Praktische Projektarbeit **Fachartikel**

Programmierparadigmen:
Prozedurale Programmierung vs. Objektorientierte Programmierung

Modulnummer: *DPL4000 0320*

Modulname: *Fachartikel*

Abgabedatum: *17.09.2020*

Abschluss: *Webdesign & Development Diploma*

Semester: *September 2020*

Name: *Peter Wagner*

Campus: *SAE Institute Zürich*

Land: *Schweiz*

Wortanzahl: *2167*

Hiermit bestätige ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken (dazu zählen auch Internetquellen) entnommen sind, wurden unter Angabe der Quelle kenntlich gemacht.

Winterthur, 07.09.2020

Ort, Datum

Unterschrift Student

Vorwort

KAMPF DER PARADIGMEN

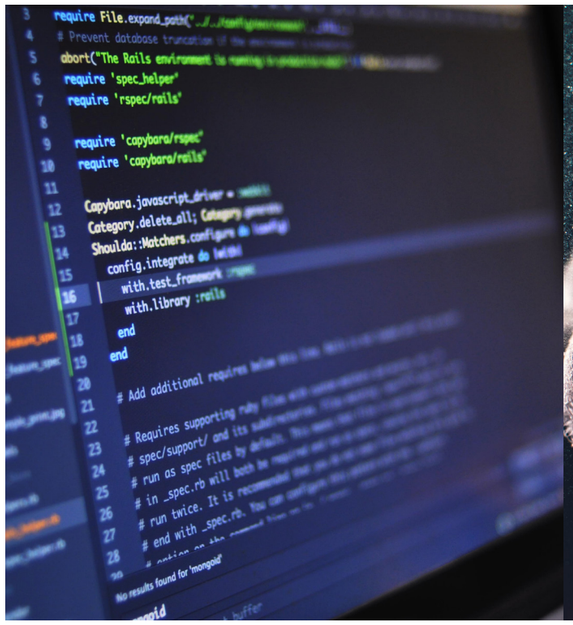

Abb. 01: Bildschirm mit Codeeditor. Datum: 04.08.17
(Foto: Luis Gomes, pexels.com)

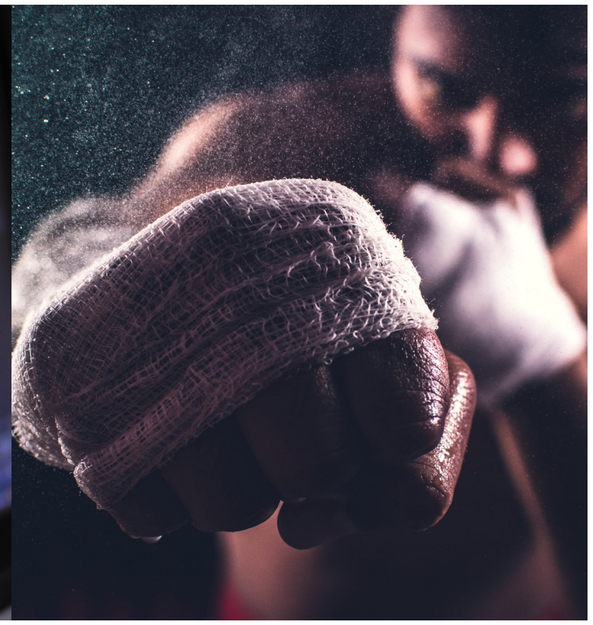

Abb. 02: Boxer ohne Handschuhe. Datum: 12.01.20
(Foto: Ashutosh Sonwani, pexels.com)

Die virtuellen Begleiter
Heutzutage sind Computerprogramme oder Apps fester Bestandteil in unserem Alltag. Wir benutzen sie, um an Informationen zu kommen, einzukaufen, Filme zu schauen oder einfach um uns zu amüsieren. Doch was steckt hinter den ganzen Apps. Wie funktionieren sie? Wie sind sie aufgebaut?

Der richtige Stil
Hier kommen die verschiedenen Programmiersprachen ins Spiel. Jede von ihnen hat ihre Eigenarten und ihren Programmierstil. Diesen Stil nennt man Programmierparadigma *(vgl. Stroustrup 2006)*.
Dieser Artikel befasst sich mit zwei sehr verbreiteten Paradigmen. Der prozeduralen - und der objektorientierten Programmierung.
Aber wo liegt der Unterschied der beiden „Stile“? Wann macht es Sinn, den einen über den anderen zu stellen? Was sind die Vor- und Nachteile der beiden? Womit wird in der Industrie gearbeitet?

Sinn und Zweck
Der Artikel soll dem Leser ein gutes Verständnis über die beiden Paradigmen vermitteln. Jeder „Stil“ wird anhand von Beispielen verständlich erklärt, um genau aufzuzeigen, wo die Stärken und Schwächen liegen. Es werden Programmiersprachen verglichen und deren Einsatzorte aufgezeigt.

Skilllevel
Der Artikel soll für alle Programmierer und Technikbegeisterte sein. Programmieranfänger werden hier viele neue, interessante Erkenntnisse sammeln. Aber auch Fortgeschrittene, können vom Artikel sicherlich profitieren.

Von Peter Wagner

Abb. 03: Peter Wagner. Datum: um 2020
(Foto: Peter Wagner)

Über den Autor

Peter Wagner, geb. 1986 in Winterthur ZH, ist Web-Entwickler und begeisterter Programmierer.
Seit 2019 studiert Peter am SAE Institut Zürich „Web-Design & Development“.
Für Ihn zählen nicht nur die äußeren (Frontend)- sondern auch die inneren Werte (Backend).

Inhaltsverzeichnis

Die Kontrahenten werden vorgestellt

WORIN LIEGT EIGENTLICH DER UNTERSCHIED ZWISCHEN PROZEDURALER UND OBJEKTORIENTIERTER PROGRAMMIERUNG

```
//fires the appear event when appropriate
var check = function() {

    //is the element hidden?
    if (!t.is(':visible')) {

        //it became hidden
        t.appeared = false;
        return;
    }

    //is the element inside the visible window?
    var a = w.scrollLeft();
    var b = w.scrollTop();
    var o = t.offset();
    var x = o.left;
    var y = o.top;

    var ax = settings.accX;
    var ay = settings.accY;
    var th = t.height();
    var wh = w.height();
    var tw = t.width();
    var ww = w.width();

    if (y + th + ay >= b &&
        y <= b + wh + ay &&
        x + tw + ax >= a &&
        x <= a + ww + ax) {

            //trigger the custom event
            if (!t.appeared) t.trigger('appear', sett

        } else {

            //it scrolled out of view
            t.appeared = false;
        }
    };

    //create a modified fn with some additional logi
    var modifiedFn = function() {

        //mark the element as visible
        t.appeared = true;

        //is this supposed to happen only once?
        if (settings.one) {

            //remove the check
            w.unbind('scroll', check);
            var i = $.inArray(check, $.fn.appear.che
            if (i >= 0) $.fn.appear.checks.splice(i,

        }

        //trigger the original fn
        fn.apply(this, arguments);

    };

    //bind the modified fn to the element
    if (settings.one) t.one('appear', settings.data,
                                settings.data, modifiedFn);
```

Abb. 04: Prozedurales Programm. Datum: 24.02.19 (Foto: Markus Spiske, unsplash.com)

In der roten Ecke: Prozedurale Programmierung
Laut *ITWissen.info (2020: https://www.itwissen.info/prozedurale-programmierung-procedural-programming.html)*, wird prozedurale Programmierung folgendermaßen beschrieben:

> ***„Die prozedurale Programmierung ist eine Art der strukturierten Programmierung. Mit ihr wird eine Gesamtaufgabe, [...], in kleinere Teilaufgaben unterteilt. Jede Teilaufgabe für sich ist einfacher zu beschreiben, programmieren und testen. Außerdem kann der entstehende Programmcode in anderen Programmen wieder verwendet werden, ein sehr wichtiger Aspekt in der Softwaretechnik.“***

Left hook....right hook...
In einem Beispiel könnte dies so aussehen: Nehmen wir an, wir haben einen Boxer, der kurz vor einem Kampf steht. In seiner Kabine wärmt er sich zuerst einwenig auf. Hier startet eine sequenzielle Abfolge von Prozeduren:

- Die erste Prozedur oder auch Funktion genannt, ist das **Aufwärmen**. Hier werden verschiedene Schlag- und Bewegungsabläufe trainiert.
- Die nächste Funktion sorgt dafür, dass der Boxer **zum Ring geht**.
- Im Ring angekommen, werden die beiden Kontrahenten vorgestellt und die Funktion **Begrüßung** wird gestartet.
- Die Ringglocke erklingt und der **Kampf beginnt.**
- Da während dem Kampf alle Schlag- und Bewegungsabläufe von **Aufwärmen** ver wendet werden, wird diese Funktion hier erneut eingesetzt.
- Der Kampf ist zu ende und es wird geprüft, ob der Boxer **gewonnen oder verloren** hat.
- Hat der Boxer gewonnen, so kann eine leicht abgeänderte Funktion von **Begrüßung** wiederverwendet werden.
- Sollte der Boxer aber verloren haben, so wird die Funktion **Enttäuschung** gestartet.

Wie im eben genannten Beispiel beschrieben, wird das Programm in Prozeduren aufgebaut. Einige Prozeduren können wiederverwendet werden, um den Aufbau zu vereinfachen und das Programm übersichtlicher zu machen. Eine Funktion kann folgendermaßen aussehen *(siehe Abb. 05)*.

```
1
2
3   let moveOne = "Right uppercut";
4   let moveTwo = "Jab";
5   let moveThree = "Backstep";
6
7   function fightCombo(move){
8       if(move === moveTwo){
9           moveThree;
10      }else{
11          moveOne;
12          console.log("KNOCK OUT!!!")
13      }
14  };
15
16  fightCombo(moveOne);
```

Abb. 05: Beispiel einer Funktion in Javascript

...und in der blauen Ecke: Objektorientierte Programmierung
Die objektorientierte Programmierung oder auch OOP genannt, ist an das menschliche Denkverhalten angelehnt. Für uns ist alles ein Objekt und so soll dieses Programmierparadigma auch funktionierten. Alles in der objektorientierten Programmierung wird durch „Objekte“ beschrieben. Wenn wir jetzt wieder auf unser Boxer-Beispiel zurück gehen, können wir folgende Objekte erstellen:

- Boxer-Objekt
- Ring-Objekt
- Arena-Objekt
- Kessel-Objekt
- Hocker-Objekt
- Ringglocke-Objekt
- usw.

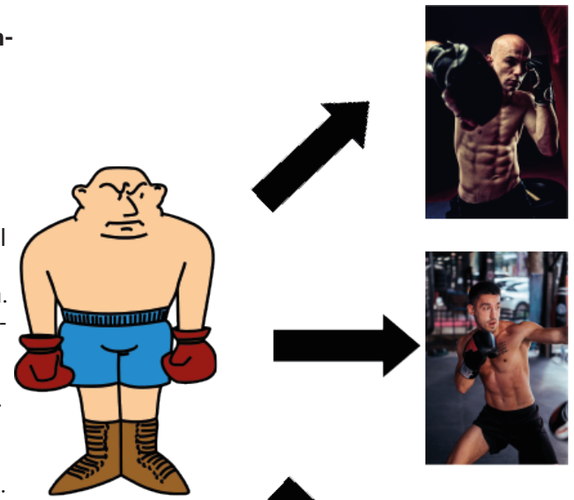

Abb. 06: OOP anhand von Boxer

So wird alles getrennt von einander programmiert.

Die Klasse Boxer
Wie im Youtube-Video von *Programmieren Starten (2018): Was ist objektorientierte Programmierung? [Youtube]* beschrieben:
„Jedes Objekt wird durch Eigenschaften beschrieben.“
Somit ist jedes Objekt einzigartig. Zusätzlich zu den Eigenschaften, können Objekte auch Funktionen enthalten. In der objektorientierten Programmierung, nennt man diese Funktionen allerdings Methoden.
Wenn wir nun mehrere Objekte vom gleichen Typ erstellen wollen, können wir eine sogenannte Klasse erstellen. Eine Klasse kann man sich als eine Art Bauplan vorstellen, aus dem wir verschiedene Objekte erstellen können.
„Sie geben die Eigenschaften und Methoden (Funktionen) vor, die ein Objekt hat.“ *(2018): Was ist objektorientierte Programmierung? [Youtube]*

Wenn wir nun eine Klasse Boxer erstellen, werden hier die Eigenschaften und Methoden festgelegt. Dies könnte so aussehen:

Eigenschaften:

- Hautfarbe
- Haare
- Farbe Boxhandschuhe
- Farbe Shorts
- Farbe Schuhe
- Gewicht

Methoden:

- leftHook()
- rightHook()
- uppercut()
- bodyShot()

Aus dieser Klasse, können wir nun verschiedene Boxer erstellen, welche diese Eigenschaften und Methoden beinhalten, aber unterschiedliche Werte haben (siehe Abb. 06).

Vererbung
In der OOP gibt es die Möglichkeit, Klassen zu vererben. Das heißt, wir erstellen eine neue Klasse mit den gleichen Eigenschaften und Methoden, der vorherigen Klasse. Nun ergänzen wir diese, mit zusätzliche Eigenschaften und Methoden. Zum Beispiel, können wir eine Boxerinnen-Klasse erstellen, welche folgende Ergänzungen haben könnte:

- Farbe Top
- Titel
- taunt()

Somit haben wir eine völlig neue Klasse erstellt, und uns die Arbeit erspart, die Klasse von Grund auf zu schreiben.
Hier kann man bereits einen Vorteil der Objektorientierung sehen. Weiteres zu den Vor- und Nachteilen der beiden Paradigmen, werden im nächsten Kapitel erklärt.

In Abb. 08 wird ein Beispiel einer Klasse und einer Vererbung dargestellt. *(siehe Abb. 08)*

Abb. 07: Boxerin von der Seite. Datum: 07.04.18
(Foto: Jermaine Ulinwa, pexels.com)

```
19  class Boxer {
20      constructor(skinColor, hair, gloveColor, weight){
21          this.skinColor = skinColor;
22          this.hair = hair;
23          this.gloveColor = gloveColor;
24          this.weight = weight;
25      }
26      leftHook();
27      rightHook();
28  }
29
30
31  class Boxerin extends Boxer{
32      constructor(topColor, titel){
33          this.topColor = topColor;
34          this.titel = titel;
35      }
36      taunt();
37  }
```
Abb. 08: Beispiel einer Klasse und einer Vererbung in Javascript

Auf wen soll ich setzen?

WAS SIND DIE VOR- UND NACHTEILE? IST EIN PARADIGMA DEM ANDEREN ÜBERLEGEN?

Als erstes ist zu sagen, dass hier, wie auch in anderen Bereichen des Lebens, alles seine Vor- und Nachteile hat. Es ist immer eine Sache der Sichtweise. Einige bevorzugen die prozedurale Programmierung, andere wiederum die objektorientierte Programmierung. Hier werden einige Punkte beschrieben und erläutert, um die Stärken und Schwächen im Paradigmenkampf aufzuzeigen.

Objektorientierte Programmierung holt zum Schlag aus
Ein grosser Vorteil bei den objektorientierten Sprachen ist, dass man mit ihnen auch prozedural programmieren kann. Anders ist es bei den prozeduralen Sprachen. Hier sind nicht alle auf Objektorientierung ausgerichtet.
Ein weiterer Pluspunkt der OOP ist ihre Flexibilität.

> ***„Durch die Aufgliederung des Programms in Objekte ist es einfacher, Änderungen an der Programmierung vorzunehmen. Oft ist es ausreichend, nur ein Objekt abzuändern, um Auswirkungen auf das gesamte Programm zu erzielen. “***

henssinger.com(http://henssinger.com/diplom/html/s32.html).

Die Fehleranfälligkeit ist geringer, da die Struktur übersichtlich und somit schnell durchleuchtet werden kann.

Prozedurale Programmierung liegt am Boden und wird angezählt....1....2....3....4
Durch den einheitlichen Aufbau, wird der Programmierablauf von anderen Entwicklern,welche am gleichen Programm arbeiten, schnell nachvollziehbar. Somit können Ergänzungen ohne großen Aufwand vorgenommen werden.

Prozedurale Programmierung kommt zurück
In Geschäftsanwendungen kann es teilweise schwierig sein, die Logik der objektorientierten Programmierung zu implementieren. Auf *entwickler.de(2020: https://entwickler.de/online/windowsdeveloper/objektorientierung-programmierung-579859284.html),* heißt es:
„Betrachten Sie Klassen als erweiterte Datentypen. Implementieren Sie Geschäftslogik als Prozesse, nicht als Verhalten in den Datenklassen.“

Soll heißen, dass hier ein Ablauf von Prozessen oder besser gesagt Prozeduren meist mehr Sinn macht als wenn man mit Objekten arbeitet.

Eine harte Rechte von prozedurale Programmierung
Die prozedurale Programmierung ist einfacher zu lernen und vermittelt die Grundprinzipien des programmierens. Alles was beim prozeduralen programmieren angewendet wird, kann man auch anschließend beim objektorientierten programmieren anwenden. Umgekehrt geht das nicht.

Ding Ding... Erste Rund ist vorbei
Wie man sieht, haben beide Paradigmen ihre Vorzüge, aber auch Schwachstellen. Am Ende des Tages, ist die Art und der Aufbau des Projekts, welches man programmieren will, entscheidend, welches Paradigma man auswählt.

Abb. 09: Ringglocke.
(Foto: unbekannt, images.pristineauction.com)

Der passende Kämpfer

PROGRAMMIERSPRACHEN, IHRE PARADIGMEN UND EINSATZORTE

Abb. 10: Logo von C. (Foto: https://de.wikipedia.org/wiki/C_(Programmiersprache))

C ein Vorbild für die jüngeren Kontrahenten

Die Programmiersprache C ist eine prozedurale Sprache, welche 1972 durch Dennis Ritchie entwickelt wurde**.** C hat viele unterschiedliche Anwendungsbereiche. Unter anderem, wird die Sprache für die Programmierung von Betriebssystemen verwendet. Wichtig zu erwähnen ist, dass einige objektorientierte Sprachen, wie zum Beispiel C#, Go, PHP, Java und C++ sich an der ***Syntax*** von C orientieren *(vgl. wikipedia.org 2020)*.

Und von der Insel Java...

Java basiert auf dem objektorientierten Paradigma. Sie ist eine sehr verbreitete Sprache und ist vielseitig verwendbar. Sie wird Serverseitig bei großen Firmen wie zum Beispiel Facebook, LinkedIn, Twitter, Amazon, eBay, usw. eingesetzt. Client-seitig wird Java eher seltener benutzt. Das Spiel Minecraft ist hier eine Ausnahme. Es wurde vollumfänglich mit Java entwickelt *(vgl. Ullenboom 2019: 55)*.

Abb. 11: Logo von Java. (Foto: https://de.wikipedia.org/wiki/Java_(Programmiersprache))

Python der Fan-Favorit

Python ist eine, in den 90er Jahren entwickelte, objektorientierte Sprache. Sie ist eine der am häufigsten verwendeten Programmiersprachen. Die Zeitschrift *iX* beschreibt den Einsatzbereich wie folgt: ***„In innovativen Bereichen wie Data Science oder maschinellem Lernen ist die Sprache inzwischen die erste Wahl."*** *(vgl. Völkl, iX Spezial 2020: 118)*

> ***Syntax:***
> ***Ein Regelsystem zur Kombination elementarer Zeichen zu zusammengesetzten Zeichen in natürlichen oder künstlichen Zeichensystemen.(vgl. wikipedia.org 2020)***

Abb. 12: Logo von Python. (Foto: https://www.festivalclaca.cat/festvi/ibwoowm_python-programming-tutorials-language-python/)

Die Syntax von Python unterscheidet sich stark von anderen Programmiersprachen. Hier wird auf geschweifte Klammern **{}** und Semikolons **;** verzichtet. Statt dessen wird mit Einrückungen strukturiert. *(siehe Abb. 13)*

```
4
5   def gcd(a, b):
6       while (b != 0):
7           t = a
8           a = b
9           b = t % b
10
11      return a
12
13
14  print(gcd(60, 96))
15  print(gcd(20, 8))
16
```

Abb. 13: Code-Beispiel und Python

Elm der Schelm

Elm wird hauptsächlich für Webanwendungen und grafische Benutzeroberflächen, sogenannte GUIs, verwerndet. Sehr positiv an dieser prozeduralen Sprache ist, dass es keine Laufzeitfehler ***„[...] Fehler die während der Laufzeit eines Computerprogramms auftreten".****(vgl. wikipedia.org 2019)* gibt. Zudem ist sie sehr einfach zu erlernen. *(vgl. Grotz, iX Spezial 2020: 98)*

Abb. 14: Logo von Elm. (Foto: https://de.wikipedia.org/wiki/Elm_(Programmiersprache))

Und der Gewinner ist.....

WELCHES PARADIGMA WIRD DENN NUN MEHR EINGESETZT?

Der Vergleich

Wenn wir nun rein von den Statistiken ausgehen, wird schnell klar, dass die objektorientierte Programmierung klar bevorzugt wird *(siehe Abb. 15 und Abb. 16)*.
Auch in der Arbeitswelt, sind viele Stellen ausgeschrieben, welche mindestens eine objektorientierte Programmiersprache voraussetzen. Die Sprachen welche hier in der Schweiz von Firmen bevorzugt werden, sind: C#, Java, C++ und Python *(vgl. jobs.ch 2020)*.

Steht der Gewinner schon fest?

Eigentlich könnte man das so sehen. Es ist aber interessant zu erwähnen, dass JavaScript, eine zum größten Teil prozedurale Programmiersprache, zu den beliebtesten gehört *(siehe Abb. 17)*. In der Schweiz teilt sie sich zeitweise sogar den zweiten Platz mit Java *(siehe Abb. 18)*. Der Grund dieser Beliebtheit ist, dass JavaScript die erste Wahl bei Webseiten und Webanwendungen im Frontend, also das, was der Benutzer sieht, ist. Aber auch im Backend *(Serverseitig, für den Benutzer nicht sichtbar)*, gewinnt JavaScript immer mehr an Popularität.

Es gibt keinen Verlierer

Selbst mit allen Statistiken, ist es nicht immer eine Frage des Paradigmas, sondern eine Frage der Sprache und des damit zu Grunde liegenden Projekts. Wie im vorhergehenden Kapitel beschrieben, sind es die Sprachen, aber auch die Entscheidung des Programmierers, welche den Unterschied machen. Allerdings gibt es nicht „die beste Programmiersprache" oder „das beste Programmierparadigma". Vielmehr stellt sich die Frage, wie gut der Programmierer sein Handwerk versteht. Denn im Moment steckt hinter den meisten Programmen noch immer ein Mensch, der es geschrieben hat.

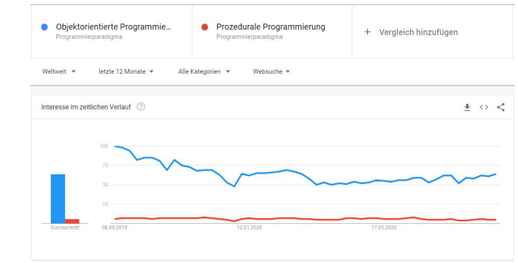

Abb. 15: Paradigmen-Vergleich weltweit (Foto: https://trends.google.de/trends/explore?q=%2Fm%2F05prj,%2Fm%2F05yd5)

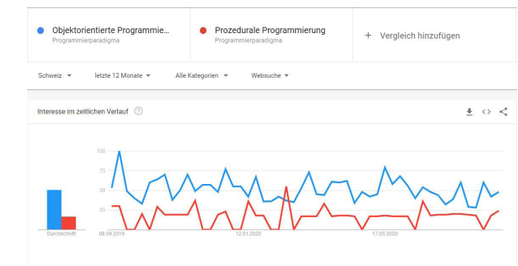

Abb. 16: Paradigmen-Vergleich Schweiz (Foto: https://trends.google.de/trends/explore?geo=CH&q=%2Fm%2F05prj,%2Fm%2F05yd5)

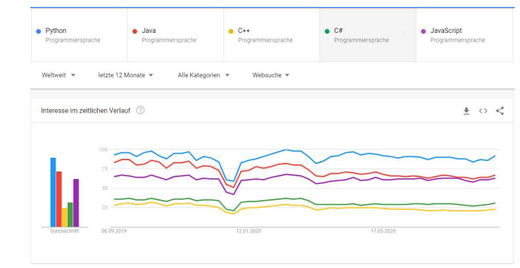

Abb. 17: Sprachen-Vergleich weltweit (Foto: https://trends.google.de/trends/explore?q=%2Fm%2F05z1_,%2Fm%2F07sbkfb,

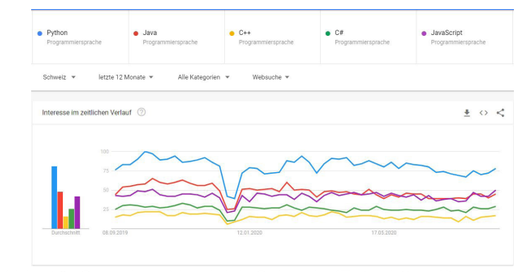

Abb. 18: Sprachen-Vergleich Schweiz (Foto: https://trends.google.de/trends/explore?geo=CH&q=%2Fm%2F05z1_,%2Fm%2F07sbkfb,%2Fm%2F0jgqg,%2Fm%2F07657k,%2Fm%2F02p97)

Fazit

DAS SCHLUSSWORT ZU DIESER ARBEIT UND MEINE PERSÖNLICHE EINSCHÄTZUNG

Schlusswort
Da das Thema Programmieren eine sehr komplexe Sache ist, war dieser Artikel auch nicht leicht zu schreiben. Es gibt bei diesem Thema sehr viele unterschiedliche Meinungen, was die Suche nach verwendbarem Material teilweise erschwerte. Trotz allem, hat mir diese Arbeit sehr geholfen, meinen Horizont zu erweitern. Durch meine Recherchen, konnte ich selbst einige neue Dinge dazu lernen. Ich bin nach dieser Arbeit noch viel mehr davon überzeugt, dass programmieren eine hohe Kunst ist. Selbst Leute, die schon seit mehreren Jahren auf diesem Gebiet arbeiten, können immer wieder was neues dazu lernen. Ein Grund dafür ist, dass fast täglich neue Technologien und Progammier-sprachen vorgestellt werden. Hier ist es wichtig, sich stehts auf dem laufenden zu halten und die Trends im Auge zu behalten.

Die Codeschnippsel
Ich bin bewusst nicht zu tief in die verschiedenen Codes eingegangen, um den Leser nicht mit Informationen zu überhäufen. Es ging mir mehr darum, einen groben Einblick zu gewähren, wie so ein Code aussehen könnte. Da die Zielgruppe für diesen Artikel aber hauptsächlich Programmierer sein werden, wird dies auch kein Problem darstellen.

Warum ein Boxwettkampf?
Die Idee hinter dieser Arbeit war, einen Fachartikel auf eine Art und Weise zu schreiben, dass selbst ein Laie sich etwas darunter vorstellen kann. Das Ganze in einen Boxwettkampf zu umhüllen, sollte das Thema einerseits verbildlichen und andererseits dem Leser auf eine spielerische Art vermittelt werden.

Die Inspiration
Inspirieren lassen, habe ich mich durch die Zeitschrift *iX* aus dem Verlag *Heise.* Die Artikel sind zweispaltig geschrieben und es werden viele Bilder und Codeschnippsel gezeigt. Auf die Idee mit dem Boxkampf kam ich, als ich das Magazin *iX Developer Winter 2019 / 20* gelesen habe. Dort wurden Artikel mit Bildern von Süßigkeiten bestückt. Dies brachte mich dazu, ein anspruchsvolles Thema in einen sportlichen Wettkampf einzubetten.

Literaturverzeichnis

WEITERFÜHRENDE INFORMATIONEN UND QUELLEN

INTERNET

entwickler.de (2019): Objektorientierung - Manchmal gehts auch ohne [online] *https://entwickler.de/online/windowsdeveloper/objektorientierung-programmierung-579859284.html [30.08.2020]*

Google Trends (2020): Diese Themen interessieren die Welt [online] https://trends.google.de/trends/?geo=DE

Henssinger.com, 3.2 Vorteile der objektorientierten Programmierung [online] *http://henssinger.com/diplom/html/s32.html [30.08.2020]*

ITWissen.info (2020): Prozedurale Programmierung [online] *https://www.itwissen.info/prozedurale programmierung-procedural-programming.html [09.08.2020]*

jobs.ch (2020): Arbeitssuche auf der führenden Schweizer Jobbörse [online] *https://www.jobs.ch/de/stellenangebote/?term=software%20entwickler*

Programmieren Starten (2018): Was ist Objektorientierte Programmierung? [online] *https://www.youtube.com/watch?v=2le2YYr3N7s&t=38s [09.08.2020]*

Stroustrup, **Bjarne,** Conference on History of Programming Languages: Evolving a language in and for the real world: C++ 1991-2006 [online] *https://www.stroustrup.com/hopl-almost-final.pdf [04.08.2020]*

wikipedia.org (2020): C(Programmiersprache) [online] *https://de.wikipedia.org/wiki/C_(Programmiersprache) [30.08.2020]*

wikipedia.org (2019): Laufzeitfehler [online] *https://de.wikipedia.org/wiki/Laufzeitfehler#:~:text=Laufzeitfehler%20(englisch%20runtime%20error)%20sind,nicht%20vorhersehbarem%20Verhalten%20des%20Programms. [01.09.2020]*

wikipedioa.org (2020): Syntax [online] *https://de.wikipedia.org/wiki/Syntax [05.09.2020]*

BÜCHER

Ullenboom, Christian (2019): *Java ist auch eine Insel*, 14. Auflage, Bonn: Rheinwerk Computing, Jg. 2019, S.55

MAGAZIN

Grotz, Martin (2020): *Elm-Apps als Web Components: iX Spezial,* Jg. 2020, S. 98

Völkl, Gerhard (2020): *Python: Vergangenheit und aktueller Erfolg,* in: *iX Spezial,* Jg. 2020, S. 118