Complete C# Masterclass

Your progress

Share

Parameter Modifiers in C# (ref, out, and in)

Parameter Modifiers in C# (ref, out, and in)

Introduction

In the previous sections, we explored functions, methods, and parameter passing in C#. While we covered the basics of how parameters work, there are a few advanced techniques that we did not introduce in the video lectures to keep things focused and streamlined. However, understanding these techniques is essential when working on real-world applications, especially when optimizing performance or ensuring specific behavior when passing arguments to methods.

This article will introduce parameter modifiers—specifically, ref, out, and in. These modifiers change how arguments are passed to methods, allowing us to pass data more efficiently or control how values are modified inside a method.

To make this concept easier to grasp, let's first look at an analogy.

1. What are Parameter Modifiers?

When you pass a value to a method in C#, it is typically passed by value—which means that the method works with a copy of the data. However, sometimes, we want to pass a reference to the original data or control how the data is passed. This is where parameter modifiers come in.

Passing Notes in a Classroom

Imagine you are in a classroom, and a teacher asks you to share some notes with a friend. There are three ways you could do this:

1. Passing a copy of your notes (pass by value) – Your friend gets a photocopy of the notes, but if they write on it, your original notes remain unchanged.
2. Passing your original notebook (pass by reference using ref) – Your friend directly writes in your notebook, modifying your notes.

3. **Giving your friend a blank sheet (out parameter) – You hand them a blank sheet, and they must write something on it before returning it to you.**
4. **Letting your friend read but not edit (in parameter) – You hand them your notes, but they can only read and not make any changes.**

Now, let's break down each of these parameter modifiers in detail.

2. ref Modifier (Passing by Reference)

What is ref?

The ref keyword allows us to pass a variable by reference rather than by value. This means that any changes made inside the method will reflect in the original variable.

Basic Syntax

```
1. void ModifyValue(ref int number)
2. {
3.     number += 10; // Modify the original value
4. }
5.
6. int myNumber = 5;
7. ModifyValue(ref myNumber);
8. Console.WriteLine(myNumber); // Output: 15
```

Step-by-Step Explanation

1. **The ModifyValue method accepts an integer by reference using ref.**
2. **The method modifies the value by adding 10.**
3. **The original myNumber is changed because we passed it by reference.**

Key Points about ref

✔ **The variable must be initialized before passing it to the method.**

✔ **Any modifications inside the method affect the original value.**

✔ **Useful when we want a method to modify existing data.**

3. out Modifier (Passing by Reference with Initialization Inside the Method)

What is out?

The out modifier is similar to ref, but with one key difference: the method must assign a value to the out parameter before returning.

Basic Syntax

```
1. void GetValues(out int result)
2. {
3.     result = 42; // Must be assigned before the method exits
4. }
5.
6. int myValue;
7. GetValues(out myValue);
8. Console.WriteLine(myValue); // Output: 42
```

Step-by-Step Explanation

1. **The GetValues method accepts an integer by reference using out.**
2. **The method assigns 42 to result before exiting.**
3. **The original variable myValue is modified with the new value.**

Key Points about out

✔ **The variable does not need to be initialized before passing it.**

✔ **The method must assign a value before returning.**

✔ **Useful when a method needs to return multiple values.**

Example: Returning Multiple Values

```
1. void Calculate(int x, int y, out int sum, out int product)
2. {
3.     sum = x + y;
```

```
    product = x * y;
}

int a = 5, b = 3, sum, product;
Calculate(a, b, out sum, out product);
Console.WriteLine($"Sum: {sum}, Product: {product}");
// Output: Sum: 8, Product: 15
```

4. in Modifier (Passing Read-Only Reference)

What is in?

The in modifier allows us to pass a variable by reference, but it cannot be modified inside the method. This is useful when passing large structures or objects efficiently without allowing them to be changed.

Basic Syntax

```
void PrintValue(in int number)
{
    Console.WriteLine(number); // Allowed
    // number += 10; // Not allowed (will cause a compile error)
}

int myNumber = 100;
PrintValue(in myNumber);
```

Key Points about in

✔ **The variable must be initialized before passing it.**

✔ **The method cannot modify the parameter.**

✔ **Useful for performance optimization when working with large objects.**

5. Comparing ref, out, and in

Feature ref out in

Requires initialization before passing? ✅ Yes ❌ No ✅ Yes

Method must assign a value? ❌ No ✅ Yes ❌ No

Can be modified inside the method? ✅ Yes ✅ Yes ❌ No

Use case Modify existing data Return multiple values Pass large objects efficiently

6. When to Use Each Modifier?

✅ Use ref when you need a method to modify an existing variable.

✅ Use out when you need a method to return multiple values.

✅ Use in when passing large objects that should not be modified.

7. Best Practices and Common Mistakes

Best Practices

✔ Use ref only when necessary to avoid unintended side effects. ✔ Use out for returning multiple values cleanly. ✔ Use in for performance benefits when passing large structs.

Common Mistakes

❌ Forgetting to initialize ref variables before passing them.

❌ Not assigning a value to an out parameter inside the method.

❌ Trying to modify an in parameter (which is read-only).

Conclusion

Parameter modifiers (ref, out, and in) provide powerful ways to control how data is passed into methods. They help optimize performance, allow for multiple return values, and enable safe modifications of variables.

If you have any questions, feel free to ask in the Q&A!.

Happy coding!

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

18 / 20 | 1hr 34min18 of 20 lectures completed1hr 34min

- **Lecture completed. Progress cannot be changed for this item.**

Play

92. Intro To Functions / Methods

7min

- **Lecture completed. Progress cannot be changed for this item.**

Play

93. Void Method without Parameters

5min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

94. Void Method with a Parameter Part 1

5min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

95. Void Method with a Parameter Part 2

7min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Start

Quiz 10: Understanding Methods in C#

- **Lecture completed. Progress cannot be changed for this item.**

Play

96. Scope of variables and parameters

3min

- **Lecture completed. Progress cannot be changed for this item.**

Start

97. Quick lesson about Argument Promotion

3min

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

Start

98. Parameter Modifiers in C# (ref, out, and in)

4min

- **Lecture completed. Progress cannot be changed for this item.**

Play

99. Moving to the classical Template - Top Level Statements

9min

- **Lecture completed. Progress cannot be changed for this item.**

Play

100. Moving our Methods outside of the Main Method

4min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

101. Fields, instance variables and how they differ from local variables

5min

- **Lecture completed. Progress cannot be changed for this item.**

Start

Quiz 11: Variable and Scope Understanding

- **Lecture completed. Progress cannot be changed for this item.**

Play

102. WeatherSimulator - Using Arrays, Random, and For Loops

10min

- **Lecture completed. Progress cannot be changed for this item.**

Play

103. WeatherSimulator - Calculating the Average Temperature

7min

- **Lecture completed. Progress cannot be changed for this item.**

Play

104. Mixing Doubles and Ints when Calculating

3min

- **Lecture completed. Progress cannot be changed for this item.**

Play

105. WeatherSimulator - Getting the Min and Max Values of an array

5min

- **Lecture completed. Progress cannot be changed for this item.**

Play

106. WeatherSimulator - Getting the Most common weather condition

12min

- **Lecture completed. Progress cannot be changed for this item.**

Start

107. CHEATSHEET - Functons and Methods in C#

0min

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

Start

108. Recursion in C#

4min

- **Lecture completed. Progress cannot be changed for this item.**

Start

Coding Exercise 9: ADVANCED EXERCISE: Calculating Average Temperature

Section 5: UPDATED: Object Oriented Programming (OOP)

18 / 43 | 3hr 10min18 of 43 lectures completed3hr 10min

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

Play

109. Objects Intro

2min

- **Lecture completed. Progress cannot be changed for this item.**

Play

110. Introduction To Classes And Objects

3min

- **Lecture completed. Progress cannot be changed for this item.**

Play

111. Creating our First own Class

8min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

112. Member Variable and Custom Constructor

7min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

113. Properties - Autogenerated - to protect our member variable

6min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

114. Defining how a property is set

8min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

115. Modifying the Get of our Property Part 1

7min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

116. Modifying the Get of our Property part 2

5min

- **Lecture completed. Progress cannot be changed for this item.**

Play

117. Having Multiple Constructors

7min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

118. Default Constructor and Use Cases

6min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Start

Quiz 12: Understanding Constructors

- **Lecture completed. Progress cannot be changed for this item.**

Play

119. Methods in Classes

7min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

120. Methods in Classes in more detail

8min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Start

121. Expression Bodied Members in C#

3min

- **Lecture completed. Progress cannot be changed for this item.**

Start

122. What are Inner Classes in C#?

3min

- **Lecture completed. Progress cannot be changed for this item.**

Start

123. Partial Classes and Methods

3min

- **Lecture completed. Progress cannot be changed for this item.**

Play

124. Optional Parameters

4min

Resources

- **Lecture completed. Progress cannot be changed for this item.**

Play

125. Named Parameters

3min

- **Lecture completed. Progress cannot be changed for this item.**

Start

126. Operator Overloading in C#

3min

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

Start

127. Passing Arguments by Value and by Reference

4min

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

Play

128. Computed Properties and No Constructor

3min

Resources

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

Play

129. Static Methods

7min

Resources

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

Start

Coding Exercise 10: Using Static Methods

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

Play

130. Static Fields

3min

Resources

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

Start

131. Static Keyword Considerations

3min

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

Start

132. The is Operator and the as Operator in C#

3min

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

Play

133. Public and Private Keywords

5min

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

Play

134. ID Key and readonly

7min

Resources

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

Play

135. Read Only Properties

3min

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

Start

Coding Exercise 11: Working with Read-Only Properties

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

Play

136. Write Only Properties

5min

Resources

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

Play

137. Const and ReadOnly

5min

Resources

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

Start

Quiz 13: Working with Read-Only Properties

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

Play

138. Quiz Project Introduction

4min

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

Play

139. QuizApp - Question Class

5min

Resources

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

Play

140. Keyword This

3min

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

Play

141. Displaying Questions

6min

Resources

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

Play

142. Displaying Answers, Console.Write and Console.ForegroundColor

7min

Resources

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

Play

143. Getting the UserInput and checking if it is right

6min

Resources

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

Play

144. Displaying Multiple Questions and if we are right or wrong

8min

Resources

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

Play

145. Displaying the Results

8min

Resources

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

Start

146. CHEATSHEET - Object Oriented Programming in C#

0min

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

Start

Coding Exercise 12: ADVANCED EXERCISE: Creating a Class with Properties and Methods

Section 6: UPDATED: Collections in C#

0 / 27 | 2hr 1min0 of 27 lectures completed2hr 1min

Section 7: UPDATED: Error Handling

0 / 14 | 45min0 of 14 lectures completed45min

Section 8: UPDATED: Inheritance

0 / 22 | 1hr 21min0 of 22 lectures completed1hr 21min

Section 9: UPDATED: Interfaces and Polymorphism

0 / 24 | 1hr 22min0 of 24 lectures completed1hr 22min

Section 10: UPDATED: Structs in C#

0 / 9 | 58min0 of 9 lectures completed58min

Section 11: UPDATED: Events and delegates

0 / 14 | 1hr 21min0 of 14 lectures completed1hr 21min

Section 12: UPDATED: Regular Expressions

0 / 11 | 43min0 of 11 lectures completed43min

Section 13: WPF - Windows Presentation Foundation

0 / 42 | 2hr 31min0 of 42 lectures completed2hr 31min

Section 14: WPF Project - Currency Converter - Part 1

0 / 8 | 1hr 14min0 of 8 lectures completed1hr 14min

Section 15: Using Databases With C#

0 / 12 | 2hr 2min0 of 12 lectures completed2hr 2min

Section 16: WPF Project - Currency Converter - Part 2

0 / 9 | 1hr 31min0 of 9 lectures completed1hr 31min

Section 17: Linq

0 / 13 | 2hr 18min0 of 13 lectures completed2hr 18min

Section 18: WPF Project - Currency Converter with GUI Database and API - Part 3

0 / 3 | 31min0 of 3 lectures completed31min

Section 19: The exercises for your coding interviews

0 / 4 | 5min0 of 4 lectures completed5min

Section 20: C# Clean Code

0 / 24 | 1hr 37min0 of 24 lectures completed1hr 37min

Section 21: C# Generics

0 / 18 | 1hr 38min0 of 18 lectures completed1hr 38min

Section 22: Threads

0 / 8 | 1hr 10min0 of 8 lectures completed1hr 10min

Section 23: Unit Testing - Test Driven Development TDD

0 / 36 | 3hr 24min0 of 36 lectures completed3hr 24min

Section 24: UNITY - Basics

0 / 16 | 1hr 35min0 of 16 lectures completed1hr 35min

Section 25: UNITY - Building the Game Pong with Unity

0 / 20 | 2hr 34min0 of 20 lectures completed2hr 34min

Section 26: UNITY - Building a Zig Zag Clone With Unity

0 / 18 | 2hr 11min0 of 18 lectures completed2hr 11min

Section 27: UNITY - Building a Fruit Ninja Clone With Unity

0 / 14 | 2hr 8min0 of 14 lectures completed2hr 8min

Section 28: Thank you for completing the course!

0 / 1 | 4min0 of 1 lecture completed4min

Section 29: Bonus

0 / 1 | 1min0 of 1 lecture completed1min

Teach the world online

Create an online video course, reach students across the globe, and earn money

Teach on Udemy

Top companies choose Udemy Business to build in-demand career skills.

Explore top skills and certifications

Certifications by Issuer

- **Amazon Web Services (AWS) Certifications**
- **Six Sigma Certifications**
- **Microsoft Certifications**
- **Cisco Certifications**
- **Tableau Certifications**
- **See all Certifications**

Web Development

- **Web Development**
- **JavaScript**
- **React JS**
- **Angular**
- **Java**

IT Certifications

- **Amazon AWS**
- **AWS Certified Cloud Practitioner**

- AZ-900: Microsoft Azure Fundamentals
- AWS Certified Solutions Architect - Associate
- Kubernetes

Leadership

- Leadership
- Management Skills
- Project Management
- Personal Productivity
- Emotional Intelligence

Certifications by Skill

- Cybersecurity Certification
- Project Management Certification
- Cloud Certification
- Data Analytics Certification
- HR Management Certification
- See all Certifications

Data Science

- Data Science
- Python
- Machine Learning
- ChatGPT
- Deep Learning

Communication

- Communication Skills
- Presentation Skills
- Public Speaking

- Writing
- PowerPoint

Business Analytics & Intelligence

- Microsoft Excel
- SQL
- Microsoft Power BI
- Data Analysis
- Business Analysis

About

- About us
- Careers
- Contact us
- Blog
- Investors

Discovery Udemy

- Get the app
- Teach on Udemy
- Plans and Pricing
- Affiliate
- Help and Support

Udemy for Business

- Udemy Business

Legal & Accessibility

- Accessibility statement
- Privacy policy
- Sitemap

- Terms