Mix Elixir build tool

CONCEPTS

Mix is task-driven: mix <task> [args]. Run mix help to see every task, including those from deps.

mix.exs deps, aliases
↓ deps.get
mix.lock pinned versions
↓ compile
_build/<env> beams per MIX_ENV
↓ release
Release self-contained

MIX_ENV picks the environment: dev (default), test (used by mix test) or prod. Each gets its own _build directory.

PROJECTS

`mix new <name>`
Create a new project

`mix new <name> --sup`
Project with a supervision tree

`mix phx.new <name>`
Phoenix project (needs the phx_new archive)

COMPILE & RUN

`mix compile`
Compile the project

`mix compile --force`
Recompile everything

`mix compile --warnings-as-errors`
Fail when there are warnings

`mix run <file.exs> [args]`
Run a script inside the app

`mix run -e "<expr>"`
Evaluate an expression

`mix run --no-halt`
Keep the app running

`iex -S mix`
Interactive shell with the project

`mix clean`
Remove compiled files

`mix do <task> + <task>`
Run several tasks in one call

TEST

`mix test`
Run the whole suite

`mix test <file>`
Run one test file

`mix test <file>:<line>`
Run the test at a line

`mix test --failed`
Re-run last failures

`mix test --stale`
Only tests for changed modules

`mix test --only <tag>`
Run tests with a tag

DEPENDENCIES

`mix deps.get`
Fetch and lock dependencies

`mix deps.update <dep>`
Update one dependency

Dependency forms in mix.exs

```
{:plug, "~> 1.16"}
{:ex_doc, "~> 0.34", only: :dev, runtime: false}
{:lib, github: "<user>/<repo>", tag: "<tag>"}
{:local, path: "../local"}
```

FORMAT & QUALITY

`mix format`
Format files per .formatter.exs

`mix format --check-formatted`
Fail if anything is unformatted

RELEASES

`MIX_ENV=prod mix release`
Build a production release

ENVIRONMENTS & TOOLING

`MIX_ENV=<env> mix <task>`
Run a task in another env

HELP

`mix help`
List available tasks

`mix help <task>`
Docs for one task