Array in Java

Array is collection of homogeneous(same) datatype elements sharing a common name.

Arrays in Java are a fundamental data structure used to store a fixed-size sequence of elements of the same type.

Array requires sequential byte of memory.

Array Indexing starts with 0, Ends at (arraysize-1).

Here's a comprehensive overview of arrays in Java:

1. Declaration and Initialization

Declaration:

You first declare an array by specifying the type of elements it will hold and its name.

```
int[] myArray; // Declaration of an integer array
```

Initialization:

You can initialize an array in several ways:

1. **Static Initialization:**

   ```
   int[] myArray = {1, 2, 3, 4, 5}; // Directly assigning values
   ```

2. **Dynamic Initialization:**

   ```
   int[] myArray = new int[5]; // Creates array with 5 elements, initialized to 0
   ```

You can also initialize an array of objects:

```
String[] names = new String[3]; // Array to hold 3 strings,initially all null
```

2. Accessing Array Elements

Array elements are accessed using an index, starting from `0` for the first element.

```
int[] numbers = {10, 20, 30, 40, 50};
System.out.println(numbers[0]); // Prints 10
```

3. Array Length

The length of an array can be obtained using the `length` property.

```
int[] numbers = {10, 20, 30, 40, 50};
System.out.println(numbers.length); // Prints 5
```

4. Multidimensional Arrays

Java supports multidimensional arrays, such as 2D arrays.

Declaration and Initialization:

```
int[][] matrix = new int[3][4]; // 3 rows and 4 columns
```

Accessing Elements:

```
matrix[0][0] = 1;
int value = matrix[0][0]; // value is 1
```

Initialization with Values:

```
int[][] matrix = {
    {1, 2, 3, 4},
    {5, 6, 7, 8},
    {9, 10, 11, 12}
};
```

5. Common Operations

Iterating Over Arrays:

You can use a `for` loop or an enhanced `for` loop (foreach loop) to iterate over array elements.

```
// Using a regular for loop
for (int i = 0; i < numbers.length; i++)
{
    System.out.println(numbers[i]);
}

// Using an enhanced for loop
for (int num : numbers)
{
    System.out.println(num);
}
```

Copying Arrays:

You can use `System.arraycopy()` or `Arrays.copyOf()` for copying arrays.

```
int[] source = {1, 2, 3, 4, 5};
int[] destination = new int[5];
System.arraycopy(source, 0, destination, 0, source.length);
```

6. Important Points

- **Fixed Size:** Once an array is created, its size cannot be changed.
- **Default Values:** Numeric arrays are initialized to `0`, `boolean` arrays to `false`, and object arrays to `null`.

- **Bounds Checking:** Java performs bounds checking, so accessing an invalid index will throw an `ArrayIndexOutOfBoundsException`.

Example - Program

Here's a simple Java program that demonstrates the usage of arrays:

```
public class ArrayExample
{
    public static void main(String[] args)
    {
        // Declare and initialize an array
        int[] numbers = {10, 20, 30, 40, 50};

        // Print array length
        System.out.println("Array length: " + numbers.length);

        // Access and print each element
        for (int i = 0; i < numbers.length; i++)
        {
            System.out.println("Element at index " + i + ": " +
            numbers[i]);
        }
    }
}
```

This covers the basics of arrays in Java.