Complete C# Masterclass

Your progress

Share

Quick lesson about Argument Promotion

Argument Promotion in C#

Introduction

In this section of the course, we have covered several key concepts related to functions and methods, including parameters, return types, and method overloading. However, one important concept that wasn’t explicitly covered in video lectures is Argument Promotion.

To keep the course streamlined, we have chosen to explain this topic here in a dedicated article. Argument Promotion is an essential feature of C# that plays a crucial role in ensuring smooth and efficient function execution, especially when dealing with overloaded methods or implicit type conversions.

This article will introduce the concept, explain why it is useful, provide an intuitive analogy, and demonstrate its application with practical examples. By the end, you’ll have a strong grasp of argument promotion and understand when and how to use it effectively.

1. What is Argument Promotion?

Definition

Argument Promotion refers to the automatic conversion of a smaller data type into a larger data type when passing an argument to a method. This happens implicitly when calling functions that require a parameter of a larger data type, ensuring that data is preserved and preventing unintended data loss.

Why is Argument Promotion Useful?

Argument promotion makes it easier to write flexible and efficient functions by allowing implicit conversions without requiring manual casting. This is particularly helpful when working with method overloading, where different method signatures may accept different parameter types.

2. A Bigger Box for Smaller Items

Imagine you have a collection of boxes in different sizes:

- A small box for storing small objects like marbles.
- A medium box for slightly bigger objects like apples.
- A large box that can hold even bigger objects like a basketball.

Now, if you need to store a small item (like a marble), you can place it in either a small box or a larger one. However, the reverse isn’t always possible—you can’t fit a basketball into a small box.

Similarly, in C#, a smaller data type (like an int) can be placed in a method that expects a larger data type (like a double), but not vice versa unless explicitly converted.

3. Declaring and Using Argument Promotion

Basic Syntax

When calling a method, C# automatically promotes arguments from smaller data types to larger data types if needed. Here’s an example:

```
void DisplayNumber(double number)
{
    Console.WriteLine("The number is: " + number);
}

int myInt = 42;
DisplayNumber(myInt); // Implicit promotion from int to double
```

Step-by-Step Implementation

Step 1: Creating a Method with a Larger Data Type Parameter

Let’s define a method that expects a double parameter:

```
void CalculateArea(double radius)
{
    double area = Math.PI * radius * radius;
    Console.WriteLine("The area is: " + area);
}
```

Step 2: Calling the Method with an int Argument

Even though the method expects a double, we can pass an int:

```
int myRadius = 5;
CalculateArea(myRadius); // Implicitly promoted from int to double
```

Expected Output

```
The area is: 78.53981633974483
```

Here, the int value 5 is automatically converted to 5.0 (a double) before the method executes.

4. Argument Promotion in Method Overloading

One of the key use cases for argument promotion is in method overloading. Consider the following overloaded methods:

```
void PrintValue(int value)
{
    Console.WriteLine("Integer: " + value);
}

void PrintValue(double value)
{
    Console.WriteLine("Double: " + value);
}
```

Now, if we call PrintValue(10), the method that accepts an int will execute. However, if we pass a float, argument promotion will take effect, and the method that accepts a double will be used:

```
float myFloat = 10.5f;
```

2. PrintValue(myFloat); // Promoted to double

Expected Output:

1. Double: 10.5

Since there isn’t an exact match for a float, the next closest match (double) is chosen due to argument promotion.

5. Comparing Argument Promotion with Explicit Casting

Feature Argument Promotion vs Explicit Casting

Automatic? ✅ Yes ❌ No

Data Loss Risk? ❌ No ✅ Yes

Syntax Simplicity ✅ Easy ❌ Requires manual casting

When using explicit casting, you are forcing a conversion, which may lead to data loss:

1. int x = 10;
2. double y = (double)x; // Explicit casting

With argument promotion, the conversion happens seamlessly without requiring explicit syntax.

6. Best Practices and Common Mistakes

Best Practices

✔ Prefer argument promotion when calling methods to avoid unnecessary type conversions.
✔ Use method overloading wisely to take advantage of argument promotion.
✔ Keep method signatures consistent to avoid confusion when dealing with implicit conversions.

Common Mistakes

❌ Expecting promotion to work in reverse (e.g., passing a double to a method expecting an int).
❌ Using explicit casting unnecessarily when argument promotion can handle the conversion.

❌ Assuming all conversions are lossless—promotion works without loss, but demotion does not.

7. Conclusion

Argument Promotion is an important concept in C# that simplifies method calls by allowing smaller data types to be automatically converted to larger compatible types. It enhances method flexibility, reduces the need for explicit casting, and plays a vital role in method overloading.

Understanding how argument promotion works can help you write cleaner and more efficient code. 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

17 / 20 | 1hr 34min17 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 incomplete. 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