

Forum:[Hack Forums](#)

Thread: [Terminal Diff Checker](#)

SarahRoseLives - July 29, 2025, 10:23 am

A single file diff checker in your terminal.

I put this together some time ago and use it frequently, thought others here might use it to.

Just do:

Code go mod init diffterm
create main.go
go mod tidy
Then build it with:

Code go build -o diffterm
Here's the code for main.go:

Code package main

```
import (  
    "strings"  
    "time"  
    "github.com/gdamore/tcell/v2"  
    "github.com/rivo/tview"  
    "github.com/sergi/go-diff/diffmatchpatch"  
    "fmt"  
)  
  
func countLines(s string) int {  
    if s == "" {  
        return 0  
    }  
    return len(strings.Split(s, "\n"))  
}  
  
// Returns the count of lines added and removed (inserted/deleted lines only)  
func diffLineCounts(orig, updated string) (added, removed int) {  
    dmp := diffmatchpatch.New()  
    diffs := dmp.DiffMain(orig, updated, false)  
    for _, d := range diffs {  
        switch d.Type {  
        case diffmatchpatch.DiffInsert:  
            added += countLines(d.Text)  
        case diffmatchpatch.DiffDelete:  
            removed += countLines(d.Text)  
        }  
    }  
    return  
}
```

```

func colorDiff(orig, updated string) string {
    dmp := diffmatchpatch.New()
    diffs := dmp.DiffMain(orig, updated, false)
    var b strings.Builder
    for _, d := range diffs {
        switch d.Type {
        case diffmatchpatch.DiffInsert:
            b.WriteString(`[green]` + tview.Escape(d.Text) + `[-]`)
        case diffmatchpatch.DiffDelete:
            b.WriteString(`[red]` + tview.Escape(d.Text) + `[-]`)
        default:
            b.WriteString(tview.Escape(d.Text))
        }
    }
    return b.String()
}

func main() {
    app := tview.NewApplication().EnablePaste(true)

    leftArea := tview.NewTextArea()
    leftArea.SetBorder(true)
    leftArea.SetTitle(" Original (Lines: 0) ")

    rightArea := tview.NewTextArea()
    rightArea.SetBorder(true)
    rightArea.SetTitle(" Updated (Lines: 0) ")

    // Short, colored legend in the bottom panel title
    bottom := tview.NewTextView().
        SetDynamicColors(true).
        SetChangedFunc(func() { app.Draw() })
    bottom.SetBorder(true).SetTitle(` [red]Removed Lines[-] | [green]Added Lines[-] | Delete to
Clear `)

    updatePanels := func() {
        origContent := leftArea.GetText()
        updatedContent := rightArea.GetText()
        leftArea.SetTitle(fmt.Sprintf(" Original (Lines: %d) ", countLines(origContent)))
        rightArea.SetTitle(fmt.Sprintf(" Updated (Lines: %d) ", countLines(updatedContent)))

        added, removed := diffLineCounts(origContent, updatedContent)
        legend := fmt.Sprintf("[red]Removed Lines: %d[-] | [green]Added Lines: %d[-] | Delete to
Clear", removed, added)
        bottom.SetTitle(" " + legend + " ")
        bottom.SetText(colorDiff(origContent, updatedContent))
    }

    leftArea.SetChangedFunc(updatePanels)
    rightArea.SetChangedFunc(updatePanels)

    flexTop := tview.NewFlex().

```

```

AddItem(leftArea, 0, 1, true).
AddItem(rightArea, 0, 1, false)
flex := tvview.NewFlex().SetDirection(tvview.FlexRow).
AddItem(flexTop, 0, 3, true).
AddItem(bottom, 0, 2, false)

```

```

// Instead of using a currentPane var and checking it in each area's SetInputCapture,
// just let tvview handle focus naturally and react to delete on whichever pane has focus.
// We track last delete time for both panes for double-tap.

```

```

var lastDelTimeLeft, lastDelTimeRight time.Time
const doubleTapThreshold = 400 * time.Millisecond

```

```

leftArea.SetInputCapture(func(event *tcell.EventKey) *tcell.EventKey {
    switch event.Key() {
    case tcell.KeyTAB:
        app.SetFocus(rightArea)
        return nil
    case tcell.KeyCtrlC, tcell.KeyEscape:
        app.Stop()
        return nil
    case tcell.KeyDelete:
        now := time.Now()
        if now.Sub(lastDelTimeLeft) < doubleTapThreshold {
            leftArea.SetText("", true)
            rightArea.SetText("", true)
        } else {
            leftArea.SetText("", true)
        }
        lastDelTimeLeft = now
        updatePanels()
        return nil
    }
    return event
})

```

```

rightArea.SetInputCapture(func(event *tcell.EventKey) *tcell.EventKey {
    switch event.Key() {
    case tcell.KeyTAB:
        app.SetFocus(leftArea)
        return nil
    case tcell.KeyCtrlC, tcell.KeyEscape:
        app.Stop()
        return nil
    case tcell.KeyDelete:
        now := time.Now()
        if now.Sub(lastDelTimeRight) < doubleTapThreshold {
            leftArea.SetText("", true)
            rightArea.SetText("", true)
        } else {
            rightArea.SetText("", true)
        }
        lastDelTimeRight = now
    }
}

```

```

        updatePanels()
        return nil
    }
    return event
})

// Global handler: TAB switches focus, Ctrl+C/Esc stops app, but delete is handled per pane.
app.SetInputCapture(func(event *tcell.EventKey) *tcell.EventKey {
    switch event.Key() {
    case tcell.KeyTAB:
        if app.GetFocus() == leftArea {
            app.SetFocus(rightArea)
        } else {
            app.SetFocus(leftArea)
        }
        return nil
    case tcell.KeyCtrlC, tcell.KeyEscape:
        app.Stop()
        return nil
    }
    return event
})

app.SetFocus(leftArea)
updatePanels()
if err := app.SetRoot(flex, true).EnableMouse(true).Run(); err != nil {
    panic(err)
}
}

```

Alerithe - July 29, 2025, 7:49 pm

Assuming this provides a side-by-side comparison, seems pretty awesome. Sadly, I am not a Go nerd, so I'm going to stick with Code git diff [options] ... :P

SarahRoseLives - July 30, 2025, 4:05 am

I couldn't post photos but yes, you get your side by side with a box below showing the final code with red/green color to show if something was added or removed.

Go has kinda become my new python, I'm addicted :)