computer jazz and electronic sound music

Udo matthias 07626-2 999 847
mobil: 017621-605276
http://www.udomatthias.com
https://www.facebook.com/udo.matthias.3
info@udomatthias.com

If it feels good, it must be in time!!

Imperative Sprachen für die Interaktion Mensch Maschine in der elektronischen Musik.

Imperative Programmierung (lateinisch *imperare* ‚anordnen', ‚befehlen') ist ein Programmierparadigma, nach dem „ein Programm aus einer Folge von Anweisungen besteht, die vorgeben, in welcher Reihenfolge was vom Computer getan werden soll". Die imperative Programmierung ist das am längsten bekannte Programmierparadigma. Diese Vorgehensweise war, bedingt durch den Sprachumfang früherer Programmiersprachen, ehemals die klassische Art des Programmierens. Sie liegt dem Entwurf von vielen Programmiersprachen, zum Beispiel ALGOL, Fortran, Pascal, Ada, PL/I, Cobol, C und allen Assemblersprachen zugrunde.[1] **Diese Sprachen kann man vereinfacht in drei Kategorien (Klassen) einteilen.Imperative Sprachen beschreiben Berechnungen als Folge von Zustandsübergängen, die durch Schrittweise Ausführung der Befehle bewirkt werden.**

Es gibt bei Ihnen 3 Programmiermethoden (Paradigmen) und 3 Sprachmechanismen.

Prozedural	Modular	Objektorientiert
Denken in Verschachtelungen von UP, d.h. Hierarchisierung von Prozessen, welche als Prozedur und/oder Funktion abgebildet werden. Sie brauchen neue Übergabewerte für ihre Berechnungen, Veränderungen usw. Deswegen stehen Konzepte der Parameterübergabe an UP im Zentrum. Des Denkens. Irgendwann wurde aber die Organisation der Daten immer wichtiger. Ich selbst habe mit **Pascal** angefangen. Diese Sprache schult das Denken und…. Die erste Sprache ist die wichtigste.	Ist ein Programmier-paradigma. Der Ansatz sieht vor, Computer-programme systematisch in logische Teilblöcke aufzuspalten, die Module genannt werden. Praktisch alle aktuellen imperativen Programmiersprachen erlauben die Umsetzung des modularen Ansatzes. **➔Modula-2** Programme bestehen aus einer Reihe von Modulen die über genau definierte Schnittstellen miteinander kommunizieren. Dieses Paradigma ist auch als **Geheimnisprinzip** bekannt, nur die in diesen Schnittstellen definierten Größen können von außerhalb des Moduls angesprochen werden. Alles andere ist von Außen unsichtbar. Durch diese **Datenkapselung** werden unbeabsichtigte Zugriffe und damit Programmierfehler vermieden. Modula kann man auch für Regelungstechnik verwenden.	Von der modularen Denk-weise zur objekt-orientierten ist es nur ein kleiner Schritt. **Bei OOP heißen die Module Klassen.** Diese Klassen sind nur noch Schablonen für die Bildung neuer Objekte. Dazu kommt dann noch das Konzept der Vererbung. Die Eigenschaften existierender Objekte können an neue davonabgeleitete Objekte weitergegeben werden. **C, Java und Assembler** waren meine weiteren Sprachen. Java ist für mich C+/-. Da OOP und OOA – Denkweisen darstellen, kann ich in jeder Sprache objektorientiert programmieren auch in Assembler. Manches steht vielleicht nicht zur Verfügung. Aber es geht ja um die Denkweise, den Ansatz mit dem ich an ein Problem gehe. Gerade in der Feinwerktechnik und Automatisierungstechnik wichtig.

Interaktion in der elektronischen Musik, das Automatisieren in z.B. Live, MAXMSP, Puredate und/oder Chuck, SuperCollider, CSound ist prinzipiell Steuerungs- und Regelungstechnik und für Ingenieure und Informatiker Grundlagenfach, ebenso wie Mathematik und Physik.