



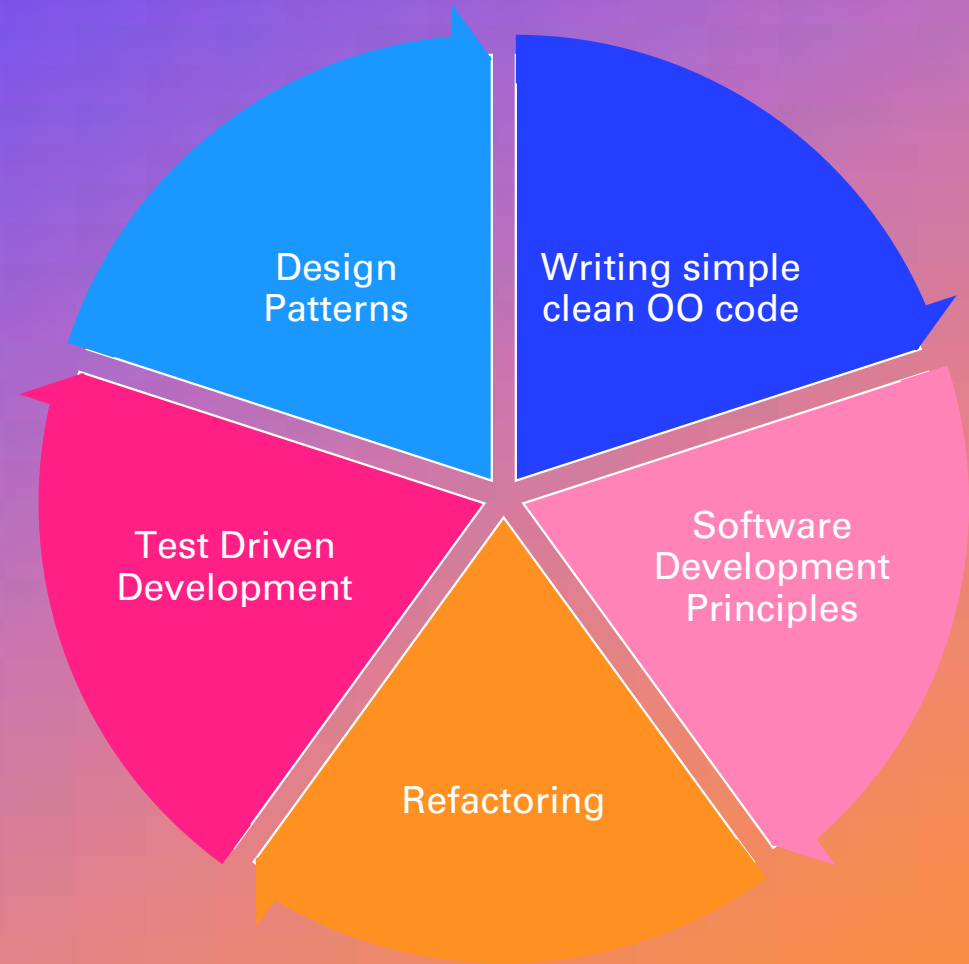
+

• CO453 APPLICATION PROGRAMMING

○

ASP.NET Core C# Applications developed using Visual Studio 2019/2022

Module Focus



Learning Outcomes



Analyse simple requirements in a structured manner



Design, document, implement and test reliable, maintainable programs as solutions to simple problems



Use structured techniques of design and implementation and good documentation practice



Use software development tools

CW1: Assessment – 5 C# Applications

CW1: 5 Applications

C# Console Apps
C# Web Apps
C# Graphic Apps

- Part in Class, Part Independent Study
- Code **reviewed** by tutor in class and feedback given
- Like a real job failure to complete the **features** on time lowers your Grade.
- Each application **feature** is marked after the App deadline for **Code Quality, Testing and Documentation**.
- **Code Quality** is judged by 34 professional best practice standards

[Clean Code Link](#)

Clean Code Quality Issues

Code Comments

Code Names

Code Layout

Code Structure

Agilealliance.org

Key Agile Concepts

Learn about Agile terminology by visiting our Agile Glossary for more terms.

Acceptance Testing
ATDD
Backlog
Backlog Grooming
BDD
Continuous Deployment
Continuous Integration

Definition of Done
Definition of Ready
Exploratory Testing
Given When Then
Incremental Development
Iterative Development
Kanban

Kanban Board
Pair Programming
Planning Poker
Product Owner
Scrum
Scrum Master
Story Mapping

TDD
Timebox
Ubiquitous Language
Unit Testing
Usability Testing
User Stories
3 C's

[Read the Agile Manifesto at AgileManifesto.org](https://agilemanifesto.org)

 **VISIT THE AGILE MANIFESTO WEBSITE**

Agile's 12 Principles

1

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

1-3 Weeks

7

Working software is the primary measure of progress.

3

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

9

Continuous attention to technical excellence and good design enhances agility.

Feature Driven
Development

10

Simplicity—the art of maximizing the amount of work not done—is essential.

Session Summary

BNU 2021

To create a simple console application using Visual Studio

To introduce the basic Object-Oriented Concepts

To Introduce input, output and arithmetic using C#

To develop and document a program top-down.

To share and document the program using Git & GitHub

Computer Programming



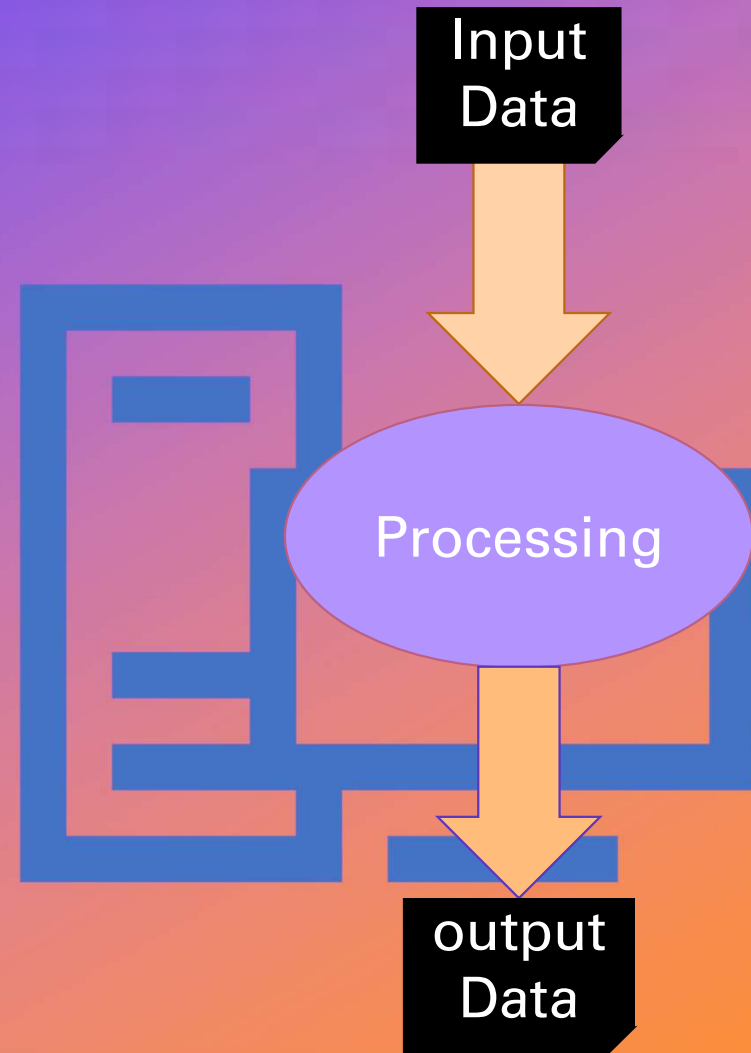
What is a Computer?
(in 4 words)



What is a Program?

WHAT IS A COMPUTER

An electronic information processing machine



What is a Program?



A coded set of instructions to complete a specific data processing task.

e.g. Convert a given distance in miles to feet.



Prompt the user to enter a distance in miles

PROGRAM DESIGN (UML: ACTIVITY DIAGRAM)

Possible alternative names?
Enter Miles, Display Feet

Classes & Objects

C# programs are divided into classes

Classes are the blueprints for creating the objects

Classes contain Data
(fields, variables or attributes)

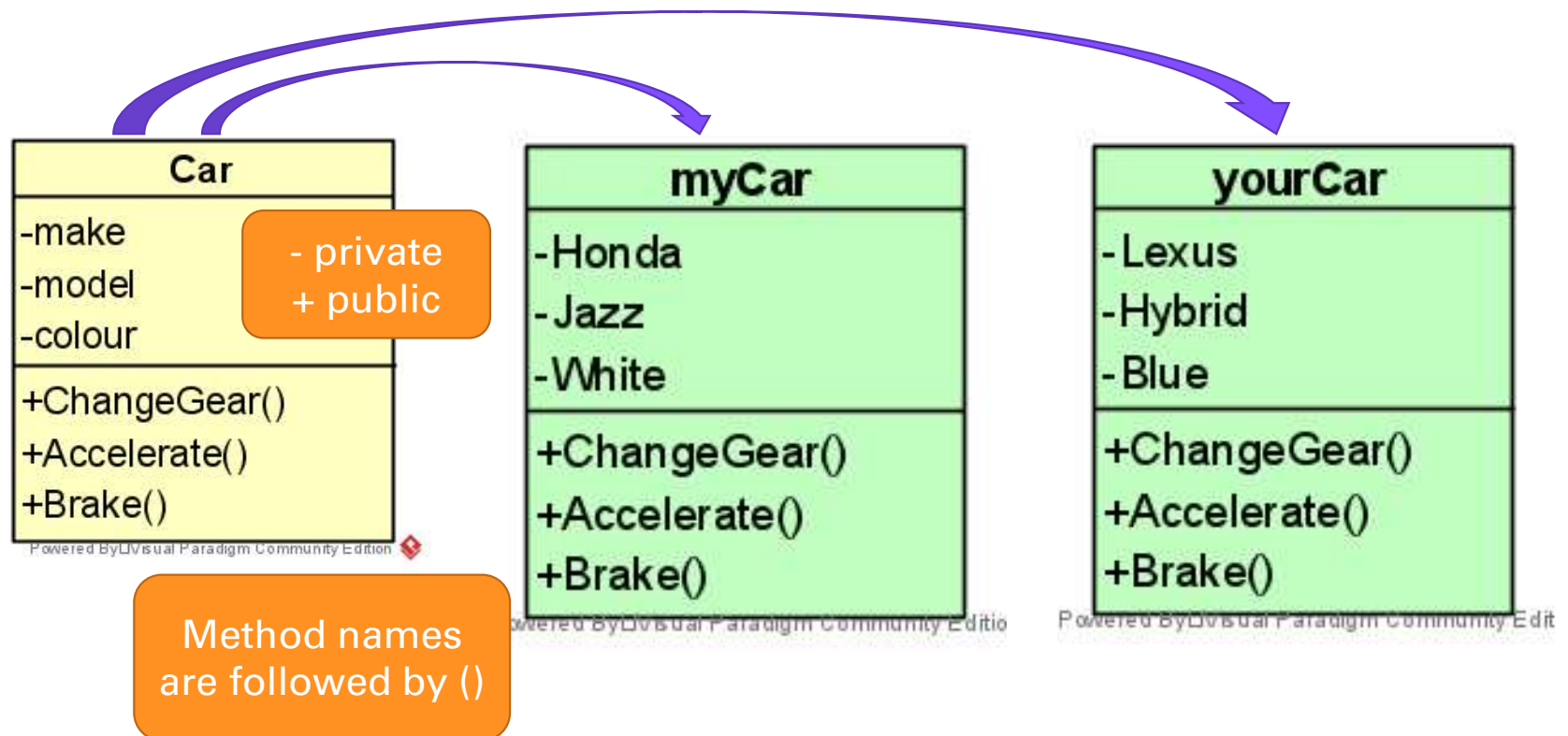
Classes contain Methods or
Functions for processing

Alternative Names

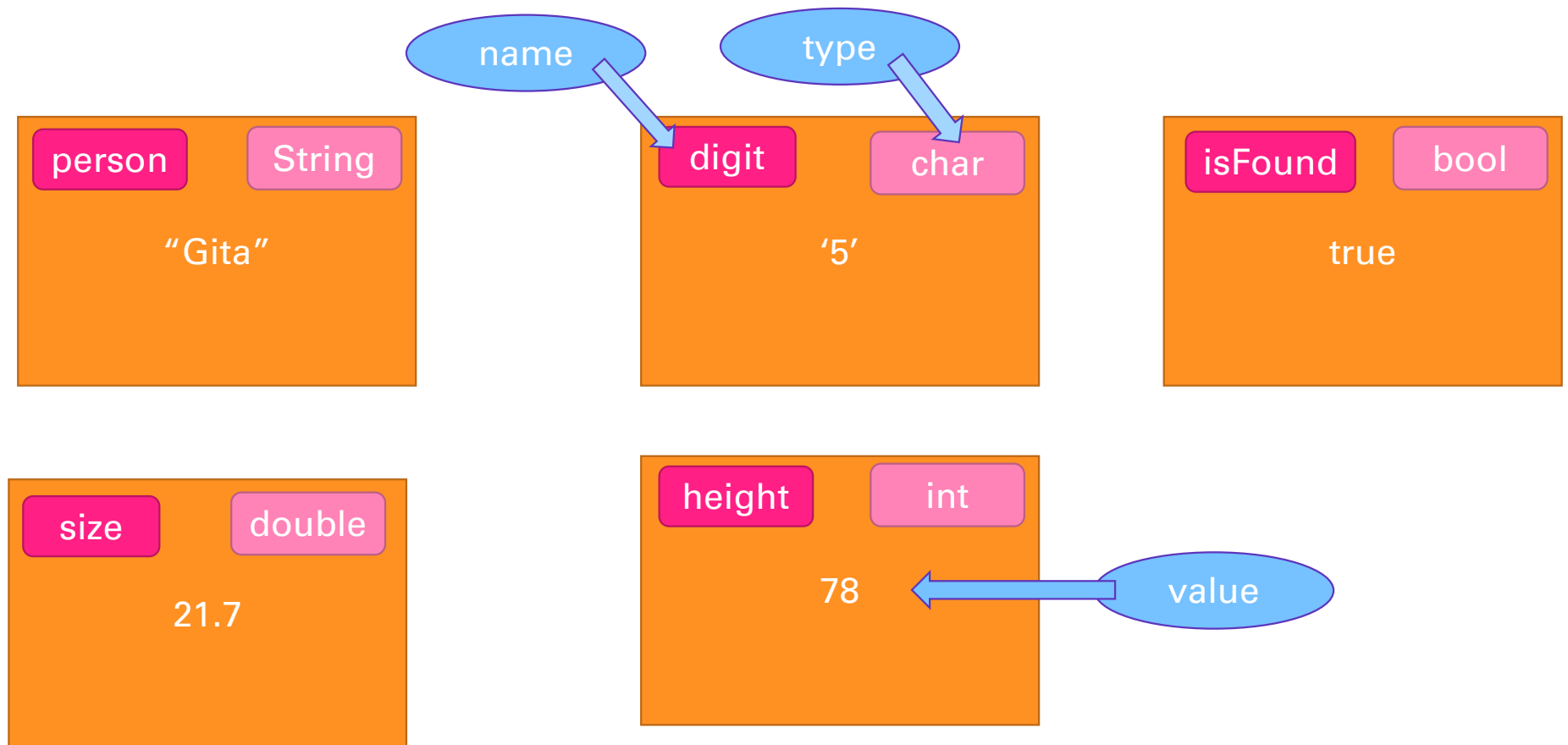
- Information
- Data
- **Attributes**
- Variables
- Properties
- Fields

- Processes
- **Methods**
- Operations
- Procedures
- Functions
- Actions

UML: Classes & Objects



Variables/Attributes: Data Types



C# Language

```
/// <summary>
/// This is a comment describing the
/// main purpose of the class
/// </summary>
```

0 references

```
public class Car
{
```

```
    // Attributes
```

```
    private string make;
```

```
    private string model;
```

```
    private string colour;
```

```
    // Methods
```

Encapsulation:
private attributes
Public methods

C# - Pascal Case
Java - Camel Case

```
// Methods
```

0 references

```
public void ChangeGear()
{
}
```

0 references

```
public void Accelerate()
{
}
```

0 references

```
public void Brake()
{
} // end of method
```

```
} // end of class
```

Microsoft Naming Conventions 2008



General
Naming
Conventions

Names of
Classes

Names of
Properties &
Methods

C# Naming Convention

BNU 2021

C# is case sensitive

All names must start with a letter

Method names start with a capital letter

Class names start with a capital (Pascal)

Names can contain letters, digits and underscores

Naming Continued



Object names start with lower case



Attribute (private variable) names start with lower case letter (camel)



Start with or contain **Nouns**

Classes
Objects
Attributes



Methods start with **Verbs**
(or imply verbs)

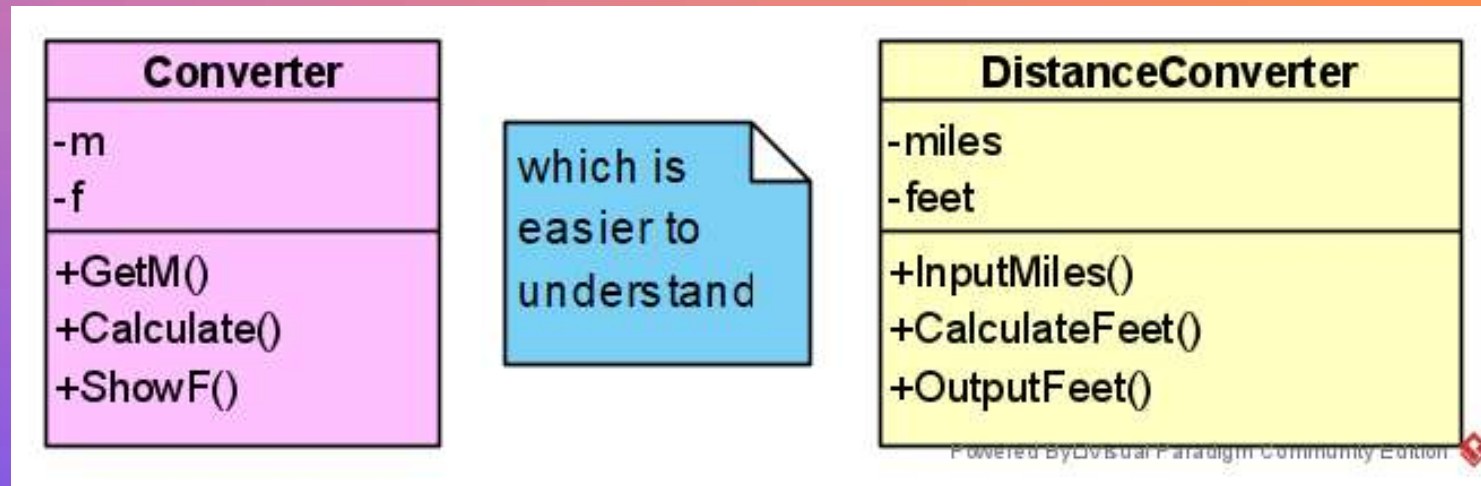
Valid Names? Good Names?

Class Names

1. Car
2. hondaCar
3. 007_Car
4. Honda Car
5. HondaCar
6. C1
7. Car1

Method Names

8. accelerate()
9. Accelerate()
10. Change Gear()
11. ChangeGear()
12. Change_Gear()
13. C2()
14. 2C()



UML: CLASS DIAGRAM

IDE: Visual Studio 2022

BNU 2021

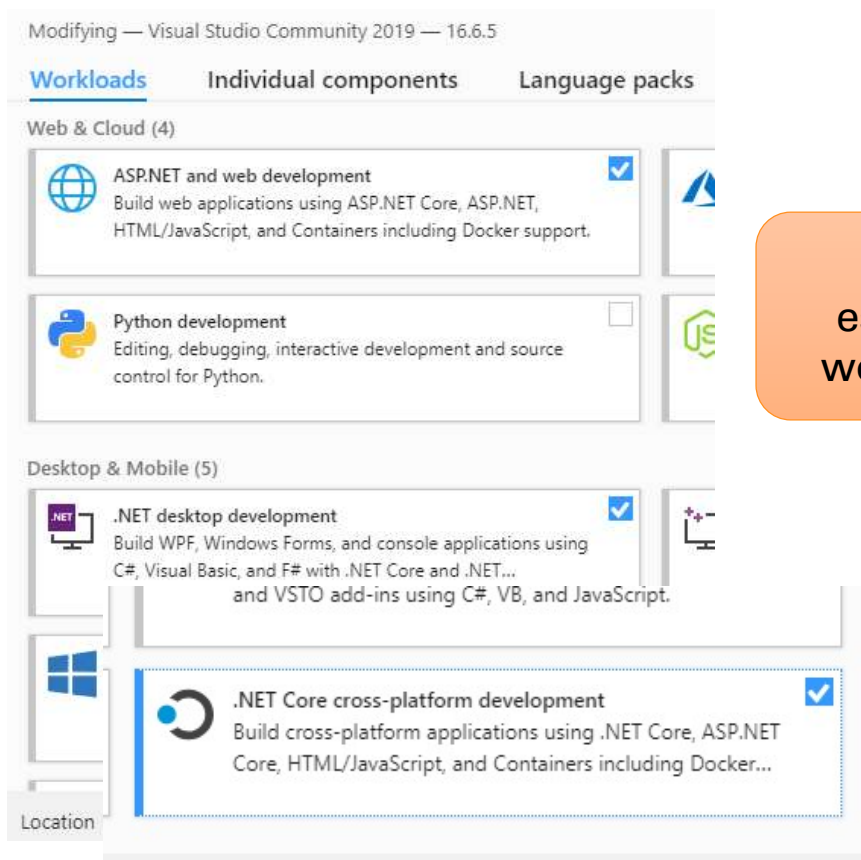
Microsoft's Integrated
Development Environment

Develop, Analyse, Debug,
Test, Collaborate, Deploy

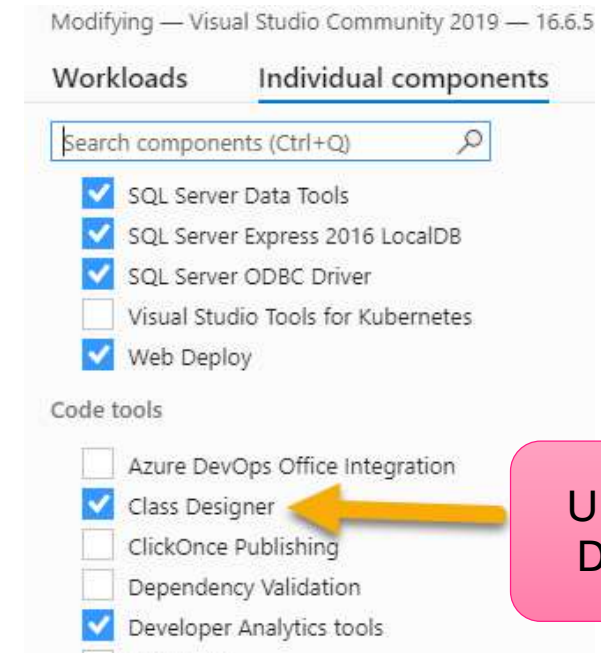
Community, Professional,
Enterprise Editions

36 Different programming
languages

Installing Visual Studio

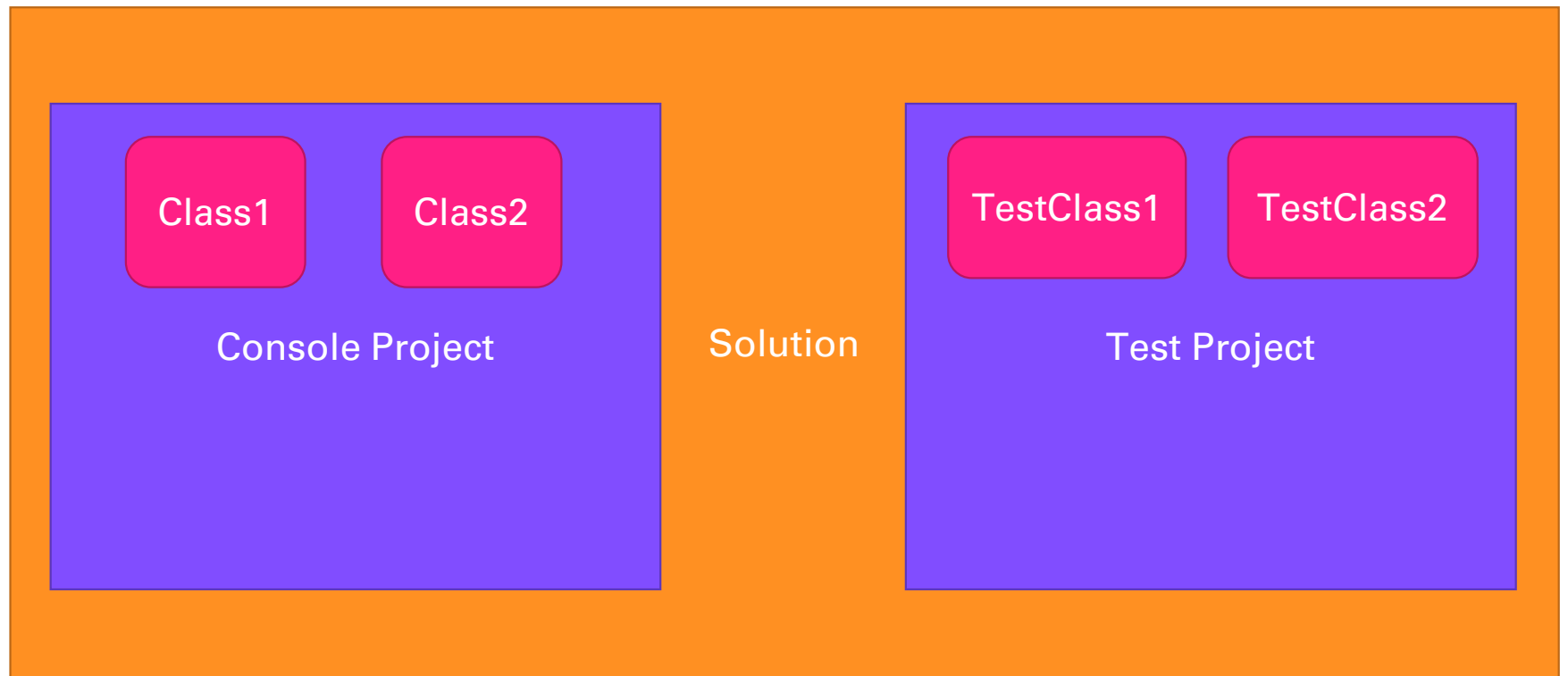


Three
essential
workloads



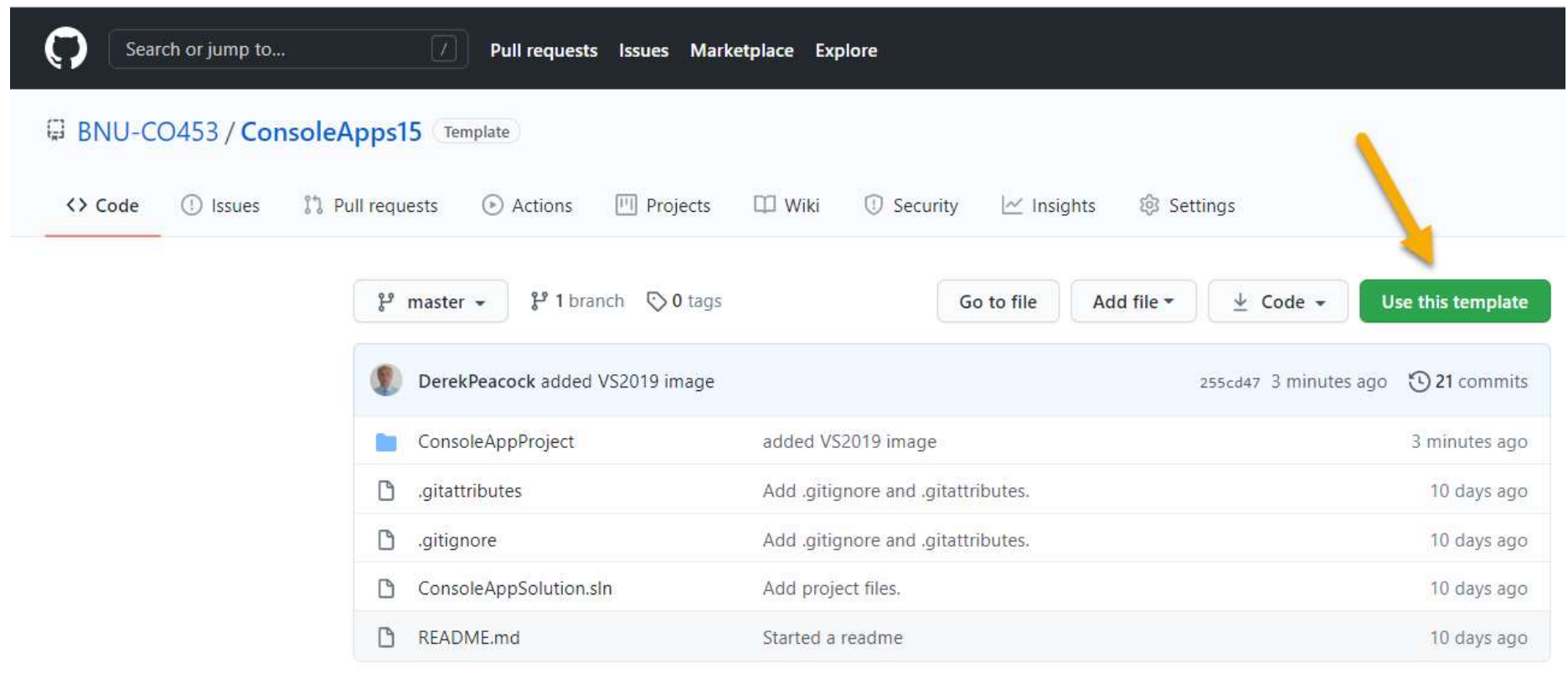
UML Class
Diagrams

Create Visual Studio Project



Find the Template

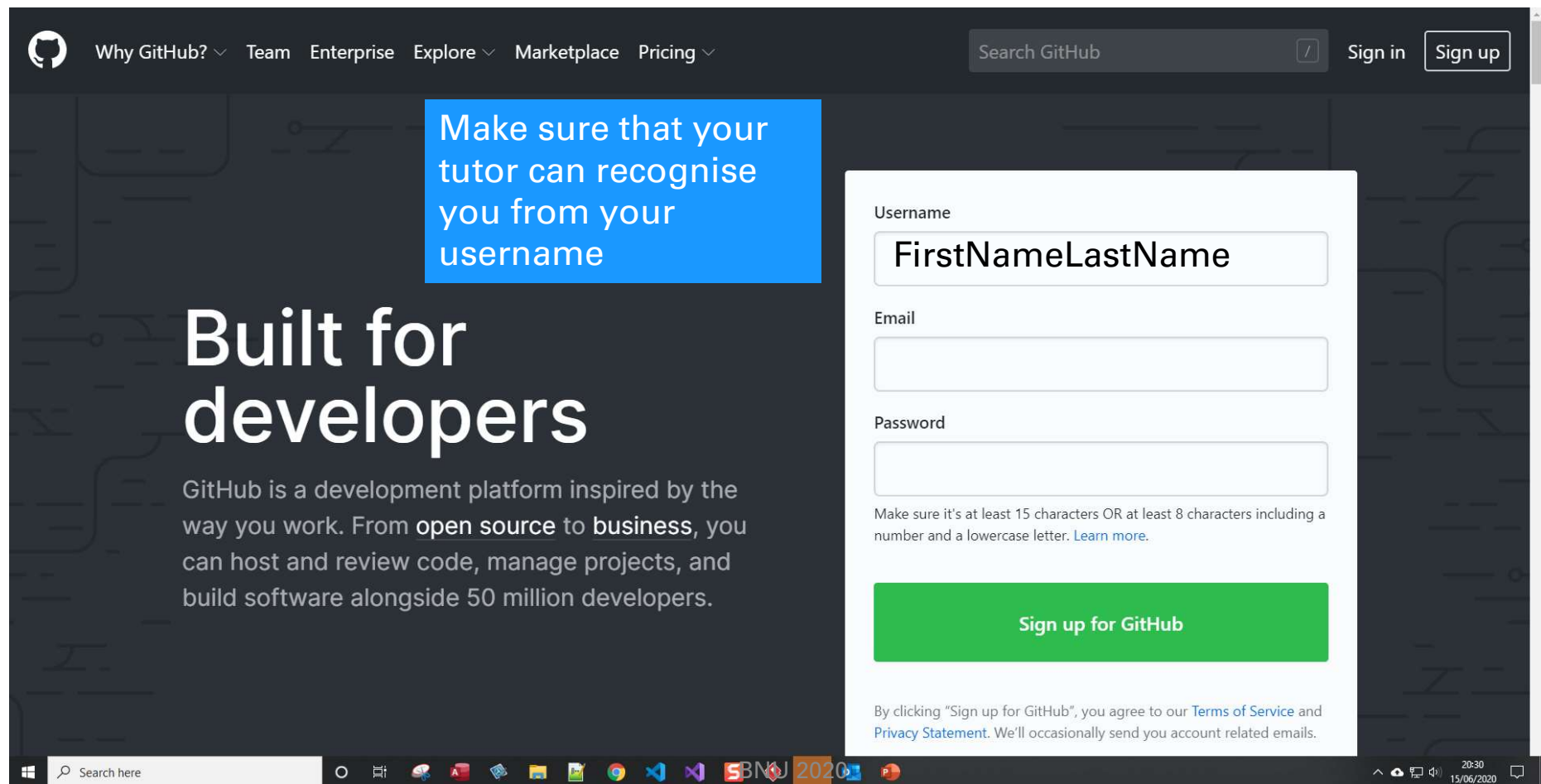
<https://github.com/BNU-CO453/ConsoleApps15>



The screenshot shows the GitHub repository page for `BNU-CO453 / ConsoleApps15`. The repository is marked as a `Template`. The navigation bar includes links for `Code`, `Issues`, `Pull requests`, `Actions`, `Projects`, `Wiki`, `Security`, `Insights`, and `Settings`. Below the navigation bar, there are buttons for `Go to file`, `Add file`, `Code`, and `Use this template`. A yellow arrow points to the `Use this template` button. Below these buttons, a commit history table is displayed, showing the commit `255cd47` by `DerekPeacock` 3 minutes ago, with 21 commits in total. The table lists the files added in this commit: `ConsoleAppProject`, `.gitattributes`, `.gitignore`, `ConsoleAppSolution.sln`, and `README.md`.

File	Commit Message	Time
ConsoleAppProject	added VS2019 image	3 minutes ago
.gitattributes	Add .gitignore and .gitattributes.	10 days ago
.gitignore	Add .gitignore and .gitattributes.	10 days ago
ConsoleAppSolution.sln	Add project files.	10 days ago
README.md	Started a readme	10 days ago

Code Sharing with GitHub



The image shows the GitHub homepage with a dark theme. The navigation bar at the top includes links for 'Why GitHub?', 'Team', 'Enterprise', 'Explore', 'Marketplace', and 'Pricing'. A search bar and 'Sign in'/'Sign up' buttons are on the right. The main heading 'Built for developers' is prominent, followed by a description of GitHub as a development platform. A blue callout box highlights the importance of a recognizable username. A white sign-up form is overlaid on the right, containing fields for Username, Email, and Password, along with a green 'Sign up for GitHub' button and a disclaimer at the bottom.

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search GitHub Sign in Sign up

Make sure that your tutor can recognise you from your username

Built for developers

GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside 50 million developers.

Username

FirstNameLastName

Email

Password

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.

Windows taskbar: Search here, task icons, SBNJ 2020, 20:30 15/06/2020

USING THE TEMPLATE

Create a new repository from ConsoleApps15

The new repository will start with the same files and folders as [BNU-CO453/ConsoleApps15](#).

Owner *



DerekPeacock ▾

Repository name *

ConsoleApps15 ✓

Great repository names are short and descriptive. [ConsoleApps15](#) is available. Inspiration? How about [psychic-goggles](#)?

Description (optional)



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.

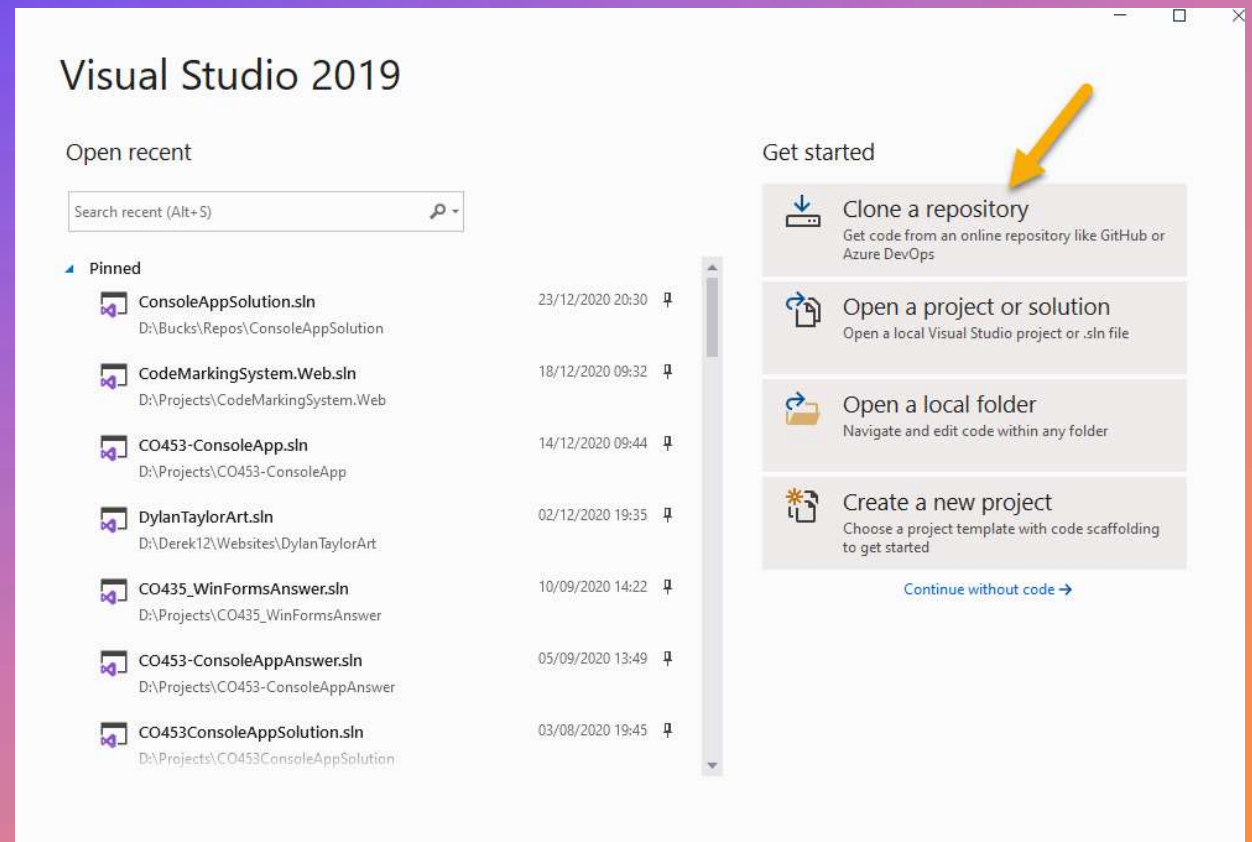


Include all branches

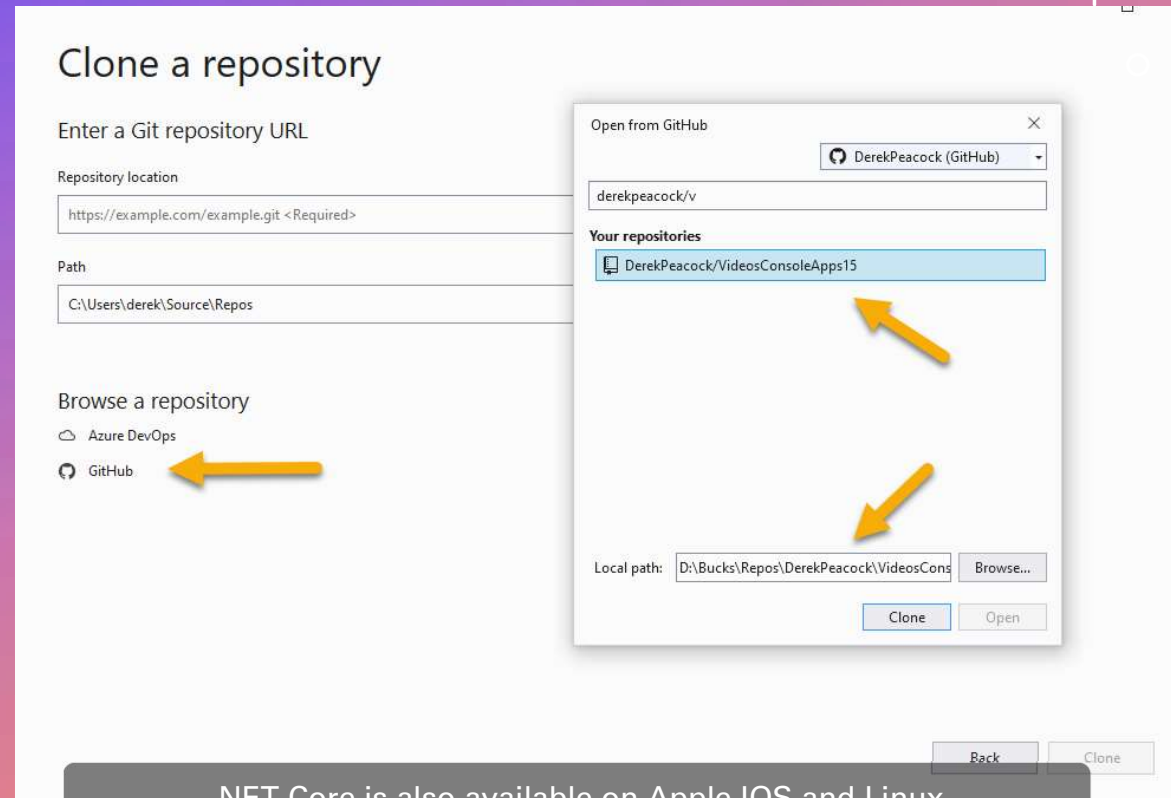
Copy all branches from [BNU-CO453/ConsoleApps15](#) and not just master.

Create repository from template

VS Startup Window



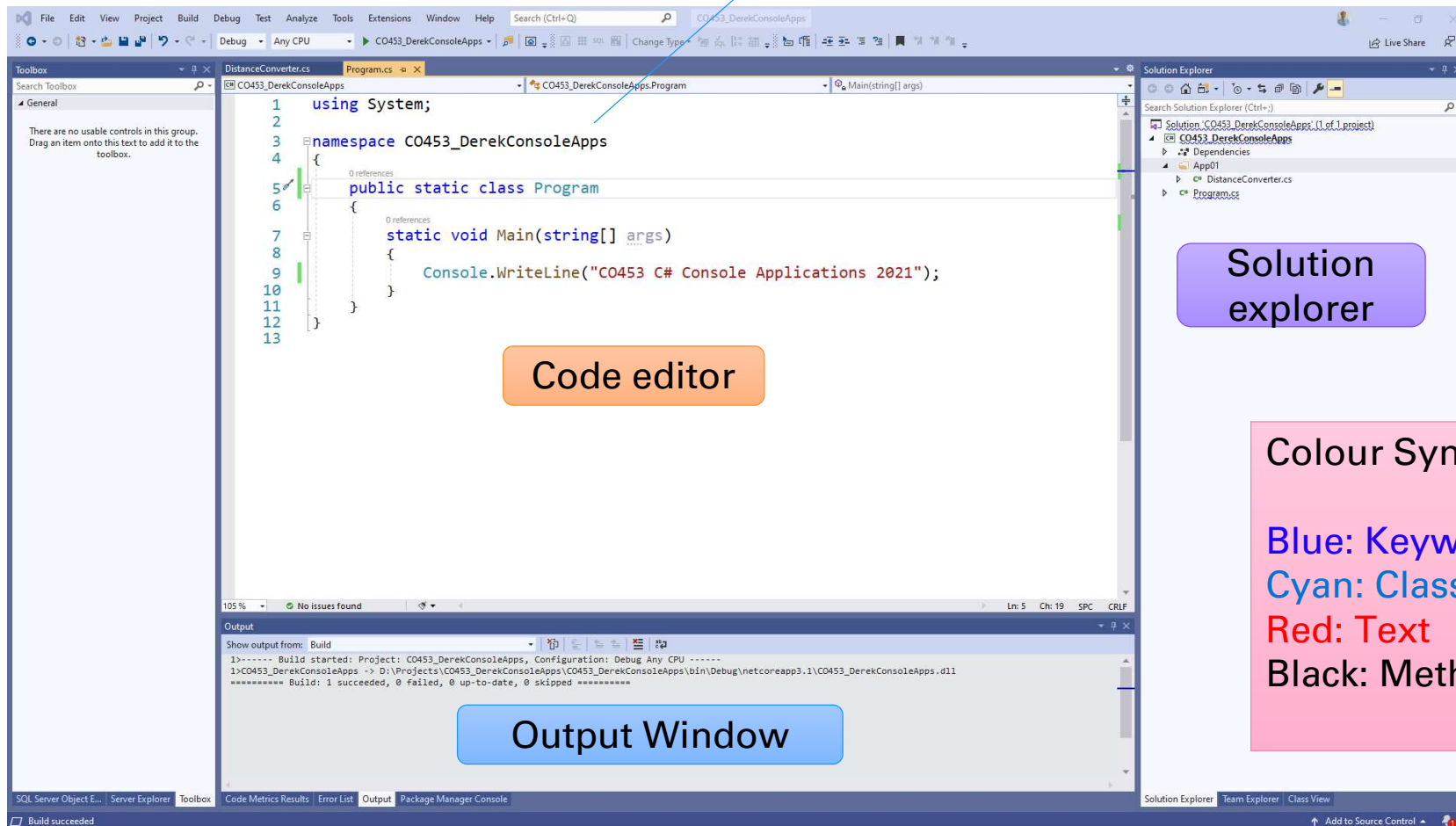
CLONING A REPOSITOTY



.NET Core is also available on Apple IOS and Linux

VS User Interface

Project name



Solution explorer

Code editor

Output Window

Colour Syntax Highlighting

Blue: Keywords

Cyan: Class names

Red: Text

Black: Method names

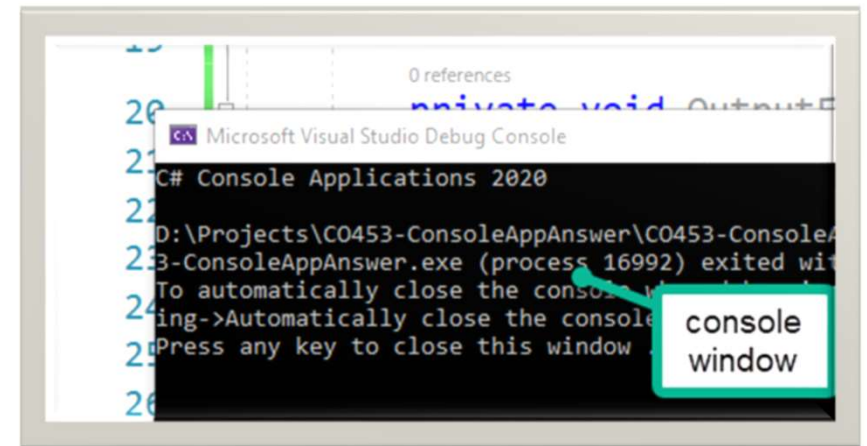
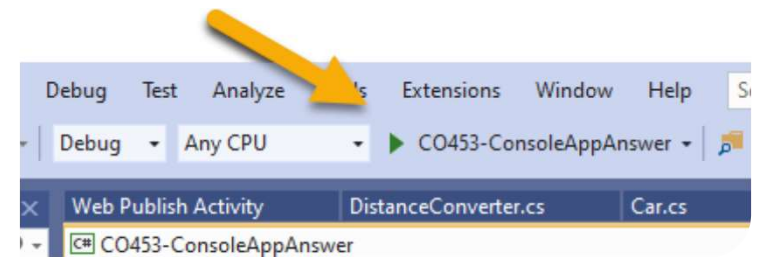
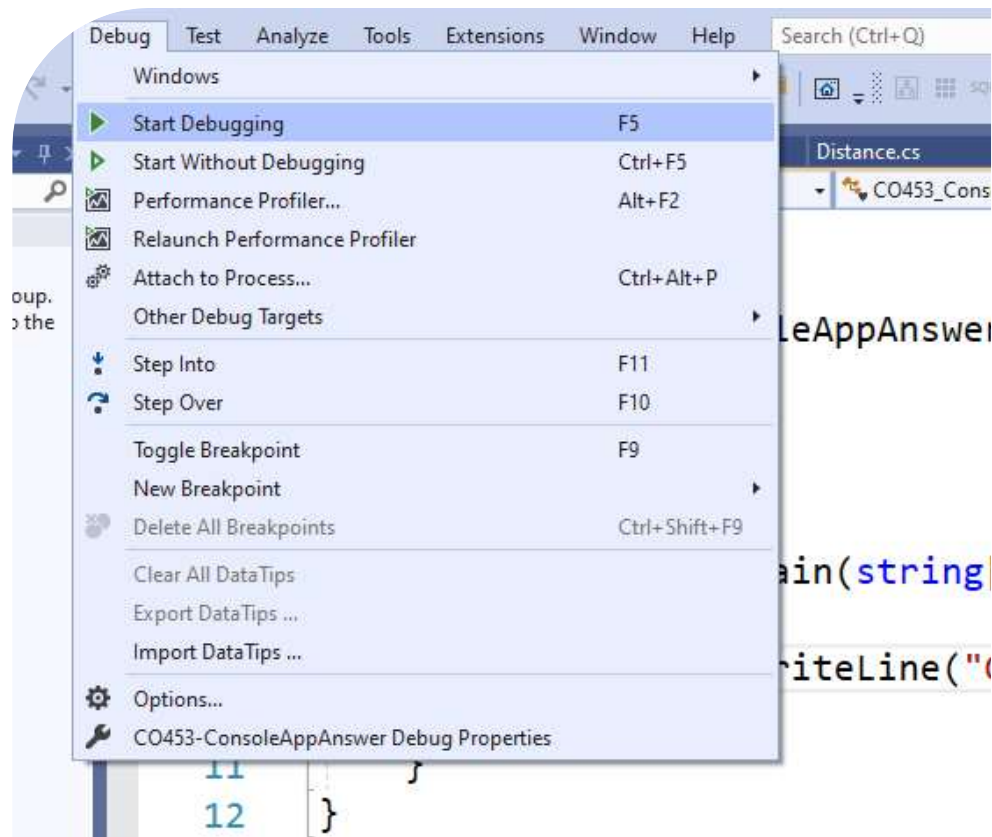
Program Class

Class contains a **Main()** method which is where the program will start

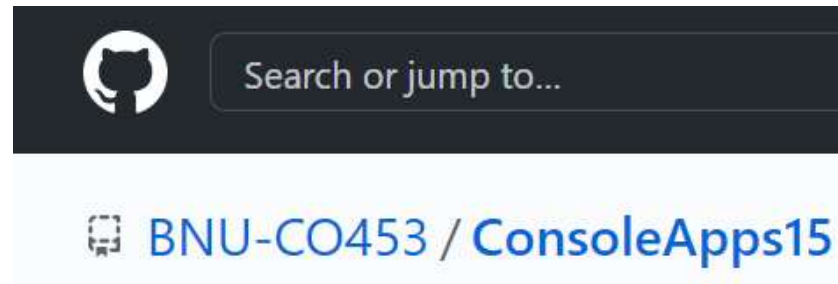
Console is a library class (using System) which contains methods

WriteLine() is a method that writes text to the console window

Start Debugging (run the program)



Using the Wiki



1

Edit Your
Readme.md

2

Add your
Home wiki
page

3

Add the
App01 wiki
page

4

Use the Wiki
information

Home Page

Wiki Home Page: Andrei [REDACTED] 7459

[BSc Data Science](#)

I am studying Data Science at Buckinghamshire new University! I 'fell in love' with programming in college while learning the basic language 'Pascal' and later on doing a course in Oracle SQL. Ever since I am using some of my free time to learn the basics of other programming languages like Python and Java, but not having a mentor and not knowing other people with similar interests had a big impact on my progression. Being enrolled at university has changed that!

CO452 Lesson: Wed 10:30 Nick (online lesson Friday 10:30)

Course Work Assessment Part A

1. [App01: LabClass](#)
2. [App02: Ticket Machine](#)
3. [App03: Student Grades](#)
4. [App04: Stock Management](#)

Course Work Assessment Part B

5. [App05: Product Stock Application](#)
6. [App06: Zuul Game Console Application](#)

▼ Pages **7**

[Home](#)

[app01 LabClass](#)

[app02 TicketMachine](#)

[App03 Student Grades](#)

[App04 Product Stock Control](#)

[App05: Product Stock Application](#)

[App06: Zuul Game Console Application](#)

Clone this wiki locally

<https://github.com/andreicruceru>



App01: User Requirements

App01: Distance Converter

Description

This App allows the user to convert distances measured in one unit of distance into another unit of distance, for example it will convert a distance measured in miles into the same distance measured in feet.

Features Stage 1

The application should:-

1. Output a heading with the name of the application and the name of the programmer.
2. Input any valid distance measured in miles
3. Convert that distance into feet
4. Output the equivalent distance in feet The input and output should accept numbers that include the decimal point, and this app can use the fact that 1.0 miles is exactly 5280 feet.

► Pages 15

CO453 Module Index

[Module Scheme](#)
[Git and GitHub](#)
[Visual Studio 2019](#)
[Coding Guide](#)
[Online Lessons](#)
[Wiki Documentation](#)
[Example Home Page](#)

Repo Part A

[App01: Distance Converter](#)
[App02: BMI Calculator](#)
[App03: Student Marks](#)
[App04: Social Network](#)
[App05: RPS Game](#)


```

public class DistanceConverter
{
    private double miles;
    private double feet;

    0 references
    public void Run()
    {
    }

    0 references
    private void InputMiles()
    {
    }

    0 references
    private void CalculateFeet()
    {
    }

    0 references
    private void OutputFeet()
    {
    }
}

```

App01: DistanceConverter

- Add the attributes and methods
- **Notice**
- Semicolons;
- the indentation
- the line spacing
- the names and use of case
- public method is black
- private methods are grey

Use Block Comments

```
namespace C0453_DerekConsoleApps.App01
{
    /// <summary>
    /// This class offers methods for converting a given
    /// distance measured in miles to the equivalent
    /// distance measured in feet
    /// </summary>
    /// <author>
    /// Derek Peacock version 0.1
    /// </author>
    0 references
    public class DistanceConverter
    {
    }
}
```

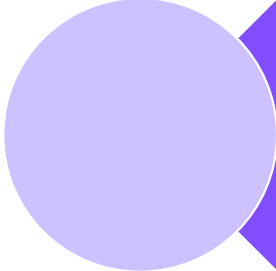
- Block comments are multiline
- Generates XML API documentation
- Every class
- Every method

API documentation is for other programmers to read

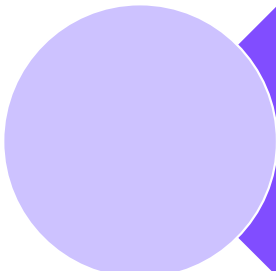
Block and Single Line Comments

```
public class DistanceConverter
{
    // Distance measured in miles
    private double miles;
    // Distance measured in feet
    private double feet;

    /// <summary>
    /// This method will input the distance measured in miles
    /// calculate the same distance in feet, and output the
    /// distance in feet.
    /// </summary>
    0 references
    public void Run()
    {
        InputMiles();
        CalculateFeet();
        OutputFeet();
    }
}
```



Only add single line comments if it would help understanding



Always add block comments to summarise what the method does

+

Top Down
Development

0 references

```
public void Run()  
{  
    InputMiles();  
    CalculateFeet();  
    OutputFeet();  
}
```

Executing this program
will not produce any
output yet!

Using Methods

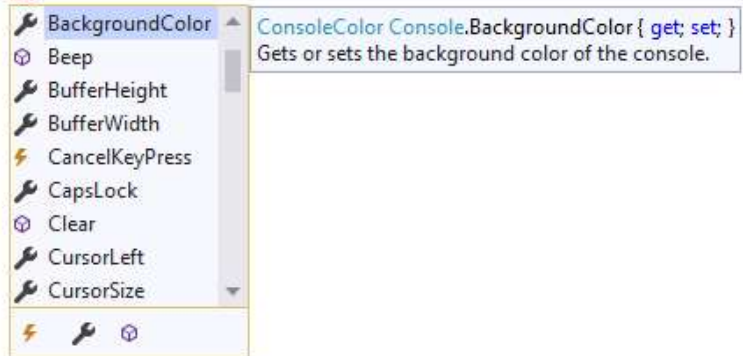
Only **public** methods can
be access by other
classes

So ConvertMilesToFeet()
calls the other **private**
methods

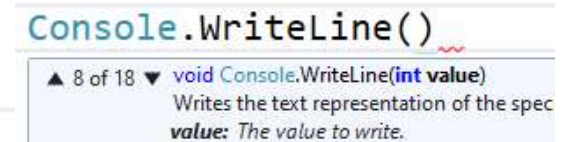
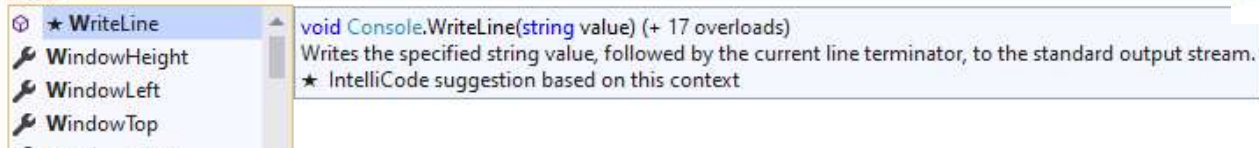
Intellisense

Your own block comments
generate intellisense

```
private void InputMiles()  
{  
    Console.  
}
```



```
Console.W
```



- Console is a class
- BackgroundColour is a Property.
- Beep is a Method
- 18 versions of WriteLine()

The InputMiles() Method

1 reference

```
private void InputMiles()
{
    Console.Write("Enter the number of miles >");
    string value = Console.ReadLine();
    miles = Convert.ToDouble(value);
}
```

ReadLine() returns whatever the user types as a string (text).

The string value has to be converted to a number (in this case a double)

Testing the DistanceConverter

```
class Program
{
    0 references
    static void Main()
    {
        Console.WriteLine(" C# Console Applications 2020");
        Console.WriteLine("");

        DistanceConverter converter = new DistanceConverter();
        converter.
```

ConvertMilesToFeet	void DistanceConverter.ConvertMilesToFeet() Calculate how many feet there are in the given miles
Equals	
GetHashCode	
GetType	
ToString	

```
D:\Projects\CO453-ConsoleAppAnswer\CO453-ConsoleApp>
C# Console Applications 2020
Enter the number of miles >2.5
```

Notice the only method that can be accessed is ConvertMilesToFeet()

The other private methods are available inside the object

```
converter.ConvertMilesToFeet();
```

SOME C# DATA TYPES

Data Type	Description	Range
int	Whole numbers	-2.1 Billion to +2.1 Billion
Float (single)	Floating decimal point	6-7 Digits
double	Floating decimal point	15 Digits
bool	Logical value	true false
char	Single character	Use single quotes marks 'A'
string	Sequence of chars	Use double quotes "Hello"
BNU 2020		44

Completing the Program

Evaluation

0 references

```
public void ConvertMilesToFeet()
{
    InputMiles();
    feet = miles * 5280;
    OutputFeet();
}
```

demonstrates input, output and arithmetic

Limited guidance to the user

The number of feet in a mile never changes

The program has to be run for each distance converted

1 reference

```
private void OutputFeet()
{
    Console.WriteLine(miles + " miles is " + feet + " feet!");
}
```

String concatenation

Microsoft Visual Studio Debug Console

```
C# Console Applications 2020
Enter the number of miles >1.0
1 miles is 5280 feet!
```

Constants & Arithmetic

Pascal case with no underbars is an alternative for constants

All constants should be declared by name

Constant names in uppercase with underbars (not MS) or Pascal case

Exceptions = 0, 1

```
public const int FEET_IN_MILES = 5280;
```

```
feet = miles * FEET_IN_MILES;
```

Operator	Description
++, --	Increment or decrement by 1
*, /	Multiplication, Division
%	Remainder after division
+, -	Addition, subtraction

Examples	Result
2 + 2 * 2	6
(2 + 2) * 2	8
C = 3; C++;	C = 4
9 / (5 / 2)	4
5.0/2.0	2.5
5 % 2	1

ENHANCE PROGRAM

```
0 references
public void ConvertMilesToFeet()
{
    OutputHeading();
    InputMiles();

    feet = miles * FEET_IN_MILES;

    OutputFeet();
}
```

```
1 reference
private void OutputHeading()
{
    Console.WriteLine();
    Console.WriteLine(" -----");
    Console.WriteLine("      Convert Miles to Feet      ");
    Console.WriteLine("      by Derek Peacock      ");
    Console.WriteLine(" -----");
    Console.WriteLine();
}
```

Microsoft Visual Studio Debug Console

C# Console Applications 2020

```
-----
      Convert Miles to Feet
      by Derek Peacock
-----

Enter the number of miles >2.5
2.5 miles is 13200 feet!
```

Independent Study



ADD METHODS SO
THAT THE USER CAN

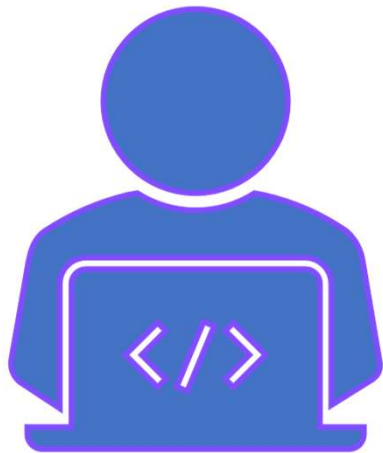


APP 01 FEATURE 02
CONVERT FEET TO
MILES



APP 01 FEATURE 3
CONVERT MILES TO
METRES

References



- C# 6 For Programmers, Deitel & Deitel (2017)
- Clean Code, Robert Martin (2009)
- [Tutorialspoint C# Classes](#)
- [Tutorialspoint C# Variables](#)
- [Visual Studio 2019](#)
- [Microsoft Tutorial Console App](#)
- [GitHub](#)
- [Markdown Cheatsheet](#)