**GAMEDEVREJECTS**

**C# Programming**

**Orlando Unity3d Development Meetup**

**Date: 22-Jan-22**

**All Copyright GAMEDEVREJCTS©2022**

**Contents**

[**9.0** **Arrays: So, What Is An Array ? ...Why Do We Use Arrays ?** 3](#_Toc93747160)

[**9.1** **Create A New Project** 3](#_Toc93747161)

[**9.2** **Declaring an Array** 4](#_Toc93747162)

[**9.2.1** **Making the Array Declaration Public** 5](#_Toc93747163)

[**Glossary** 7](#_Toc93747164)

[**References** 8](#_Toc93747165)

# **9.0 Arrays: So, What Is An Array ? ...Why Do We Use Arrays ?**

An array is simply **a collection of variables of the same data type.** So, say we had a bunch of values or parameters of the same type. Instead of constantly declaring and defining the variables individually. What we can do is create an array or container of integers of integers of the same type. For example, the dimensions for a cube in a scene. It has a length, height, width, thickness. For simplicity’s sake let’s use define its dimensions in the same units, say metres.

Rather than declaring & defining the parameters as separate variables. We can create an array as a container for the cube's parameter of the same data type. Then in our script specify the individual position of the element in the array to both retrieve or update the value of the variable. Okay, so let's look at the syntax for creating an array. The first thing we do is specify the data type:-

1. **int**
2. **Then add a pair of square brackets, which indicates to the compiler we are adding an array**
3. **Then we now specify a unique name. For example, arr (short for array), which is the array declaration. We get a squiggly line under the name because we haven't used the array yet**
4. **Now to DEFINE the array we place an equal’s sign, followed by two curly braces, which store the values for the array => It could be 0,1,2,3,4,5**

Now to access, all these values. In C# and most other programming languages. We use indexing. We have an index number or position for all the values stored in this array. Now the thing to remember is that the first index number or position always starts at 0. 2nd value is at position or index 1, 3rd value is at position 2 , etc. and so on. So, if we want to output to the console the 1ST value of the array. We would write the following:

# **9.1 Create A New Project**

Create a new Project, Scene then attach a Script. See Chapter 00.0 Create A Scene.

Or create an empty object, then attach a new script, since we are just using the Console Output anyway as follows

(NavigateTo).SCENE -- (IN).HIERARCHY -- / (Right Click).**Create Empty** /

(IN).INSPECTOR – Transform – (3 dots) => / **(Select).Reset** /

(IN).PROJECT -- Assets -- \_SCRIPTS -- (Right Click).Create.C# Script => / **NewBehaviourScript** /

(IN).\_SCRIPTS -- (Select).**NewBehaviourScript**.(Drag and Drop) – HIERARCHY => / (Onto).**GameObject** /

(IN).\_SCRIPTS => / (Select).**NewBehaviourScript**.(Double Left Click) – VSCODE /

Ok inside of the void Start() function, we write:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class NewBehaviourScript : MonoBehaviour

{

    // Start is called before the first frame update

    void Start()

    {

        int[] arr = {0,1,2,3,4,5};

        Debug.Log("1st array element = "+arr[0]);

        Debug.Log("2nd array element = "+arr[1]);

    }

    // Update is called once per frame

    void Update()

    {

    }

}

Graphical user interface, application

Description automatically generated

# **9.2 Declaring an Array**

Now there are several different ways to declare an array. Now supposing we didn't know or calculate the values of the array.

This is how we create a new array of size 5 => Write int arr[];

So next to define the values inside of the array. > We state the index on the left then assign it a value to the right, as follows => arr = new Int[5];

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class NewBehaviourScript : MonoBehaviour

{

    // Start is called before the first frame update

    void Start()

    {

        //int[] arr = {0,1,2,3,4,5};

        int[] arr;

        arr = new int[5];

        arr[0] = 1;

        arr[1] = 2;

        Debug.Log("1st array element = "+arr[0]);

        Debug.Log("2nd array element = "+arr[1]);

    }

    // Update is called once per frame

    void Update()

    {

    }

}

Graphical user interface, application

Description automatically generated

# **9.2.1 Making the Array Declaration Public**

Now another way to define the values in an array is by making the array declaration public. So, if we comment all the code in the void start for now. Then write…**public int[] arr** > Navigate to your inspector > You will see the arr we've created or declared in the Script area of your inspector window. By adding the word public, makes it accessible to everyone, and visible via the inspector. It is what is known as an access modifier.

Great so now we have the capability to define the size of the area. And add values if we expand the toggle triangle. If you manually entered the size. You will automatically expand to the number of elements in the array. So now I can put values directly into the array via the inspector. I don't need to update the script.

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class NewBehaviourScript : MonoBehaviour

{

    public int[] arr;

    // Start is called before the first frame update

    void Start()

    {

        //int[] arr = {0,1,2,3,4,5};

        //int[] arr;

        arr = new int[5];

        arr[0] = 1;

        arr[1] = 2;

        Debug.Log("1st array element = "+arr[0]);

        Debug.Log("2nd array element = "+arr[1]);

    }

    // Update is called once per frame

    void Update()

    {

    }

}

# **Glossary**

# **References**

/End