Toggly Elixir / Phoenix / LiveView

OTP supervision · Phoenix Plug · LiveView · signed defs

Install

MIX.EXS

```
{:toggly, "~> 0.7"}
{:toggly_phoenix, "~> 0.3"}   # optional
{:toggly_live_view, "~> 0.1.0"} # LV 1.2+
```

Backend App Key only — never a management credential. Signed definitions are on by default.

Supervise

```
# Application.start/2 — one named client
{Toggly,
 name: MyApp.Flags,
 app_key: System.get_env("TOGGLY_APP_KEY"),
 environment: "Production",
 defaults: %{"new-dashboard" => false}}
```

No key: defaults stay usable; `refresh/1` returns `:missing_app_key`. Never derive `name` from users.

Evaluate

```
ctx = %{
  "identity" => "alice", "groups" => ["beta"],
  "claims" => %{"role" => "admin"},
  "request" => %{"country" => "US"}
}
Toggly.enabled?(MyApp.Flags, "new-dashboard", ctx)
Toggly.enabled?(MyApp.Flags, ["a", "b"], ctx,
  requirement: :any)
Toggly.enabled?(MyApp.Flags, "maint", ctx,
  negate: true)
```

Multi-key default is `:all`. Empty key list is false before negation. Boolean branches only — no variant API.

Context & entities

String-keyed maps, per evaluation. No global identity setter.

```
ctx = Toggly.Context.from_headers(
  conn.req_headers, %{"identity" => user.id})
order = Map.put(ctx, "entity", %{
  "kind" => "Order", "key" => "ord-vip",
  "attributes" => %{"Vip" => true}})
Toggly.enabled?(MyApp.Flags,
  "ExpressCheckout", order)
```

ContextProperty needs a matching entity even if AlwaysOn user conditions pass. Kind is exact; attribute names are case-insensitive. Unknown filters fail closed.

Refresh & snapshots

```
Toggly.refresh(MyApp.Flags)
# :ok | {:error, reason}
Toggly.snapshot(MyApp.Flags, ctx)
Toggly.subscribe(MyApp.Flags)
# {:toggly_updated, MyApp.Flags, rev}
```

Add `snapshot_path` and `max_signature_age_seconds` on the child spec. `signed: false` opts into unsigned defs.

Reliability

- Evaluations read ETS and do not wait on HTTP.
- Failed refresh keeps last-good flags. Poll default 60s; WebSocket invalidation when an app key exists.
- Partial Percentage/Targeting without identity fails closed. Prefer a stable `"identity"`.

Toggly Elixir / Phoenix / LiveView

OTP supervision · Phoenix Plug · LiveView · signed defs

Phoenix Plug

```
# After authentication.
plug Toggly.Phoenix.Plug,
  client: MyApp.Flags,
  flags: ["new-dashboard", "api-v2"],
  context: &MyAppWeb.FeatureContext.from_conn/1
# optional: gate: "beta-access", status: 404
```

Callback returns a string-keyed map. Plug adds `request.userAgent`, `acceptLanguage`, `country` without replacing explicit context. Assigns: `toggly_client`, `toggly_context`, `toggly_flags`.

LiveView

```
use Phoenix.LiveView
import Toggly.LiveView, only: [feature: 1]
on_mount {Toggly.LiveView,
  {MyApp.Flags, ["new-dashboard", "api-v2"]}}
```

HEEX — PAIR FEATURE + NEGATE

```
<.feature flags={@toggly_flags}
          feature="new-dashboard">
  <p>New dashboard</p>
</.feature>
<.feature flags={@toggly_flags}
          feature="new-dashboard"
          negate={true}>
  <p>Classic dashboard</p>
</.feature>
```

Hook reads `socket.assigns.toggly_context` or signed session `"toggly_context"`. HTTP Plug assigns do not become LiveView session data. After identity change: `assign_feature_flags/4`.

Usage & telemetry

```
Toggly.record_usage(MyApp.Flags, key, on)
Toggly.flush(MyApp.Flags) # before shutdown
Toggly.metric(MyApp.Flags, :counter, key, 1, %{})
# Local Telemetry only — no gRPC metrics.
```

Batches POST `/api/usage/stats` as `enabled` / `disabled` counters. Never identity, claims, or unique-user hashes.

Pitfalls

- Context maps use **string keys**. No process-wide identity.
- Gates are presentation only — they do not replace authorization.
- Disabled LiveView content is a second `negate={true}` block with the same flags.
- Do not send backend App Keys or raw definitions to browsers.
- `snapshot/2` is diagnostics — do not return backend definitions to clients.

Resources

- Docs: `docs.toggly.io/sdks/elixir`
- Sample: `Toggly.Samples/elixir-phoenix-sdk`
- Hex: `toggly`, `toggly_phoenix`, `toggly_live_view`