



David Odejobi



# A QUICK GUIDE TO TOP LIST FUNCTIONS IN FLUTTER/DART

---

@iamDavidOdejobi

# map()

This function is used to transform each element of a list into a new element. It takes a callback function as an argument, and this function is applied to each element of the list. The callback function takes one argument, the current element, and returns the new element.



```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    List<int> squaredNumbers = numbers.map((number) => number * number).toList();  
    print(squaredNumbers); // [1, 4, 9, 16, 25]  
}
```

03

# where()

This function is used to filter elements of a list based on a certain condition. It takes a callback function as an argument, and this function is applied to each element of the list. The callback function takes one argument, the current element, and returns a boolean indicating whether the element should be included in the new list or not.

```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    List<int> evenNumbers = numbers.where((number) ⇒ number % 2 == 0).toList();  
    print(evenNumbers); // [2, 4]  
}
```

# forEach()

This function is used to iterate over the elements of a list and perform some action on each element. It takes a callback function as an argument, and this function is applied to each element of the list. The callback function takes one argument, the current element.



```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    numbers.forEach((number) => print(number));  
    // Output: 1, 2, 3, 4, 5  
}
```

05

# reduce()

This function is used to combine all elements of a list into a single value. It takes two arguments: a callback function and an initial value.

The callback function takes two arguments, the accumulated value and the current element, and returns the new accumulated value.

```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    int sum = numbers.reduce((acc, curr) => acc + curr);  
    print(sum); // 15  
}
```

# any()

This function is used to check if any elements of a list satisfy a certain condition. It takes a callback function as an argument, and this function is applied to each element of the list. The callback function takes one argument, the current element, and returns a boolean indicating whether the element satisfies the condition or not. It returns true if any elements satisfy the condition, otherwise false.



```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    bool anyEven = numbers.any((number) => number % 2 == 0);  
    print(anyEven); // true  
}
```

07

# firstWhere()

This function is used to find the first element in a list that satisfies a certain condition. It takes two arguments: a callback function and an optional default value. The callback function takes one argument, the current element, and returns a boolean indicating whether the element satisfies the condition or not. It returns the first element that satisfies the condition or the default value if not found.



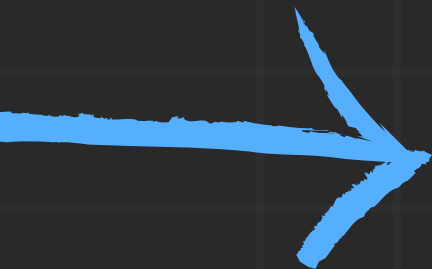
```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    int firstEven = numbers.firstWhere((number) => number % 2 == 0, orElse: () => -1);  
    print(firstEven); // 2  
}
```

# lastWhere()


This function is used to find the last element in a list that satisfies a certain condition. It takes two arguments: a callback function and an optional default value. The callback function takes one argument, the current element, and returns a boolean indicating whether the element satisfies the condition or not. It returns the last element that satisfies the condition or the default value if not found.

```
void main() {  
    List<int> numbers = [1, 2, 3, 4, 5];  
    int lastEven = numbers.lastWhere((number) ⇒ number % 2 == 0, orElse: () ⇒ -1);  
    print(lastEven); // 4  
}
```





**DONT FORGET  
TO LIKE, SHARE  
AND SAVE IF  
YOU LIKE THIS  
POST**



@iamDavidOdejob

