|  |
| --- |
|  |
|  |
| **Milestone 312 Ace und Lua**Januar 2017. |

Milestone im Haus bläst Trübsinn hinaus. Noch nie war dieser Spruch so aktuell wie jetzt, wo sich Milestone auf einfache Weise von Nutzern selbst programmieren lässt. Was lässt sich damit erreichen? Wenn das entsprechende Lua-Programm gestartet ist, hat es die komplette Kontrolle über Tasten, Lautsprecher, Mikrofon und mehr. Es lassen sich also eigene Anwendungen erstellen wie das Abspielen von Play-Listen, ein Adress-Verzeichnis, Spiele und vieles mehr.

Lua-Programme sind nichts weiter als Text-Dateien mit der Endung .lua anstatt .txt. Das heisst Programme können mit jedem Text-Editor erstellt, angesehen und verändert werden. Ebenso können sie verteilt werden, per Email oder Speicherkarte.

Die Möglichkeiten von Lua auf Milestone sind mächtig. Es wurden für die Milestone-Ace Hardware Treiber entwickelt, welche Lua auf Milestone Zugriff geben auf das UKW-Radio, den Neigungs-Sensor, die rote Leuchtdiode und vieles mehr. Dadurch lässt sich beispielsweise eine einfache sprechende Wasserwaage programmieren oder mit der Möglichkeit UKW zu senden ein Babyfon realisieren. Als Schulungsbeispiel mit zusätzlicher Hardware wollen wir ein sprechendes Fleischthermometer vorstellen.

1. Was ist Programmieren?

Programmieren mag für Menschen, die ihre Lebensführung bislang davon fern hielten, abschreckend klingen. Vielleicht kommen dabei Assoziationen hoch von mysteriösen Ingenieuren in weissen Kitteln, die den ganzen Tag kaum sprechen und ständig mit einem PC beschäftigt sind. Nun, wir hoffen wir können allfällige Bedenken dass Programmieren nur etwas für einen ganz speziellen Kreis von Menschen ist, zerstreuen. Programmieren bedeutet, bei einer Maschine Hebel zu betätigen, damit diese das dafür Vorgesehene tut. Bei einem Computer-Programm sind die Möglichkeiten uferlos - was gleichermassen ein Segen ist sowie Menschen verunsichern kann. Vergleichen wir es mit dem Aufenthalt in einer Stadt, in der die Möglichkeiten sich zu bewegen unendlich vielfältig sind, die Möglichkeiten sich zu verlaufen aber ähnlich gross. Zwei Dinge sollten bei dieser Betrachtung aber nicht vergessen werden: Erstens gibt es ein eindeutiges Ziel, welches man erreichen will, und zweitens gibt es sehr viele Hilfen um das Ziel auf einfache Weise zu erreichen - sowohl als Stadtbesucher wie auch als Programmierer.

Beim Programmieren kommt sehr gewinnbringend hinzu, dass sich bereits bestehender Programm-Code wiederverwenden lässt. Dies bedeutet, dass man sich um die Tiefen der Details bei korrektem Vorgehen nur ein einziges mal kümmern muss und danach darauf aufbauen kann. Ähnlich wie einem Kleinkind das Anziehen von Kleidern erst beigebracht werden muss, kann später darauf aufgebaut werden, etwa mit dem Satz: „Zieh dich nun bitte an“. Etwas später im Leben ist das Binden von Schnürsenkeln an der Reihe, welches ebenfalls erst detailliert beigebracht werden muss. Später macht das Kind auch dies von alleine. Noch später heisst es: "Zeit für die Schule“, und das Kind erledigt darauf hin alles selbstständig - so lässt sich zumindest hoffen. Wir wollen nicht die Parallele machen, dass ein Kind programmiert wird, sondern dass auf Abläufe und Fertigkeiten nach und nach abstrakter zugegriffen wird. Man kümmert sich nicht mehr um jedes Detail sondern erledigt Arbeiten auf höherer Stufe. Dadurch wird man viel effizienter beim Programmieren und es entwickelt sich zunehmend Freude, weil gesteckte Ziele schneller erreicht werden. Ein erfahrener Programmierer steht auf der Schulter eines Riesen, weil ihm für jegliche Aufgabe ein Sack voller Hilfsmittel bereit steht, auf den er zurück greifen kann.

1. Wie lernt man Programmieren?

Programmieren wird von Mensch zu Mensch unterschiedlich erlernt. Wir würden grob zwischen zwei Vorgehensweisen unterscheiden: Es gibt Menschen, welche Anleitungen und Definitionen studieren, bevor sie praktische Erfahrungen sammeln, und es gibt Menschen welche sich anhand zahlreicher Beispiele sowie Versuch-und-Irrtum ihr inneres Bild einer Programmiersprache aufbauen. Der Autor dieses Textes gehört zur zweiten Kategorie. Dies sind typischerweise auch jene Menschen, welche nach Anschaffung eines neuen Gerätes vor Inbetriebnahme keine Gebrauchsanleitung lesen sondern gleich loslegen, und erst wenn sich die Bedienung des Geräts hoffnungslos gibt, einen Versuch mit der Anleitung starten. Da eine Programmiersprache aber komplex genug ist, dass ein kompletter Neueinsteiger im Programmieren bei einer abstrakten Erläuterung anhand Definitionen nur Bahnhof verstehen würde, raten wir Ihnen, zumindest mit einem Online-Interpreter erste Versuche zu machen, bevor Sie sich an Milestone wagen.

* 1. Illustrative Beispiele am Online-Interpreter

Was ist das nun wieder - ein Online-Interpreter? Da ist ein Lua-Interpreter auf dem Internet, anhand welchem sich einfache Lua Programme starten lassen. Ein solcher Interpreter findet sich beispielsweise hier: www.lua.org/cgi-bin/demo

Öffnen Sie obigen Link und begeben Sie sich in das Textfenster hinein. Dort könnten Sie etwa eingeben:

print('Heute ist ein guter Tag')

Danach drücken Sie unten das Kästchen welches mit „run“ beschriftet ist. Es öffnet sich eine zweite Box, in welcher die Ausgabe Ihres Programms sichtbar wird. Darin steht „Heute ist ein guter Tag“. Sie werden die überragende Leistung dieses Befehls sicher zu würdigen wissen, denn schliesslich war dies Ihr erstes Lua Programm! Herzlichen Glückwunsch dazu. Erahnen Sie, was dieses Programm als Ausgabe bringt?

print(1+2\*3)

Falls Sie mit 9 antworten sind Sie ein ebenso guter Programmierer wie jemand der mit 7 geantwortet hat. Sie müssen nur noch beachten, dass Lua die Punkt-kommt-vor-Strich-Regel beherzigt, wie es in der Schule mal gepaukt wurde. Das heisst, multipliziert und geteilt wird jeweils mit höherer Priorität als addiert und subtrahiert. Die richtige Antwort lautet 7, da 2 mal 3 gleich 6 ist, plus 1 macht 7 .

Sie könnten nun beliebig lange Texte ausgeben über Lua, aber das würde in dieser Form noch keinen Mehrwert schaffen. Gehen wir einen Schritt weiter: Schleifen. Sehr häufig verbringen Programme Zeit in sogenannten Schleifen. Dies sind Programmteile, welche mehrfach durchlaufen werden, mit jeweils kleinen Anpassungen wie etwa einer Zahl, die sich verändert. Beispiel: Sie möchten alle Zahlen von 1 bis 100 aufzählen. Dies könnten Sie am Online-Interpreter wie folgt erreichen:

for zahl=1,100,1 do

 print(zahl)

end

Toll, was? So schnell haben Sie noch nie auf 100 gezählt.

Sie werden den Sinn dieser 3 Programmierzeilen vielleicht bereits erahnen. Die erste Zeile besagt, dass für (for) den Platzhalter mit Namen zahl gezählt werden soll, und zwar von 1 bis 100, mit jeweils 1 Unterschied. Sollten Sie also gerade mal die 13er-Reihe benötigen, können Sie dies so erreichen:

for zahl=13,100,13 do

 print(zahl)

end

Dies hat zur Ausgabe 13, 26, 39, und so weiter. Bis 91. Die nächste 13er-Zahl wäre 104, aber wir sagten dem Computer ja, er solle nur bis 100 zählen.

All dies hat Sie vielleicht noch keineswegs aus der Reserve gelockt. Dann sollten wir noch das folgende Beispiel aus der täglichen Praxis ansehen. Nehmen wir an, Sie möchten mal wieder all Ihre Schallplatten Ihrer gepflegten Sammlung alphabetisch ordnen. Lua könnte hier auf folgende Weise helfen:

alben = {"Pink Floyd", "The Beatles", "Abba", "Peter Alexander", "Heintje", "Reinhard Mey" }

table.sort(alben)

for k,v in ipairs(alben) do

 print(v)

end

Es funktioniert. Sie sehen die angeführten Popstars nun in alphabetischer Ordnung auf dem Ausgabe-Schirm:

Abba

Heintje

Peter Alexander

Pink Floyd

Reinhard Mey

The Beatles

Der Befehl

table.sort(alben)

scheint die eingegebene Liste zu sortieren. Die folgende Schleife ("for k, v …") gibt die sortierten Werte dann nur noch an den Bildschirm aus. Wir wollen nicht im einzelnen die Funktion der verwendeten Befehle ergründen. Wichtiger an diesem kurzen Beispiel ist, dass Sie die Mächtigkeit erahnen, wenn auf vordefinierte Codestücke zurück gegriffen wird. Der Befehl table.sort(), der ja die eigentliche Sortierarbeit erledigt, ist nichts weiter als ein Aufruf eines bereits existierenden Programm-Stücks, welches ein anderer Programmierer irgendwo geschrieben hat und uns freundlicherweise zur Verfügung stellt. Was genau sich hinter table.sort() verbirgt kann uns egal sein; Hauptsache unsere Daten werden klaglos sortiert. Nebst etwas logischem Denkvermögen sollte ein Programmierer also auch Wissen haben über bereits vorhandene Funktionen. Nennen wir sie Bibliotheksfunktionen. Diese Bibliotheksfunktionen sind zahlreich und sie nehmen mitunter viel Platz ein in den dicken Wälzern über Computersprachen. Dabei bilden sie nichts weiter als eine schnöde Liste von Funktionen. Allerdings werden sie zumindest thematisch gruppiert. So gehört table.sort() zu den Tabellen-Funktionen, was durch den Punkt zwischen table und sort gekennzeichnet wird. "table" stellt dabei den Namen des Bibliothekmoduls dar, und "sort" will ausdrücken, dass sortiert wird. Falls Sie sich fragen, was es in dieser "table" Bibliothek sonst noch geben könnte ausser sortieren wären da:

table.getn() —- Gibt die Groesse einer Tabelle zurueck

table.setn() —- Legt die Groesse einer Tabelle fest

table.insert() —- Fuegt in eine bestehende Tabelle ein

table.remove() —- Entfernt aus einer bestehenden Tabelle

Hier offenbart sich ein weiterer schwieriger Punkt des Programmierens: Wie soll man diese knochentrockenen Funktionen im Kopf präsent halten, wenn einem im Moment des Lesens nicht im entferntesten eine praktische Anwendung einfällt, bei der man sie nutzen würde? Hier helfen Erfahrung sowie Fleissarbeit im Durchgehen von Bibliotheks-Einträgen.

* 1. Lua auf Milestone 312 Ace

Lua Programme werden auf Milestone 312 Ace ausnahmslos in der Anwendung Audio gestartet, weil dort die Möglichkeit herrscht, sich frei in Ordner-Strukturen zu bewegen. Trotzdem stehen einem laufenden Lua-Programm sämtliche Möglichkeiten der Milestone-Hardware offen, also etwa das Erfragen der Uhrzeit und Datum, Starten des Radios und anderes mehr. Nach Beenden eines Lua-Programms befindet sich Milestone an derselben Stelle in der Anwendung Audio, wie zu Beginn des Programms.

Wie starten Sie ein Lua-Programm? Bewegen Sie sich in der Anwendung Audio im internen Speicher auf den Ordner Lua und drücken Sie die Modus-Taste (Kreuztaste) um in den Ordner einzutauchen. Anschliessend können Sie mittels Links- und Rechtspfeil in diesem Ordner vorhandene Lua-Programme suchen. Diese werden angesagt mit dem Wort "Lua" vorweg. Beispiel: "Lua: Wasserwaage". Play startet das Programm.

In aller Einfachheit können Sie so vorgehen, um das Beispiel der zu sortierenden Musikalben auf Milestone zu generieren und zu starten.

1. Lua Anwendungen müssen in einem Ordner namens Lua im internen Speicher abgelegt sein. Jede Anwendung ist wiederum in einem eigenen Ordner. Legen Sie im Ordner Lua also den Ordner „Alben sortieren“ oder etwas in dieser Art an.
2. Ergänzen Sie das Programm mit einer "require" Anweisung wie folgt:

require "BonesMilestone"

Jetzt sieht das Gesamtprogramm wie folgt aus:

require "BonesMilestone"

alben = {"Pink Floyd", "The Beatles", "Abba", "Peter Alexander", "Heintje", "Reinhard Mey" }

table.sort(alben)

for k,v in ipairs(alben) do

 print(v)

end

Speichern Sie dies mit Namen main.lua im Ordner "Alben sortieren".

3. Legen Sie zusätzlich eine Datei mit Namen Manifest.lua an, mit dem folgenden Minimal-Inhalt:

name = 'Alben sortieren'

Das war es. Wenn Sie nun in der Anwendung Audio Ihres Milestone den Ordner Lua aufsuchen und mit der Modus-Taste in diesen hinein gehen, können Sie mit Links- und Rechtspfeil die Lua Anwendung "Alben sortieren" suchen. Wenn gefunden, drücken Sie Play und das Programm startet. Sie hören auf dem Lautsprecher "Abba, Heintje, Peter Alexander, Pink Floyd, Reinhard Mey, The Beatles". Danach stoppt die Ausführung automatisch und Milestone ist wieder im Leerlauf.

Falls das soeben erwähnte Beispiel nicht läuft kann dies daran liegen, dass die stets benötigten Minimalstrukturen für Lua noch nicht auf Ihrem Milestone angelegt sind. Dies behandeln wir im folgenden Unterkapitel.

* 1. Minimalstrukturen auf Milestone 312 Ace / manifest.lua

Folgende Strukturen sind auf Milestone 312 Ace im internen Speicher Ordner notwendig für Lua, in einem Ordner Lua auf Wurzel-Ebene:

* + Die allgemeinen Milestone-spezifischen Bibliotheken in einem Ordner namens "Libs". Dort drin befinden sich diverse Dateien wie etwa "serialize.lua", "str.lua", etc. sowie ein Ordner namens "BonesMilestone" mit Dateien wie init.lua, language.lua und milestone.lua. Sie brauchen zu keinem Zeitpunkt in diese Dateien hinein zu schauen aber können dies aus Neugierde natürlich tun. Ohne diese Standardstrukturen ist ein ordnungsgemässer Ablauf und die Integrität von Lua-Programmen nicht gewährleistet. Bones AG hält auf seiner Homepage eine PAK Datei bereit, welche diesen Ordner Libs mit sämtlichen Unterstrukturen jederzeit neu erzeugen kann.
	+ Pro Lua-Anwendung ein separater Programm-Ordner mit beliebigem Namen, beispielsweise "Senso Spiel",
	+ Innerhalb dieses Programm-Ordners mindestens das Programm selbst, mit Namen "main.lua",
	+ ebenfalls innerhalb des Programm-Ordners eine Datei "manifest.lua". Wie oben bereits erwähnt ist der minimale Inhalt dieser Datei der Name des Programms, beispielsweise name = 'Alben sortieren'
	+ Stellen Sie sicher, dass Sie die Firmware 5.6 oder grösser auf Ihrem Milestone 312 Ace installiert haben.

Welchen Zweck hat die Datei manifest.lua? Sie dient der korrekten Ansage des Programms in der jeweilig eingestellten Sprache von Milestone. Dies ist möglich aufgrund mehrerer Text-Einträge in unterschiedlichen Sprachen innerhalb der Datei, beispielsweise wie folgt:

{

 name = "Race Bat",

 version = "1.0.2",

 vendor = "Company Inc.",

 license = "GPL",

 desc = {

 en = "A racing game that needs heaphones on. Have Fun.",

 de = "Ein Rennsportspiel. Bitte Kopfhörer anziehen."

 }

}

Sie sehen beim Feld "desc" für Description (Beschreibung) einen Eintrag in Englisch sowie einen in Deutsch. Beim Suchen des entsprechenden Programms wird nun möglichst der sprachlich passende Ausdruck verwendet, oder ansonsten der englische. Falls kein englischer Ausdruck vorhanden ist und keine übereinstimmende Sprache gefunden wird, wird der erstbeste Ausdruck gewählt.

* 1. Fehlerbehandlung

Programmieren lernen heisst aus Fehlern lernen. Auch bei einfachen Programmen gelingt es nur selten, dass sie nach Erstprogrammierung fehlerfrei anlaufen. Bei komplexen Programmen dürfte dies unmöglich sein. Lua führt bei jedem Befehl eine Reihe von Überprüfungen durch und beendet ein Programm unverzüglich, wenn eine weitere Verarbeitung von Programm-Code nicht möglich ist. Häufige Gründe dafür sind Tippfehler in Schlüsselwörtern, beispielsweise wenn statt print() geschrieben wird: pirnt(). Auch das offen lassen von geöffneten Klammern ist unverzeihlich.

Das Programm Sortieren ist ein sehr kurzes Programmier-Beispiel, wie Text-Felder sortiert werden können. Etwas längere Beispielprogramme sind him nächsten Kapitel beschrieben.

* 1. Einige Tipps, wie man den Programmablauf beenden sollte

Erstens, geben sie dem Nutzer die Möglichkeit, sich von ihrem genialen Programm zu verabschieden. Jederzeit. Mit der Tasten-Kombination "Selektor-Record". Auf diese Tasten-Kombination sollte ihr Programm reagieren und sich gepflegt verabschieden. Nichts ist lästiger als ein Programm, welches sich partout nicht vertschüssen möchte. Sie müssen allerdings selbst dafür sorgen, dass diese Tastenkombination in ihrem Programm etwas bewirkt. Sinnvoll ist dies beim Warten auf eine Nutzereingabe oder bei einer längeren Rechnung, bei der Milestone ansonsten nicht ansprechbar wäre.

Zweitens, lassen sie Milestone nicht ins Leere laufen. Mit Lua haben sie ziemlich viel Kontrolle über Milestone. Wenn sie eine Endlos-Schleife programmieren oder unendlich lange auf eine Eingabe warten, wird Milestone sie nicht enttäuschen: Er wird diszipliniert unendlich lange warten. Bis die Batterie entladen ist und das Lua Programm abgebrochen wird. Sie sollten es nicht soweit kommen lassen und nach fünf Minuten erfolglosem Warten ihr Programm beenden. Sie könnten auch erst nachfragen, ob da noch etwas kommt? Falls ja bitte Rechtspfeil und dann wird erneut 5 Minuten gewartet, oder etwas in dieser Art.

Die etwas härtere Tour ist Selektor-Links-Play-Rechts, also 4 Tasten gleichzeitig. Dies beendet jedes Lua-Programm. Dafür haben wir gesorgt. Allerdings wird das Programm relativ jäh beendet, ohne Rücksicht auf gegenwärtige laufende Aktionen. Es ist möglich, dass "halbe Dateien" zurück blieben, falls gerade beim Schreiben in eine Datei beendet wird. Das ist allerdings ziemlich selten.

Natürlich gibt es die noch die ganz harte Tour: Drücken der Milestone Reset Kombination mit den vier Tasten rund um Play. Diese sollten Sie aber wirklich nur im Notfall benutzen, wenn gar nichts anderes mehr geht. Sie stellen ihr Radio ja auch nicht aus, indem sie die Hauptsicherung des Hauses ausschalten. Das Drücken dieser Tastenkombination tut aber genau dies. Es löst einen Hardware-Reset aus, ungeachtet dem Zustand des Betriebssystems. Abgefragt wird diese Tastenkombination rund 1000 mal pro Sekunde, direkt vom Tastatur-Treiber der Milestone Haupt-Firmware. Dieser Treiber hat sehr hohe Priorität, weshalb es ihnen wahrscheinlich auch noch nie gelungen ist, mit sehr schnellen Tastendrücken Milestone auszutricksen - er konnte ihnen immer folgen. Sollte der innere Kern des Betriebssystems abgestürzt sein würde allerdings auch diese Tastenkombination nichts mehr nützen. In so einem Fall tritt eine letzte Sicherung in Kraft: Der Hardware-Watchdog. Der erwähnte Tastatur-Treiber muss spätestens einmal pro 20 Sekunden sagen, dass er ordnungsgemäss läuft, indem er ein spezielle Kombination in eine Speicherzelle schreibt. Sollte dies ausbleiben, löst eine unbestechliche und sehr robuste Hardware-Schaltung einen Neustart des Hauptprozessors aus. Das entspricht dem Drücken einer Reset-Taste (dies es deshalb bei Milestone nie gab und nie geben wird).

1. Beispielprogramme

Folgende Beispielprogramme finden Sie auf der Website [www.bones.ch](http://www.bones.ch), falls diese noch nicht auf Ihrem Milestone 312 Ace vorliegen.

* 1. Black Jack

Spielen Sie Black Jack wie in einem Casino. Das Ziel ist, möglichst nahe an 21 Punkte heran zu kommen, wobei ein As entweder 1 oder 11 Punkte zählt, und jede Bildkarte wie King, Queen oder Jack zählt 10 Punkte. Wenn Sie mehr als 21 Punkte beisammen haben, haben Sie diese Runde verloren.

* 1. Limonade

Ein Simulations-Spiel aus der Betriebswirtschaft. Versuchen Sie möglichst viel Gewinn mit dem Verkauf von Limonade zu erwirtschaften, über einen Zeitraum von 10 Tagen. Trotzen Sie Überraschungen wie Wetter, Preiserhöhungen und Baustellen.

* 1. Finde deinen nächsten Drachen

Das Programm Finde deinen nächsten Drachen ist ein komplexeres Beispiel anhand eines kleinen Abenteuer-Spieles. Aufgabe im Spiel ist es, auf einer Insel einen Drachen zu zähmen, um mit ihm nach Hause fliegen zu können. Damit der Drache gefügig wird ist es ratsam, ihm etwas zu Fressen zu geben. Das Futter müssen Sie aber erst finden, und dazu ist Einfallsreichtum und etwas Fleiss angesagt.

Der Programm-Code ist gut strukturiert und kann als Vorlage für eigene, weitaus komplexere Abenteuer-Spiele benutzt werden. Das Grundgerüst des Spiels kann dabei beibehalten werden. Verändert oder ergänzt sollten werden: Die Räume, in denen man sich aufhalten kann, die herum liegenden Gegenstände, die Aktionen die sich mit den Gegenständen machen lassen sowie die Reaktionen darauf. Zudem sind einige Spezialitäten im Spiel eingebaut; beispielsweise ein Zufalls-Element beim Fische fangen, ein gangbarer Weg der erst bei zweitem Umschauen auffällt oder ein Wegstück, welches sich unendlich hin zieht, ohne jemals zu einem Ende zu gelangen. Diese Spezialitäten wurden eingebaut um zu zeigen dass sich die Hauptstruktur nach Bedarf ausprägen lässt. Das Spiel macht Gebrauch von Hintergrundgeräuschen, was die Spiel-Atmosphäre stark verbessert.

Sie sollten sich als Neuling in Programmiertechnik nicht als erstes mit dem Code dieses Spiels befassen, da er stellenweise hohe Komplexität aufweist.

* 1. Fleischthermometer

Das Programm Fleischthermometer arbeitet mit externer Hardware zusammen, welche Sie von Ihrem Milestone Händler beziehen können. Geliefert wird eine Fleischthermometer-Nadel vom Hersteller Electrolux sowie ein von Bones AG entwickeltes Adapterstück um die Nadel an Milestone anzuschliessen. Das Programm Fleischthermometer arbeitet mittels dem Adapterstück mit der Nadel zusammen, um Ihnen Ihren Braten fein zuzubereiten. Das Adapterstück zwischen der Nadel und Milestone hat ein elektronisches Innenleben. Der elektrische Widerstand der Nadel, welcher sich entsprechend der Temperatur ändert, wird mit einem Lua-Programm ausgewertet und daraus die momentan herrschende Braten-Temperatur errechnet. Natürlich kann auch ein Alarm-Signal aktiviert werden bei Erreichen einer vorgegebenen Temperatur.

1. Weiterführend
	1. Weiterführende Literatur

Über Lua wird sehr viel im Internet geschrieben. Wir empfehlen jedermann und jederfrau, die sich für eigene Programmerstellung interessiert, den entsprechenden Wikipedia-Artikel zu lesen: <https://de.wikipedia.org/wiki/Lua> .

Danach ist bereits Schluss mit allgemeinen Empfehlungen, da wie oben erwähnt Menschen sehr verschieden vorgehen beim Lernen neuer Dinge.

Für Programmier-Einsteiger empfehlen wir einen Lehrgang auf PC zu absolvieren, bevor auf Milestone programmiert wird. Ansonsten könnte das Frustrations-Potenzial beim Programmieren zu gross werden, da einem Milestone im Gegensatz zu einer ausgewachsenen Entwicklungsumgebung einigermassen im Dunkeln lässt, wenn sich das Programm verrannt hat und Milestone womöglich nur noch per Reset-Kombination reanimiert werden kann (die Reset-Kombination ist weiterhin das Drücken aller 4 Tasten um die mittlere Play-Taste herum). Eine Entwicklungsumgebung auf PC hat die Vorteile, dass Befehle Schritt für Schritt abgearbeitet werden können, unter Einblick in den Zustand von Variablen sowie expliziten Fehler-Rückmeldungen oder Warnungen. Eine gut ausgebaute Entwicklungs-Umgebung ist beispielsweise ZeroBrane Studio, kostenlos und auf verschiedenen Plattformen erhältlich (Windows, Mac, Linux). Beispielsweise wurde das Spiel "Finde deinen nächsten Drachen" in grossen Teilen erst auf ZeroBrane entwickelt und danach auf Milestone adapiert.

Sollten Sie ein erfahrener Programmierer in anderen Programmiersprachen sein, also wenn Sie zufälligerweise Wolfgang Hubert, Dietmar Segbert oder Toralf Schulz heissen, könnte diese Diplomarbeit der RWTH Aaachen interessant zu studieren sein:

<https://www.matse.itc.rwth-aachen.de/dienste/public/show_document.php?id=10360> .

Oder womöglich gleich der hochkomprimierte 15 Minuten Crash-Kurs über Lua, in Englischer Sprache:<http://tylerneylon.com/a/learn-lua/> .

Eine umfassende deutsche Dokumentation von Lua findet sich beispielsweise hier:

<http://lua.coders-online.net>

1. Tipps die man ausnahmsweise lesen sollte
	1. Klare Anweisungen geben

Gehen Sie beim Programmieren vom dümmsten anzunehmenden Anwender aus. Instruieren Sie eher zuviel als zuwenig. Nichts ist lästiger als ein geniales Programm, welches sich aber leider nicht nutzen lässt weil man nicht weiss wie. Vergessen Sie nicht, auch zu beschreiben wie man das Programm wieder verlässt.

* 1. Struktur

Überlegen Sie sich vor dem Programmieren, was Sie genau wollen und wie dies zu erreichen ist. Jedes nicht-triviale Problem muss stückweise in kleinere Probleme zerlegt werden, bevor es gelöst werden kann. Diese Teilstücke können ziemlich grosse Komplexität erreichen. Durch Strukturierung von Programm-Code behält man die Übersicht. Ein Beispiel: Bei einer Adresskarte sollen Einträge für Privatanschrift sowie Geschäftsanschrift verwendet werden. Man könnte nun Programm-Code für die eine Anschrift sowie Programm-Code für die andere Anschrift erstellen. Wenn später noch eine dritte Anschrift hinzu kommt, wird wiederum Programm-Code dafür erstellt, und so weiter. Oder man erstellt universellen Programm-Code, der sich für allerlei Anschriften verwenden lässt. Unsere Präferenz ist eindeutig. Universeller, strukturierter Code ist kürzer, klarer und lässt sich einfach warten.

* 1. Der 2.0 Effekt

Während dem Erstellen eines Programms müssen Sie sich in viele Abläufe und Probleme hinein denken, von denen Sie zu Beginn Ihrer Arbeit nicht mal etwas ahnten. Im Nachhinein ist man schlauer und hätte die Struktur des Gesamtprogramms in den meisten Fällen mehr oder weniger stark anders gebaut. Unser Ratschlag: Wenn Sie mit Ihrem Programm fertig sind, löschen Sie es wieder und bauen Sie es neu. Oder verstecken Sie Ihr fertiges Programm zumindest für 2 Tage und denken Sie sich neu hinein. In aller Regel erlebt man das blaue Wunder, wie rasch man ein Programm beim zweiten Wurf erstellt hat, und vor allem in welcher Einfachheit.

* 1. Famous Last Words

Sollte ein Lua-Programm auf Milestone unerwartet beendet werden müssen, im Strassengebrauch nennt man dies auch "Abstürzen", versucht der Lua-Interpreter einen Hinweis zu geben, worin das Problem beim Programm bestand. Dies wird in einer Datei "lua\_log.txt" im entsprechenden Programm-Ordner abgelegt. Die Datei wird jeweils ergänzt, anstatt neu geschrieben. Seien Sie also nicht erstaunt, wenn Sie bei einem erneuten Absturz immer auch die Fehlermeldungen vorheriger Abstürze darin finden.

* 1. Ihr geschützter Arbeitsplatz

Haben Sie keine Angst vor Fehlern! Der Grund, weshalb jedes Lua-Programm in einem eigenen Ordner stehen muss ist, dass nur in diesem Ordner Schreibrechte herrschen. Das bedeutet, Sie können ausserhalb des Programm-Ordners nichts verändern und somit auch nichts kaputt machen. Gelesen werden kann aber alles.

* 1. Alles hat ein Ende

Wenn Sie ein laufendes Lua-Programm beenden wollen halten Sie bitte diese Hierarchie ein:

* + Beenden Sie das Programm in der vom Programm-Autor vorgesehenen Weise,
	+ Sollten Sie diese Weise nicht kennen oder sollte diese Weise nicht funktionieren, versuchen Sie den Lua-Warmstopp: Selektor-Taste plus Links/Play/Rechts. Sie drücken also 4 Tasten gleichzeitig: Selektor plus der Balken der drei waagrechten Tasten. Dies hilft beispielsweise, eine Endlos-Schleife zu beenden, die aus Sicht des Interpreters korrekt läuft, aber nichts bringt.
	+ Wenn auch dies nicht möglich ist, scheint die Lage etwas verfahren. Was bleibt ist der harte Reset mit den 4 Tasten rund um Play herum. Dies startet Milestone erneut. Sie sollten dies aber erst versuchen, wenn die ersten beiden Möglichkeiten nicht gangbar sind.

Jetzt wünschen wir Ihnen viel Erfolg und Freude bei der Entdeckung von Lua auf Milestone, mit einem ungeahnten Meer von Möglichkeiten.

Bones AG, Schweiz

Stephan Knecht