CHEATSHEET - Object Oriented Programming in C#

Core Concepts

Component	Description	Example / Syntax
Classes and Objects	Templates for creating objects, encapsulating data and methods to manipulate data.	'class Car { string color; int speed; }
Inheritance	Allows a class to inherit behavior and attributes from another class.	'class ElectricCar : Car { int batteryLevel; }'
Polymorphism	Ability to present the same interface for differing underlying data types.	'override void Drive() { /* Implementation */}'
Encapsulation	Hides the internal state of the object and requires all interaction to be performed through an object's methods.	'public void SetSpeed(int spd) { speed = spd; }'
Abstraction	Reduces complexity by hiding details and showing only essentials, often using interfaces and abstract classes.	'abstract class Vehicle { abstract void Drive(); }'

Constructors

Component	Description	Example / Syntax
Multiple Construction	Supports multiple ways of initializing objects through different sets of parameters.	'Car(string c) { color = c; } Car(string c, int s) { color = c; speed = s; }'
Default Constructor	Automatically provided if no constructors are explicitly defined; initializes objects with default values	'Car() { color = "white"; speed = 0;}'

Methods

Component	Description	Example / Syntax
Method Overloading	Methods with the same name but different parameters to perform closely related functions.	'void Print(int num) {} void Print(sting msg) {}'
Optional Parameters	Allows methods to be involved with optional arguments by specifying default values.	'void Drive(int speed, boolheadlights = false) {}'
Named Parameters	Enhances method readability by allowing parameters to be passed in any order using their names.	Drive(headlightsOn: true, speed: 60);

Properties

Component	Description	Example / Syntax
Computed Properties	Does not store a value but calculates it based on other data.	'public int Speed { get { return mph; } set { mph = value; } }'
Read-Only Properties	Allows the property to be read but not modified after initialization	'public int MaxSpeed { get; }'
Write-Only Properties	Allows the property to be written but not read externally.	'public string Password { set { passwordHash = Hash(value); } }'

Class Members and Modifiers

Component	Description	Example / Syntax
Static Members	Belong to the class, rather than instances of the class; includes static method and fields.	'static int NumberOfCars;'
Access Modifiers	Control access levels for classes and class members: public, private, protected, and internal	'private string owner; public void SetOwner(string name) { owner = name; },

Advanced Usage

Component	Description	Example / Syntax
Delegates and Events	Delegates are type-sage function pointers. Events are notifications sent by objects when something important happens.	'delegate void UpdateStatus(string status); event UpateStatus OnUpdate;'
Composition	Describes a class that references one or more objects of other classes in instance variables	'class Engine { } class Car { Engine engine; }'
Const and ReadOnly	Use 'const' for compile-time constants and 'readOnly' for runtime constants that are set in the constructor.	'const double Pi = 3.14; readonly double circumference;'

AI Assistant & Course content

Course content

Overview

Q&AQuestions and answers

Notes

Announcements

Reviews

Learning tools

Section 1: UPDATED: Introduction, Overview of Visual Studio, DataTypes And Variables

51 / 56 | 3hr 6min51 of 56 lectures completed3hr 6min

Section 2: UPDATED: Making Decisions

20 / 28 | 1hr 33min20 of 28 lectures completed1hr 33min

Section 3: UPDATED: Loops

22 / 24 | 1hr 37min22 of 24 lectures completed1hr 37min

Section 4: UPDATED: Functions and Methods

20 / 20 | 1hr 34min20 of 20 lectures completed1hr 34min

Section 5: UPDATED: Object Oriented Programming (OOP)

41 / 43 | 3hr 10min41 of 43 lectures completed3hr 10min

- Lecture incomplete. Progress cannot be changed for this item.