

Python Arrays

Python does not have built-in support for Arrays, but [Python Lists](#) can be used instead.

Arrays are used to store multiple values in one single variable:

Example

Create an array containing car names:

```
cars = ["Ford", "Volvo", "BMW"]
```

Access the Elements of an Array

You refer to an array element by referring to the *index number*.

Example

Get the value of the first array item:

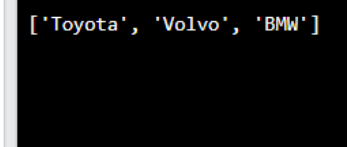
```
cars = ["Ford", "Volvo", "BMW"]  
x = cars[0]  
print(x)
```

A terminal window with a black background and a light gray border. The word "Ford" is printed in white text at the top of the window.

Example

Modify the value of the first array item:

```
cars = ["Ford", "Volvo", "BMW"]  
cars[0] = "Toyota"  
print(cars)
```

A terminal window with a black background and a light gray border. The list ["Toyota", "Volvo", "BMW"] is printed in white text at the top of the window.

The Length of an Array

Use the `len()` method to return the length of an array (the number of elements in an array).

Example

Return the number of elements in the `cars` array:

```
cars = ["Ford", "Volvo", "BMW"]  
  
x = len(cars)  
  
print(x)
```

A terminal window with a black background and white text showing the output of the code, which is the number 3.

3

Looping Array Elements

You can use the `for in` loop to loop through all the elements of an array.

Example

Print each item in the `cars` array:

```
cars = ["Ford", "Volvo", "BMW"]  
  
for x in cars:  
    print(x)
```

A terminal window with a black background and white text showing the output of the code, which are the elements of the cars array: Ford, Volvo, and BMW.

Ford
Volvo
BMW

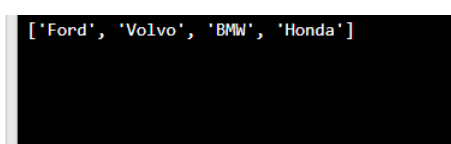
Adding Array Elements

You can use the `append()` method to add an element to an array.

Example

Add one more element to the `cars` array:

```
cars = ["Ford", "Volvo", "BMW"]  
  
cars.append("Honda")  
  
print(cars)
```

A terminal window with a black background and white text showing the output of the code, which is the updated cars array: ['Ford', 'Volvo', 'BMW', 'Honda'].

```
['Ford', 'Volvo', 'BMW', 'Honda']
```

Removing Array Elements

You can use the `pop()` method to remove an element from the array.

Example

Delete the second element of the `cars` array:

```
cars = ["Ford", "Volvo", "BMW"]  
cars.pop(1)  
print(cars)
```

```
['Ford', 'BMW']
```

You can also use the `remove()` method to remove an element from the array.

Example

Delete the element that has the value "Volvo":

```
cars = ["Ford", "Volvo", "BMW"]  
cars.remove("Volvo")  
print(cars)
```

```
['Ford', 'BMW']
```

Array Methods

Python has a set of built-in methods that you can use on lists/arrays.

Method	Description
append()	Adds an element at the end of the list
clear()	Removes all the elements from the list

<code>copy()</code>	Returns a copy of the list
<code>count()</code>	Returns the number of elements with the specified value
<code>extend()</code>	Add the elements of a list (or any iterable), to the end of the current list
<code>index()</code>	Returns the index of the first element with the specified value
<code>insert()</code>	Adds an element at the specified position
<code>pop()</code>	Removes the element at the specified position
<code>remove()</code>	Removes the first item with the specified value
<code>reverse()</code>	Reverses the order of the list
<code>sort()</code>	Sorts the list

While True Statement

In Python, the `True` keyword is a boolean expression. It's used as an alias for `1`, and the `while` keyword is used to specify a loop. The statement `while True` is used to specify an infinite `while` loop.

An infinite loop runs indefinitely until the end of time or when the program is forcefully stopped. The following code example below shows us how we can create an infinite loop with the `while True` statement.

```
while True:  
  
    print("Hello World")
```

Output:

```
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World  
  
Hello World
```

We created an infinite `while` loop that prints `Hello World` every time it is executed by using the `while True` statement in the code above. This approach is not recommended because it stops the code from its completion.

One workaround is the use of the `break` statement inside the infinite loop to stop the process when a particular condition is satisfied. This approach is demonstrated in the following program below.

```
i = 0

while True:

    print("Hello World")

    i+=1

    if i == 10:

        break
```

Output:

[illegible]

We stopped the infinite `while` loop by using the `break` statement in the code above. The execution of the infinite loop was stopped after the value of the integer variable `i` becomes equal to `10`.