiplication of strings
- Python: FOR/IN and IF/IN
- technique: pile