



# Migrating from rebar2.x to rebar3.x

---

# Content

- $3 \neq 2 + 1$
- Commands and configuration
- Installation
- Cleaning up existing projects
- First steps for K2
- References

$$3 \neq 2 + 1$$

- rebar3 is not the next iteration of rebar2 !!!
- different workflow
- different logic
- different behaviour
- treat it like migrating to a different build tool !!!

## 3 $\neq$ 2 + 1 – Basic usage

- rebar3 **only** handles **OTP**-structured projects
- the **new** command creates a project from a template
  - 1. argument: **lib**, **app**, **release** or **plugin**
  - 2. argument: name
- dependencies are listed in the **rebar.config** file under the **deps** key

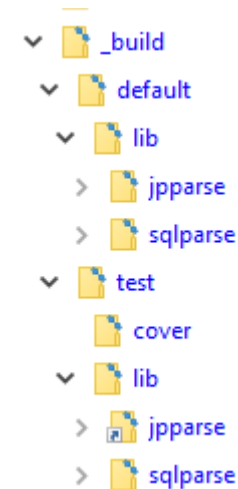
```
{deps, [{cowboy, "1.0.1"},  
        {cowboy, {git, "git://github.com/ninenines/cowboy.git", {tag, "1.0.1"}}}  
]}.
```

```
{cowboy, "*", {git, "git://github.com/ninenines/cowboy.git", {tag, "1.0.1"}}}
```

- **compile** fetches dependencies and compiles all applications
  - no **rebar get-dependencies** command
  - no abbreviated commands

## 3 $\neq$ 2 + 1 – Basic usage

- output is found in the `_build` directory at the root of the project
  - installing dependencies
  - building releases
  - any other output written to disk
- tests by default are expected to be found under the `test/` directory, aside from `eunit` found within individual modules



# Commands and configuration

- `clean` - removes compiled beam files from apps
  - `rebar3 clean` - only cleans the default profile
  - `rebar3 as test clean` - only cleans the test profile
  - `--all` – can be added to clear dependencies' beams
- `compile` - ensures / fetches all dependencies and compiles the needed dependencies and the project's apps `.app.src` and `.erl` files
  - `{erl_opts, ... / {erl_first_files, ... / {xrl_opts, ... / {yrl_opts, ...`
- `cover` - coverage analysis on modules called by CT or Eunit
  - `{cover_enabled, ... / {cover_excl_mods, ...`
- `ct` - runs common tests located under the `test/` directory
  - `{ct_cover, ... / {ct_use_short_names, ... / {ct_verbose, ...`
- `deps` - lists source and package dependencies
- `dialyzer` - builds the PLT and carries out success typing analysis
  - `{dialyzer_plt_location, ... / {dialyzer_warnings, ...`

# Commands and configuration

- `edocs` - generates documentation using `doc`
- `eunit t` - runs eunit tests on project apps
  - `{eunit t_opts, ...}`
- `new`

# Installation

- create a new directory e.g. `c:\Software\rebar3`
- download the latest rebar3 from <https://www.rebar3.org/>
- create a script called `rebar3.cmd` in the rebar3 directory:

```
@echo off
setlocal
set rebarscript=%~f0
escript.exe "%rebarscript: .cmd=%" %*
```
- add the rebar3 directory to your `PATH` variable



# Cleaning up existing projects

- remove the following directories:

`.rebar`

`_rel`

`ebin`

`deps`

- `.gitignore`: remove these entries and add new `_build`
- `rebar.config`: remove the entries `sub_dirs` and `lib_dirs`
- review the `.app.src` files
  - applications you depend on are listed in the `applications` tuple
  - no circular dependencies
  - the following fields are mentioned (releases):  
`applications`, `description`, `env`, `registered`, `vsns`

# Cleaning up existing projects

- handles releases and OTP applications only
- dependencies should only be OTP applications
- one of the following directory structures:

`src/*. {erl , app. src}`

- or

`apps/app1/src/*. {erl , app. src}`

`apps/app2/src/*. {erl , app. src}`

`apps/app3/src/*. {erl , app. src}`

- or

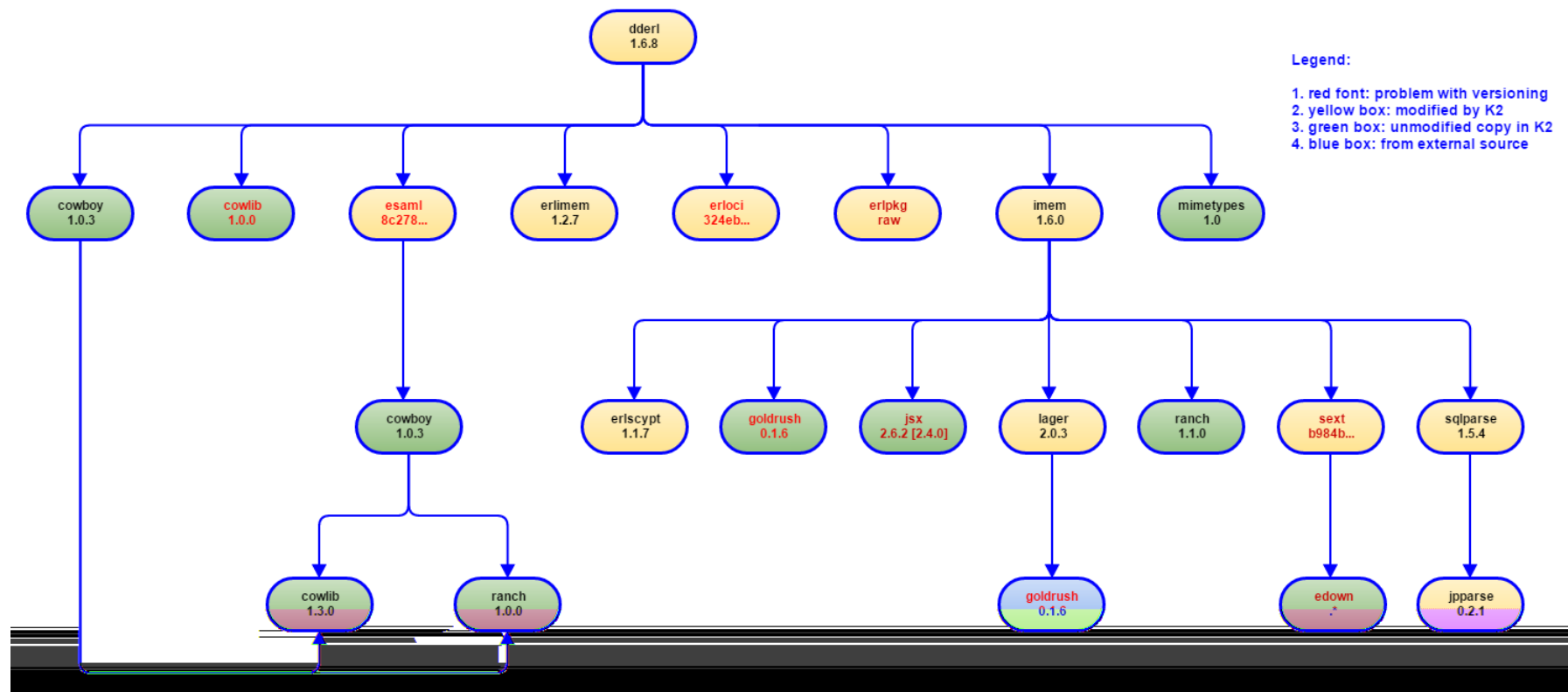
`lib/app1/src/*. {erl , app. src}`

`lib/app2/src/*. {erl , app. src}`

`lib/app3/src/*. {erl , app. src}`

- no `get-dependencies` command
  - no longer re-compiles dependencies once it has done so before
  - no longer checks or enforces dependencies – instead uses a ‘nearest to the root’ dependency algorithm

# First steps for K2



# First steps for K2

- Project sqlparse - .gitignore

```
_build  
erl_crash.dump  
rebar.lock  
src/*.dot  
src/sql_lex.erl  
src/sql_parse.erl  
test/*.beam
```

# First steps for K2

- Project sqlparse – `rebar.config`

```
{cover_enabled, true}.
{cover_excl_mods, [sql_lex, sqlparse]}.

{ct_cover, true}.
{ct_verbose, true}.

{deps, [{jpparse, {git, "https://github.com/K2InformaticsGmbH/jpparse.git",
{tag, "0.2.1"}}}]}.

{dialyzer_plt_location, local}.
{dialyzer_warnings, [error_handling,
overspecs,
race_conditions,
specdiffs,
underspecs,
unknown,
unmatched_returns
]}.
...
```

# First steps for K2

- Project sqlparse – `rebar.config`

```
...
{eunit_opts, [
    {skip_deps, true},
    verbose
]}.

{xref_checks, [
    deprecated_functions,
    deprecated_function_calls,
    exports_not_used,
    locals_not_used,
    undefined_functions,
    undefined_function_calls
]}.

{xrl_opts, [dfa_graph, verbose]}.

{yrl_opts, [verbose]}.
```



# First steps for K2

- Project sqlparse – `sql parse. app. src`
- `{mod, {CallbackMod, Args}}`

Defines a callback module for the application, using the application behaviour. This tells OTP that when starting your application, it should call `CallbackMod:start(normal, Args)`. This function's return value will be used when OTP will call `CallbackMod:stop(StartReturn)` when stopping your application. People will tend to name *CallbackMod* after their application.

- `{modules, [ModuleList]}`

Contains a list of all the modules that your application introduces to the system. A module always belongs to at most one application and can not be present in two applications' app files at once. This list lets the system and tools look at dependencies of your application, making sure everything is where it needs to be and that you have no conflicts with other applications already loaded in the system. If you're using a standard OTP structure and are using a build tool like *rebar3*, this is handled for you.



# First steps for K2

- Project sqlparse – `.travis.yml`

```
language: erlang
notifications:
  email:
    - xxx
otp_release:
  - 18.0
  - 18.1
  - 18.2
  - 18.2.1
  - 18.3
after_success:
  "./rebar3 ct"
```

# References

- Website

<https://www.rebar3.org/>

- Github

<https://github.com/erlang/rebar3>