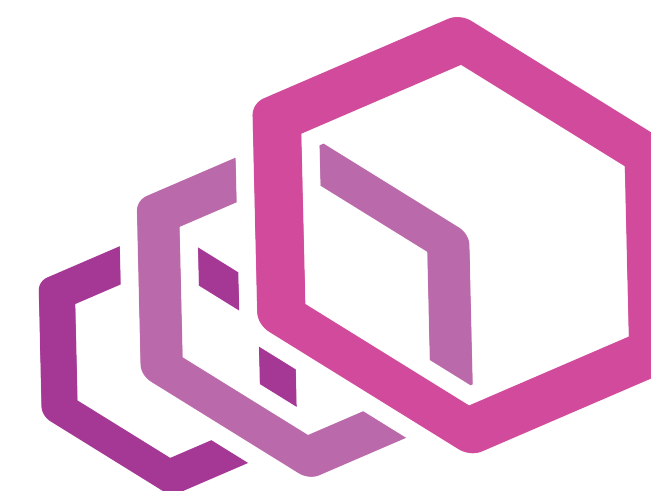


# Switch between projects like a Ninja 🥷

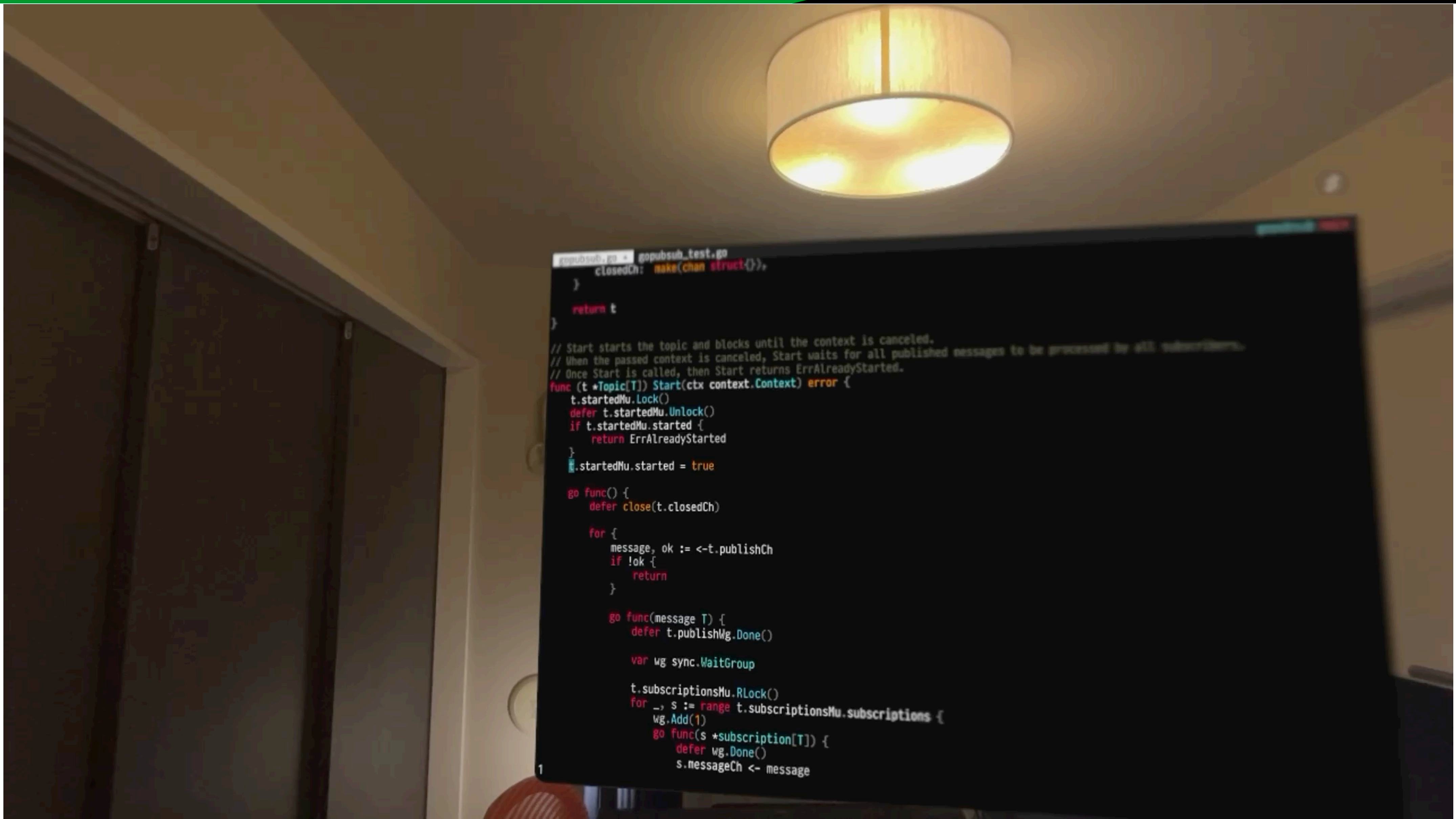


**newmo**  
Architect

**Yuki Ito**  
**@mrno110**



# Vim on Apple Vision Pro



```
gopubsub_test.go
closedCh: make(chan struct{}),
}
return t
}
// Start starts the topic and blocks until the context is canceled.
// When the passed context is canceled, Start waits for all published messages to be processed by all subscribers.
// Once Start is called, then Start returns ErrAlreadyStarted.
func (t *Topic[T]) Start(ctx context.Context) error {
    t.startedMu.Lock()
    defer t.startedMu.Unlock()
    if t.startedMu.started {
        return ErrAlreadyStarted
    }
    t.startedMu.started = true
    go func() {
        defer close(t.closedCh)
        for {
            message, ok := <-t.publishCh
            if !ok {
                return
            }
            go func(message T) {
                defer t.publishWg.Done()
                var wg sync.WaitGroup
                t.subscriptionsMu.RLock()
                for _, s := range t.subscriptionsMu.subscriptions {
                    wg.Add(1)
                    go func(s *subscription[T]) {
                        defer wg.Done()
                        s.messageCh <- message
                    }
                }
            }
        }
    }
}
```

# Agenda

- ✓ **My moves for switching between projects**
- ✓ **Dive into "Session" in Vim**
- ✓ **Advanced usage of "Session"**

# Agenda

- ✓ **My moves for switching between projects**
- ✓ **Dive into "Session" in Vim**
- ✓ **Advanced usage of "Session"**

```
func main() {
    run.Run(start)
}

func start(ctx context.Context) int {
    env := new(environment)
    if err := envconfig.Process(ctx, env); err != nil {
        fmt.Fprintf(os.Stderr, "failed to load environment variables: %s", err.Error())
        return 1
    }

    var group servergroup.Group

    group.Add(&server{})

    if err := group.Start(ctx); err != nil {
        fmt.Fprintf(os.Stderr, "failed to start or stop the server: %s", err.Error())
        return 1
    }

    return 0
}
```

# Listing Projects

```
main.go  go.mod  vimconf-demo-app main
  if err := envconfig.Process(ctx, env); err != nil {
    fmt.Fprintf(os.Stderr, "failed to load environment variables: %s", err.Error())
    return 1
  }

  var group servergroup.Group

  group.Add(&server{})

  if err := group.Start(ctx); err != nil {

```

---

```
> | < 66/66
go/src/github.com/110y/bootes
go/src/github.com/110y/echoserver
go/src/github.com/110y/glm
go/src/github.com/110y/glm.nvim
go/src/github.com/110y/go-decls
go/src/github.com/110y/go-expr-completion
go/src/github.com/110y/go-func-info
go/src/github.com/110y/go-package-info
go/src/github.com/110y/gophercon24-demo
go/src/github.com/110y/go-return
go/src/github.com/110y/goreturn
```

# Filtering Projects

```
main.go  go.mod  vimconf-demo-app main
if err := envconfig.Process(ctx, env); err != nil {
    fmt.Fprintf(os.Stderr, "failed to load environment variables: %s", err.Error())
    return 1
}

var group servergroup.Group

group.Add(&server{})

if err := group.Start(ctx); err != nil {

> confi < 2/66
go/src/github.com/110y/vimconf-demo-config
repos/github.com/eBay/firebase-remote-config-monitor
```



# Filtering Projects

The logo for the fzf tool, featuring a pink chevron symbol pointing right, followed by the letters 'fzf' in a bold, lowercase, sans-serif font. The 'f' and 'z' are light green, and the second 'f' is light gray.

*command-line fuzzy finder*

# Saving Session

**TODO : explain my usage of "mksession"**



# Restoring Session

**TODO : explain my usage of "vim -S"**

# Agenda

- ✓ **My moves for switching between projects**
- ✓ **Dive into "Session" in Vim**
- ✓ **Advanced usage of "Session"**

# What is "Session" ?

**TODO : explain basics of "Session" in Vim**

# mksession

**TODO : explain about mksession**

# Session file

**TODO : explain contents of the Session file**



# sessionoptions

**TODO : explain sessionoptions and its behaviour**

# Agenda

- ✓ **My moves for switching between projects**
- ✓ **Dive into "Session" in Vim**
- ✓ **Advanced usage of "Session"**

# Complements of Session

**TODO : explain about undofile/viminfo/shada**

# Session Plugins

**TODO : introduce some Session related plugins**

# git-worktree

**TODO : explain my setup for git-worktree + Session**

# Agenda

- ✓ **My moves for switching between projects**
- ✓ **Dive into "Session" in Vim**
- ✓ **Advanced usage of "Session"**

VimConf 2024

Thank you 🥷

Yuki Ito