

Advanced Strings

String objects have a variety of methods we can use to save time and add functionality. Lets explore some of them in this lecture:

```
In [75]: s = 'hello world'
```

Changing case

We can use methods to capitalize the first word of a string, change cases to upper and lower case strings.

```
In [76]: # Capitalize first word in string  
s.capitalize()
```

```
Out[76]: 'Hello world'
```

```
In [77]: s.upper()
```

```
Out[77]: 'HELLO WORLD'
```

```
In [78]: s.lower()
```

```
Out[78]: 'hello world'
```

Location and Counting

```
In [80]: s.count('o')
```

```
Out[80]: 2
```

```
In [81]: s.find('o')
```

```
Out[81]: 4
```

Formatting

The center() method allows you to place your string 'centered' between a provided string with a certain length. Personally, I've never actually used this in code as it seems pretty esoteric...

```
In [83]: s.center(20, 'z')
```

```
Out[83]: 'zzzzhello worldzzzzz'
```

`expandtabs()` will expand tab notations `\t` into spaces:

```
In [84]: 'hello\t hi'.expandtabs()
```

```
Out[84]: 'hello  hi'
```

is check methods

These various methods below check if the string is some case. Lets explore them:

```
In [40]: s = 'hello'
```

`isalnum()` will return `True` if all characters in `S` are alphanumeric

```
In [41]: s.isalnum()
```

```
Out[41]: True
```

`isalpha()` will return `True` if all characters in `S` are alphabetic

```
In [43]: s.isalpha()
```

```
Out[43]: True
```

`islower()` will return `True` if all cased characters in `S` are lowercase and there is at least one cased character in `S`, `False` otherwise.

```
In [44]: s.islower()
```

```
Out[44]: True
```

`isspace()` will return `True` if all characters in `S` are whitespace.

```
In [45]: s.isspace()
```

```
Out[45]: False
```

`istitle()` will return `True` if `S` is a title cased string and there is at least one character in `S`, i.e. uppercase characters may only follow uncased characters and lowercase characters only cased ones. Return `False` otherwise.

```
In [47]: s.istitle()
```

```
Out[47]: False
```

isupper() will return True if all cased characters in S are uppercase and there is at least one cased character in S, False otherwise.

```
In [35]: s.isupper()
```

```
Out[35]: False
```

Another method is endswith() which is essentially the same as a boolean check on s[-1]

```
In [69]: s.endswith('o')
```

```
Out[69]: True
```

Built-in Reg. Expressions

Strings have some built-in methods that can resemble regular expression operations. We can use split() to split the string at a certain element and return a list of the result. We can use partition to return a tuple that includes the separator (the first occurrence) and the first half and the end half.

```
In [52]: s.split('e')
```

```
Out[52]: ['h', 'llo']
```

```
In [72]: s.partition('e')
```

```
Out[72]: ('h', 'e', 'llo')
```

```
In [58]: s
```

```
Out[58]: 'hello'
```

Great! You should now feel comfortable using the variety of methods that are built-in string objects!