

Go Workshop

Rumen Mitov

February 11, 2025

Go: A Quick Introduction

- ▶ Systems programming language
- ▶ Garbage-collected
- ▶ Large standard library
- ▶ Great developer tooling

Installation

Golang Installation

Tour of Go

Start the tour!

Workshop: Simple Go Endpoint

Run from the root directory of this repo: `go run ./src`

Further Web Examples

Go web examples

Cover at Home!

- ▶ Formatting
- ▶ Naming Convention
- ▶ Doc Comments
- ▶ Unit Tests (up to Benchmarks for now)

Organization of our Repo

Directory Structure

- ▶ **src** - for main endpoint handling / routing
- ▶ **src/utils** - common utility functions
- ▶ **sql** - sql scripts to initiate database tables / populate database with dummy data
- ▶ **assets** - media (videos, images, etc.)
- ▶ **docs** - documentation

Branch Policy

- ▶ each new feature / fix is its own branch
- ▶ commit freely to that branch
- ▶ feature must work in order to be merged with the dev branch
- ▶ developers submit pull requests to dev branch (meaningful commit message)
- ▶ backend lead is responsible for making sure dev works before merging with master
- ▶ **NOTE:** Only dev merges with master

Commit Message Prefix

This applies to final commits (i.e. commits when merging):

- ▶ **feat(<feature>):** description
- ▶ **refactor(<feature>):** description
- ▶ **fix(<feature>):** description
- ▶ **hotfix(<bug>):** description
- ▶ **bug(<bug>):** description
- ▶ **tests(<feature>):** description
- ▶ **misc:** description
- ▶ **docs:** description
- ▶ **update:** description

`feat(login): created login endpoint`

Merging

- ▶ **Squash merge** if your feature is ready to be merged upstream!
- ▶ The developer is responsible for resolving any merge conflicts!

Rebasing

- ▶ **Rebase** if you are still working on your code, but you need to pull the latest changes!