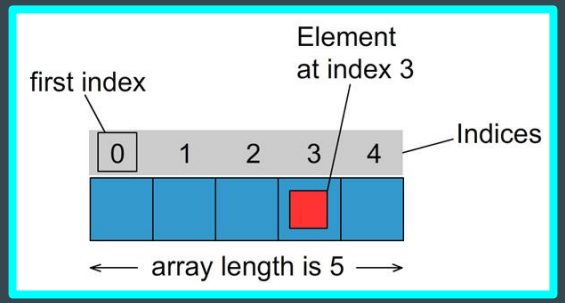
Arrays and ArrayLists

visual	representation	of	an	array

A tutorial by Joshua Rovner

The Basics

- An array/arraylist is a data structure which holds a group of elements with the same data type - arrays can hold any data type, but arraylists can only hold non-primitive types.
- Each element has its own index in the array - the first element has index 0, and the last element has index array length - 1.
- For example, in an array of 6 elements, the third element would have index 2 and the last element would have index 5.



Array Specifics

- At any time, an array retains the same number of elements as it was declared with.
- To delete or add an element, a new array must be made and the contents of the old one must be copied into the new one.
- **length** field - printing out the statement ***yourArrayName*.length** will give you an integer representing the length of the array.
- **Traversing** an array - going through and performing some sort of action on each element.

```
public static void main (String[] args)
{
    int sum = 0;
    int[] exampleArray = {1, 7, 3, 9, 6};

    for (int i = 0 ; i < exampleArray.length ; i++)
    {
        sum += exampleArray [i];
        System.out.println ("Adding " + exampleArray [i] + " at index " + i + " : current sum = " + sum + ".");
    }
}
```

```
Adding 1 at index 0 : current sum = 1.
Adding 7 at index 1 : current sum = 8.
Adding 3 at index 2 : current sum = 11.
Adding 9 at index 3 : current sum = 20.
Adding 6 at index 4 : current sum = 26.
```

Java Syntax for Arrays

- **Declaration** - dataType[] arrayName = new dataType[length]; **or** dataType[] arrayName = new dataType[] {put, contents, here};
  - **Shorthand** - dataType[] arrayName = {put, contents, here};
- When accessing an index, use square brackets around the index value - arrayName[integerIndex].

```
int[] exampleArray = new int[] {1, 7, 3, 9, 6};
```

```
int[] exampleArray2 = {1, 7, 3, 9, 6};
```

```
int valueAtIndex2 = exampleArray[2];
```

ArrayList Specifics

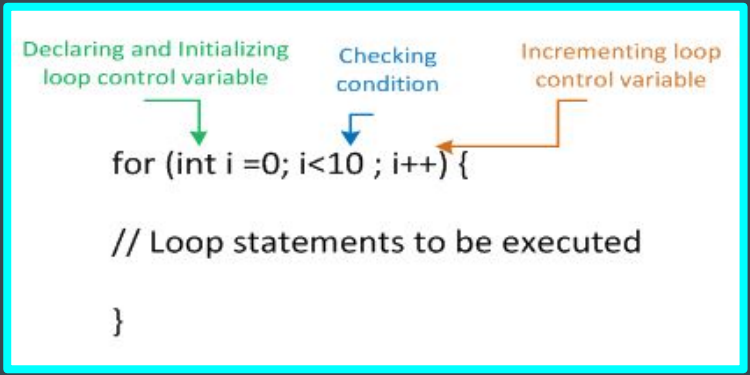
- An arraylist has a malleable number of elements.
- To delete or add an element, a method call can be used (more on this later).
- **size ()** method - printing out the statement ***yourArrayListName*.size ()** will give you an integer representing the length of the array.
- Syntax for declaring an arraylist - ArrayList<ContentsType> arrayName = new ArrayList<ContentsType>();
- To add elements during declaration, you must use the Arrays.addList () method, and.put each desired element into the parentheses with a comma between them (see code example).

ArrayList Methods

- ArrayLists have several methods which are extremely useful:
  - **boolean add (E e)** - adds the desired element to the end of the list and returns true if successful.
  - **void add (int index, E element)** - adds the desired element to the specified index and pushes all elements afterwards one place back.
  - **E remove (int index)** - removes the contents at the specified index, shifts the elements behind the index one place forward, and returns the data being removed.
  - **E set (int index, E element)** - sets the element at the specified index to the specified value and returns the data being replaced.
  - **E get (int index) -** returns the contents of the specified index.

Accessing Elements

- **For loop** - a loop that iterates from a certain start point until it reaches a certain condition. When used on arrays and arraylists, this can access only certain index values.
- **For-each loop** - a loop that performs the same action on **every** element in a list. This should **never** be used when creating new elements or objects.




```
for (String s : array)
    s += "for-each";
```