

Java vs C#

Presented by :

Ahmet Eren Çiçekdal

B231202020

Ebru Çağlar

B231202064

Ceren Güleşen

B221202060

Muhammet Karaman

B221202052

Ayşenur Yılmaz

B231202019



Overview

Java and C# (C Sharp) are two of the most popular and widely used object-oriented programming languages in the field of software development. In this presentation, we will examine the core features, differences, and appropriate use cases of these two languages.

What is The Object Orianted Programming ?

Object-Oriented Programming (OOP) is a programming paradigm that organizes and structures code around objects. In this approach, objects encapsulate data (attributes) and the operations (methods) that can be performed on that data. OOP is particularly useful for developing, managing, and extending complex systems efficiently.

Advantages of OOP:

- Code Reusability: Through inheritance, code written once can be reused elsewhere.
- Ease of Maintenance and Scalability: Modular code structure makes modifications easier.
- Real-World Modeling: Objects represent real-world entities, making the code more intuitive.
- Security: Encapsulation protects data from unauthorized access.

Java

Java is a powerful and easy-to-use object-oriented programming language developed by Sun Microsystems. It was originally created to be simple, portable, and reliable, allowing programs to run on any device with the Java Virtual Machine (JVM). Over time, Java has become one of the most popular programming languages, used in everything from mobile apps and web development to enterprise systems. Its "write once, run anywhere" feature makes it a versatile choice for developers.



- Initial Release Date: 1995
- Developer: Sun Microsystems (later acquired by Oracle)
- Platform: Platform-independent, "Write Once, Run Anywhere" (WORA)
- Application Areas: Web applications, mobile applications (Android), enterprise software

Top Java Applications



The Mars Rover



Google



eBay



Minecraft



Netflix



Uber



Eclipse IDE



Twitter



Spotify



CashApp



Signal



NASA WorldWind

C#

C# is a modern, powerful, and object-oriented programming language developed by Microsoft. It was designed to be simple, flexible, and efficient, making it ideal for building a wide range of applications, from desktop and web apps to games and enterprise systems. As part of the .NET framework, C# allows developers to create reliable and scalable software that can run on different platforms. Its clean syntax and rich features make it a popular choice for both beginners and experienced developers.



- Initial Release Date: 2000
- Developer: Microsoft
- Platform: .NET Framework and .NET Core (now unified as .NET 5/6)
- Application Areas: Windows applications, game development (Unity), web applications

Top C# Applications



Microsoft Offices



Visual Studio



Norton Antivirus



Unity



Github



Stackoverflow

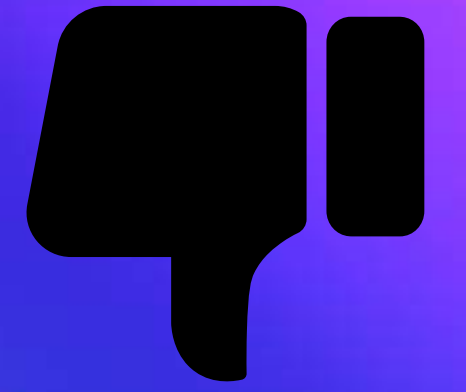
Advantages



Java

- Can run seamlessly on various operating systems thanks to the JVM.
- Widely used in large-scale enterprise applications and systems because of its robustness.
- Supports scalable applications and architectures.
- Versatile, finding applications from web and mobile development to embedded systems.
- Boasts a large collection of libraries, surpassing C# in terms of the number of packages available.
- It has been around for a very long time, which means it has one of the largest and most knowledgeable communities in the IT world.
- Enforces strong typing, enhancing code reliability.
- Rooted in the principles of object-oriented programming while also embracing functional concepts.
- A new version of the language every 6 months.

Disadvantages



Java

- Verbose and less intuitive syntax compared to many languages, leading to an increased likelihood of boilerplate code.
- Tends to be more CPU and memory-intensive than several languages, including C#.
- Requires compilation, potentially introducing delays in development and testing workflows.
- Lacks support for operator overloading and other modern features.
- Does not provide support for nullable references, limiting flexibility in handling null values.
- May not be the optimal choice for small, lightweight projects.

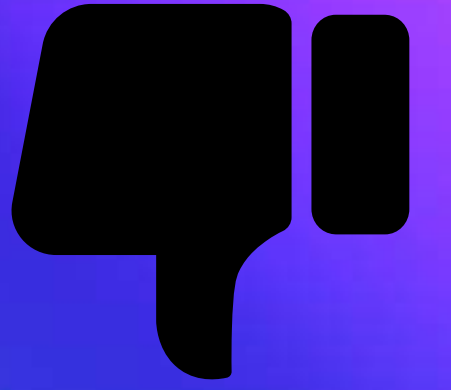
Advantages



C#

- Open-source approach to design.
- Easier syntax than Java.
- Executable on multiple operating systems, due to the .NET runtime.
- Great for improvable, large enterprise applications thanks to its flexibility and integration with the Microsoft ecosystem.
- Facilitates the development of improvable applications.
- Applicable across diverse domains, spanning web and mobile development to embedded systems.
- Supports operator overloading, structs, nullable reference types and other cool features.
- Enforces strong typing for raised code reliability.
- A well-established programming language, fostering a community rich in expertise.
- Rooted in object-oriented principles while also embracing functional programming concepts.








Disadvantages



C#

- Allows you to write unsafe code due to pointers, the goto statement, and unmanaged memory allocation.
- Especially supports unchecked exceptions, leading to less sturdy error handling as developers are not compelled to handle certain types of exceptions explicitly.
- Poses challenges for beginners due to its complex setup requirements.
- Not the best choice for small, lightweight projects.
- Involves compilation, introducing potential delays in development and testing workflows.
- Still too tied to the Windows ecosystem.

Technical Features

Category	Java	C#
 Syntax	Detailed, strict, with a lot of rules	A bit detailed, but clean and expressive
 Performance	Fast, but not faster than C# in most cases	Fast
 Memory usage	High	Good
 Ecosystem	Extensive, with millions of libraries available	Vast, with hundreds of thousands of libraries available
 Community	Several millions of users	A few millions of users
 Improvability	Very high, especially in the enterprise field	Very high, especially in Azure cloud platform
 Web Scraping	Supported by some libraries and many resources	Supported by some libraries



Application Areas

Area	Java	C#
Web Development	Spring, Hibernate	ASP.NET
Mobile Applications	Android Studio	Xamarin, MAUI
Game Development	Less commonly used	Unity, Godot
Enterprise Software	Java EE	.NET

Java "Hello World" Code

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

C# "Hello World" Code

```
using System;  
  
class Program {  
    static void Main(string[] args) {  
        Console.WriteLine("Hello, World!");  
    }  
}
```

Importing Libraries in Java

Java: `import` keyword

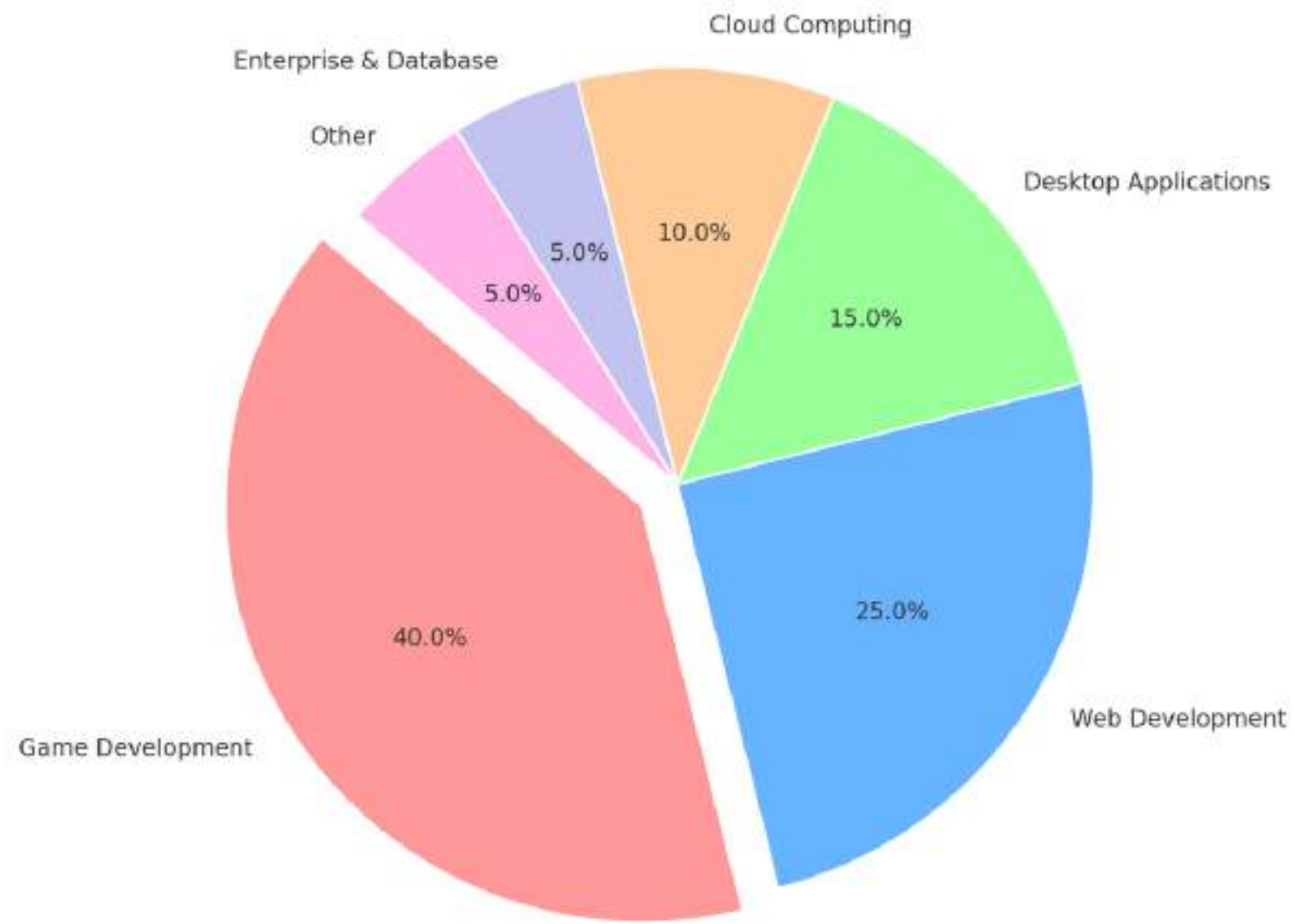
```
import java.util.*;  
import java.io.*;
```

Importing Libraries in C#

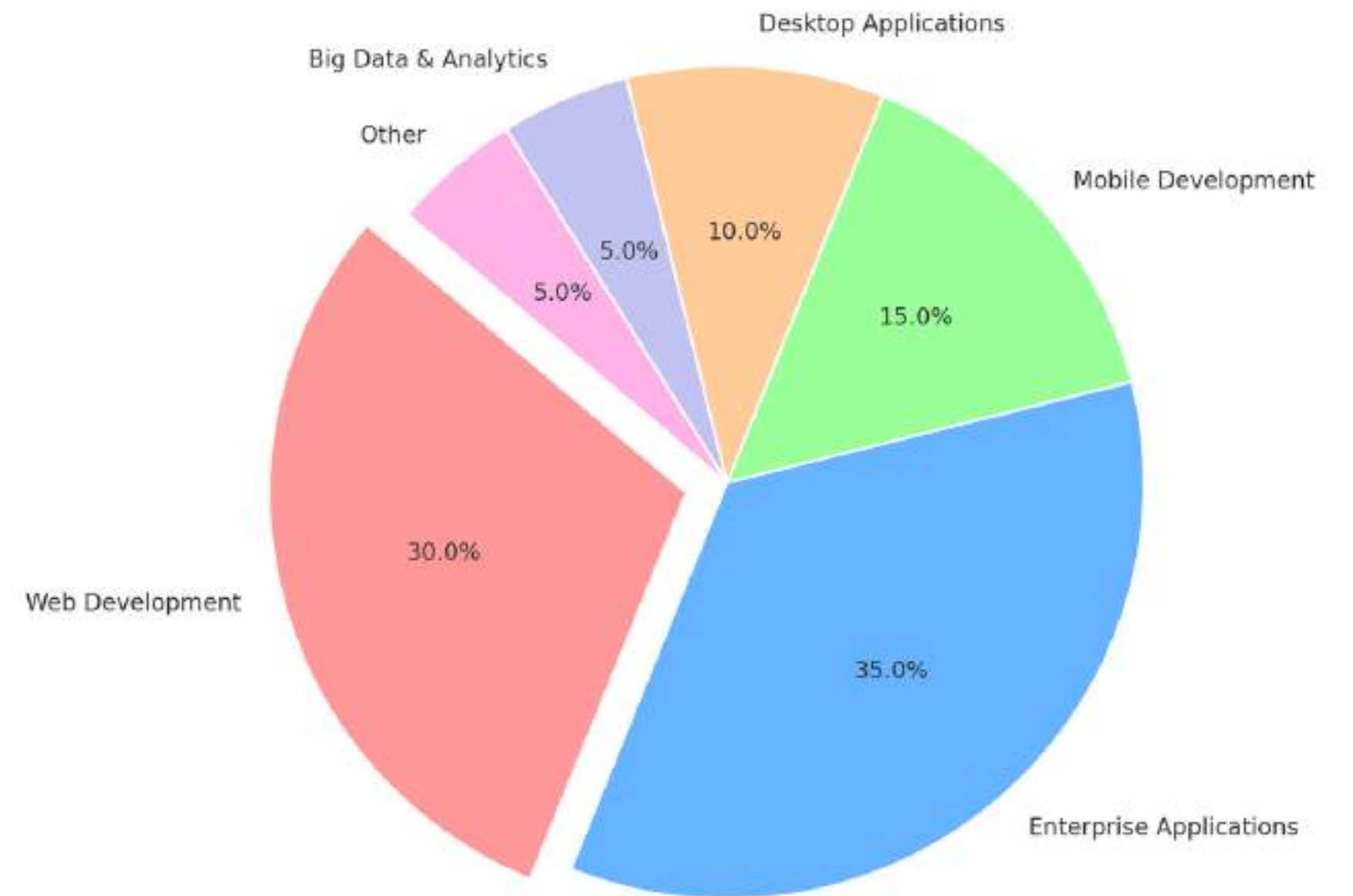
C#: `using` keyword

```
using System;  
using System.IO;  
using System.Reflection;
```

C# Usage Areas



Java Usage Areas



Which to Choose?

Java

- If platform independence is a priority.
- When developing for Android.
- For large-scale enterprise applications.

C#

- When building Windows or Microsoft-based applications.
- For game development using Unity.
- To take advantage of modern language features.

Conclusion

Java and C# are two powerful programming languages that excel in different domains. The choice between them depends on the project requirements and developer needs. Both languages hold a significant position in the modern software development landscape.

References

<https://www.slideshare.net>

<https://brightdata.com>

<https://www.sphinx-solution.com>

<https://hackr.io/>

<https://www.techcareer.net/>

<https://softteco.com/>