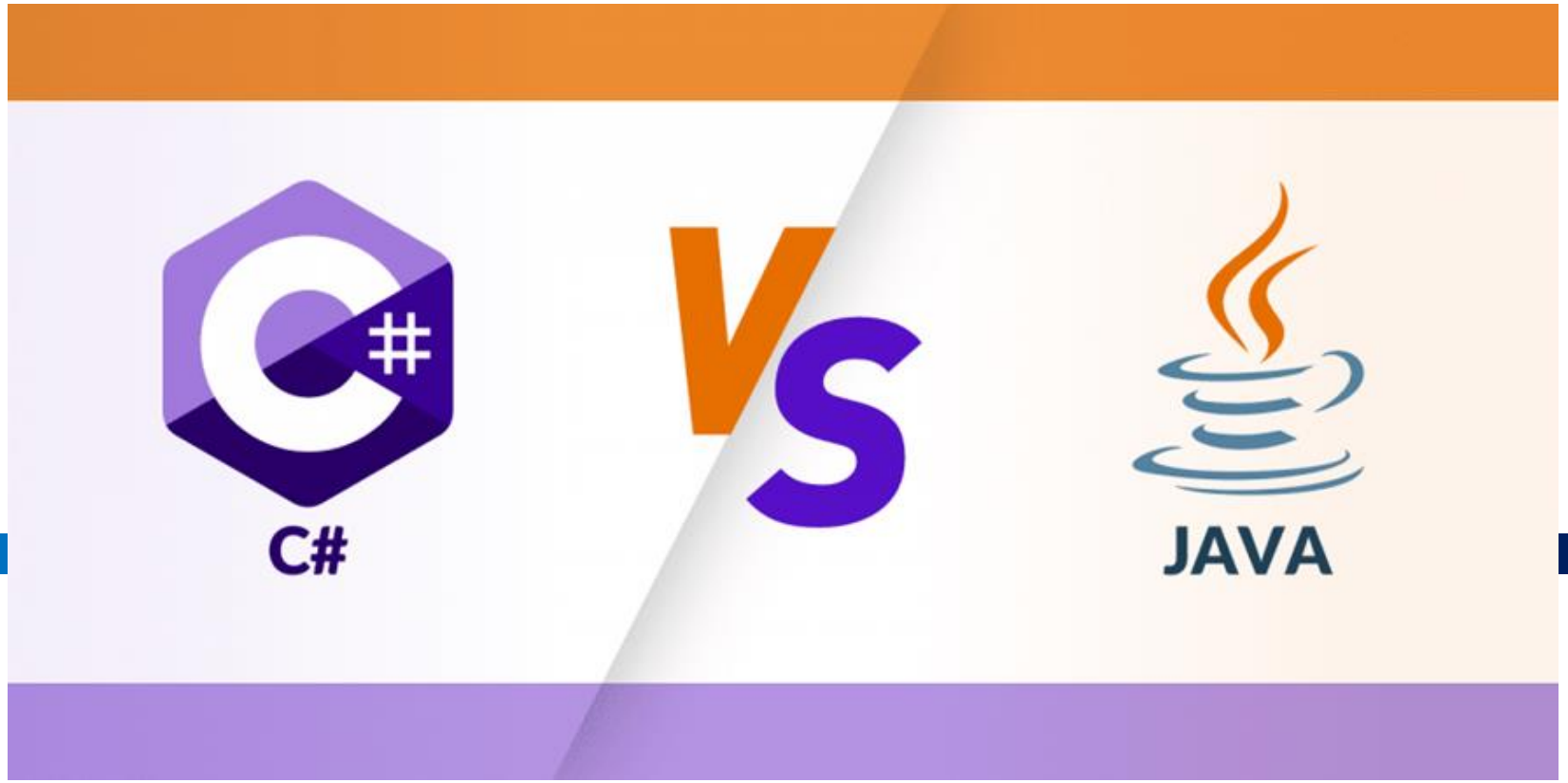

Unit 1

Migrating from C# to JAVA

C#

- Developed by Microsoft -> .NET
- Derived from C++ and Java
- Object-Oriented
- No pointers / explicit garbage collection

JAVA

- Developed by SUN Microsystems (currently updated by Oracle)
- Platform independent
- Object-Oriented
- No pointers / explicit garbage collection
- Large pool of software tools / libraries

Main Function

C#

```
static void Main(string[] args)
```

Java

```
public static void main(String[]
args)
```

Print statements

C#

```
System.Console.WriteLine("Hello world!");

Console.WriteLine("Hello again!");
```

Java

```
System.out.println("Hello world!");
```

Jose Ramón García Sevilla

jrgarcia@iesmarenostrum.com

Constants

C#

```
const int K = 100;
```

Java

```
static final int K = 100;
```

Inheritance

C#

```
class B:A, IComparable
{
    ……
}
```

Java

```
class B extends A implements IComparable
{
    ……
}
```

Jose Ramón García Sevilla

jrgarcia@iesmarenostrum.com

Overloading / overriding

C#

Virtual methods must be explictly marked using the keywords `virtual / override`. Use `base` to call a base class method.

Java

All methods are virtual. Use `super` to call a base class method.

Jose Ramón García Sevilla

jrgarcia@iesmarenostrum.com

Classes in a file

C#

Multiple public classes allowed in a single file. No name restrictions.

Java

Only one public class in a single file. File name and public class name must match.

Switch statements

C#

Do not allow fall-through statements

```
 /* ERROR: Won't compile due to fall-through at
case "D" */
   case "D": Console.WriteLine("D seen");
   case "E": Console.WriteLine("E seen");
             break;
```

Java

Allow fall-through statements

Jose Ramón García Sevilla jrgarcia@iesmarenostrum.com

Libraries

C#

```
using System;

using System.IO;

using System.Reflection;
```

Java

```
import java.util.*;

import java.io.*;
```

Properties

C#

```
public int Size {
  get { return size; } / get -> size;
  set { size = value; } / set -> size = value;
}
```

Java

```
public int getSize() {
  return size;
}
public void setSize ( int value ) {
  size = value;
}
```

Strings

C#

```
string s1, s2;
if ( s1 == s2 )
while ( s1 < s2 )
```

Java

```
String s1, s2;
if ( s1.equals(s2) )
while ( s1.compareTo(s2) < 0 )
If ( s1.compareToIgnoreCase(s2) == 0 )
```

Parameters

C#

Both by value or reference.

Java

Just by value.

Exceptions

C#

Only handles unchecked exceptions

Java

Handles both checked and unchecked exceptions

https://www.geeksforgeeks.org/checked-vs-unchecked-exceptions-in-java/

Collections

C#

Can be directly instantiated

Java

Collections are interfaces and cannot be directly instantiated

```
Set<String> mySet = new HashSet<>();
```

Jose Ramón García Sevilla *jrgarcia@iesmarenostrum.com*

Operator overloading

C#

Supported

Java

Not supported 

Jose Ramón García Sevilla *jrgarcia@iesmarenostrum.com*

Coding conventions

C#

https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/coding-conventions

Java

https://www.oracle.com/technetwork/java/codeconventions-150003.pdf