**Dart collections:**

A [collection](https://api.dart.dev/stable/dart-core/dart-core-library.html#collections) is an object that contains other objects. It allows us to store, retrieve data in the form of a collection per say. For example:

* List: an ordered collection of objects with a length (also called an *array*).
* Set: an unordered collection of unique objects.
* Map: an unordered collection of key-value pairs.

One thing that is common between these collection types is that all of them implements an Iterable. An Iterable is just another object capable of returning its element in iterations or in easy words, one-by-one.

Also, Dart supports syntax for constructing three types of collections: list literals ([]), map literals ({}) and set literals (also {}).

**List:**

List, also known as an Array in some languages is simply is an ordered collection of objects which has a length. The list can be iterated through indices, starting off with the zero index and ending at length-1. To define a list in Dart, we enclose its elements with square brackets [].

Example:

var list = [1, 2, 3];

 Dart Lists offers a bunch of functions and properties. As adding and removing elements from a list by using the **Add()** and **Remove()** methods.

Moreover, it allows you to access and assign values in any element in the list using **subscript notation** where the index goes within the square brackets after the list name.

Immutable list offered by using the **final keyword** and checking the first and last element using the **first** and **last** properties.

Check what the list contains use **isEmpty** or **isNotEmpty** property and iterating over the list using the **for** statement.

**Set:**

A Set-in dart is an unordered collection of unique objects. Since the collection is of unordered objects, we cannot iterate through the elements using Indices like we can in other Collection Types such as Lists. To define a set in Dart, we enclose its elements with square brackets {}.

Example:

Var set = {1,2,3};

Dart set also has its own functions and properties. To find the number of elements of asset we use the **length property** and to access an element at an index we can use **elementAt().**As list we can use the **first** and **last** property to access the first and last element.

In addition**, add()** and **addAll()** are used to add an element or more than an element to the set and for checking the existence of an element we use **contains()** method. The intersection and union of two sets are done using the. **intersection()** and .**union()** property and to iterate over elements in asset we use the **for-in** loop.

**Map:**

The Map object is a simple key/value pair. Keys and values in a map may be of any type. A Map is a dynamic collection. In other words, Maps can grow and shrink at runtime. To define a Map in Dart, we enclose its elements with square brackets {}.

Example:

Var string = {‘tea’:1, ’milk’:2, ‘sugar’:3};

For the Dart map functions and properties,

We can access an element using the **subscript notation** also and to **add** a new element to a map we have to assign the element with a key that is not available on the map, thus we can also **remove** an element with its key by using the remove method ().

Added to that, for **updating** any assign value to an element with a key already exist we just update its value.

To sum it up, Collections are powerful tools for managing and manipulating data and selecting the appropriate collection type is essential for efficient and effective programming. Consider the requirements of your program and choose between lists, sets, and maps accordingly.