



Logging

## Übung 9

# Serilog

- <https://github.com/serilog/serilog/wiki/Structured-Data>
- Wenn wir ein Object Foo serialisieren wollen:

```
Log.Error("{@Foo}", foo);
```

# Setup

- 14 Tage Test-Account anlegen: [logit.io](https://logit.io)
- Projekt ELK.Core
- Folgende NuGet Pakete sind relevant:
  - Microsoft.Extensions.Configuration.Json
  - Serilog
  - Serilog.Settings.Configuration
  - Serilog.Sinks.Network

# appsetting.json hinzufügen

- Über Projekt > Hinzufügen > Neues Element

Neues Element hinzufügen - ELK.Core

The screenshot shows the 'Add New Item' dialog in Visual Studio. The left sidebar shows a tree view with 'Visual C#-Elemente' expanded. The main area displays a list of items to add, sorted by 'Standard'. The 'JavaScript-JSON-Konfigurationsdatei' item is selected and highlighted in blue. The right pane, titled 'konfig', shows the details for the selected item.

Icon	Name	Category
	Anwendungskonfigurationsdatei	Visual C#-Elemente
	JavaScript-JSON-Konfigurationsdatei	Visual C#-Elemente
	Regelsatz für Codeanalyse	Visual C#-Elemente
	TypeScript-JSON-Konfigurationsdatei	Visual C#-Elemente

**konfig**

**Typ:** Visual C#-Elemente  
JSON-Konfigurationsdatei für JavaScript Language Service

# Neuen Stack anlegen

The screenshot shows the logit.io dashboard. At the top, a green banner indicates "You have 14 days left in your trial" with a yellow "Subscribe" button. The left sidebar contains the logit.io logo, a user profile with ID "5c3cf2a8", and navigation links for "New Account", "Account Settings", "Support", and "What's new?". The main content area is titled "Stacks" and features a central graphic of three stacks of server racks. Below the graphic, the text reads "You haven't created a Stack yet!" followed by "Start sending and monitoring your data within minutes. Build your first Stack!". A yellow "Create Stack" button is positioned at the bottom of this section.

# Konfiguration extrahieren

- <https://logit.io/sources/configure/dotnetcore>
- Konfiguration kopieren
- Mit „Launch Kibana“ im Cockpit kommt man zu den Logs

# Setup

```
var configuration = new ConfigurationBuilder()  
    .AddJsonFile("appsettings.json")  
    .Build();  
  
Log.Logger = new LoggerConfiguration()  
    .ReadFrom.Configuration(configuration)  
    .CreateLogger();
```

# Aufgabe 1

- Erstellen von 2 Logg-Objekten:
  - CustomerAddressChanged
    - FirstName: string
    - LastName: string
    - Address
      - Street: string
      - City: string
  - UserLoginFailed
    - UserId: int
    - Reason: string
- Objekte mit Zufallswerten befüllen (Random Strings)
- Objekte als JSON serialisieren
- Jeweils 10 Log-Einträge



# Aufgabe 1

- Was beobachtet man in Kibana? („Launch Kibana“ im Stack-Menü)
  - Kann Kibana das Objekt serialisieren?
- Konkrete Aufgabe:
  - Suchen nach `UserLoginFailed` (type) nach einer spezielle `UserId` in Kabana
  - Kopieren Sie das Query + Result nach Fragen.txt

Antworten in Fragen.txt