# Introduction

## Extension Methods

Extension methods allow adding methods to existing types without creating a new derived type, recompiling, or otherwise modifying the original type. They are defined as static methods but are called by using instance method syntax. Their first parameter is preceded by the this modifier, and specifies which type the method operates on, and are brought into scope at the namespace level.

public static int WordCount(this string str)

{

return str.Split().Length;

}

"Hello World".WordCount();// => 2

## Instructions

In this exercise you'll be processing log-lines.

Each log line is a string formatted as follows: "[<LEVEL>]: <MESSAGE>".

There are three different log levels:

* INFO
* WARNING
* ERROR

You have several tasks, each of which will take a log line and ask you to do something with it.

### 1. Allow retrieving the string after a specific substring

Looking at the logs of the last month, you see that the test message is always located after a specific substring. As you're anticipating having to extract the test message sometime in the near future, you decide to create a helper method to help you with that.

Implement the (*static*) LogAnalysis.SubstringAfter() extension method, that takes in some string delimiter and returns the substring after the delimiter.

var log = "[INFO]: File Deleted.";

log.SubstringAfter(": "); *// => returns "File Deleted."*

### 2. Allow retrieving the string in between two substrings

On further inspection, you see that the log level is always located between square brackets ([ and ]). As you're also anticipating having to extract the log level sometime in the near future, you decide to create another helper method to help you with that.

Implement the (*static*) LogAnalysis.SubstringBetween() extension method that takes in two string delimiters, and returns the substring that lies between the two delimiters.

var log = "[INFO]: File Deleted.";

log.SubstringBetween("[", "]"); *// => returns "INFO"*

### 3. Parse message in a log

Implement the (*static*) LogAnalysis.Message() extension method to return the message contained in a log.

var log = "[ERROR]: Missing ; on line 20.";

log.Message(); *// => returns "Missing ; on line 20."*

### 4. Parse log level in a log

Implement the (*static*) LogAnalysis.LogLevel() extension method to return the log level of a log.

var log = "[ERROR]: Missing ; on line 20.";

log.LogLevel(); *// => returns "ERROR"*