leren. durven. doen.

SYNTRA WEST

Programmeren 1 C#

C# Programmeur

west-vlaanderen POM

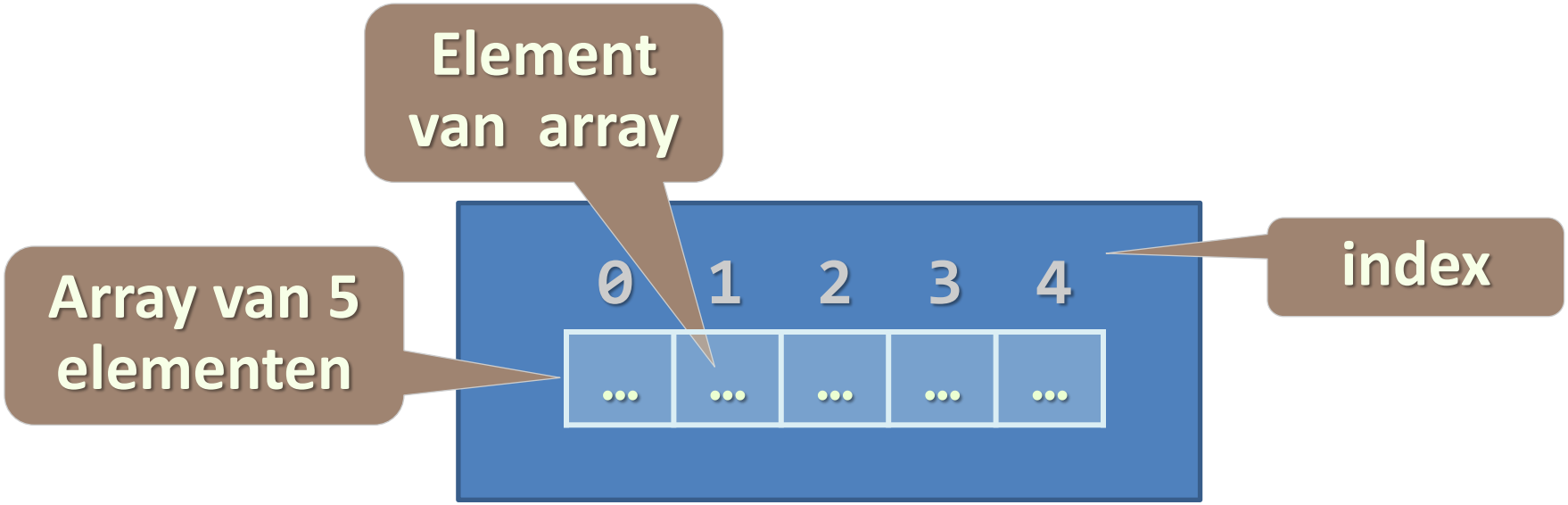
Eéndimensionale Arrays

Inhoud Eéndimensionale Arrays

1. Declaratie en initialisatie van 1-dim Arrays
2. Toegang (lees/schrijf) tot Array Elementen
3. Console Input en Output van Arrays
4. Elementen van Arrays overlopen met `for` en `foreach` lus
5. Kopiëren van Arrays
6. Sorteren van array elementen

Inleiding: Wat zijn Arrays?

- Een array is een reeks van elementen
  - Alle elementen zijn van hetzelfde type
  - Vaste volgorde van de elementen
  - Heeft vaste grootte (Array.Length)

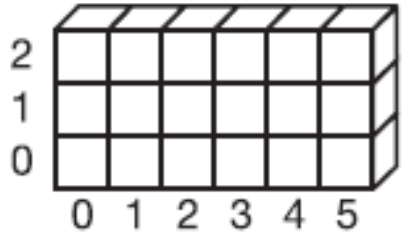


Soorten Arrays

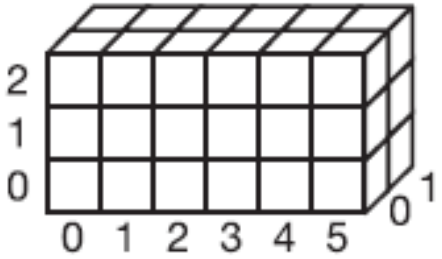
One-Dimensional Arrays

One-Dimensional
`int[5]`

Rectangular Arrays

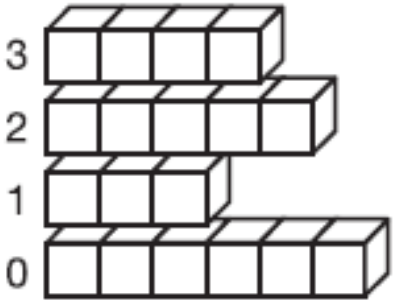


Two-Dimensional
`int[3,6]`



Three-Dimensional
`int[3,6,2]`

Jagged Arrays



Jagged Array
`int[4][]`

Declaratie van ééndimensionale Array

- Definiëren van het type van de elementen
- Vierkante haakjes [ ] betekent "array"
- Voorbeelden:
  - Declaratie van array van integers:

```
int[] myIntArray;
```

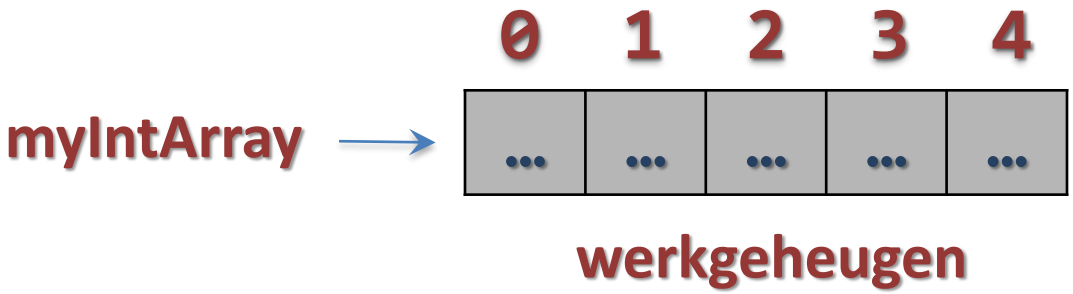
  - Declaratie van array van strings:

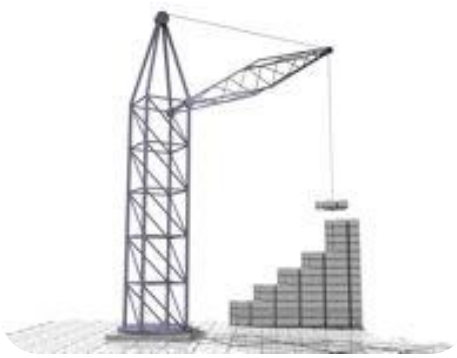
```
string[] myStringArray;
```

Creatie van ééndimensionale array

- Gebruik van operator `new`
  - Specifieer array lengte
- Voorbeeld creatie array van 5 int elementen:

```
myIntArray = new int[5];
```

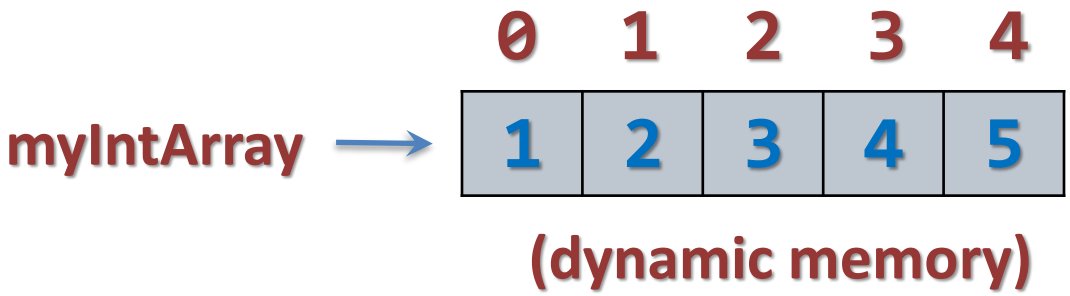




Creatie en Initialisatie van Arrays

- Creatie en initialisatie van Array kan tegelijkertijd:

```
int[] myIntArray = {1, 2, 3, 4, 5};
```



- De new operator mag, maar is niet vereist wanneer initialisatie met {} gebeurt:

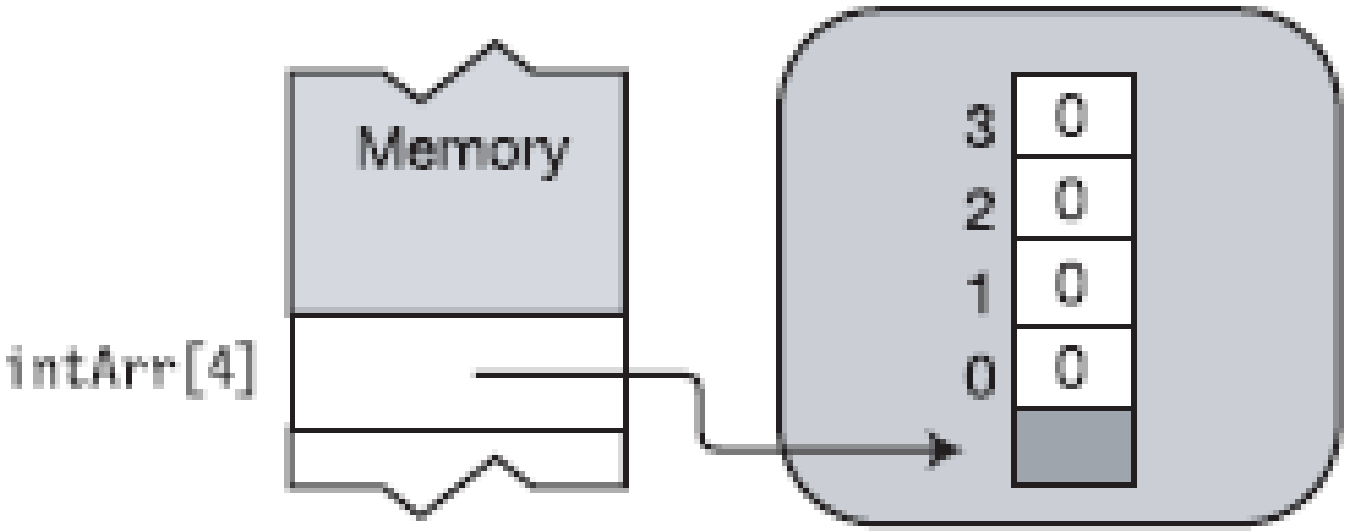
```
int[] myIntArray = new int[] {1, 2, 3, 4, 5};
```

Voorbeeld creatie van Array

- Creatie van array die de dagen van de week bevat

```
string[] DagenVanWeek =
{
    "Maandag",
    "Dinsdag",
    "Woensdag",
    "Donderdag",
    "Vrijdag",
    "Zaterdag",
    "Zondag"
};
```

Geheugen allocatie van 1-dimensionale Arrays



Toegang tot Array Elementen

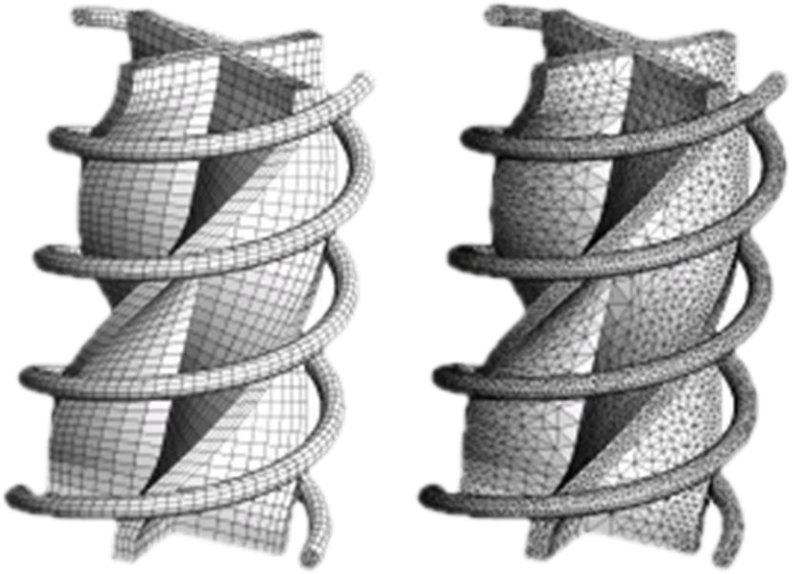
Lezen en schrijven van Array-elementen

Toegang tot Array-elementen

- Toegang tot Array elementen gebeurt door de operator `[ ]` (index-operator)
  - Het eerste element heeft index `0`
  - Het laatste element heeft index `Length-1`

- Array elementen kunnen worden gelezen en gewijzigd door de `[ ]` operator

Aflopen van Array Elementen met `for` en `foreach`



`for` Statement

- Gebruik `for` lus om array af te lopen wanneer:
  - Het nodig is om de index bij te houden
  - Het niet noodzakelijk is om de array sequentieel van het eerste naar het laatste element af te lopen
  - Waarden van elementen moeten worden aangepast
- Gebruik in de loop-body (`array[index]`)

```
int[] arr = new int[] { 1, 2, 3, 4, 5 };
int[] power2Array = new int[5];
for (int index = 0; index < arr.Length; index++)
{
    power2Array[index] = arr[index] * arr[index];
    Console.WriteLine(power2Array[index]);
}
```

Voorbeelden: Arrays aflopen met `for` Loop

- Uitschrijven van array van integers in omgekeerde volgorde:

```
Console.WriteLine("Element in omgekeerde volgorde: ");
for (int i = arr.Length - 1; i >= 0; i--)
{
    Console.Write(arr[i] + " ");
}
// Result: 5 4 3 2 1
```

- Waarde van elk array-element op index zetten:

```
for (int index = 0; index < arr.Length; index++)
{
    arr[index] = index;
    Console.Write(arr[index] + " ");
}
```

Overlopen van Arrays met `foreach`

- Hoe werkt de `foreach` loop?

```
foreach (type variable in array)
```

  - `type` – het data type van het element
  - `Variable` – naam van variable
  - `array` – naam van array

- Nuttig wanneer index niet nodig is
  - Alle elementen worden afzonderlijk overlopen
  - Opgelet: warden van elements kunnen **niet** worden gewijzigd in een foreach lus

Voorbeeld Arrays aflopen met `foreach`

- Uitschrijven van alle elementen van een `string[]` array:

```
string[] steden =
{
    "Antwerpen",
    "Brugge",
    "Kortrijk",
    "Gent"
};
foreach (string stad in steden)
{
    Console.WriteLine(stad);
}
```

Schrijven van Array naar de Console

- Overloop alle elementen van de array
- Schrijf elk element naar de console
- Scheid de elementen met spatie of nieuwe lijn

```
//vervolg code voorgaande slide
Console.WriteLine("De elementen van de array zijn:");
for (int i = 0; i < aantalElementen; i++)
{
    Console.Write(arr[i] + " ");
}
```

Voorbeeld schrijven van string Array naar de Console

```
string[] array = { "één", "twee", "drie" };

// Process all elements of the array
for (int index = 0; index < array.Length; index++)
{
    // Print each element on a separate line
    Console.WriteLine("element[{0}] = {1}",
        index, array[index]);
}
```

Sorteren van Arrays

- Class Array heeft een methode Sort() waarmee de elementen kunnen worden gesorteerd:

```
Array.Sort(array);
```

```
6     public static void Main()
7     {
8         int[] getallen = {5,4,2,3,1};
9         Console.WriteLine("Array met oorspronkelijke volgorde van elementen:");
10        foreach(int getal in getallen)
11        {
12            Console.Write(getal + " ");
13        }
14        Console.WriteLine();
15        Console.WriteLine("Array na sortering van elementen:");
16        Array.Sort(getallen);
17        foreach(int getal in getallen)
18        {
19            Console.Write(getal + " ");
20        }
21    }
```

Samenvatting

- Arrays zijn sequenties (reeksen) van vaste lengte met elementen van **hetzelfde type**
- Array elementen zijn toegankelijk via index
  - Kunnen uitgelezen en gewijzigd worden
- Iteratie over array elementen kan worden gedaan met `for` en `foreach` lussen

Methoden

leren. durven. doen.

SYNTRA WEST

Vragen?

west-vlaanderen POM

Referenties

http://learncs.org/
https://www.w3schools.com/cs

Fundamentals of computer programming with c#
© svetlin nakov & co