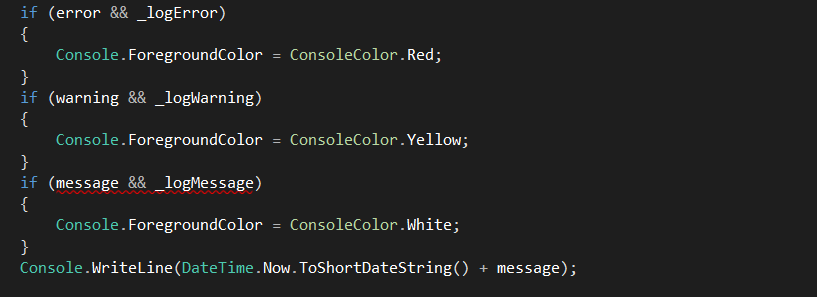
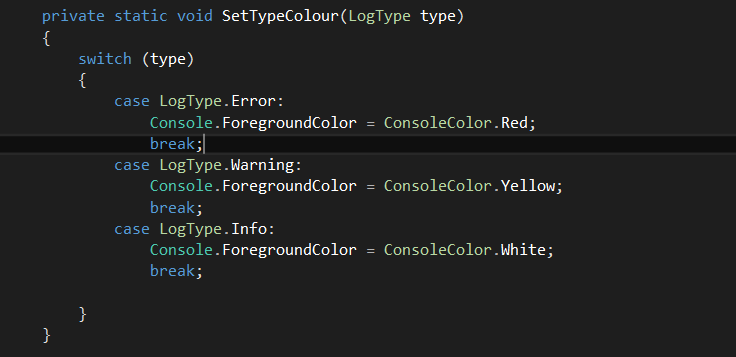
**Code Review**

**If you were to review the following code, what feedback would you give? Please be specific and indicate any errors that would occur as well as other best practices and code refactoring that should be done.**

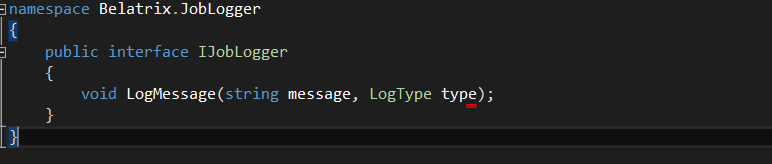
1. Hay un error de compilación, debido a que el nombre de la variable message esta duplicado tanto como de tipo bool y como tipo string en los parámetros de entrada del método original **LogMessage**
2. El código no es reusable y si se agregar algún tipo nuevo de log, se tendría que agregar más código al método existente que duplicaría la estructura actual de los sources de logs(Archivo,Base de datos,Consola).
3. Log Message es un método demasiado largo. Se debe descomponer o dividir el método LogMessage



Por ejemplo, este pequeño segmento de código podríamos extraerlo en un nuevo método. Se debe tomar en cuenta las variables de dentro del alcance del código. Podemos poner el tipo de log (Error,Warning o Message) como parámetro de entrada para el nuevo método. Y de igual forma, las variables que son modificadas como la variable message.



1. Podemos crear una interface la cual sea implementada por los tres tipos o formar de logs que se manejan en el ejercicio. Entonces podemos tener una interface IJobLogger como se muestra a continuación, en la cual se exponga el método que permitirá la diferentes implementaciones para los diferentes sources definidos.



1. Se deben definir también las tres clases referentes a los tipos de logs que se presentan en el ejercicio que implementen la interfaz anteriormente descrita.
2. Se implementara una clase que permita realizar el seteo de los log sources (Db,File,Console) y de los tipos, asi como realizar explícitamente el Log dependiendo de los settings en sí.