



Programmierparadigmen

📅 2. Juli 2016 (<http://sebinside.de/2016/07/02/programmierparadigmen/>) 👤 seb
(<http://sebinside.de/author/seb/>) ➦ Studium (<http://sebinside.de/category/studium/>)

Programmierparadigmen ist nicht nur der Name der letzten Pflichtvorlesung eines Informatik-Studenten am KIT in Karlsruhe, sondern auch einer der spannendsten Bereiche der Programmierung. Paradigma bedeutet: *Grundsätzliche Denkensart* oder *Muster*. Und im Bereich der Programmierparadigmen sieht man deshalb auch, wie *vielfältig* heutige Programmiersprachen sind.

„Die Grenzen meiner Sprache sind die Grenzen meiner Welt“ – Ludwig Wittgenstein

Mit diesem Zitat beginnt die besagte Programmierparadigmen-Vorlesung – und bringt es damit ziemlich genau auf den Punkt. Abhängig von der gewählten Programmiersprache stehen viele Türen offen, vieles andere ist aber wiederum gar nicht erst möglich hinzuschreiben. Und das ist auch gut so: Denn angelegt an das Zitat stellt die Programmiersprache die Welt im Idealfall genau so dar, wie der

Entwickler es gerade benötigt. Das klärt an dieser Stelle auch gleich die Frage, ob es eine „perfekte“ oder die „beste“ Sprache gibt: Nein, es gibt nur die „am besten geeignete“ Sprache für einen Anwendungsfall.

Anfänger kennen meist nur Java oder C++ – aber die Welt der Programmiersprachen ist viel größer als das. Um einen Überblick zu bekommen, kann man verschiedene Konzepte einer Sprache (Also was man wie hinschreiben muss, um etwas zu bewirken) in Kategorien einteilen. Womit wir wieder beim Ausgangspunkt wären: Den Programmierparadigmen.

Im Folgenden möchte ich einige, wichtige Paradigmen kurz erklären. Die Abgrenzung ist manchmal nicht so einfach, und soll auch gar nicht das Ziel sein, denn nicht mal die Sprachen selbst halten sich daran. Außen vorlassen möchte ich allerdings die logische Programmierung. Solltet ihr jemals mit logischen Sprachen wie Prolog zu tun haben, rate ich euch: Lauft! Und zwar so schnell ihr könnt!

Imperative Programmierung

Imperative Programmierung ist das klassischste Paradigma und das, was man auch intuitiv vor Augen hat, wenn man an Programmierung denkt: Ein Befehl nach dem Anderen. Es wird also **sequentiell** beschrieben, was getan werden soll: Zum Beispiel das Einlesen einer Variablen oder deren Verarbeitung. Beispiele hierfür sind Assemblersprachen.

Der nächste Schritt ist die Anreicherung um **Kontrollstrukturen** zur Steuerung des Kontrollflusses. Dazu gehören zum Beispiel Bedingungen wie if-then-else oder Schleifen. Man spricht hier vom **strukturellen Paradigma**.

Für die **prozedurale Programmierung** fehlen jetzt noch – wie der Name schon sagt – Prozeduren, also Methoden und Funktionen. Die zugrunde liegende Idee ist die erweiterte Abstraktion des Quellcodes. Ein weiteres Ziel hiervon ist die verbesserte **Codewiederverwendung**. Anstatt sich zu wiederholen, kapselt man den Code besser in einer eigenen Methode. Prinzipien hierfür gibt es viele – genug für ein eigenen Artikel oder eher ein Buch.

Ein Beispiel für eine Sprache, die alle diese Paradigmen vereint, ist C.

```
1 // Prozedural
2 void sayHello(int choice) {
3
4     // Strukturell
5     if (choice == 0) {
6         printf("Hello, World!\n");
7     } else {
8         printf("Hi, World!\n");
9     }
10
11 }
12
13 int main()
14 {
15     //Imperativ
16     sayHello(1);
17     return 0;
18 }
```

Objektorientierte Programmierung

Kommen wir nun zum Superstar unter den Programmierparadigmen: Der **objektorientierten Programmierung**. Ihr fragt euch, wo Java und C++ bis jetzt waren? Beides sind objektorientierte Sprachen. C++ hieß ursprünglich auch „C with Classes“, also „C mit Klassen“.

Aber was sind Klassen? Die objektorientierte Denkweise ist schnell erklärt: Alles ist ein Objekt. Der Computer, das Tablet oder das Handy auf dem ihr gerade diesen Satz hier lest, ist ein Objekt. Ein Apfel, ein Schreibtisch, ein Auto... alles Objekte. Ein Objekt zeichnet sich durch zwei Dinge aus: Zunächst hat es Eigenschaften. Zum Beispiel die Farbe, Gewicht oder die Sorte eines Apfels. Hinzu kommt das Verhalten: Ein Apfel kann reifen oder vom Baum fallen (Salopp gesagt: Der Baum ruft auf einem Apfel die Funktion `falleRunter()` auf).

Sowohl Eigenschaften (auch Attribute genannt) als auch Verhalten wird in Klassen definiert. Eine Klasse ist der Prototyp eines Objektes, ein Objekt ist die konkrete Instanz einer Klasse. Mit anderen Worten: Die Klasse ist der Bauplan, z.B. eines Autos. Hier wird nur definiert, dass Autos z.B. eine Farbe und einen Preis haben. Der konkrete Preis wird erst im Objekt festgelegt.

Das Ziel der **Objektorientierung** ist eine weitere Abstraktionsebene, analog zum Verpacken von repetitiven Quellcode in Funktionen im prozeduralen Paradigma. Durch die Aufteilung von Code in Klassen und Paketen wird sowohl der Überblick als auch die Arbeit im Team gefördert.

Bekannte Sprachen dieses Paradigmas sind Java, C++, Scala, C#, VB.NET und so viele mehr...

```
1 public abstract class Pokemon {  
2  
3     public int size;  
4     public abstract void attack();  
5  
6 }  
7  
8 public class Charmander extends Pokemon {  
9  
10    @Override  
11    public void attack() { System.out.println("BURN!"); }  
12  
13 }
```

Übrigens: Objektorientierung bringt noch viel mehr mit, als nur die Kapselung von Code. Gerade im Bereich der Vererbung gibt es noch so einiges, was wiederum Bücher füllen würde 😊

Deklarative Programmierung

Deklarative Programmierung steht im krassen Gegensatz zum imperativen Paradigma. Anstatt zu beschreiben, **wie** etwas erreicht wird, also z.B. durch eine Abfolge von Befehlen, wird hier beschrieben, **was** erreicht werden soll. Es liegt also von Haus aus schon eine ganz andere Abstraktion zu Grunde.

Ein gutes Beispiel hierfür ist Datenbanksprache SQL. Eine Abfrage enthält keine Information, wie die Software den Befehl optimieren soll, was als erstes ausgewertet werden soll oder wo genau im Speicher die Informationen liegen. Stattdessen wird beschrieben, was das Ergebnis ausdrücken soll. Zum Beispiel die Namen aller Studenten mit einem Schnitt von 2,5 oder besser.

```
1 SELECT Name FROM Students WHERE Grade <= 2.5
```

Funktionale Programmierung

Kommen wir zum Schluss noch zur *funktionalen Programmierung*. Bis vor einem Jahr hätte ich auch hier noch geraten, möglichst schnell zu laufen. Aber von wegen, funktionale Programmierung ist doch ziemlich cool! 😊

Die funktionale Sichtweise ist wesentlich anders, als die imperative. Eigentlich ist es wie in der Mathematik: Alles ist eine *Funktion*. Diese Funktion kann Eingabeparameter haben und hat Rückgabeparameter. Insbesondere besteht eine Funktion aber nicht aus einer Reihe von Befehlen und kann nur die Variablen verändern, mit der sie auch aufgerufen wurde. Das hat den Vorteil, dass gefährliche Seiteneffekte per Definition ausgeschlossen sind: Wird eine Funktion mit denselben Parametern aufgerufen, gibt sie auch immer das gleiche zurück.

Okay, ich gebe zu: Das ist wirklich nicht so einfach zu verstehen. Deshalb hier mal zwei Beispiele in der Programmiersprache Haskell.

```
1 -- Alle Fibonacci-Zahlen als unendliche Liste
2 fibs = zipWith (+) (0:fibs) (0:1:fibs)
3
4 -- Test, ob jedes Listen-Element den gleichen Inhalt hat
5 isSame (x:xs) = 0 == (length $ dropWhile (==x) xs)
```

In funktionaler Programmierung kann man dasselbe ausdrücken wie in imperativer. Es gibt nur andere Einschränkungen bzw. Anforderungen, ein anderes Paradigma eben. Ein Gewinn davon ist, dass der Code wesentlich kompakter sein kann, bei gleicher oder sogar besserer Performance. Wobei das zu erörtern... wäre wieder ein eigenes Buch. Ebenfalls möchte ich hier nicht auf das Lambda-Kalkül oder Funktionen höherer Ordnung eingehen. Nur so viel: Funktionale Programmierung kann unglaublich mächtig sein. Auch wenn man schon etwas mehr Übung dafür benötigt.

Fazit

So viel zu Programmierparadigmen – einem wirklich riesigen Themengebiet. Es gibt noch viel mehr Ansätze, als ich hier besprochen habe. In jedem Fall gilt aber: Die Entwicklung ist noch lange nicht abgeschlossen und es kommen nach wie vor neue Sprachen und Konzepte auf den Markt. Ein Beispiel für eine „relativ“ neue Sprache ist Scala. Sie ist ein **Hybrid** und vereint sowohl die Objektorientierung von Java als auch den funktionalen Aspekt von Haskell. Du sagst nach dem Lesen dieses Artikels, das geht doch gar nicht? Dann freue dich auf die nächsten Posts in diesem Blog 😊

Teilen mit:

✉ E-Mail (<http://sebinside.de/2016/07/02/programmierparadigmen/?share=email&nb=1>)

Facebook 3 (<http://sebinside.de/2016/07/02/programmierparadigmen/?share=facebook&nb=1>)

🐦 Twitter (<http://sebinside.de/2016/07/02/programmierparadigmen/?share=twitter&nb=1>)

Gefällt mir:

★ Gefällt mir



5 Bloggern gefällt das.

c (<http://sebinside.de/tag/c/>) haskell (<http://sebinside.de/tag/haskell/>) java (<http://sebinside.de/tag/java/>)

paradigma (<http://sebinside.de/tag/paradigma/>) programmierung (<http://sebinside.de/tag/programmierung/>)

scala (<http://sebinside.de/tag/scala/>) sql (<http://sebinside.de/tag/sql/>)

studium (<http://sebinside.de/tag/studium/>)

17 Gedanken zu „Programmierparadigmen“

JEMAND

2. Juli 2016 um 11:50 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-42>)

Eine Frage: Bei dem SQL Beispiel werden alle Studenten mit einem Schnitt von 2,5 oder besser rausgesucht aber im Beispiel wird \geq genutzt, also größer gleich, müsste es nicht wenn es um besser geht kleiner gleich genutzt werden?

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=42#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=42#respond))

SEB (HTTP://SEBINSIDE.DE)

2. Juli 2016 um 12:04 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-43>)

ja, war genau falsch rum 😊

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=43#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=43#respond))

FABIANEBRAUN

2. Juli 2016 um 12:32 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-44>)

Seb ich denke du hast versucht es für jeden menschen so einfachb wie möglich zu erklären allerdings hab ich leider und ich betone leider mit meinen 14 jahren nur bahnhof verstanden
da dies eine informatik vorlesung am kit zusammengefasst hat.
ansonsten war das ein sehr lesenswerter artikel und sehr interessant
lg Fabian 😊

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=44#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=44#RESPOND))

**SPARXDEV ([HTTP://SPARXDEV.DE](http://sparxdev.de))**

2. Juli 2016 um 22:35 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-47>)

Ich bin auch 14 und hab schon eine eigene Website, mehrere Tools Programmiert und schon so etwas wie eine art neben Job in dem bereich 😊

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=47#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=47#RESPOND))

TOM

9. Juli 2016 um 14:12 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-48>)

Ich bin 12 und versteh nur Bahnhof hoch 2, Programmier aber auch.

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=48#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=48#RESPOND))

**CTEXXX**

7. August 2016 um 18:59 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-51>)

Geht mir genauso!

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=51#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=51#RESPOND))

CTEXXX/CRAFTBOX555

19. Juli 2016 um 11:51 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-49>)

Ach hallo Sparxdev, dich kenne ich vom Discord-Server von Halbzwillig!

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=49#RESPOND](http://sebinside.de/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=49#RESPOND))

MAX

3. August 2016 um 14:54 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-50>)

Sag, was du nicht verstanden hast, ich kann es dir (glaube ich) erzählen. 😊 (es gibt keine Dummen Fragen =))

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=50#RESPOND](http://sebinside.de/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=50#RESPOND))

OLAGINO

2. Juli 2016 um 12:55 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-45>)

Ich muss sagen, Sebastian, du hast das supergenial zusammengefasst! Vor allem lesen sich deine Artikel wie weiche Butter durch, da denkt man sich, huh, ich habe doch gerade erst angefangen zu lesen. Mehr davon. Ich finde es super, dass du hier diesen Blog aufgemacht hast.

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=45#RESPOND](http://sebinside.de/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=45#RESPOND))

**SPARXDEV ([HTTP://SPARXDEV.DE](http://sparxdev.de))**

2. Juli 2016 um 22:33 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-46>)

Interessant... 😊

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=46#RESPOND](http://sebinside.de/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=46#RESPOND))

Pingback: Von imperativ zu funktional – Ein Scala-Beispiel – SebInside

(<http://sebinside.de/2016/08/26/von-imperativ-zu-funktional-ein-scala-beispiel/>)

EMBIE27 ([HTTPS://PLUS.GOOGLE.COM/114064511144603262997](https://plus.google.com/114064511144603262997))

11. September 2016 um 16:53 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-59>)

Ich (15), rebellisch(:D): Ich lese den Artikel und sehe: „Solltet ihr jemals mit logischen Sprachen wie Prolog zu tun haben, rate ich euch: Lauft! Und zwar so schnell ihr könnt!“ Mein erster Gedanke: „Ah, klingt interessant. Direkt mal genauer anschauen.“ Jetzt lerne ich Prolog. 😊

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=59#RESPOND](http://sebinside.de/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=59#RESPOND))

MAX

25. September 2016 um 16:23 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-68>)

Als funktionale Sprache halte ich auch Erlang für erwähnenswert.

Alleine die Laufzeitumgebung (eigene scheduler und garbage collection) ist nen Blick wert.

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=68#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=68#respond))

SEB ([HTTP://SEBINSIDE.DE](http://sebinside.de))

26. September 2016 um 11:09 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-69>)

Ja, es gibt definitiv noch[^]viel mehr in die Richtung. Wirklich beschäftigt habe ich mich bis jetzt leider nur mit Haskell ^^

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=69#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=69#respond))

LEON EMMERICH ([HTTPS://WWW.FACEBOOK.COM/APP_SCOPED_USER_ID/1740047886285284/](https://www.facebook.com/app_scoped_user_id/1740047886285284/))

17. April 2017 um 19:07 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-101>)

Hallo alle zusammen,

ich bin 15 und habe mich jetzt über drei Jahre hinweg mit den typischen Einsteigersprachen C++ und Java beschäftigt. Ich finde Sprachen wie Haskell oder Scala sehr interessant, nunja, was empfiehlt ihr mir, zu „lernen“? LG

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=101#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=101#respond))

MAX

18. April 2017 um 22:39 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-102>)

Haskell und Skala sind interessant gegenüber typischen Objekt/Klassen-orientierten Programmiersprachen.

Außerdem empfehle ich immer wieder einen Blick auf Erlang zu werfen (hier in den Kommentaren jetzt zum 2. Mal), das ist einfach eine etwas andere Welt, von den Build tools bis hin zur Laufzeit selbst.

Wenn du so richtig viel Zeit hast und Lust etwas in die Tiefe zu gehen, schau dir mal assembly languages an (x86, arm).

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=102#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=102#respond))

**COERNERBROT**

28. Januar 2018 um 17:53 Uhr (<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-136>)

Warum ist das hier nicht in der Kategorie „Coding“?

ANTWORTEN ([HTTP://SEBINSIDE.DE/2016/07/02/PROGRAMMIERPARADIGMEN/?REPLYTOCOM=136#RESPOND](http://sebinside.de/2016/07/02/programmierparadigmen/?replytocom=136#respond))

KOMMENTAR VERFASSEN

Gib hier deinen Kommentar ein ...

Von imperativ zu funktional – Ein Scala-Beispiel ➤ (<http://sebinside.de/2016/08/26/von-imperativ-zu-funktional-ein-scala-beispiel/>)

SOCIAL MEDIA

(htt (htt (htt (htt
p://t p://f p://y (htt
witt aceb outu p://g
er.c ook. be.c ithu
om/ com om/ b.co
skat /seb skat m/s
e70 insid e70 ebin
2) e) 2) side)

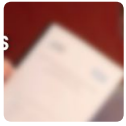
BLOG VIA E-MAIL ABONNIEREN

Gib Deine E-Mail-Adresse an, um diesen Blog zu abonnieren und Benachrichtigungen über neue Beiträge via E-Mail zu erhalten.

E-Mail-Adresse

[ABONNIEREN](#)

BELIEBTE BEITRÄGE



Rückblick auf den Bachelor (<http://sebinside.de/2017/04/13/rueckblick-auf-den-bachelor/>)
13 Apr , 2017



Programmierparadigmen
(<http://sebinside.de/2016/07/02/programmierparadigmen/>)
02 Jul , 2016



Webseiten kommen und gehen (<http://sebinside.de/2016/06/15/webseiten-kommen-und-gehen/>)
15 Jun , 2016

KATEGORIEN

📁 Allgemein (<http://sebinside.de/category/allgemein/>)

📁 Coding (<http://sebinside.de/category/coding/>)

📁 Minecraft (<http://sebinside.de/category/minecraft/>)

📁 Studium (<http://sebinside.de/category/studium/>)



NEUESTE KOMMENTARE

💬 Bowser bei Hallo. äh.. Welt! (<http://sebinside.de/2016/03/22/hallo-aeh-welt/#comment-138>)

💬 daeaevaeaed bei Rückblick auf den Bachelor (<http://sebinside.de/2017/04/13/rueckblick-auf-den-bachelor/#comment-137>)

💬 coernerbrot bei Programmierparadigmen
(<http://sebinside.de/2016/07/02/programmierparadigmen/#comment-136>)

💬 seb (<http://sebinside.de>) bei Eine neue Plugin-Architektur für Code Overflow
(<http://sebinside.de/2018/01/16/eine-neue-plugin-architektur-fuer-code-overflow/#comment-135>)

💬 Sven (<https://usacookie.de>) bei Rückblick auf den Bachelor
(<http://sebinside.de/2017/04/13/rueckblick-auf-den-bachelor/#comment-134>)

ARCHIV

Januar 2018 (<http://sebinside.de/2018/01/>)

Juni 2017 (<http://sebinside.de/2017/06/>)

Mai 2017 (<http://sebinside.de/2017/05/>)

April 2017 (<http://sebinside.de/2017/04/>)

Oktober 2016 (<http://sebinside.de/2016/10/>)

September 2016 (<http://sebinside.de/2016/09/>)

August 2016 (<http://sebinside.de/2016/08/>)

Juli 2016 (<http://sebinside.de/2016/07/>)


Juni 2016 (<http://sebinside.de/2016/06/>)

April 2016 (<http://sebinside.de/2016/04/>)


März 2016 (<http://sebinside.de/2016/03/>)

SOCIAL MEDIA



(http (http (http 
 p://t p://f p://y (http
 witt aceb outu p://g
 er.c ook. be.c ithu
 om/ com om/ b.co
 skat /seb skat m/s
 e70 insid e70 ebin
 2) e) 2) side)

FEED

 (<http://sebinside.de/feed/>) RSS - Beiträge (<http://sebinside.de/feed/>)

 (<http://sebinside.de/comments/feed/>) RSS - Kommentare (<http://sebinside.de/comments/feed/>)



Ich heiße Sebastian, studiere Informatik am KIT in Karlsruhe und betreibe den YouTube Kanal skate702 (<http://youtube.com/skate702>). Auf diesem Blog geht es um Technik, Software und Minecraft!

Impressum (<http://skate702.de/impressum/>) © 2016 skate702 – Theme von Colorlib (<http://colorlib.com/>) Powered by
 WordPress (<http://wordpress.org/>)