C# Tracing intro

Logging

Tracing / logging information

- Instead of using Console.WriteLine use tracing / logging for released Systems.
- You can setup the log to write to:
  - The Console
  - A File, in different formats
  - Windows Notification

- The Tracing can have several output channels
- The Trace level can be changed (actual at runtime)

How to Choose Output Chanel

- The Trace class can write to “TraceListener”
- The ”TraceListener” is an abstract class
  i.e. you need concrete TraceListener class.
- C# have some buildt in classes like:
  - TextWriterTraceListener
  - XmlWriterTraceListener
  - EventLogTraceListener
- They work like observers i.e. you can add them to theTrace class like:

```
Trace.Listeners.Add(objOfTraceListener);
```

Make your own TraceListener class

- You can design and implement you own Listener by Inherits from TraceListener and override:
  - **public override void Write(string message)**
  - **public override void WriteLine(string message)**

Trace Level

- Trace works with diff. Levels
  - Off
  - Verbose
  - Info
  - Warning
  - Error

- But in real life only ‘info’,’warning’, ‘error’ exists
  - Trace.TraceInformation(-- string -)
  - Trace.TraceWarning(-- string -)
  - Trace.TraceError(-- string -)

Trace Example

```
TraceListener tl = new TextWriterTraceListener(Console.Out)
        { Filter = new EventTypeFilter(SourceLevels.Warning) };

Trace.Listener.Add(tl);

Trace.TraceInformation("This is info");    // not printet
Trace.TraceWarning("This is warning");   // is printet
Trace.TraceError("This is error");          // is printet
```