

CSCI 141

Scott Wehrwein

Importing Modules
The random module

Goals

- Know how to `import` a `module` and call its functions.
- Know how to generate random numbers using the `random` module's `randint` function.

Other Peoples' Code

We've already used code other people wrote by calling built-in Python functions:

- `print, input, type`

Built-in functions are special because they're always available.

Many other functions exist in the Python Standard Library, which is a collection of **modules** containing many more functions.

Other Peoples' Code

An example: I want to generate a random integer between 0 and 10.

I don't know how to do this.

Someone who does has written some functions for me. They live in the `random` module:

```
import random
```

I could go look at the source code...

Ar
int

l d

Sc

Tr

l c

```
197
198 ## ----- integer methods -----
199
200 def randrange(self, start, stop=None, step=1, _int=int):
201     """Choose a random item from range(start, stop[, step]).
202
203     This fixes the problem with randint() which includes the
204     endpoint; in Python this is usually not what you want.
205
206     """
207
208     # This code is a bit messy to make it fast for the
209     # common case while still doing adequate error checking.
210     istart = _int(start)
211     if istart != start:
212         raise ValueError("non-integer arg 1 for randrange()")
213     if stop is None:
214         if istart > 0:
215             return self._randbelow(istart)
216         raise ValueError("empty range for randrange()")
217
218     # stop argument supplied.
219     istop = _int(stop)
220     if istop != stop:
221         raise ValueError("non-integer stop for randrange()")
222     width = istop - istart
223     if step == 1 and width > 0:
224         return istart + self._randbelow(width)
225     if step == 1:
226         raise ValueError("empty range for randrange() (%d, %d, %d)" % (istart, istop, width))
227
228     # Non-unit step argument supplied.
229     istep = _int(step)
230     if istep != step:
231         raise ValueError("non-integer step for randrange()")
```

e.

Other Peoples' Code

An example: I want to generate a random integer between 0 and 10.

I don't know how to do this.

Someone who does has written some functions for me. They live in the `random` module:

```
import random
```

I could go look at the source code... but I'd rather just use their functions without knowing **how** they work.

```
num = random.randint(0, 10)
```

Other Peoples' Code

```
import random  
num = random.randint(0, 10)
```

Two questions:

1. **What is this syntax about?**
2. How do I know what the function does?

Using Modules: Syntax

The Python Standard Library is a collection of **modules** containing many more functions.

To use functions in a module, you need to **import** the module using an **import statement**:

```
import module
```

(replace the text in *this font* with the specific module name)

By convention, we put all import statements at the **top** of programs.

Using Modules: Syntax

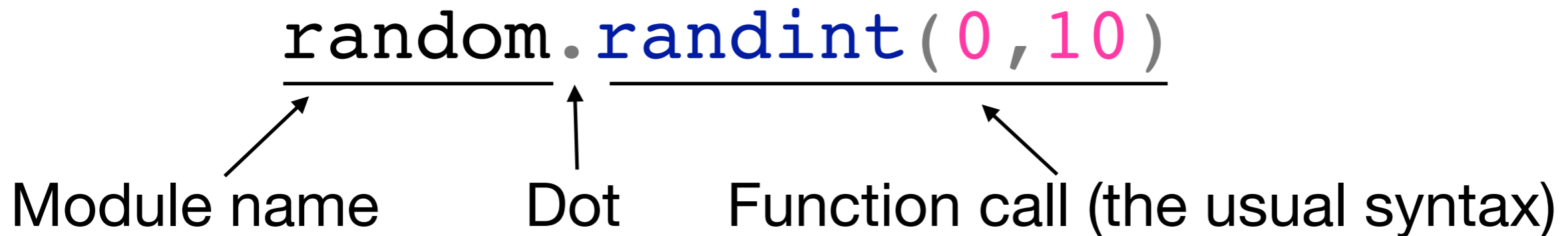
Once you've imported a module:

```
import random
```

you can call functions in that module using the following syntax:

random.randint(0,10)

Module name Dot Function call (the usual syntax)

A diagram illustrating the syntax of a module function call. The code 'random.randint(0,10)' is shown with horizontal lines underlining 'random', 'randint', and '(0,10)'. Three arrows point from labels below to these underlined parts: 'Module name' points to 'random', 'Dot' points to the period between 'random' and 'randint', and 'Function call (the usual syntax)' points to '(0,10)'. The word 'randint' is colored blue, '0' is pink, and '10' is pink.

Other Peoples' Code

```
import random  
num = random.randint(0, 10)
```

Two questions:

1. What is this syntax about?
- 2. How do I know what the function does?**

Other Peoples' Code

```
import random  
num = random.randint(0, 10)
```

Two questions:

1. What is this syntax about?
- 2. How do I know what the function does?**

Read about it in the Python documentation.

My approach, in practice:

1. Google “python 3 <whatever>”
2. Make sure the URL is from python.org and has version python 3.x

example

Demo

Demo

- `guess.py`
 - Pick a random number between 1 and 10
 - Count how many tries it takes a user to guess it.

math module

- The math module has useful stuff!
- You can read about it in the [documentation](#).
- logarithms, trigonometry, ...
- Modules can also contain values:

```
>>> import math
>>> math.pi
3.141592653589793
>>> math.e
2.718281828459045
>>>
```

More on import statements

- Import the entire module:

```
import random  
num = random.randint(1, 10)
```

- Import a specific function:

```
from math import sin  
sin0 = sin(0)
```

- Don't need module name dot notation
- Other math functions are not accessible

Demo

- Thonny shell:
 - `import math`
 - `from math import`
- `wave.py`
 - draw a sine-wavy picture using text art