Important points about Map in Apex

sweety kumari

In Apex, a map is a collection that associates a set of keys with their corresponding values. Each key in a map is unique, and maps do not preserve the order of their elements.

A map in Apex is declared using the syntax:

```
Map<KeyType, ValueType> mapName = new
Map<KeyType, ValueType>();
```

where KeyType is the data type of the key and ValueType is the data type of the value.
For example, a map that associates String keys with Integer values can be declared as:

```
Map<String, Integer> myMap = new
Map<String, Integer>();
```

map Methods

- **put(Object key, Object value): Associates the specified value with the specified key in the map.**

Code:

```
Map<String, Integer> myMap = new
Map<String, Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
```

- **keySet(): Returns a set of all keys in the map.**

Code:

```
Map<String, Integer> myMap = new Map<String,
Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
Set<String> keySet = myMap.keySet(); // returns a
set of all keys
```

- **clear(): Removes all key-value pairs from the map.**

Code:

```
Map<String, Integer> myMap = new Map<String, Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
myMap.clear(); // removes all key-value pairs from the map
```

- **clone(): Creates a shallow copy of the map.**

Code:

```
Map<String, Integer> myMap = new Map<String, Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
Map<String, Integer> clonedMap = myMap.clone(); // creates
a shallow copy of myMap
```

- **containsKey(Object key): Returns true if the map contains the specified key**

Code:

```
Map<String, Integer> myMap = new Map<String, Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
Boolean containsKey = myMap.containsKey(‘apple’); // returns true
```

- **containsValue(Object value): Returns true if the map contains the specified value**

Code:

```
Map<String, Integer> myMap = new Map<String, Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
Boolean containsValue = myMap.containsValue(20); // returns true
```

- **entrySet(): Returns a set of all key-value pairs in the map**

Code:

```
Map<String, Integer> myMap = new Map<String,
Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
Set<Map.Entry<String, Integer>> entrySet =
myMap.entrySet(); // returns a set of all key-value pairs
```

- **get(Object key): Returns the value associated with the specified key**

Code:

```
Map<String, Integer> myMap = new Map<String,
Integer>();
myMap.put(‘apple’, 10);
myMap.put(‘banana’, 20);
Integer value = myMap.get(‘apple’); // returns 10
```

- **isEmpty(): Returns true if the map is empty.**

Code:

```
Map<String, Integer> myMap = new Map<String, Integer>
();
Boolean isEmpty = myMap.isEmpty(); // returns true
```

- **putAll(Map< extends Object, extends Object> m): Copies all key-value pairs from the specified map to this map.**

Code:

```
Map<String, Integer> myMap = new Map<String,
Integer>();
Map<String, Integer> anotherMap = new Map<String,
Integer>();
anotherMap.put('pear', 30);
anotherMap.put('orange', 40);
myMap.putAll(anotherMap); // copies all key-value
pairs from anotherMap to myMap
```

- **remove(Object key): Removes the key-value pair associated with the specified key from the map.**

Code:

```
Map<String, Integer> myMap = new Map<String,
Integer>();
myMap.put(’apple’, 10);
myMap.put(’banana’, 12);
myMap.remove(’apple’);
```

Overall, maps are a powerful tool in Apex that allow you to store and retrieve data efficiently using key-value pairs.

Hope you found this useful and handy to use for your day to day life coding