



UWP



Choose a Page



WINDOWS > GUIDES

How do I implement custom logging?

The Catalog app has a complete code example demonstrating [custom logging](#).

This article shows how to implement logging to the Visual Studio output window.

Implement a class derived from `ICustomLogger`, overriding the various log methods for each log level (`Warn`, `Error`, etc.):

```
1 class CustomLogger : ICustomLogger
2 {
3     private static void Log(string tag, string message)
4     {
5         System.Diagnostics.Debug.WriteLine($"{tag}: {message}");
6     }
7
8     public void Critical(string tag, string message)
9     {
10        Log(tag, message);
11    }
12
13    public void Error(string tag, string message)
14    {
15        Log(tag, message);
16    }
17
18    public void Warn(string tag, string message)
19    {
20        Log(tag, message);
21    }
22
23    public void Info(string tag, string message)
24    {
25        Log(tag, message);
```



ASK AI

```

26     }
27
28     public void Debug(string tag, string message)
29     {
30         Log(tag, message);
31     }
32
33     public void Trace(string tag, string message)
34     {
35         Log(tag, message);
36     }
37 }

```

Then, assign an instance of it to the Nutrient `Logger` :

```

1  // Set up your own custom logger.
2  var customLogger = new CustomLogger();
3
4  // Set it in the Nutrient logger.
5  var logger = Logger.Instance;
6  logger.CustomLogger = customLogger;
7
8  // Choose your own log level. Debug is useful for sending reports to us.
9  logger.CurrentLevel = Level.Debug;
10
11 // You can use the logger for your own logs too.
12 logger.Info("MYAPP", "Created customer logger and set log level to Debug");

```

Consider logging your application to a file that users can send to you. This typically saves our customers a lot of time when diagnosing issues.

Was this helpful?

✓ YES

✗ NO

Questions? [Contact us](#)