About the Latest Fyne CLI Tool Changes

Insights into major changes to the Fyne command line tools

Simon Dassow

Consult With Simon

September 19, 2025

Topics

- Background
- Repository sync/migration
- Code cleanups
- Unified flags
- Improved help
- New commands
- Creating new apps
- Handling translations
- Open issues
- Next steps

Background

A Few Personal Details

- Software Developer: 26+ years
- House Music: 21+ years
- Dad: 18+ years
- Calisthenics: 10+ years
- Vegan: 9+ years
- Certified Calisthenics Trainer: 3+ years
- Independent Consultant: 2+ years

Some Professional Things

- Web, RAC Export Trading (1998)
- Web/admin, L&B GmbH (1998)
- Web/backend, Flipside (1999)
- Admin/backend, Vivendi Universal Games (2000)
- Web/backend/admin, Delphi Management Beratung (2003)
- Web/backend, RealNetworks (2006)
- Founder/dev/admin, PuzzWorks (2008)
- Lead/backend/dev, Booking (2010)
- Founder/consultant, Consult With Simon (2023)

Open Source Journey

- Perl/PHP: early web development
- Apache: httpd and mod_perl related patches
- Perl: various modules on CPAN
- PostgreSQL: pg-ruid extension
- OpenBSD: many ports, some userland changes, some xenocara changes, hackathons
- Various: patches with fixes or new features
- Conferences/user groups: present at/on BSD, Perl, Fyne events
- Fyne: OpenBSD support (go-glfw), bug reports, PRs, core contributor, lately top tools contributor

Step By Step

Tools Repository Migration

- Done (inter) VCS migrations before
- Changes on both sides
- Tracking based on dates and commit messages
- Once no problem, twice tho?
- Ended up manually going through commits twice
- First time Oct 17, 2024
- And again Jan 12, 2025 to get recent changes

Code Cleanups

- Clear and concise and wording
- Consistent phrasing and interpunctation
- Intuitive order of commands
- Short aliases

Unified Flags

- Reduce repetition
- Centralize flags
- Simplify developer experience
- Ease maintenance
- Increase readability

Central Flags

```
var intFlags = map[string]func(*int) cli.Flag{
        "app-build": func(dst *int) cli.Flag {
                return &cli.IntFlag{
                        Name:        "app-build",
                        Usage:       "set build number (integer >0, increasing with
                        Destination: dst,
                }
        },
        // ...
}
```

Flag Usage

```go
func Package() *cli.Command {
        p := NewPackager()

        return &cli.Command{
                Name:        "package",
                Aliases:     []string{"p"},
                Usage:       "Packages an application for distribution",
                Description: "You may specify the --executable to package, otherwis
                Flags: []cli.Flag{
                        // ...
                        intFlags["app-build"](&p.AppBuild),
                        // ...
                },
                Action: func(_ *cli.Context) error {
                        // ...
                },
        }
}
```

Improved Help

- Cleanup ouput
- Align style and naming
- Increase readability

fyne help

```
@@ -13,11 +13,12 @@
 COMMANDS:
-   bundle   Embeds static content into your go application.
-   env      The env command prints the Fyne module and environment information
-   get      Downloads and installs a Fyne application
-   install  Packages an application and installs an application.
-   package  Packages an application for distribution.
-   release  Prepares an application for public distribution.
-   version  Shows version information for fyne.
-   serve    Package an application using WebAssembly and expose it via a web serve
-   build    Build an application.
+   init                 Initializes a new Fyne project
+   env, e               Prints the Fyne module and environment information
+   build, b             Builds an application
+   package, p           Packages an application for distribution
+   release, r           Prepares an application for public distribution
+   install, get, i  Packages and installs an application
+   serve, s             Packages an application using WebAssembly and exposes it via a
+   translate, t         Scans for new translation strings
+   version, v           Shows version information for fyne
+   bundle               Embeds static content into your go application
    help, h         Shows a list of commands or help for one command
```

fyne help build

```
@@ -10,10 +10,10 @@
 OPTIONS:
-   --target value, --os value      The mobile platform to target (android, android
-   --sourceDir value, --src value  The directory to package, if executable is not
-   --tags value                    A comma-separated list of build tags.
-   --release                       Enable installation in release mode (disable de
-   -o value                        Specify a name for the output file, default is
-   --pprof                         Enable pprof profiling. (default: false)
-   --pprof-port value              Specify the port to use for pprof profiling. (d
-   --metadata value                Specify custom metadata key value pair that you
+   --target value, --os value       set operating system to target (android, andro
+   --src value, --source-dir value  set directory to package, if executable is not
+   --tags value                     set comma-separated list of build tags
+   --release                        enable installation in release mode, disable d
+   --output value, -o value         specify name for the output file (default: bas
+   --pprof                          enable pprof profiling (default: false)
+   --pprof-port value               specify pprof profiling port (default: 6060)
+   --metadata value                 specify custom metadata key value pair that yo
    --help, -h                      show help
```

Install Command Tags

- Build and package already support it
- Noticed when installing on local machine
- Easy and consistent change thanks to unifed flags

New Parameter

```
@@ -17,2 +17,3 @@
    --release                        enable installation in release mode, disable de
+   --tags value                     set comma-separated list of build tags
    --verbose, -v                    show details when running (default: false)
```

```
$ fyne install --tags beta
$
```

Bundle Multiple Files

- Previouly would only take one argument
- Now allows multiple
- And supports globbing without shell
- Useful in generator scripts

Generator Usage

```
package main

//go:generate fyne bundle -o resources.go *.png

// ...
```

Natively Embed Files

- Previouly was reading files and embedding bytes
- Now using Go's embed package
- Files in source tree instead of code

The Difference

```diff
@@ -5,8 +5,12 @@

-import "fyne.io/fyne/v2"
+import (
+        _ "embed"
+        "fyne.io/fyne/v2"
+)

+//go:embed foo.txt
+var resourceFooTxtData []byte
 var resourceFooTxt = &fyne.StaticResource{
         StaticName: "foo.txt",
-        StaticContent: []byte(
-                "42\n"),
+        StaticContent: resourceFooTxtData,
 }
```

Creating New Apps

Creating New Apps

- As easy as it gets
- Hello world in one command
- Fantastic for bug reports

Help Creating Apps

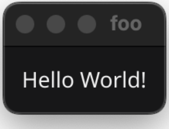
```
$ fyne help init
NAME:
   fyne init - Initializes a new Fyne project

USAGE:
   fyne init [command options] [module-path]

DESCRIPTION:
   Initializes a new Fyne project in the current directory, including
   a go.mod, main.go, and FyneApp.toml file (unless existing).

OPTIONS:
   --app-id value, --id value  set app-id in reversed domain notation for android,
   --name value                set name of the application (default: executable fil
   --icon value                set name of the application icon file
   --verbose, -v               show details when running (default: false)
   --help, -h                  show help
$
```

Starting Project Foo

```
$ mkdir foo
$ cd foo
$ fyne init -v
go: creating new go.mod: module foo
go: to add module requirements and sums:
        go mod tidy
go: finding module for package fyne.io/fyne/v2/widget
go: finding module for package fyne.io/fyne/v2/app
go: found fyne.io/fyne/v2/app in fyne.io/fyne/v2 v2.6.3
go: found fyne.io/fyne/v2/widget in fyne.io/fyne/v2 v2.6.3
go: finding module for package github.com/kr/text
go: found github.com/kr/text in github.com/kr/text v0.2.0
Your new app is ready. Run it directly with: go run .
$
```

Project Foo

```
package main

import (
        "fyne.io/fyne/v2/app"
        "fyne.io/fyne/v2/widget"
)

func main() {
        a := app.NewWithID("foo")
        w := a.NewWindow("foo")

        w.SetContent(widget.NewLabel("Hello World!"))
        w.ShowAndRun()
}
```



Translations

New Command: Translate

- Easy to use
- Utilizes Go’s AST parser to find translations
- Can scan imports
- Example following the hello world app...

Help Translate

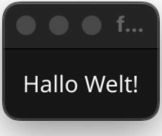
```
$ fyne help translate
NAME:
   fyne translate - Scans for new translation strings

USAGE:
   fyne translate [command options] translationsFile [source ...]

DESCRIPTION:
   Recursively scans the current or given directories/files for
   translation strings, and creates or updates the translations file.

OPTIONS:
   --imports, -i  enable scanning of all imports (slow) (default: false)
   --update, -u   update existing translations (use with care) (default: false)
   --verbose, -v  show details when running (default: false)
   --help, -h     show help
$
```

Integrate Translations

```
@@ -3,3 +3,7 @@ package main
 import (
+       "embed"
+
+       "fyne.io/fyne/v2"
        "fyne.io/fyne/v2/app"
+       "fyne.io/fyne/v2/lang"
        "fyne.io/fyne/v2/widget"
@@ -7,2 +11,5 @@ import (

+//go:embed translations/*.json
+var translationsFS embed.FS
+
 func main() {
@@ -11,3 +18,7 @@ func main() {

-       w.SetContent(widget.NewLabel("Hello World!"))
+       if err := lang.AddTranslationsFS(translationsFS, "translations"); err != nil
+               fyne.LogError("failed to load translations", err)
+       }
+
+       w.SetContent(widget.NewLabel(lang.X("hello", "Hello World!")))
        w.ShowAndRun()
```

Scanning And Translating

```
$ mkdir translations
$ fyne translate -v translations/en.json
scanning files: [main.go]
scanning code ...
adding: hello
$ cat translations/en.json
{
    "hello": "Hello World!"
}
$ sed -e 's!Hello World!Hallo Welt!' translations/en.json > translations/de.json
$
```



Hallo Welt!

Other Things

Improved Install Command

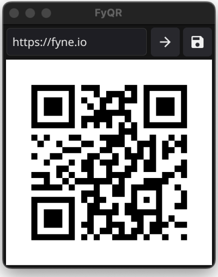
- Does what you want
- Works like Go
- Local installation
- Remote with optional git tag

fyne help install

```
$ fyne help install
NAME:
   fyne install - Packages and installs an application

USAGE:
   fyne install [command options] [remote[@version]]

DESCRIPTION:
   The install command packages an application for the current platform and copies
   into the system location for applications by default.

OPTIONS:
   --target value, --os value       set operating system to target (android, androi
   --dst value, --installDir value  specify install destination, instead of the OS
   --icon value                     set name of the application icon file
   --use-raw-icon                   skip any OS-specific icon pre-processing (defaul
   --app-id value, --id value       set app-id in reversed domain notation for andro
   --release                        enable installation in release mode, disable de
   --tags value                     set comma-separated list of build tags
   --verbose, -v                    show details when running (default: false)
   --help, -h                       show help
$
```

Install QR Code Generator

```
$ fyne install github.com/andydotxyz/fyqr@latest
$
```



Migrations Enabled For New Apps

- New threading model since Fyne 2.6
- Applies to new apps created with fyne init
- Wrap calls to Fyne code in fyne.Do or fyne.DoAndWait to call Fyne code from goroutines

Calling Fyne Code

```go
// ...
fyne.Do(func() {
        someLabel.Enable()
})
// ...
```

Looking Forward

Open Issues

- Still things open
- WASM optimizations
- App metadata for linux packages
- Bunch of smaller issues

Next Steps

- Basic translation support in new apps
- Pass GOFLAGS in release command
- Support for pre-release version tags
- Integration of fyne-cross
- Removal of old/deprecated code

Questions

- Ask now
- Or later
- Community chat

Thanks

- Core team for excellent input and feedback
- Andy for the constant engagement
- Fyne Labs for sponsoring
- Community members for contributing
- Audience and you for listening