Config as Code

Railway supports defining the configuration for a single deployment in a file alongside your code. By default, we will look for a `railway.toml` or `railway.json` file.

Everything in the build and deploy sections of the service settings can be specified in this configuration file.

How does it work?

When a new deployment is triggered, Railway will look for any config files in your code and combine these values with the settings from the dashboard.

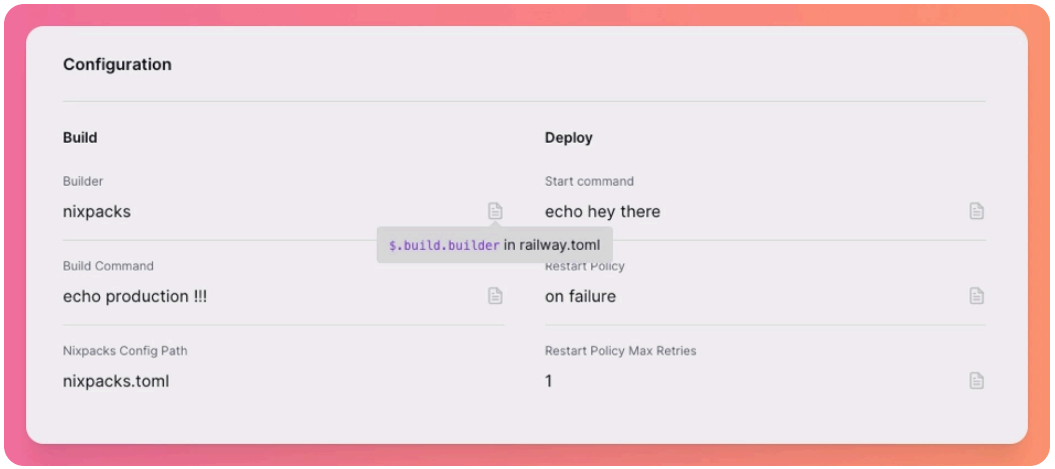
The resulting build and deploy config will be used **only for the current deployment**.

The settings in the dashboard will not be updated with the settings defined in code.

Configuration defined in code will always override values from the dashboard.

Config Source Location

On the deployment details page, all the settings that a deployment went out with are shown. For settings that come from a configuration file, there is a little file icon. Hovering over the icon will show exactly what part of the file the values originated from.



Configurable Settings

Specify the Builder

Set the builder for the deployment.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
    "builder": "NIXPACKS"
  }
}
```

Possible values are:

- `NIXPACKS`
- `DOCKERFILE`

Note: Railway will always build with a Dockerfile if it finds one. To build with nixpacks, you can remove or rename the Dockerfile.

Read more about Builds here.

Watch Patterns

Array of patterns used to conditionally trigger a deploys.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
    "watchPatterns": ["src/**"]
  }
}
```

Read more about watch patterns here.

Build Command

Build command to pass to the Nixpacks builder.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
    "buildCommand": "yarn run build"
  }
}
```

This field can be set to `null`.

Read more about the build command here.

Dockerfile Path

Location of non-standard Dockerfile.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
    "dockerfilePath": "Dockerfile.backen
  }
}
```

This field can be set to `null`.

More about building from a Dockerfile here.

Nixpacks Config Path

Location of a non-standard Nixpacks config file.

```
{
  "$schema": "https://railway.com/railwa
```

```
  "build": {
    "nixpacksConfigPath": "backend_nixpa
  }
}
```

This field can be set to `null`.

Nixpacks Plan

Full nixpacks plan. See the Nixpacks documentation for more info.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
    "nixpacksPlan": {
      "providers": ["python", "node"],
      "phases": {
        "install": {
```

```
        "dependsOn": ["setup"],
        "cmds": ["npm ci"]
      }
    }
  }
}
```

This field can be set to `null`.

You can also define specific options as follows.

setup phase.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
```

```
    "nixpacksPlan": {
      "phases": {
        "setup": {
          "nixPkgs": ["...", "ffmpeg"]
        }
      }
    }
  }
```

Custom Install Command

install command.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
    "nixpacksPlan": {
      "phases": {
```

```
      "install": {
        "dependsOn": ["setup"],
        "cmds": ["npm install --legacy
      }
    }
  }
}
```

Nixpacks Version

EXPERIMENTAL: USE AT YOUR OWN RISK!.

Version of Nixpacks to use. Must be a valid Nixpacks version.

```
{
  "$schema": "https://railway.com/railwa
  "build": {
```

```
    "nixpacksVersion": "1.30.0"
  }
}
```

This field can be set to `null`.

You can also use the `NIXPACKS_VERSION` configuration variable to set the Nixpacks version.

Start Command

The command to run when starting the container.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
```

```
    "startCommand": "node index.js"
  }
}
```

This field can be set to `null`.

Read more about the start command here.

Number of Replicas

For horizontal scaling, the number of instances to run for the deployment.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "numReplicas": 5
```

```
    }
  }
```

This field can be set to `null`.

Read more about horizontal scaling here.

Multi-region Configuration

Horizontal scaling across multiple regions, with two replicas in each region.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "multiRegionConfig": {
      "us-east4": {
        "numReplicas": 2
      },
```

```
      "us-west1": {
        "numReplicas": 2
      },
      "europe-west4": {
        "numReplicas": 2
      },
      "asia-southeast1": {
        "numReplicas": 2
      }
    }
  }
}
```

This field can be set to `null`.

Read more about horizontal scaling with multiple regions here.

Healthcheck Path

Path to check after starting your deployment to ensure it is healthy.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "healthcheckPath": "/health"
  }
}
```

This field can be set to `null`.

Read more about the healthcheck path here.

Healthcheck Timeout

Number of seconds to wait for the healthcheck path to become healthy.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "healthcheckPath": "/health",
    "healthcheckTimeout": 300
  }
}
```

This field can be set to `null`.

Read more about the healthcheck timeout here.

Restart Policy Type

How to handle the deployment crashing.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "restartPolicyType": "ALWAYS"
  }
}
```

Possible values are:

- `ON_FAILURE`
- `ALWAYS`
- `NEVER`

Read more about the Restart policy here.

Restart Policy Max Retries

Set the max number of retries for the restart policy.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "restartPolicyType": "ALWAYS",
    "restartPolicyMaxRetries": 5
  }
}
```

This field can be set to `null`.

Cron Schedule

Cron schedule of the deployed service.

```
{
  "$schema": "https://railway.com/railwa
  "deploy": {
    "cronSchedule": "*/15 * * * *"
  }
}
```

This field can be set to `null`.

Setting Environment Overrides

Configuration can be overridden for a specific environment by nesting it in a `environments.[name]` block.

When resolving the settings for a deployment, Railway will use this priority order:

1. Environment specific config in code
2. Base config in code
3. Service settings

The following example changes the start command just in the production environment.

```
{
  "$schema": "https://railway.com/railwa
  "environments": {
    "staging": {
      "deploy": {
        "startCommand": "npm run staging
      }
    }
  }
}
```

PR Environment Overrides

Deployments for pull requests can be configured using a special `pr` environment. This configuration is applied only to deploys that belong to an ephemeral environment. When resolving the settings for a PR deployment, the following priority order is used:

1. Environment with the name of the ephemeral environment
2. Environment with the hardcoded name "pr"
3. Base environment of the pull request
4. Base config as code
5. Service settings

```
{
  "$schema": "https://railway.com/railwa
  "environments": {
    "pr": {
      "deploy": {
        "startCommand": "npm run pr"
      }
    }
  }
}
```

Configuring a Build provider with Nixpacks

To define a build provider ahead of time, create a `nixpacks.toml` file and configure it like so:

```
providers = ["...", "python"]
```

10 People are discussing this topic in Forums.

Join the Discussion

Prev
Build and Start Commands

Next
Cron Jobs

Edit this file on GitHub