
ffmpeg-go

ffmpeg-go is go lang port of <https://github.com/kkroening/ffmpeg-python>
check examples/example_test.go and ffmpeg_test.go for more examples.

How to get and use

You can get this package via:

```
1 go get -u github.com/u2takey/ffmpeg-go
```

Note: `ffmpeg-go` makes no attempt to download/install FFmpeg, as `ffmpeg-go` is merely a pure-Go wrapper - whereas FFmpeg installation is platform-dependent/environment-specific, and is thus the responsibility of the user, as described below.

Installing FFmpeg

Before using `ffmpeg-go`, FFmpeg must be installed and accessible via the `$PATH` environment variable.

There are a variety of ways to install FFmpeg, such as the official download links, or using your package manager of choice (e.g. `sudo apt install ffmpeg` on Debian/Ubuntu, `brew install ffmpeg` on OS X, etc.).

Regardless of how FFmpeg is installed, you can check if your environment path is set correctly by running the `ffmpeg` command from the terminal, in which case the version information should appear, as in the following example (truncated for brevity):

```
1 $ ffmpeg
2 ffmpeg version 4.2.4-1ubuntu0.1 Copyright (c) 2000-2020 the FFmpeg
   developers
3 built with gcc 9 (Ubuntu 9.3.0-10ubuntu2)
```

Note: The actual version information displayed here may vary from one system to another; but if a message such as `ffmpeg: command not found` appears instead of the version information, FFmpeg is not properly installed.

Examples

```

1 split := Input(TestInputFile1).VFlip().Split()
2   split0, split1 := split.Get("0"), split.Get("1")
3   overlayFile := Input(TestOverlayFile).Crop(10, 10, 158, 112)
4 err := Concat([]*Stream{
5   split0.Trim(KwArgs{"start_frame": 10, "end_frame": 20}),
6   split1.Trim(KwArgs{"start_frame": 30, "end_frame": 40})}).
7   Overlay(overlayFile.HFlip(), "").
8   DrawBox(50, 50, 120, 120, "red", 5).
9   Output(TestOutputFile1).
10  OverWriteOutput().
11  Run()

```

Transcoding From One Codec To Another

```

1 err := ffmpeg.Input("./sample_data/in1.mp4").
2   Output("./sample_data/out1.mp4", ffmpeg.KwArgs{"c:v": "libx265"
3     }).
4   OverWriteOutput().ErrorToStdOut().Run()

```

Cut Video From Timestamp

```

1 err := ffmpeg.Input("./sample_data/in1.mp4", ffmpeg.KwArgs{"ss": 1}).
2   Output("./sample_data/out1.mp4", ffmpeg.KwArgs{"t": 1}).
3   OverWriteOutput().Run()
4 assert.Nil(t, err)

```

Add Watermark For Video

```

1 // show watermark with size 64:-1 in the top left corner after seconds
2   1
3 overlay := ffmpeg.Input("./sample_data/overlay.png").Filter("scale",
4   ffmpeg.Args{"64:-1"})
5 err := ffmpeg.Filter(
6   []*ffmpeg.Stream{
7     ffmpeg.Input("./sample_data/in1.mp4"),
8     overlay,
9   }, "overlay", ffmpeg.Args{"10:10"}, ffmpeg.KwArgs{"enable": "gte(t
10   ,1)"}).
11   Output("./sample_data/out1.mp4").OverWriteOutput().ErrorToStdOut().
12   Run()

```

result:



Cut Video For Gif

```
1 err := ffmpeg.Input("./sample_data/in1.mp4", ffmpeg.KwArgs{"ss": "1"}).  
2   Output("./sample_data/out1.gif", ffmpeg.KwArgs{"s": "320x240", "  
   pix_fmt": "rgb24", "t": "3", "r": "3"}).  
3   OverWriteOutput().ErrorToStdOut().Run()
```

result:



Task Frame From Video

```
1 func ExampleReadFrameAsJpeg(inFileName string, frameNum int) io.Reader
2 {
3     buf := bytes.NewBuffer(nil)
4     err := ffmpeg.Input(inFileName).
5         Filter("select", ffmpeg.Args{fmt.Sprintf("gte(n,%d)", frameNum)
6             }).
7         Output("pipe:", ffmpeg.KwArgs{"vframes": 1, "format": "image2",
8             "vcodec": "mjpeg"}).
9         WithoutOutput(buf, os.Stdout).
10        Run()
11    if err != nil {
12        panic(err)
13    }
14    return buf
15 }
16
17 reader := ExampleReadFrameAsJpeg("./sample_data/in1.mp4", 5)
18 img, err := imaging.Decode(reader)
19 if err != nil {
20     t.Fatal(err)
21 }
22 err = imaging.Save(img, "./sample_data/out1.jpeg")
23 if err != nil {
24     t.Fatal(err)
25 }
26 }
```

result :



Get Multiple Output

```
1 // get multiple output with different size/bitrate
2 input := ffmpeg.Input("./sample_data/in1.mp4").Split()
3 out1 := input.Get("0").Filter("scale", ffmpeg.Args{"1920:-1"}).
4 Output("./sample_data/1920.mp4", ffmpeg.KwArgs{"b:v": "5000k"})
5 out2 := input.Get("1").Filter("scale", ffmpeg.Args{"1280:-1"}).
6 Output("./sample_data/1280.mp4", ffmpeg.KwArgs{"b:v": "2800k"})
7
8 err := ffmpeg.MergeOutputs(out1, out2).OverWriteOutput().ErrorToStdOut
9 ().Run()
```

Show FFmpeg Progress

see complete example at: [showProgress](#)

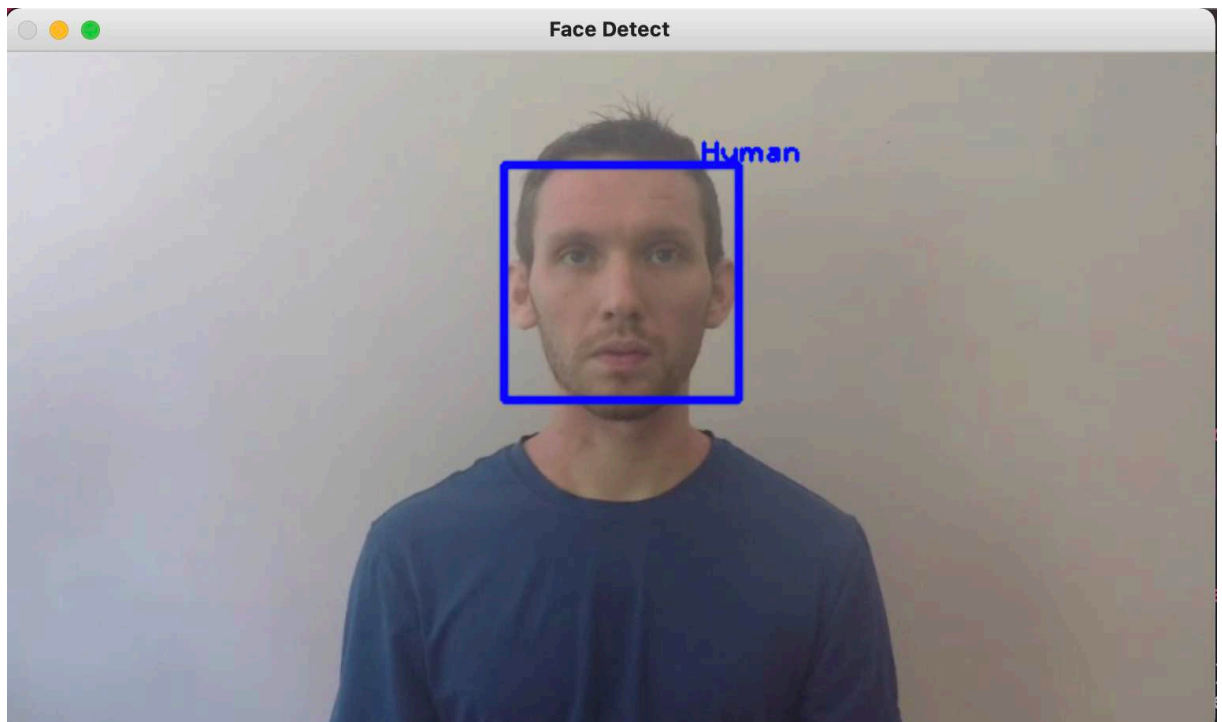
```
1 func ExampleShowProgress(inFileName, outFileName string) {
2     a, err := ffmpeg.Probe(inFileName)
3     if err != nil {
4         panic(err)
5     }
6     totalDuration := gjson.Get(a, "format.duration").Float()
7
8     err = ffmpeg.Input(inFileName).
9         Output(outFileName, ffmpeg.KwArgs{"c:v": "libx264", "preset": "
10             veryslow"}).
11         GlobalArgs("-progress", "unix://" + TempSock(totalDuration)).
12         OverWriteOutput().
13         Run()
14     if err != nil {
15         panic(err)
16     }
17     ExampleShowProgress("./sample_data/in1.mp4", "./sample_data/out2.mp4")
18 }
```

result

```
1 progress: .0
2 progress: 0.72
3 progress: 1.00
4 progress: done
```

Integrate FFmpeg-go With Open-CV (gocv) For Face-detect

see complete example at: [opencv](#)



result:

Set Cpu limit/request For FFmpeg-go

```
1 e := ComplexFilterExample("./sample_data/in1.mp4", "./sample_data/
  overlay.png", "./sample_data/out2.mp4")
2 err := e.RunWithResource(0.1, 0.5)
3 if err != nil {
4     assert.Nil(t, err)
5 }
```

result from command top: we will see ffmpeg used 0.5 core as expected.

```
1 > top
2 PID      USER      PR  NI   VIRT   RES    SHR  S  %CPU  %MEM    TIME+
   COMMAND
3 1386105  root      20   0 2114152 273780 31672 R   50.2   1.7
   0:16.79 ffmpeg
```

View Progress Graph

function view generate mermaid chart, which can be use in markdown or view online

```
1 split := Input(TestInputFile1).VFlip().Split()
2 split0, split1 := split.Get("0"), split.Get("1")
```

```

3     overlayFile := Input(TestOverlayFile).Crop(10, 10, 158, 112)
4 b, err := Concat([]*Stream{
5     split0.Trim(KwArgs{"start_frame": 10, "end_frame": 20}),
6     split1.Trim(KwArgs{"start_frame": 30, "end_frame": 40})}).
7     Overlay(overlayFile.HFlip(), "").
8     DrawBox(50, 50, 120, 120, "red", 5).
9     Output(TestOutputFile1).
10    OverWriteOutput().View(ViewTypeFlowChart)
11 fmt.Println(b)

```

