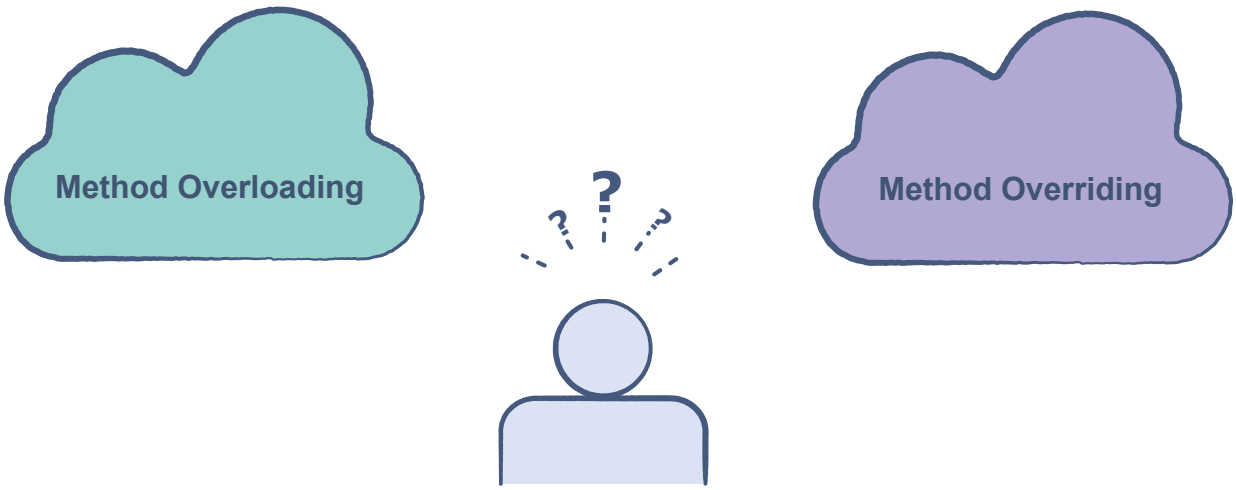
Difference Between Overloading and Overriding

In this lesson, you will get familiar with the differences between method overloading and method overriding.

WE'LL COVER THE FOLLOWING

Method Overloading & Method Overriding

Beginner developers often get confused regarding the terms method overloading and method overriding. These are two completely different concepts.



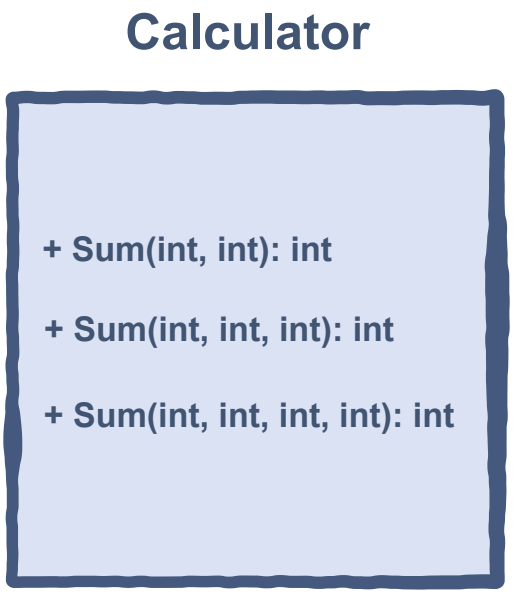
Let's compare the differences below:

Method Overloading	Method Overriding
It is done inside the same class	Base and derived class(es) are

It is done inside the same class.	required here.
Overloading happens at **compile time.**	Overriding happens at **runtime**
Gives better performance because the binding is being done at compile time.	Gives worse performance because the binding is being done at run time.
Private and **sealed** methods can be overloaded.	**Private** and **sealed** methods can not be overridden.
Return type of the method does not matter in case of method overloading.	Return type of the method must be the same in the case of overriding.
Arguments must be different in the case of overloading.	Arguments must be the same in the case of overriding.
Mostly used to increase the readability of the code.	Mostly used to have a separate implementation for a method that is already defined in the base class.
This happens at the compile time, so it can be referred to as *static or compile-time* polymorphism.	This happens at run time, so it can be called *dynamic or runtime polymorphism*.

Method Overloading Example

Let's implement the calculator class in C#:

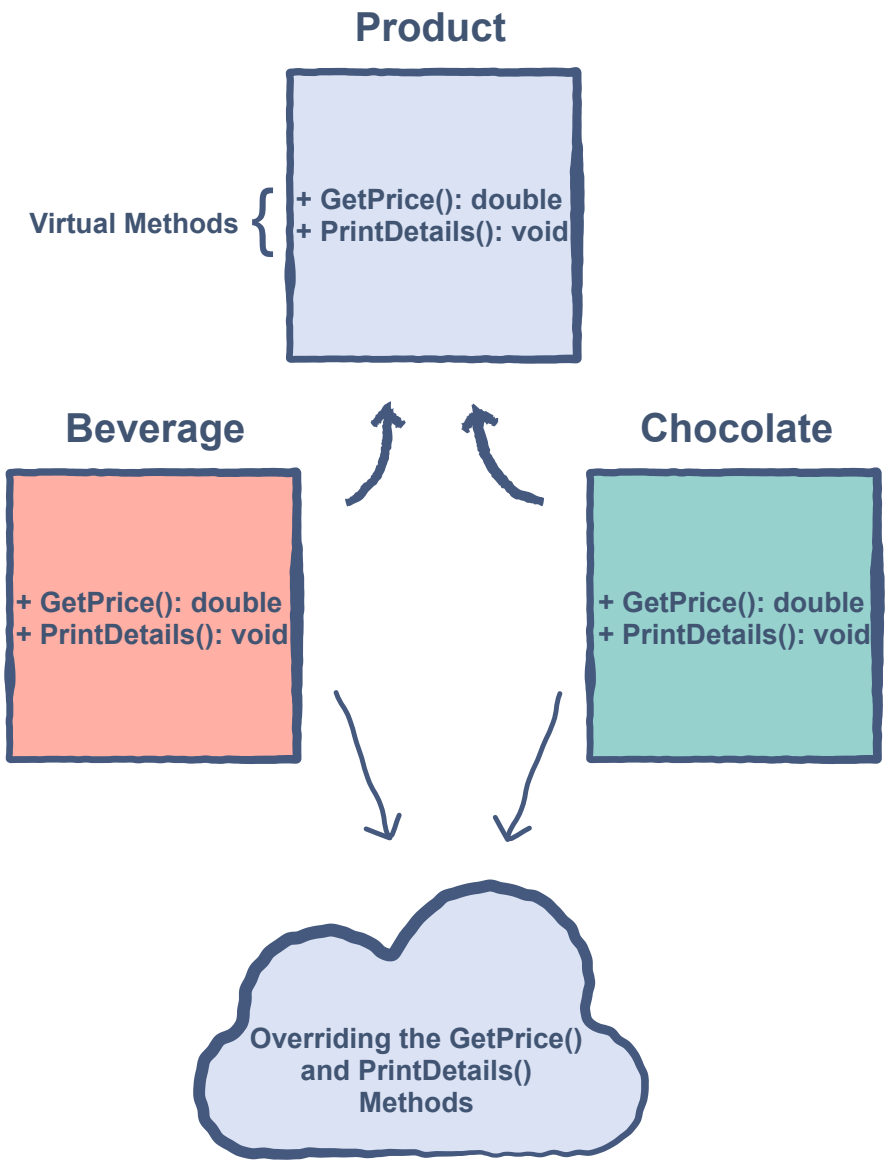



```
//Calculator Class
class Calculator {

  // Sum funtions with two parameters
  int Sum(int num1, int num2) {
    return num1 + num2;
  }

  // Sum funtions with three parameters
  int Sum(int num1, int num2, int num3 ) {
    return num1 + num2 + num3;
  }

  // Sum funtions with four parameters
  int Sum(int num1, int num2, int num3, int num4 ) {
    return num1 + num2 + num3 + num4;
  }

  public static void Main(string[] args) {
    var cal = new Calculator();

    Console.WriteLine("10 + 20 =  " + cal.Sum(10, 20));
    Console.WriteLine("10 + 20 + 30 =  " + cal.Sum(10, 20, 30));
    Console.WriteLine("10 + 20 + 30 + 40 =  " + cal.Sum(10, 20, 30, 40));
  }

}
```

Here we have **3** different versions of the `Sum()` method. The `Sum()` method is overloaded here.

Method Overriding Example

We have already looked at an example of method overriding in the previous lesson:



We had a base class, `Product`, and two derived classes `Beverage` and `Chocolate.` Here, the `GetPrice()` and `PrintDetails()` methods of the Product class were overridden in the Beverage and the Chocolate class.

We've learned the differences between method overloading and method overriding. In the upcoming lessons, you can test your knowledge of polymorphism with a quiz and coding exercises.