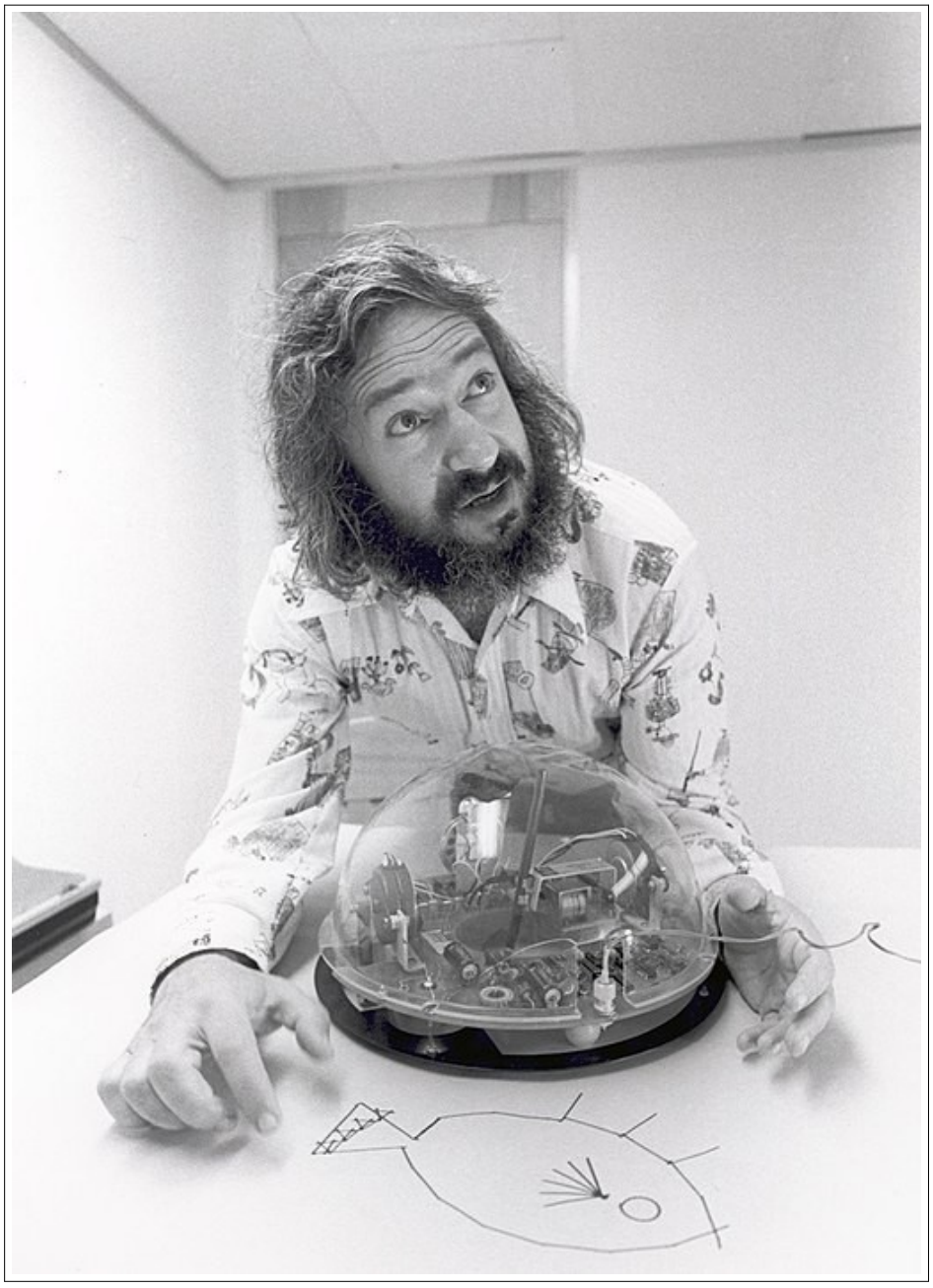
Logo und NetLogo



Seymour Papert

1928 (Südafrika) - 2016 (USA)

Studium (1950er Jahre)

- Mathematik in Cambridge/GB
- Psychologie in Genf bei Jean Piaget (bekannter Entwicklungspsychologe)

Professor für Mathematik und Erziehungswissenschaften am MIT

Wissenschaftliche Arbeit

- seit den 60er Jahren am MIT in Boston/USA
- Mitbegründer des Labors für KI
- Thema: Kinder und Computer
- Programmiersprache `Logo`
- Vater der `LEGO Mindstorms`

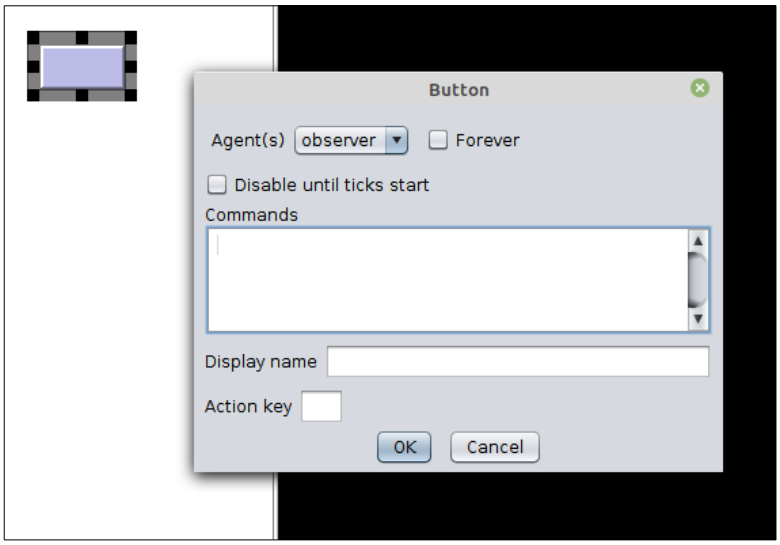
NetLogo ist eine modernere Entwicklung auf Basis von Logo.

Tutorial: Erste Schritte in NetLogo

NetLogo-Programme bestehen aus drei Teilen:

- Das **Interface** enthält die Welt aus quadratischen Feldern (patches) mit den Turtles sowie alle Steuer- und Anzeigeelemente.
- Die **Dokumentation** beschreibt Interface und Programm und gibt Hinweise zur Bedienung. Spätestens bei Veröffentlichung sollte sie vorhanden sein.
- Der **Code** enthält schließlich das eigentliche Programm.

Interface



Das Interface ist die Verbindung zwischen Mensch und Computer, es kümmert sich um die Ein- und Ausgaben. Das häufigste Interface-Element dürften Buttons sein.

- *Starte NetLogo und erstelle einen Button!*

In der Online-Version muss dazu vom `interactive` in den `authoring mode` gewechselt werden (oben links).

- *Konfiguriere den Button!*

Der Eintrag `observer` im bedeutet, dass dieser Button das Programm „von außen“ steuert und ist der Standardfall. Als Kommandos sind einzutragen

```
clear-all          ; vollständiger Reset
create-turtles 1   ; eine Turtle erzeugen
reset-ticks        ; den Tickzähler (die Uhr des Programms) starten
```

Der `Display name` des Buttons sollte `setup` sein.

- *Erstelle einen zweiten Button namens go mit den Kommandos*

```
ask turtle 0 [          ; bitte, Turtle 0:
  right 5 - random 11   ;
  forward 0.2           ;
]
tick                    ;
```

Setze den Haken `Disable until ticks start`, damit dieser Button erst nach dem `setup` benutzt werden kann.

An den Kommandos sieht man schon: NetLogo ist sehr net(t) – alle werden immer gebeten, etwas zu tun (`ask`).

- *Nutze die Buttons `setup` und `go` und ermittle, was die Kommandos in `go` bedeuten.*

Die Klammern [und] dienen dazu, mehrere Kommandos zu einem Block zusammenzufassen. Sie werden uns in jedem Programm begegnen.

Man kann in `setup` die Turtle noch ein wenig verändern. Bitte die Turtle doch um Dinge wie...

- `pen-down`
- `set size 4`
- `set shape "turtle"` (sicher findest du weitere Möglichkeiten)
- `set color green`

Auch in `go` ist noch Platz zum Experimentieren: Ändere Drehrichung, Winkelbereich und Schrittweite.

Funktionen

Die beiden Buttons für `setup` und `go` enthält fast jedes NetLogo-Programm. Für ein brauchbares Programm müsste man den `go`-Button allerdings sehr häufig drücken. Deshalb kann man in den Einstellungen der Buttons den kleinen Haken `Forever` setzen. Damit bleibt der Button bis zur nächsten Betätigung gedrückt.

- *Erzeuge einen weiteren Button `go-forever` mit forever-Haken, aber ohne Kommandos!*

Die beiden `go`-Buttons müssten jetzt dieselben Kommandos enthalten. Bei Programmänderungen erzeugt das auf Dauer unnötigen Aufwand. Deswegen ist nicht gut, Anweisungen direkt in die Buttons und damit ins Interface zu schreiben. Statt dessen sollte dort nur eine einzige Anweisung stehen, mit der eine entsprechende Funktion aufgerufen wird. Funktionen sind im Grunde nichts anderes als selbst geschriebene Kommandos.

Öffne die Eigenschaften des `setup`-Buttons und kopiere alle Anweisungen. Wechsle in den Reiter `Code` und schreibe dort eine `setup`-Funktion!

```
to setup
  hier müssen die kopierten Anweisungen hinein
end
```

In den `setup`-Button kommt nun als einzige Anweisung der Befehl `setup`. Verfahre mit dem `go`-Button entsprechend.

Jetzt steht alles Wichtige in einer einzigen Programmdatei und man muss sich nicht durch alle Buttons klicken, um das Programm zu erfassen. Außerdem könnte man die Buttons auch löschen – das Programm wäre trotzdem noch intakt[1].

→ Es ist fast immer eine schlechte Idee, mehr als ein Kommando in einen Button zu schreiben.

1 Dieses Prinzip ist dasselbe wie bei HTML und CSS: Trenne Form und Inhalt! Der Button (Form) steuert das Programm (Inhalt). Die Funktionen des Programms könnten aber genauso von anderen Buttons oder vom Anwender direkt genutzt werden. Nur so kann man das Interface ohne Änderungen am Programm nach Belieben umgestalten.