









Along the way, we'll learn about TCP connections, HTTP headers, HTTP verbs, handling multiple connections and more.

-  Bind to a port ✓
-  Respond with 200 VERY EASY
-  Extract URL path EASY
-  Respond with body EASY
-  Read header EASY
-  Concurrent connections EASY
-  Return a file MEDIUM
-  Read request body MEDIUM

### HTTP Compression

-  Compression headers EASY
-  Multiple compression schemes MEDIUM
-  Gzip compression MEDIUM

## Build your own Interpreter →



0/48 stages












This challenge follows the book [Crafting Interpreters](#) by Robert Nystrom.

In this challenge you'll build an interpreter for [Lox](#), a simple scripting language. Along the way, you'll learn about tokenization, ASTs, tree-walk interpreters and more.

Before starting this challenge, make sure you've read the "Welcome" part of the book that contains these chapters:

- [Introduction](#) (chapter 1)
- [A Map of the Territory](#) (chapter 2)
- [The Lox Language](#) (chapter 3)

These chapters don't involve writing code, so they won't be covered in this challenge. This challenge will start from chapter 4, [Scanning](#).

 Scanning: Empty file	VERY EASY
 Scanning: Parentheses	MEDIUM
 Scanning: Braces	EASY
 Scanning: Other single-character tokens	MEDIUM
 Scanning: Lexical errors	MEDIUM
 Scanning: Assignment & equality Operators	MEDIUM
 Scanning: Negation & inequality operators	MEDIUM
 Scanning: Relational operators	MEDIUM
 Scanning: Division operator & comments	MEDIUM
 Scanning: Whitespace	MEDIUM
 Scanning: Multi-line errors	MEDIUM