TOG: A Toggle-Oriented Paradigm for Computational Minimalism

Author: L.A.x. (Lane A. Seals)
Institutional Affiliation: Independent Researcher

Abstract

This paper introduces TOG, a new toggle-oriented programming language that makes programming simpler by using ON/OFF states instead of long and complex syntax. Unlike most languages that use many words and structures, TOG reduces everything to toggles. This makes it easier to learn, faster to prototype, and fun for creative coding. This paper explains how TOG works, shows examples, and argues why this approach is valuable.

1. Introduction

Most programming languages use complicated rules and words like if/else, while, or function. TOG asks a simpler question: what if programming was just about flipping states ON and OFF? This approach makes coding more direct, more intuitive, and easier to learn. TOG is designed to combine simplicity with expressive power.

2. Syntax of TOG

TOG uses a small set of symbols instead of long keywords:

- `?` = condition (if)
- `->` = then
- `<>` = else
- `~` = loop until OFF
- `!` = toggle the state of a variable

Example Program

Here is a short program written in TOG:

```
engine = OFF
? engine -> print("running") <> print("stopped")
engine!
? engine -> print("running") <> print("stopped")
```

3. Toggle Semantics

In TOG, variables are always in one of two states: ON or OFF. The toggle operator `!` flips the state. This makes TOG closer to real-world

switches, circuits, and systems where states matter. For example:
light = OFF
light!
? light -> print("The light is ON") <> print("The light is OFF")
4. Use Cases
1. Teaching: TOG helps beginners understand programming logic faster.
2. Prototyping: Developers can write ideas quickly without worrying about complex syntax.
3. Art and Creative Code: TOG's minimal style makes it great for experiments and generative art.
5. Conclusion
TOG reduces programming to the simple act of flipping states. This minimal design makes it powerful for teaching, prototyping, and creativity. By recognizing the power of toggles, L.A.x. has introduced a new way to think about coding: not through long Truth and False rules, but through the elegance of ON and OFF (may be better in some regards).

TOG 2.23 rev1a' expansion possibilities (8-30-2025)

//introduction of the nu keyword.

Example

Obj = nu dog(return 1);

Obj();

//returns 1

———————————————————-

//loop number

~5(print("5")

//should print number "5" 5 times

———————————————————

Url string passing api example:

www.guitarmapfree.com/A2B2A2B2|~3(song)|”yeeeehaw”!