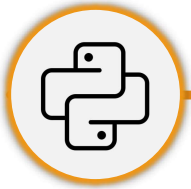


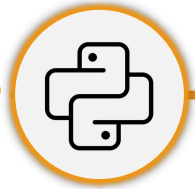
# Data Types in Python

# Data types



In this section we'll introduce **Python datatypes**, review their properties and learn how to identify and convert between them

TOPIC WE'LL COVER	GOALS FOR THIS SECTION
Data types	• Basics of data types in Python
The type function	• Learn how to determine an object's data type
Type conversion (Type casting)	• Learn to convert different types of data
Iterables	• Understand the concept of iterability
Mutability	• Understand the concept of mutability



# Python Data Types



Python has a series of **built-in data types**. Each data type has its own properties and use cases

- These data types can be grouped into the **following categories**:



Numeric



None



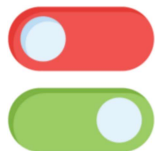
Set



Text



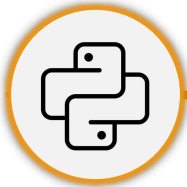
Sequence



Boolean



Mapping



# Python Data Types



## Single Value Data Types



### Numeric:

Represents numeric values

- Integers(**int**): -4, 0, 15
- Floats(**float**): 3.14, 0.0
- Complex(**complex**): 3 + 2j



### Text:

Represents sequences of characters (text)

- String(**str**): "Python", '@%\$!\*&'



### Boolean:

Represents True and False values

- Boolean(**bool**): True, False



### None:

Represents the absence of a value

- **NoneType**: None

## Multiple Value Data Types



### Sequence:

Represents sequence of values (Text / Numeric)

- List(**list**): [0, 10, 15], [1, True]
- Tuple(**tuple**): ('pants', 'cap')
- Range(**range**): range(0, 10, 1)



### Mapping:

Maps keys to values for efficient information retrieval

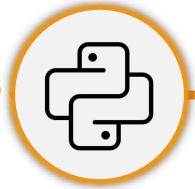
- Dictionary(**dict**): {'Black Cap': 10, 'Shirt': 15.99}



### Set:

Represents collection of unique non-duplicate data type

- Set(**set**): {'shirt', 'cap'}
- Frozenset(**frozenset**): {'shirt'}



# The `type()` function



The `type()` function will return the **data type** of the object passed to it

```
type(1)
```

int

```
type(1.5)
```

float

```
type("Hello World!")
```

str

```
type(None)
```

NoneType

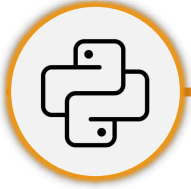
```
type([5, 10, 15])
```

list

```
type({"a":1, "b":2})
```

dict

💡 **DevTip:** Use `type()` function if you're getting **TypeError**.



# Type Conversion



Convert data types by using this notation : **name of data type** ( **object** )

**Example** Converting data into an **integer** data type

```
int("123")
```

123

→ `int( )` : function → `"123"` : Argument


Using `int()` converts the text **string** of `"123"` into an **integer** data type

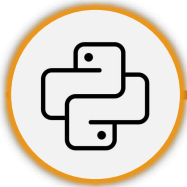
```
int([1, 2, 3])
```

**TypeError**

**TypeError** : `int( )` argument must be string, a bytes-like object or a number, not 'list'.

Using `int()` we want to convert a **list** into an **integer**, but we got **TypeError**. This operations isn't valid because the list has multiple values, while integer is just a simple one value!

 **Errors in Python will cause the code to stop running.**  
It's important to learn to diagnose and fix them.



# Let's Test What We've learned!



- Working with **type( )**
- Performing Type Conversion



Type\_Function\_&\_Type\_Conversion\_01.ipynb

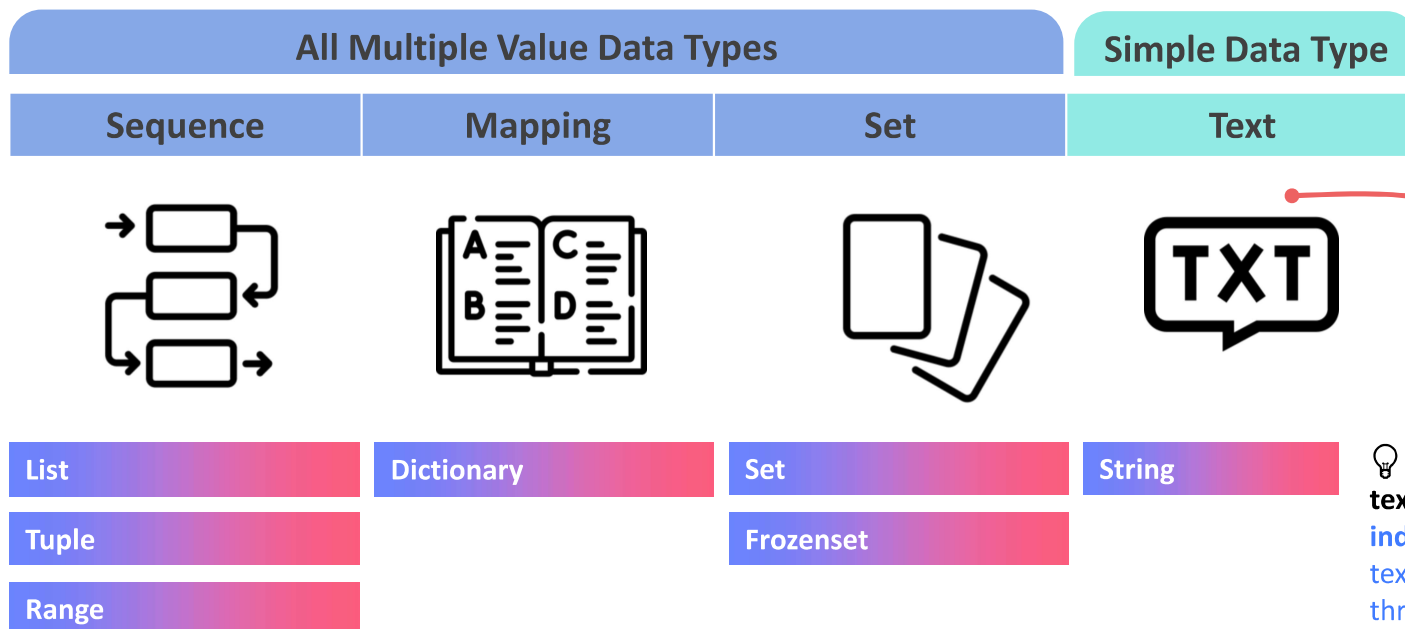


# Iterables



**Iterables** are data types that can **be iterated**, or **looped through**, allowing you to move from one value to the next one

- These data types are considered as **iterable**:



💡 Although we considered **text** as a **single value**, **individual characters** in a text string can be iterated through!



# Mutables



A data type is **mutable** if it can be **modified** after its creation

## Mutable Data Types

**Flexible** : can add, remove, change values in the object after creation!

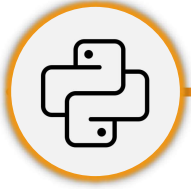
- Lists
- Dictionaries
- Sets

## Immutable Data Types

**Inflexible** : to Modify a value must delete and recreate the entire object.

- Integers
- Floats
- Strings
- Booleans
- Tuples
- Frozenset

💡 Immutable data types **quicker to access** (more efficient)



## Section Wrap-Up



- ✓ Python has a wide variety of **data types** with different properties
  - integer, float, string, Boolean, list, dictionary, tuple, set, frozenset
- ✓ The **type( ) function** allows you to specify an object's data type
  - You can change between different data types using **type conversion**
- ✓ **Iterables** are data types with values that can loop through
  - All multiple value data types; **list, tuple, range, dictionary, set, frozenset** and just **string** in single value data type
- ✓ Data types can be **mutable** or **immutable**
  - Some data can be modified after creation, while some can not (you need to **delete** and **recreate** them)