Module 5

Arrays

Objectives

- Declare and create arrays of primitive, class, or array types
- Explain why elements of an array are initialized
- Explain how to initialize the elements of an array
- Determine the number of elements in an array
- Create a multidimensional array
- Write code to copy array values from one array to another

Relevance

What is the purpose of an array?

Declaring Arrays

- Group data objects of the same type.
- Declare arrays of primitive or class types:

```
char s[];
Point p[];

char[] s;
Point[] p;
```

- Create space for a reference.
- An array is an object; it is created with `new`.

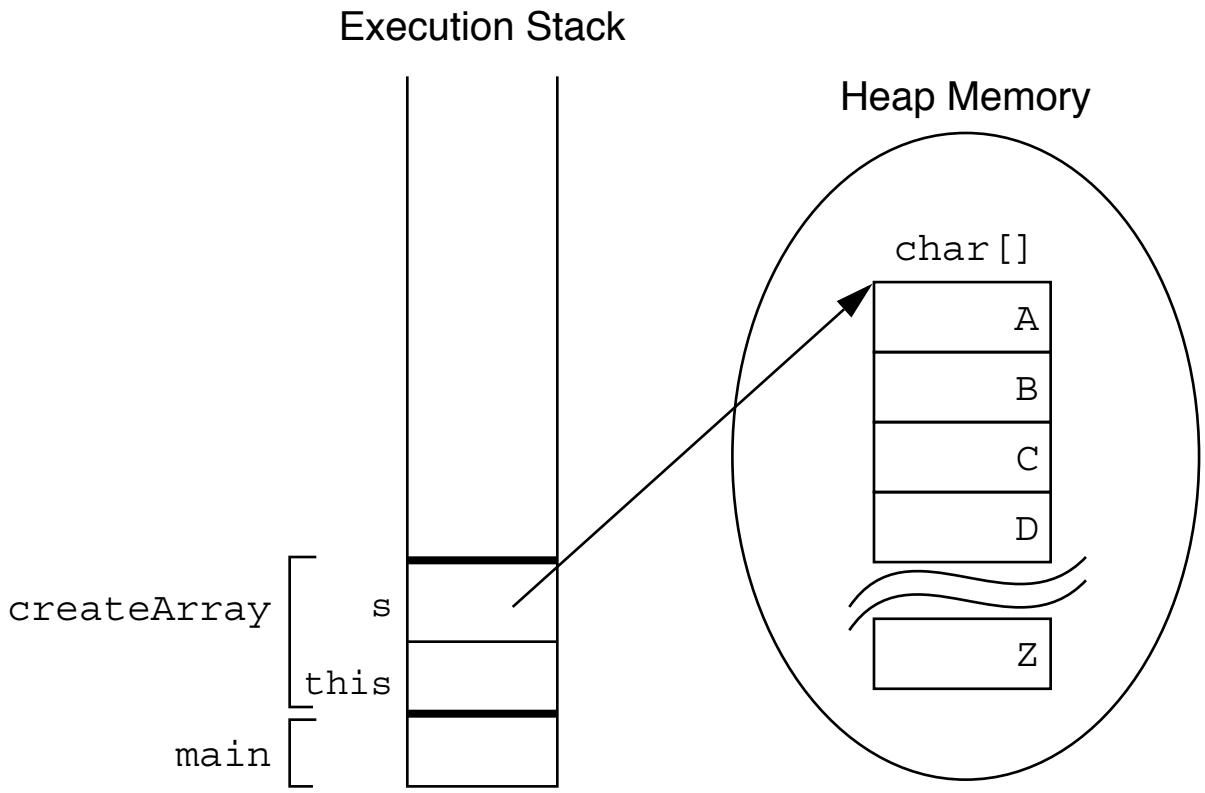
Creating Arrays

Use the `new` keyword to create an array object.

For example, a primitive (`char`) array:

```
1   public char[] createArray() {
2     char[] s;
3
4     s = new char[26];
5     for ( int i=0; i<26; i++ ) {
6       s[i] = (char) ('A' + i);
7     }
8
9     return s;
10  }
```

Creating an Array of Character Primitives

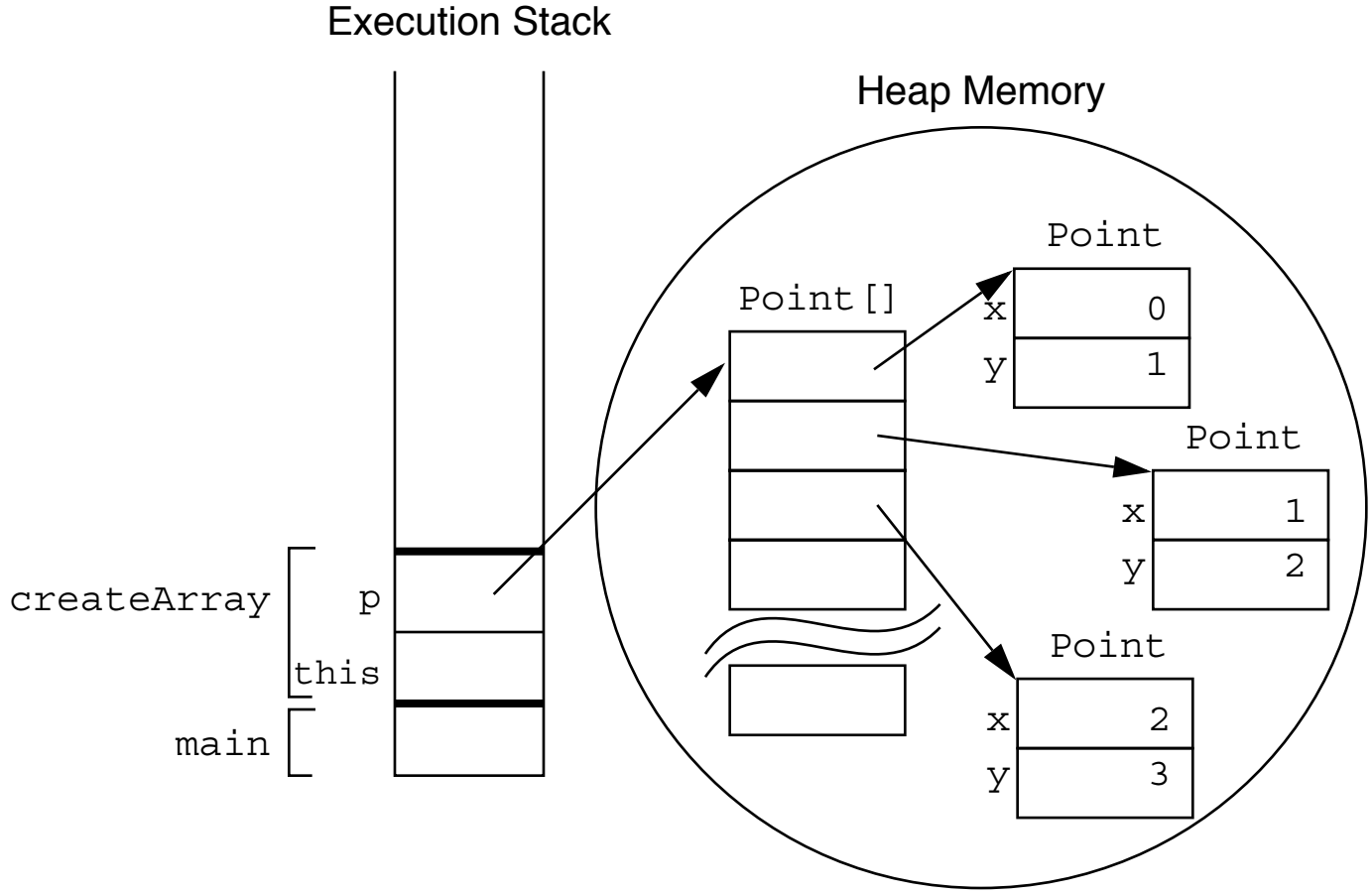


Creating Reference Arrays

Another example, an object array:

```
1   public Point[] createArray() {
2     Point[] p;
3
4     p = new Point[10];
5     for ( int i=0; i<10; i++ ) {
6       p[i] = new Point(i, i+1);
7     }
8
9     return p;
10  }
```

Creating an Array of Character Primitives With `Point` Objects



Initializing Arrays

- Initialize an array element.
- Create an array with initial values.

```
String[] names;
names = new String[3];
names[0] = "Georgianna";
names[1] = "Jen";
names[2] = "Simon";
```

```
String[] names = {
    "Georgianna",
    "Jen",
    "Simon"
};
```

```
MyDate[] dates;
dates = new MyDate[3];
dates[0] = new MyDate(22, 7, 1964);
dates[1] = new MyDate(1, 1, 2000);
dates[2] = new MyDate(22, 12, 1964);
```

```
MyDate[] dates = {
    new MyDate(22, 7, 1964),
    new MyDate(1, 1, 2000),
    new MyDate(22, 12, 1964)
};
```

Multidimensional Arrays

Arrays of arrays:

```
int[][] twoDim = new int[4][];
twoDim[0] = new int[5];
twoDim[1] = new int[5];

int[][] twoDim = new int[][4]; // illegal
```

Multidimensional Arrays

- Non-rectangular arrays of arrays:

```
twoDim[0] = new int[2];
twoDim[1] = new int[4];
twoDim[2] = new int[6];
twoDim[3] = new int[8];
```

- Array of four arrays of five integers each:

```
int[][] twoDim = new int[4][5];
```

Array Bounds

All array subscripts begin at 0:

```
public void printElements(int[] list) {
  for (int i = 0; i < list.length; i++) {
    System.out.println(list[i]);
  }
}
```

Using the Enhanced `for` Loop

Java 2 Platform, Standard Edition (J2SE™) version 5.0 introduced an enhanced `for` loop for iterating over arrays:

```
public void printElements(int[] list) {
  for ( int element : list ) {
    System.out.println(element);
  }
}
```

The `for` loop can be read as *for each* `element` *in* `list` *do*.

Array Resizing

- You cannot resize an array.
- You can use the same reference variable to refer to an entirely new array, such as:

```
int[] myArray = new int[6];
myArray = new int[10];
```

Copying Arrays

The `System.arraycopy()` method to copy arrays is:

```
1   //original array
2   int[] myArray = { 1, 2, 3, 4, 5, 6 };
3
4   // new larger array
5   int[] hold = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 };
6
7   // copy all of the myArray array to the hold
8   // array, starting with the 0th index
9   System.arraycopy(myArray, 0, hold, 0, myArray.length);
```