Section 2

April 29, 2021

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Today’s Agenda

- Social Time (5 minutes)
- Recap/New (10 minutes)
- ohyay Dev Environment Explainer
- Mars Weight problem (5-10 minutes)
- 8-ball problem (20 minutes)
- Questions/comments/discussion (5 min)

Where We Are At

- This is the first week of Python
- Assignment 1 was due yesterday
- Lessons 1-5 have been published
- Perhaps I will publish my solutions

Social Time

- Triumphs?
- Challenges?
- Doesn’t have to be about Code in Place!

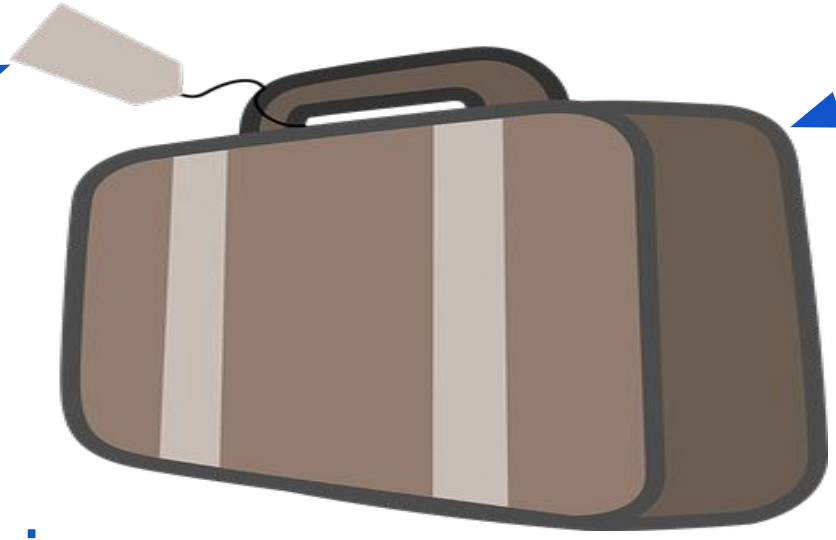
Recap

- Variables
- Types
- Type “casting”
- “=”, “==”, and “!=”
- Random module

Recap: Variables

X = 5

- Variable is like a piece of luggage
- The name is the tag



name

value

Stored in memory

Variable becomes a Python object, stored in memory

Every Python object has:
id
data type
value

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Recap: Variables

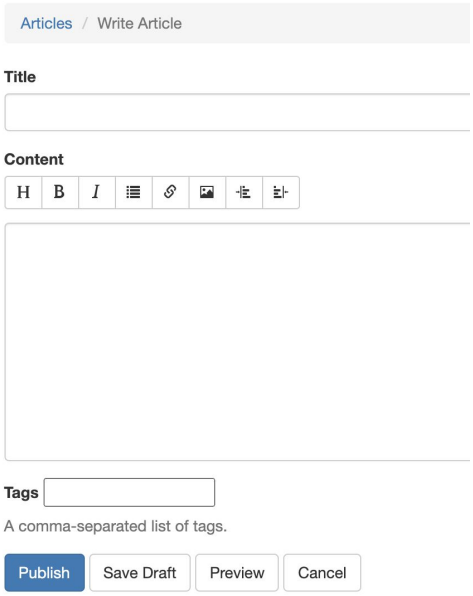
- Real world example: Django website

Variables represent unknown future data

```
class Article(models.Model):
    DRAFT = "D"
    PUBLISHED = "P"
    STATUS = ((DRAFT, _("Draft")), (PUBLISHED, _("Published")))

    user = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        null=True,
        related_name="author",
        on_delete=models.SET_NULL,
    )
    image = models.ImageField(
        _("Featured image"), upload_to="articles_pictures/%Y/%m/%d/"
    )
    timestamp = models.DateTimeField(auto_now_add=True)
    title = models.CharField(max_length=255, null=False, unique=True)
    slug = models.SlugField(max_length=80, null=True, blank=True)
    status = models.CharField(max_length=1, choices=STATUS, default=DRAFT)
    content = MarkdownxField()
    edited = models.BooleanField(default=False)
    tags = TaggableManager()
    objects = ArticleQuerySet.as_manager()

    class Meta:
        verbose_name = _("Article")
        verbose_name_plural = _("Articles")
        ordering = ("-timestamp",)
```



Form fields map to variables, which map to a database

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Recap: Type Examples

int is a number without a decimal point (**x = 5**)

float is a number with a decimal point (**x = 5.0**)

string is a word within single/double quotes (**x = “hello”**)

bool is a Boolean logic value (**True or False**)

Recap: Type Casting

- Use a built in function to convert a variable from one data type to another

  - int() to convert to integer
  - str() to convert to a string
  - float() to convert to a float

Recap: Type Casting

- How data works in the real world
- We have words, numbers, True/False, etc.
- You cannot do the same things to every data type

You can subtract one number from another number

10 - 5

You cannot subtract one word from another word

word - word

Recap: Type Casting

- The operations that Python can do for each data type are different; The operations Python can...

...do for strings

`str.capitalize()`
Return a copy of the string with its first character capitalized and the rest lowercased.

Changed in version 3.8: The first character is now put into titlecase rather than uppercase. This means that characters like digraphs will only have their first letter capitalized, instead of the full character.

`str.casefold()`
Return a casefolded copy of the string. Casefolded strings may be used for caseless matching.

Casefolding is similar to lowercasing but more aggressive because it is intended to remove all case distinctions in a string. For example, the German lowercase letter `'ß'` is equivalent to `"ss"`. Since it is already lowercase, `lower()` would do nothing to `'ß'`; `casefold()` converts it to `"ss"`.

The casefolding algorithm is described in section 3.13 of the Unicode Standard.

New in version 3.3.

`str.center(width[, fillchar])`
Return centered in a string of length *width*. Padding is done using the specified *fillchar* (default is an ASCII space). The original string is returned if *width* is less than or equal to `len(s)`.

`str.count(sub[, start[, end]])`
Return the number of non-overlapping occurrences of substring *sub* in the range [*start*, *end*]. Optional arguments *start* and *end* are interpreted as in slice notation.

`str.encode(encoding="utf-8", errors="strict")`
Return an encoded version of the string as a bytes object. Default encoding is `'utf-8'`. *errors*

are different than for integers

Operation	Result	Notes	Full documentation
`x + y`	sum of *x* and *y*		
`x - y`	difference of *x* and *y*		
`x * y`	product of *x* and *y*		
`x / y`	quotient of *x* and *y*		
`x // y`	floored quotient of *x* and *y*	(1)	
`x % y`	remainder of `x / y`	(2)	
`-x`	*x* negated		
`+x`	*x* unchanged		
`abs(x)`	absolute value or magnitude of *x*		`abs()`
`int(x)`	*x* converted to integer	(3)(6)	`int()`
`float(x)`	*x* converted to floating point	(4)(6)	`float()`

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Recap: Type Casting

1. input() accepts 2 integers as strings.

2. The strings are converted into integers and added.

Terminal

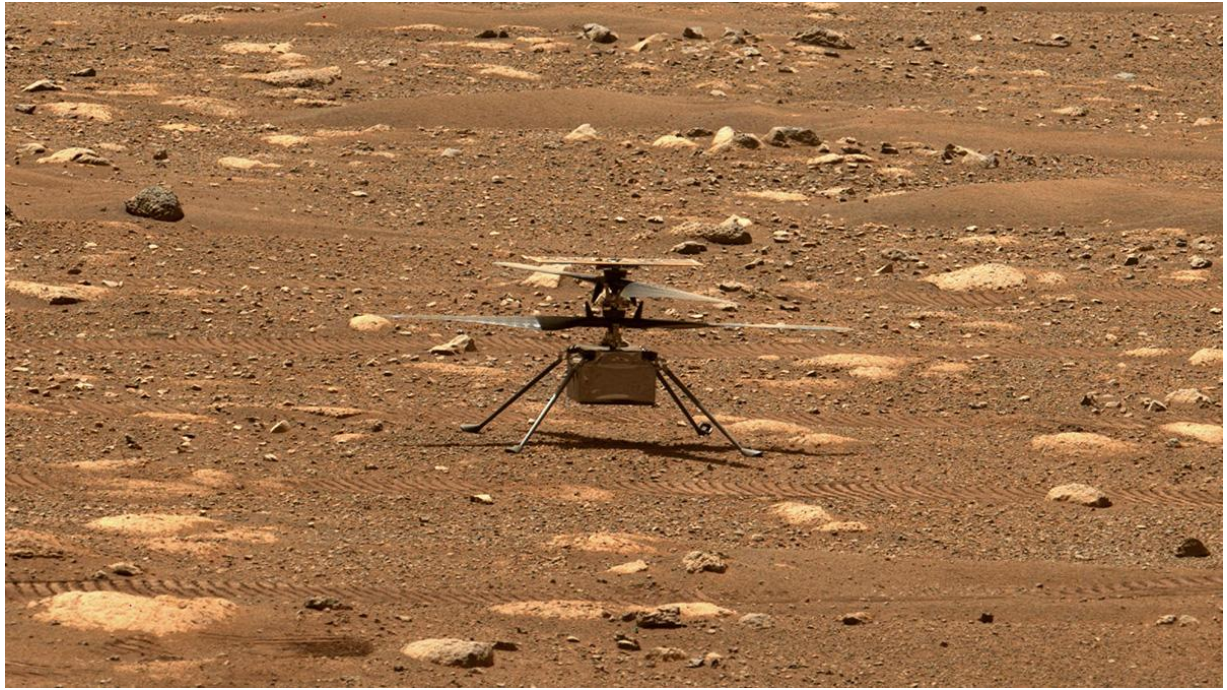
```
[user@sahara ~]$ python add2numbers.py
This program adds two numbers.
Enter first number: 1
Enter second number: 2
The total is 3.
[user@sahara ~]$
```

```
6  def main():
7      print("This program adds two numbers.")
8      num1 = input("Enter first number: ")
9      num1 = int(num1)
10     num2 = input("Enter second number: ")
11     num2 = int(num2)
12     total = num1 + num2
13     print("The total is " + str(total) + ".")
14
15 if __name__ == '__main__':
16     main()
```

3. The total is converted back into a string, to be displayed in the terminal.

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Mars Weight Problem



Katherine Michel, Section Leader, Stanford Code in Place, 2021

Open Source Python :)

python / cpython

Sponsor | Watch 1.4k

Code | Pull requests 1.5k | Actions | Security | Insights

master | 8 branches | 469 tags | Go to file | Add file | Code

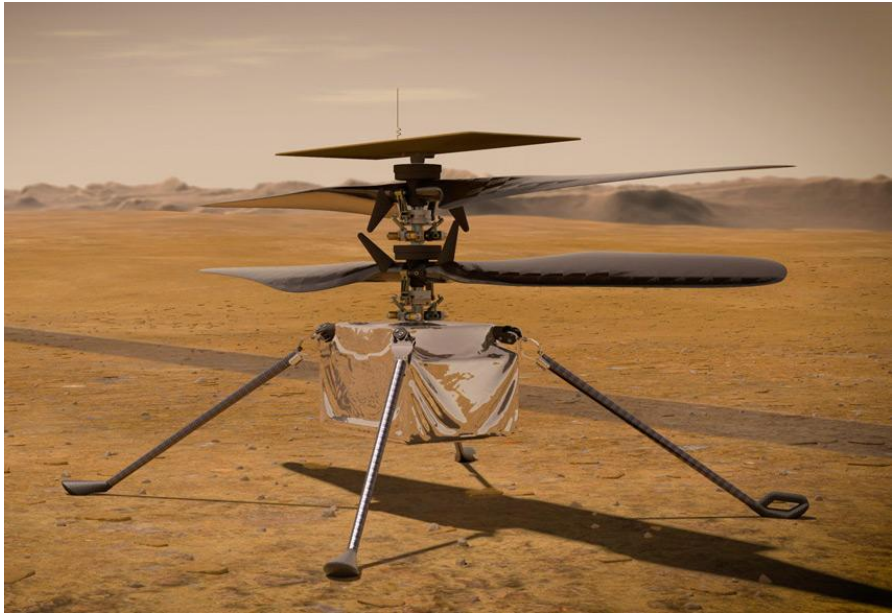
vstinner bpo-28254: _posixsubprocess uses PyGC_Enable/Py... 103d5e4 4 hours ago 109,735 commits

.azure-pipelines	bpo-43774: Doc job of Azure Pipelines uses Doc/requiremen...	19 days ago
.github	Restrict GITHUB_TOKEN permissions for the 'stale' workflow ...	5 days ago
Doc	bpo-43908: Add Py_TPFLAGS_IMMUTABLETYPE flag (GH-2...	4 hours ago
Grammar	bpo-43914: Highlight invalid ranges in SyntaxErrors (#25525)	5 days ago
Include	bpo-43908: Add Py_TPFLAGS_IMMUTABLETYPE flag (GH-2...	4 hours ago

Mars 2020 Helicopter Contributor

@KatherineMichel contributed code to 1 repository used in the **Mars 2020 Helicopter Mission:**

python/cpython



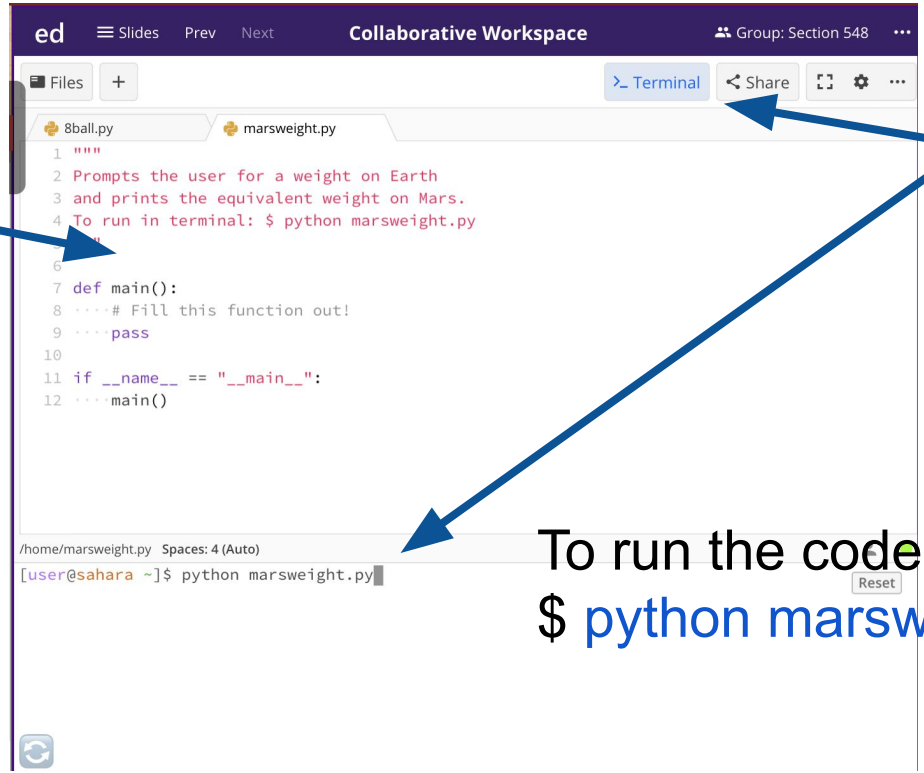
Katherine Michel, Section Leader, Stanford Code in Place, 2021

Open Source Python :)

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Collaborative Workspace

Everyone can see the same code and edit collaboratively



The "Terminal" button can be used to open a terminal…

To run the code type file name
$ python marsweight.py

Katherine Michel, Section Leader, Stanford Code in Place, 2021

New: “=”, “==”, and “!=”

“=” (assignment)	Use to assign a variable to a value (x = 5)
“==” (equals)	Use to check if left side and right side are equal (if variable == 1:)
“!=” (is not equal to)	Use to check if left side and right side are not equal (if variable != 2:)

New: “==” Example

ANSWER = "Hello world!" ← Example of a global constant

```
def main():
    user_response = input("Enter a number: ")
    response_string = int(user_response)
    if response_string == 1:
        print(ANSWER)
```

```
Enter a number: 1
Hello world!
```

By the way… if and while statements are the same in Karel and Python

New: Random

```
import random

random_number = random.randint(1, 42)
print(random_number)
```

8-Ball Problem

Katherine Michel, Section Leader, Stanford Code in Place, 2021

Closing… Thoughts, Questions, Discussion?