Objects and Classes

Using Objects and Classes
Defining Simple Classes

greenwich.edu.vn

UNIVERSITY of GREENWICH

Alliance with FPT Education

UNIVERSITY of GREENWICH
Alliance with FPT Education

Table of Contents

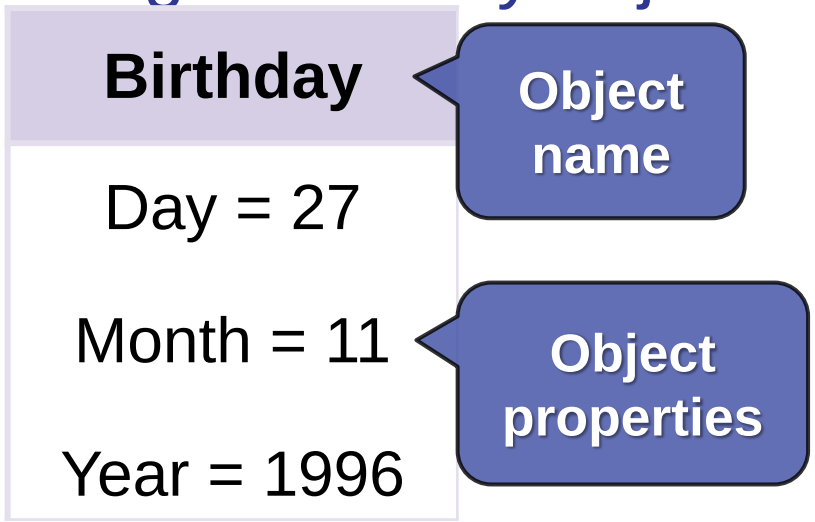
- Objects
- Classes
- Built in Classes
- Defining Simple Classes
  - Properties
  - Methods
  - Constructors

OBJECTS AND CLASSES

What is an Object? What is a Class?

Objects

- An object holds a set of named values
  - E.g. birthday object holds day, month and year
  - Creating a birthday object:



Object name

Object properties

Create a new object of type DateTime

```
var day = new DateTime(
  2017, 6, 19);
Console.WriteLine(day);
```

The new operator creates a new object

```
var birthday = new { Day = 27, Month = 11, Year = 1996 };
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Classes

- In programming, classes provide the structure for objects
  - Act as template for objects of the same type
- Classes define:
  - Data (properties), e.g. Day, Month, Year
  - Actions (behavior), e.g. AddDays(count), Subtract(date)
- One class may have many instances (objects)
  - Sample class: DateTime
  - Sample objects: peterBirthday, mariaBirthday

UNIVERSITY of GREENWICH
Alliance with FPT Education

Objects – Instances of Classes

- Creating the object of a defined class is called instantiation
- The instance is the object itself, which is created runtime
- All instances have common behaviour

```
DateTime date1 = new DateTime(2018, 5, 5);
DateTime date2 = new DateTime(2016, 3, 5);
DateTime date3 = new DateTime(2013, 3, 2);
```

Classes vs. Objects

- Classes provide structure for creating objects

```
class
DateTime
```

Class name

```
Day: int
Month: int
Year: int
```

Class data (properties)

```
AddDays(…)
Subtract(…)
```

Class actions (methods)

- An object is a single instance of a class

```
object
peterBirthday
```

Object name

```
Day = 27
Month = 11
Year = 1996
```

Object data

Problem: Day of Week

- You are given a date in format day-month-year
  - Calculate and print the day of week in English

18-04-2016	⇨	Monday		27-11-1996	⇨	Wednesday

```
string dateAsText = Console.ReadLine();
DateTime date = DateTime.ParseExact(dateAsText, "d-M-yyyy",
  CultureInfo.InvariantCulture);
Console.WriteLine(date.DayOfWeek);
```

ParseExact(…) needs a format string + culture (locale)

USING THE BUILT-IN API CLASSES

Math, Random, BigInteger, ...

UNIVERSITY of GREENWICH
Alliance with FPT Education

Built-in API Classes in .NET

- NET provides thousands of ready-to-use classes
  - Packaged into namespaces like System, System.Text, System.Collections, System.Linq, System.Net, etc.
- Using static .NET class members:

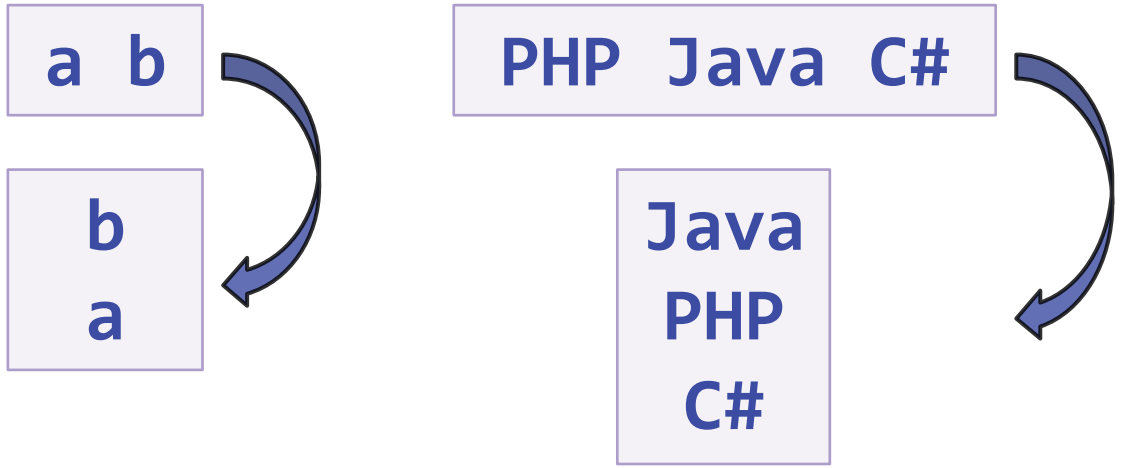
```
DateTime today = DateTime.Now;
double cosine = Math.Cos(Math.PI);
```

- Using non-static .NET classes

```
Random rnd = new Random();
int randomNumber = rnd.Next(1, 99);
```

Problem: Randomize Words

- You are given a list of words
  - Randomize their order and print each word at a separate line



Note: the output is a sample. It should always be different!

UNIVERSITY of GREENWICH
Alliance with FPT Education

Solution: Randomize Words

```
string[] words = Console.ReadLine().Split(' ');
Random rnd = new Random();
for (int pos1 = 0; pos1 < words.Length; pos1++)
{
    int pos2 = rnd.Next(words.Length);
    // TODO: Swap words[pos1] with words[pos2]
}
Console.WriteLine(string.Join(Environment.NewLine, words));
```

Problem: Big Factorial

- Calculate n! (n factorial) for very big n (e.g. 1000)

5 → 120

10 → 3628800

12 → 479001600

50 →
```
30414093201713378043612608166064768844377641568960512000000000000
```

88 →
```
185482642257398439114796845645546284380220968949399346684421580986889562184028199319100141244804501828416633516851200000000000000000000
```

Solution: Big Factorial

Use the .NET API class System.Numerics .BigInteger

```
using System.Numerics;
...
int n = int.Parse(Console.ReadLine());
BigInteger f = 1;
for (int i = 2; i <= n; i++)
    f *= i;
Console.WriteLine(f);
```

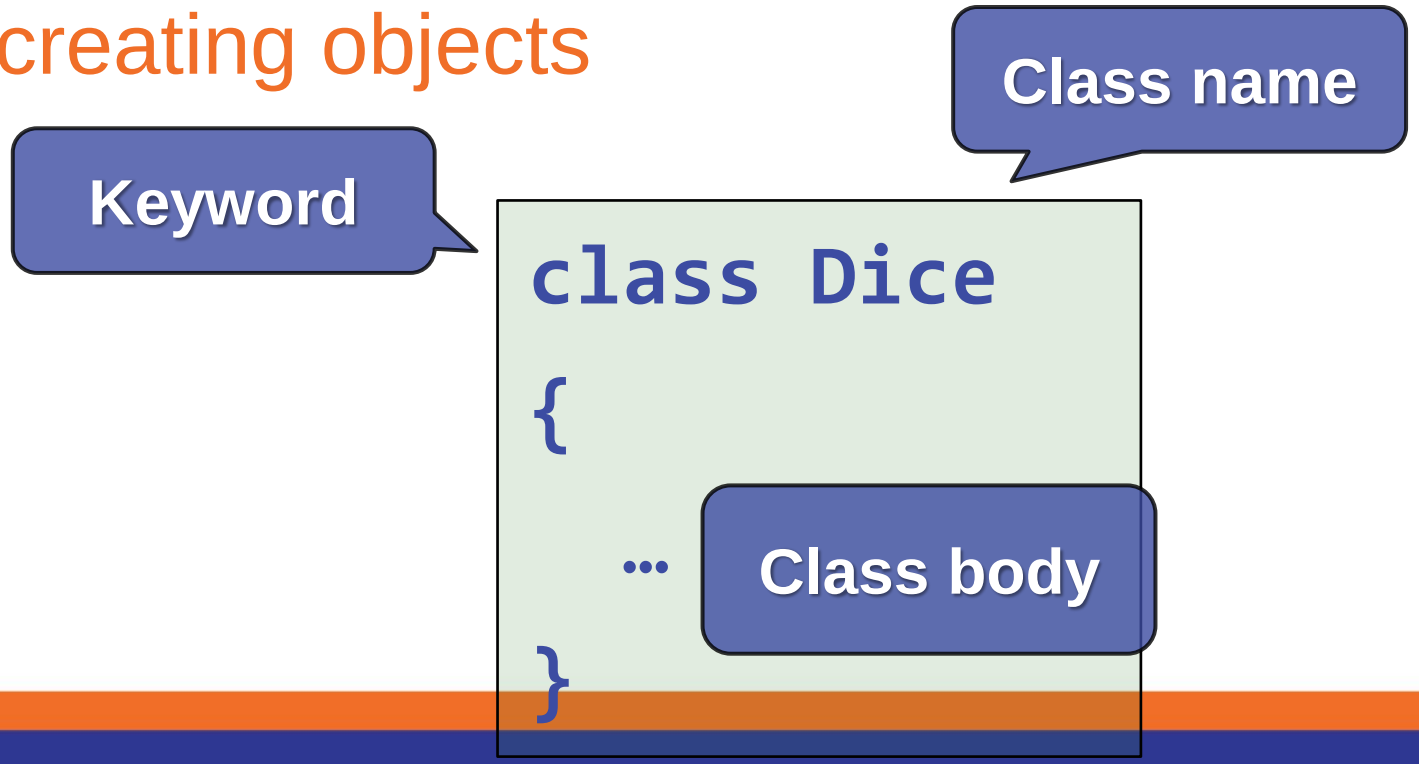
N!

DEFINING CLASSES

Creating Custom Classes

Defining Simple Classes

- Specification of a given type of objects from the real-world
- Classes provide structure for describing and creating objects



Naming Classes

- Use PascalCase naming
- Use descriptive nouns
- Avoid abbreviations (except widely known, e.g. URL, HTTP, etc.)

```
class Dice { … }
class BankAccount { … }
class IntegerCalculator { … }
```

```
class TPMF { … }
class bankaccount { … }
class intcalc { … }
```

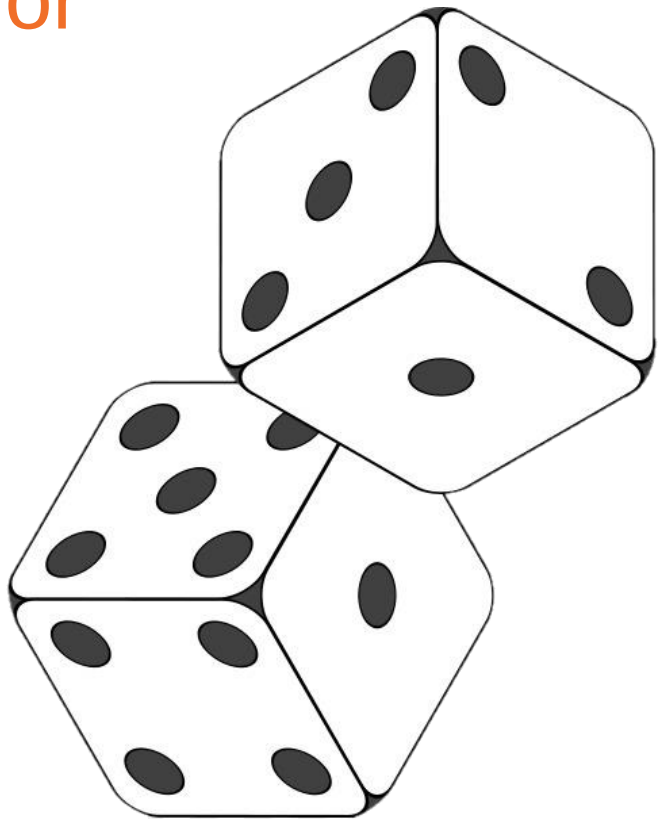
Class Members

- Class is made up of state and behavior
- Properties store state
- Methods describe behaviour

```
class Dice
{
  public int Sides { get; set; }
  public string Type { get; set; }
  public void Roll() { }
}
```

Properties

Method



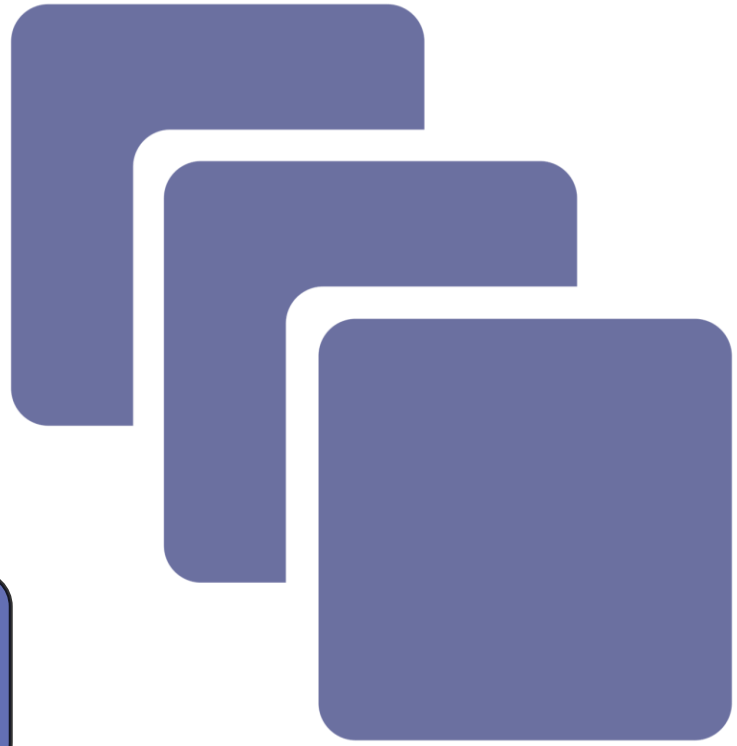
Creating an Object

- A class can have many instances (objects)

```
class Program
{
  public static void Main()
  {
    Dice diceD6 = new Dice();
    Dice diceD8 = new Dice();
  }
}
```

Variable stores a reference

Use the new keyword



UNIVERSITY of GREENWICH
Alliance with FPT Education

Properties

- Describe the characteristics of a given class

```
class Student
{
  public string FirstName { get; set; }
  public string LastName { get; set; }
  public int Age { get; set; }
}
```

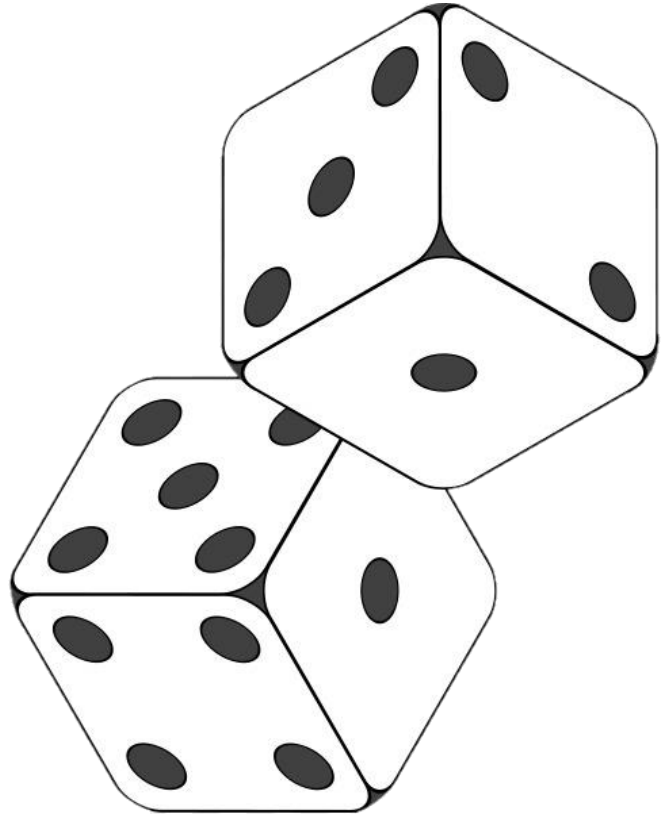
The getter provides access to the field

The setter provides field change

Methods

- Store executable code (algorithm)

```
class Dice
{
  public int Sides { get; set; }
  public int Roll()
  {
    Random rnd = new Random();
    return rnd.Next(1, Sides + 1);
  }
}
```



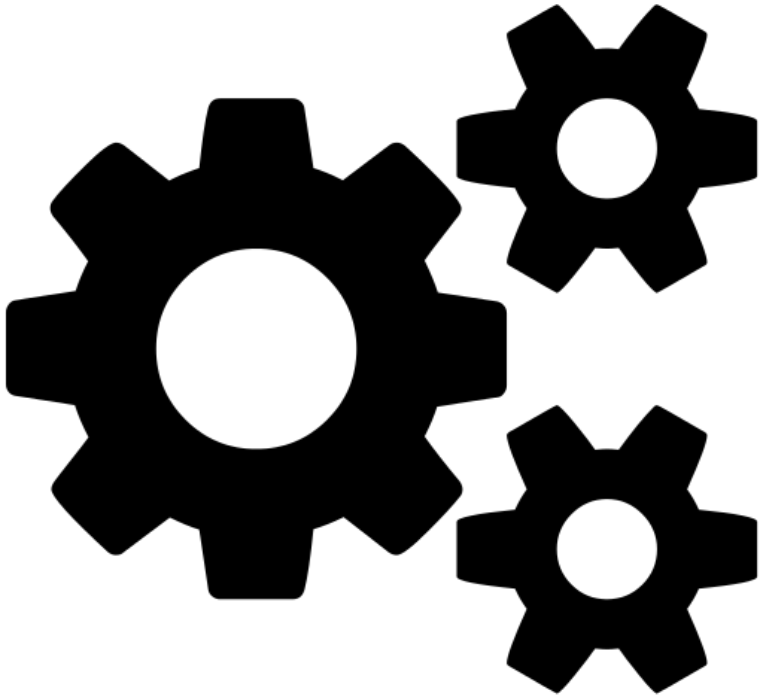
Constructors

- Special methods, executed during object creation

```
class Dice
{
    public int Sides { get; set; }
    public Dice()
    {
        this.Sides = 6;
    }
}
```

Constructor name is the same as the name of the class

Overloading default constructor



UNIVERSITY of GREENWICH
Alliance with FPT Education

Constructors (2)

- You can have multiple constructors in the same class

```
class Dice
{
  public Dice() { }
  public Dice(int sides)
  {
    this.Sides = sides;
  }
  p int Sides { get; set; }
}
```

```
class StartUp
{
 public static void Main()
 {
    Dice dice1 = new Dice();
    Dice dice2 = new Dice(7);
  }
}
```

UNIVERSITY of GREENWICH
Alliance with FPT Education

Summary

- Objects
  - holds a set of named values
  - Instance of a class
- Classes define templates for object
  - Methods
  - Constructors
  - Properties