








Python


Tuples, Sets and Dictionaries

Session Expectations:

- Cameras on 
- Mics muted 
- Raise Hand 
 - React 
 - Respond **"?"**



Learning Objectives

- 
- **Intent:** To explore the **additional data types** available in Python.
 - **Implementation:** To work with the different data types Python offers us.
 - **Impact:** To contextualise when and why we would utilise a **Tuple**, a **Set** or a **Dictionary**.

New Data Types:

We have covered the similarities between **JavaScript** and **Python**. However, **Python** offers us some new Data Types that we have not worked with before.


The Data Types in question are **Tuples** and **Sets**.

Tuple:

A tuple in Python is a fixed, ordered collection of items, written with parentheses.

Once created, you can't change its elements, making it useful for data that shouldn't change.


Think of a tuple like a list that will not be changed.

 example.py

```
# Creating a tuple with different types of data
person = ('John Doe', 30,
          'john.doe@example.com', True)

# Accessing elements of the tuple
print("Name:", person[0])      # Output:
Name: John Doe
print("Age:", person[1])       # Output:
Age: 30
print("Email:", person[2])     # Output:
Email: john.doe@example.com
print("Is Active:", person[3]) # Output:
Is Active: True
```

Set:

 example.py

```
# Creating a set
my_set = {1, 2, 3, 4, 5}

# Adding elements to a set
my_set.add(6)
my_set.add(3) # Adding a duplicate element
              (will not be added again)
```

A set in Python is an unordered collection of unique elements.

Unlike lists or tuples, sets do not maintain any specific order of elements.

Sets automatically handle duplicate values by only keeping unique ones.

You can add or remove elements from a set using methods.

Useful for operations that require unique elements.

Dictionary (Object):

A dictionary in Python is an unordered collection of key-value pairs, written with curly braces.


Each key in a dictionary must be unique and is used to access the corresponding value.

Once created, you can add, modify, or remove key-value pairs as needed, making dictionaries extremely flexible and useful for storing and managing dynamic data.

Think of a dictionary like a JavaScript object.

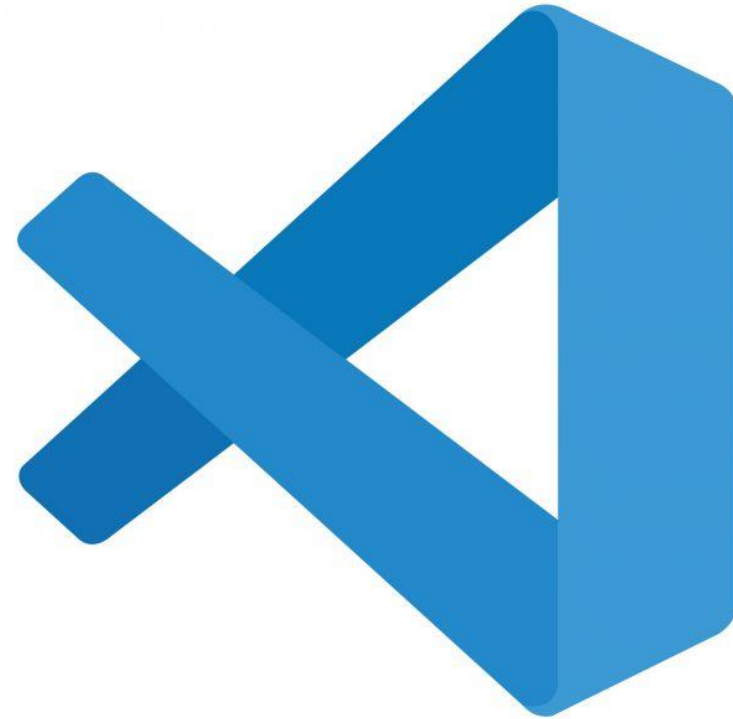
 example.js

```
// JavaScript object example
const person = {
  name: "Alice",
  age: 30,
  city: "New York"
};
```

 example.py

```
# Python dictionary example
person = {
    "name": "Alice",
    "age": 30,
    "city": "New York"
}
```

**Let's move over
to VS Code to
work through
some examples**



Reference Links:

Tuple: <https://www.programiz.com/python-programming/tuple>

Set: <https://www.programiz.com/python-programming/set>

Dictionary: <https://www.programiz.com/python-programming/dictionary>