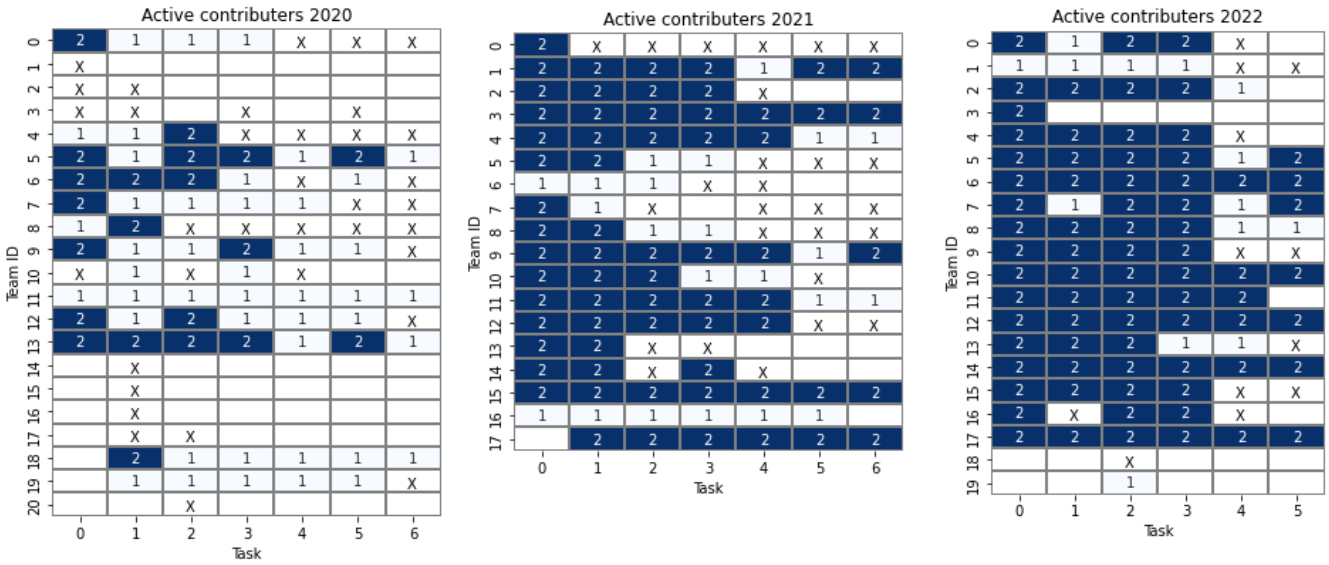
Anwendungsbeispiele

Parameter	Kursinformationen
Veranstaltung:	`Vorlesung Softwareentwicklung`
Teil:	`27/27`
Semester	`Sommersemester 2023`
Hochschule:	`Technische Universität Freiberg`
Inhalte:	Zusammenfassung und Ausblick, Anwendung von C# in der Godot Engine
Link auf den GitHub:	https://github.com/TUBAF-IfI-LiaScript/VL_Softwareentwicklung/blob/master/27_Anwendungen.md
Autoren	Sebastian Zug, Galina Rudolf & André Dietrich

Auswertung der Gitaktivitäten



Vergleich der studentischen Teamaktivitäten in Git 2020-2022

Secrets

Wie gehen wir mit Schlüsseln, Passwörtern usw. in unseren Codes um?

Zielstellung: + Komfortable Handhabung im Projekt + Projektübergreifende Verwendung (?) + Speicherung ohne Weiterleitung an Repositories

Ein Lösungsansatz ist die Verwendung von Microsoft.Extensions.Configuration.UserSecrets

```
dotnet new console -o secret_example
dotnet add package Microsoft.Extensions.Configuration.UserSecrets
dotnet user-secrets init
dotnet user-secrets set "ServiceAPIKey" "1213234435"
```

Das war es schon. Nun finden Sie unter

- `~/.microsoft/usersecrets/<user_secrets_id>/secrets.json` (Linux/macOS)
- `%APPDATA%\Microsoft\UserSecrets\<user_secrets_id>\secrets.json` (Windows)

den Eintrag

```
{
  "ServiceAPIKey": "1213234435"
}
```

Aus dem Programm heraus können Sie darauf unmittelbar zurückgreifen.

```
using Microsoft.Extensions.Configuration;

var config = new ConfigurationBuilder().AddUserSecrets<Program>().Build();
```

```
string APIsecret = config["ServiceAPIKey"];

Console.WriteLine(APIsecret);
```

Anwendungsbeispiel

Lassen Sie die Inhalte der Lehrveranstaltung anhand eines Codereviews Revue passieren lassen.

https://cobwebsonmymind.wordpress.com/2011/04/13/thingspeak-net-class/

Wir fokussieren uns auf zwei Methoden für das grundsätzliche Schreiben eines Wertes auf den Server.

Aufgabe: Bewerten Sie den Code im Hinblick auf:

- Verwendbarkeit des Beispiels
- Entwurfsqualität
- Implementierung

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Text;
using System.Net;
using System.IO;

namespace ThingSpeak
{
    public class ThingSpeak
    {
        private const string _url = "http://api.thingspeak.com/";
        private const string _APIKey = "YOUR_KEY_HERE";

        public static Boolean SendDataToThingSpeak(string field1, string f
          string field3, string field4, string field5, string field6, strin
          field7, string field8, out Int16 TSResponse)
        {
            StringBuilder sbQS = new StringBuilder();

            // Build the querystring
            sbQS.Append(_url + "update?key=" + _APIKey);
```

```
            if (field1 != null) sbQS.Append("&field1=" + HttpUtility.UrlEnc
              (field1));
            if (field2 != null) sbQS.Append("&field2=" + HttpUtility.UrlEnc
              (field2));
            if (field3 != null) sbQS.Append("&field3=" + HttpUtility.UrlEnc
              (field3));
            if (field4 != null) sbQS.Append("&field4=" + HttpUtility.UrlEnc
              (field4));
            if (field5 != null) sbQS.Append("&field5=" + HttpUtility.UrlEnc
              (field5));
            if (field6 != null) sbQS.Append("&field6=" + HttpUtility.UrlEnc
              (field6));
            if (field7 != null) sbQS.Append("&field7=" + HttpUtility.UrlEnc
              (field7));
            if (field8 != null) sbQS.Append("&field8=" + HttpUtility.UrlEnc
              (field8));
            // The response will be a "0" if there is an error or the entry
              > 0
            TSResponse = Convert.ToInt16(PostToThingSpeak(sbQS.ToString()))
            if (TSResponse > 0)
            {
                return true;
            }
            else
            {
                return false;
            }

        }

        private static string PostToThingSpeak(string QueryString)
        {
            StringBuilder sbResponse = new StringBuilder();
            byte[] buf = new byte[8192];

            // Hit the URL with the querystring and put the response in
              webResponse
            HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create
              (QueryString);
            HttpWebResponse webResponse = (HttpWebResponse)myRequest.GetRes
              ();
            try
            {
                Stream myResponse = webResponse.GetResponseStream();
                int count = 0;
                // Read the response buffer and return
                do
                {
                    count = myResponse.Read(buf, 0, buf.Length);
                    if (count != 0)
                    {
```

```
                                sbResponse.Append(Encoding.ASCII.GetString(buf, 0,
                                  ));
                            }
                        }
                        while (count > 0);
                        return sbResponse.ToString();
                    }
                    catch (WebException ex)
                    {
                        return "0";
                    }

                }
            }
        }
```

Ablauf eines Schreibprozesses:

1. Initiierung:
    - der Nutzer spezifiziert die Kanalkonfiguration und den Kanalnamen
2. Laufzeit:
    - Schreiben der Werte
    - Versenden
    - Evaluation des Erfolgs und "Markierung" der bereits versandten Daten

Resumee

Woche	Tag	Inhalt der Vorlesung
1	4. April	Organisation, Einführung von GitHub und LiaScript
	8. April	Softwareentwicklung als Prozess
2	11. April	Konzepte von Dotnet und C#
	15. April	*Karfreitag*
3	18. April	*Ostermontag*
	22. April	Elemente der Sprache C# (Datentypen)
4	25. April	Elemente der Sprache C# (Forts. Datentypen)
	29. April	Elemente der Sprache C# (Ein-/Ausgaben)
5	2. Mai	Programmfluss und Funktionen
	6. Mai	Strukturen / Konzepte der OOP
6	9. Mai	Säulen Objektorientierter Programmierung
	13. Mai	Klassenelemente in C# / Vererbung
7	16. Mai	Klassenelemente in C# / Vererbung
	20. Mai	Versionsmanagement im Softwareentwicklungsprozess
8	23. Mai	UML Konzepte
	27. Mai	UML Diagrammtypen
9	30. Mai	UML Anwendungsbeispiel
	3. Juni	Testen
10	6. Juni	*Pfingstmontag*
	10. Juni	Dokumentation und Build Toolchains

11	13. Juni	Continuous Integration in GitHub
	17. Juni	Generics
12	20. Juni	Container
	24. Juni	Delegaten
13	27. Juni	Events
	1. Juli	Threadkonzepte in C#
14	4. Juli	Taskmodell
	8. Juli	Language Integrated Query
15	11. Juli	Design Pattern
	15. Juli	Anwendungungsfälle

> **Frage**: Und was kann ich jetzt damit anfangen?

> **Und was kann ich jetzt damit anstellen?**

Siehe Mini-Godot Projekt im Projektordner 🙂

Evaluation der Lehrveranstaltung

> **Danke für Ihr Interesse! Viel Erfolg bei den Prüfungen**