

01 - Collection Classes - Maps

A Map is a Collection class that allows storing **key-value** pairs

The *key* - is a unique identifier to associate with a value
The *value* - the data to be associated with a key

Map Name	
key	value

A Map is also known as "associative array"

Use a Map when you need to associate a piece of data (value) with an unique identifier (key):

- StateCode - StateName
- Student # - Student
- Name - where they live

Map Name	
<i>key</i>	<i>value</i>
OH	Ohio
AK	Alaska

- the content of the key must be unique
- the content of the value does not have to be unique

Types of Maps:

- HashMap - entries are stored in an unknown order using Hash Code
- TreeMap - entries are stored in key sequence order
- LinkedHashMap - entries are stored in the order in which they were added

Two common ways of defining a map:

```
Map<key-data-type, value-data-type> nameOfMap = new typeOfMap<>();  
  
typeOfMap<key-data-type, value-data-type> nameOfMap = new typeOfMap<>();
```

Define a Map where the key is a String and the value is String

```
Map<String, String> myMap = new HashMap<>();  
HashMap<String, String> myMap = new HashMap<>();  
Map<String, String> myMap = new HashMap<String, String>();  
HashMap<String, String> myMap = new HashMap<String, String>();  
  
Map<String, String> myMap = new TreeMap<>();  
TreeMap<String, String> myMap = new TreeMap<>();  
Map<String, String> myMap = new TreeMap<String, String>();  
TreeMap<String, String> myMap = new TreeMap<String, String>();  
  
Map<String, String> myMap = new LinkedHashMap<>();  
LinkedHashMap<String, String> myMap = new LinkedHashMap<>();  
Map<String, String> myMap = new LinkedHashMap<String, String>();  
LinkedHashMap<String, String> myMap = new LinkedHashMap<String, String>();
```

.put(key, value) will add an entry to the Map or update the value if key already exists

```
// Define a Map with associate a learner # with a Learner.  
// key is an Integer and the value is a String  
// key-type, value-type  
  
Map<Integer, String> learners = new HashMap<Integer, String>();  
  
learners.put(9, "Andrew");  
learners.put(3, "Giang");  
learners.put(6, "Casey");  
learners.put(9, "Kathy");
```

learners	
6	"Casey"
9	"Kathy"
3	"Giang"

Since primitives cannot be stored a a Collection class object (only Objects are allowed),

A set of "Wrapper Classes" are provided by Java to represent primitives as objects:

<u>primitive</u>	<u>Wrapper</u>
int	Integer
double	Double
float	Float
boolean	Boolean
char	Character

Wrapper classes also provide methods we can find useful:

- Integer.parseInt(string)** - Convert a String to an int
- Double.parseDouble(string)** - Convert a String to an double
- Integer.toString()** - Convert an Integer object to a String

For more information on Java Wrapper classes:

https://www.w3schools.com/java/java_wrapper_classes.asp