roc-wasm4

Project documentation
0.8.0

Contents

Release 0.8.0 3
- Build a game 3
- Choose an example 3
- Use the platform API 3
- Distribute a game 3
- Work on the platform 4

roc-wasm4

roc-wasm4 is a Roc platform for WASM-4 games. It provides drawing, input, audio, disk persistence, netplay state and sprites.

Release 0.8.0

Use this release's platform bundle and examples archive. The API archive restores this release's reference. Download the PDF manual. Each asset has checked SPDX and CycloneDX sidecars beside it.

Build a game

Install a current Roc nightly and the WASM-4 CLI (`w4`). For a checkout, install Blueprint 0.11.0, then build the host and a cart:

```
blueprint run build-host
roc build examples/basic/main.roc --target=wasm32 --opt=size --output=basic.wasm
w4 run basic.wasm
```

Checkout examples use the working-tree platform. For a standalone game, copy an entire example directory and replace its platform path with the immutable bundle URL from the selected release. Follow that release's compiler requirements; older releases may require a specific nightly.

Choose an example

Example	What it demonstrates
`examples/basic/main.roc`	Drawing, text, input, mouse, trace and tones
`examples/snake/main.roc`	A playable snake game
`examples/rocci-bird/main.roc`	Rocci Bird, by Brendan Hansknecht with art by Luke DeVault
`examples/sound/main.roc`	A tone parameter playground

Play Rocci Bird online. The repository README links animated demonstrations.

Use the platform API

Import `w4.W4` for drawing, input, audio, persistence and utility functions. Use `w4.Sprite` for sprite data and blitting; `w4.Host` contains the lower-level hosted effects used by the platform. See the generated API reference for function signatures and usage.

Distribute a game

Build with size-oriented optimization and use the WASM-4 CLI to produce an HTML game:

```
roc build app.roc --target=wasm32 --opt=size --output=app.wasm
w4 bundle app.wasm --title "My Game" --html my-game.html
```

Release 0.8.0

For native distribution:

```
w4 bundle app.wasm --title "My Game" --windows my-game-windows.exe --mac my-game-mac --linux my-game-linux
```

Work on the platform

Run `blueprint run check` from the repository root. Blueprint downloads and verifies the fixed tooling compiler and Zig; `roc` on PATH is the compiler under test. `blueprint run bundle` bundles an already-built host.

CI runs all four public carts for sixty deterministic frames in the official WASM-4 web runtime under locked headless Chromium on Linux. It checks framebuffer hashes, simulated input, trace output and host import calls. Play-test audible sound and native runtime behavior before release. The host reserves 32768 bytes for dynamic allocation by default; build with `zig build -Dmem-size=<bytes>` inside `blueprint shell` to change that limit.

Run `blueprint run docs` to render the HTML, PDF and Markdown manual and API reference.