The Twelve-Factor App

Mailovemisa

Onnet - AHT

Jul 28th, 2022

What is 12-Factor App?

The 12 factors

What is 12-Factor App?

Problem

Making applications that run at web-scale is hard work.

Systems that claim to be **web-scale** are able to handle rapid growth efficiently and not have bottlenecks that require re-architecting at critical moments

What is 12-Factor App

- The 12 Factor App methodology is an influential pattern to designing scalable application architecture.
- published in 2011 by **Adam Wiggins**
- a set of design principles for making application horizontally scalable

Source: https://12factor.net

Who

Any developer building applications which run as a service. Ops engineers who deploy or manage such applications.

Why

- scalable
- enable modern agile workflows
- portability
- set baseline expectations for others
- avoid common problems

The 12 factors

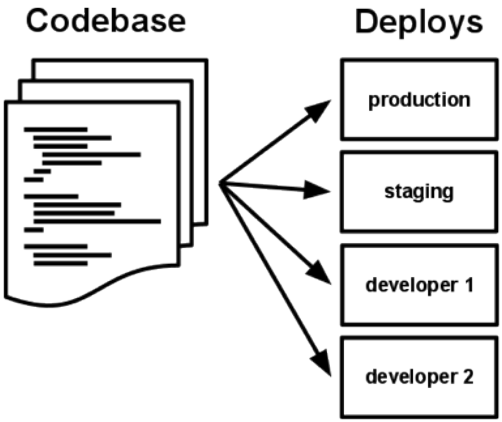
Overview

Codebase
One codebase tracked in revision control, many deploys
Dependencies
Explicitly declare and isolate dependencies
Configuration
Store config in the environment
Backing Services
Treat backing services as attached resources
Build, release, run
Strictly separate build and run stages
Processes
Execute the app as one or more stateless processes
Port binding
Export services via port binding
Concurrency
Scale out via the process model
Disposability
Maximize robustness with fast startup and graceful shutdown
Dev/prod parity
Keep development, staging, and production as similar as possible
Logs
Treat logs as event streams
Admin processes
Run admin/management tasks as one-off processes

1. Codebase

One codebase tracked in revision control, many deploys

- Only one codebase per app
  - If there are multiple codebases, it's not an app
  - Multiple apps sharing the same code is a violation of twelve-factor.
- Many deployes of one app



1. Codebase - Q

- Example of violation?
- Odoo EE vs Odoo CE?
- Git submodules?

2. Dependencies

Explicitly declare and isolate dependencies

- Never relies on implicit existence of system-wide packages
- Declares all dependencies, completely and exactly, via a dependency declaration manifest
- Dependency declaration and isolation must always be used together

3. Config

Store config in the environment

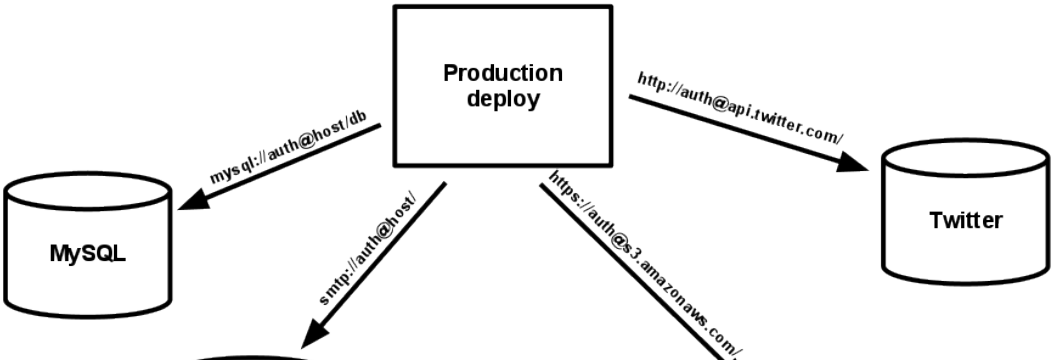
Strict separation of config from code

An app's config is everything that is likely to vary between deploys - resource handles to database, memory - credentials to external services - per-deploy values

4. Backing Services

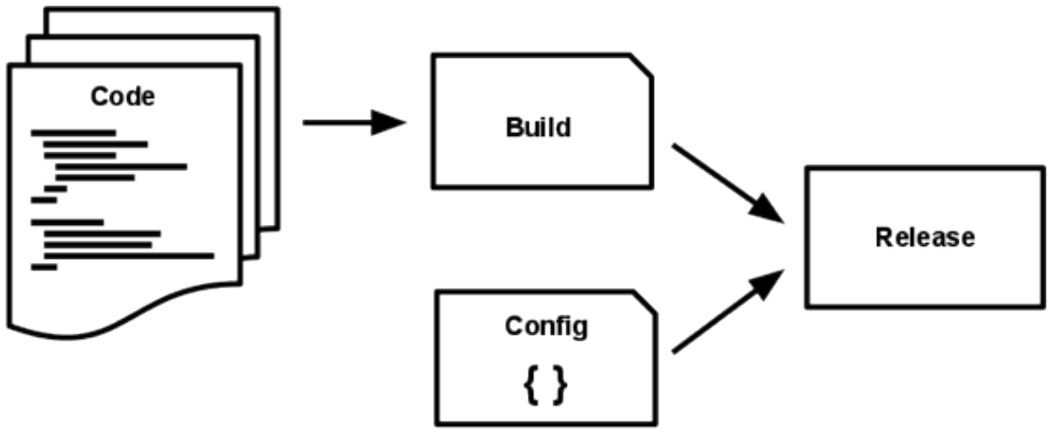
Treat backing services as attached resources

A backing service is any service the app consumes over the network as part of its normal operation. The code for a twelve-factor app makes no distinction between local and third party services. Swap the application from one provider to another without making any further modifications to the code base



5. Build, Release, Run

Strictly separate build and run stages



maximize your delivery speed while keeping high confidence through automated testing and

8. Concurrency