**Using "for" Loops In Lists**

Just as you can also use the if keyword inside of lists (to add elements conditionally), you can also use the for keyword to add multiple items into a list:

1. final numbers = [5, 6];
2. final myList = [
3. 1,
4. 2,
5. for (final num in numbers)
6. num
7. ];

In this example, the numbers 5 and 6 will be added to myList (hence myList thereafter is [1, 2, 5, 6]).

This for ... in syntax is a special variation of the for loop that loops through multiple items in a list. You will see it again later in the course - both outside and inside of a list. It will also be explained again later.

The idea behind this loop is to simplify the process of performing some operation on all items in a list.

When used in a list, it's essentially an alternative to the spread operator (...):

1. final numbers = [5, 6];
2. final myList = [
3. 1,
4. 2,
5. ...numbers
6. ];

It can be useful in scenarios where values must be transformed before being added to a list - the for ... in loop can then be used instead of map() + spread operator:

1. final numbers = [5, 6];
2. final myList = [
3. 1,
4. 2,
5. ...numbers.map((n) {
6. return n \* 2;
7. }) // adds 10 and 12
8. ];

can be replaced with:

1. final numbers = [5, 6];
2. final myList = [
3. 1,
4. 2,
5. for (final num in numbers)
6. num \* 2 // adds 10 and 12
7. ];

As mentioned, you will learn more about for later in the course.

You can also learn more about for ... in inside of lists here: <https://github.com/dart-lang/language/blob/master/accepted/2.3/control-flow-collections/feature-specification.md#repetition>