



**CATCHING RENDERING
REGRESSIONS ON ALL
PLATFORMS**

François Mockers

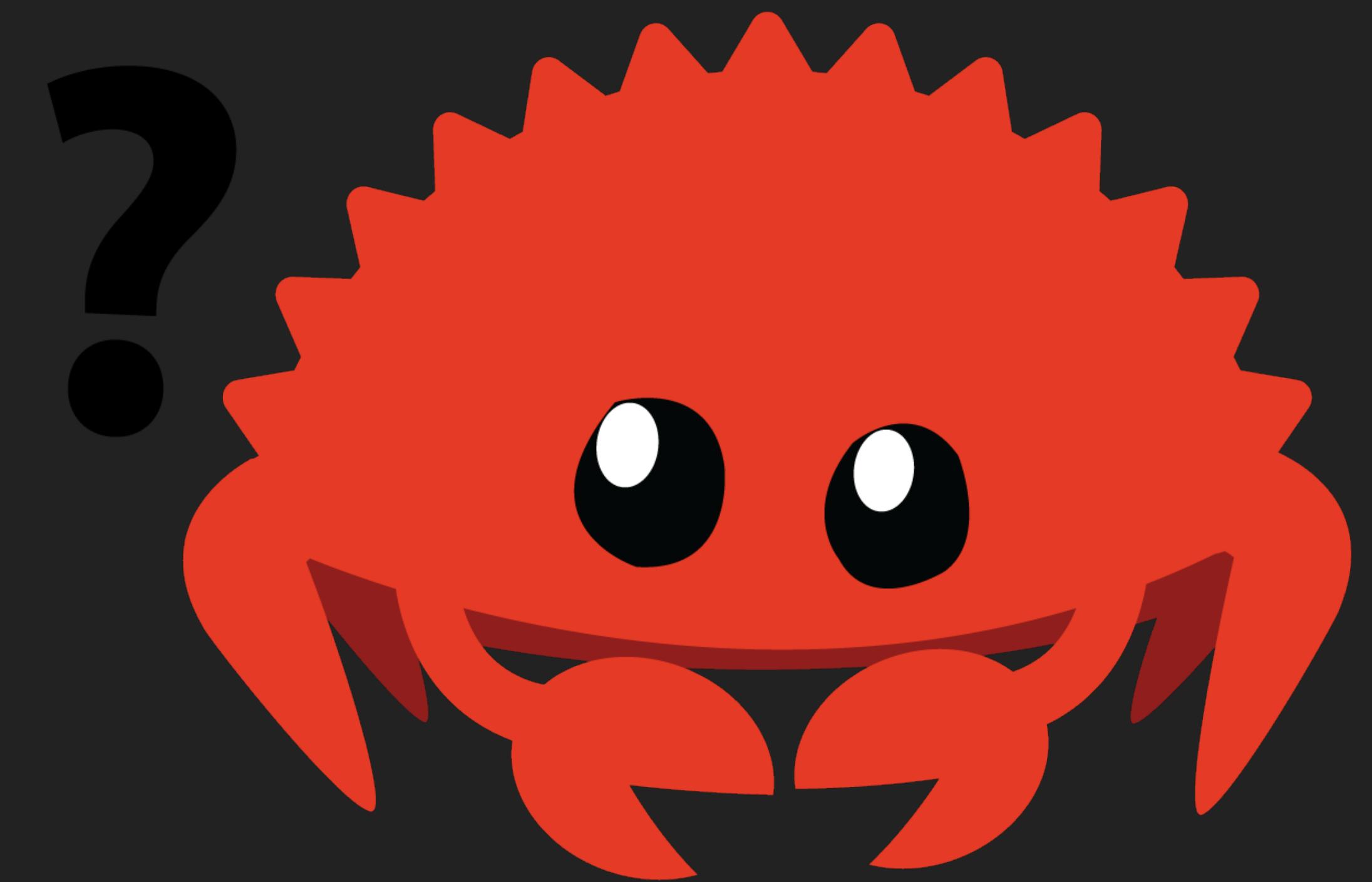
Bevy Game Development Meetup #4 - 2024-05-24

HOW TO TEST A GAME ENGINE ON MACOS, LINUX, WINDOWS, IOS, ANDROID, WEBGPU, WEBGL2, ... IN CONTINUOUS INTEGRATION

WHO AM I

- ▶ François Mockers, mockersf on GitHub
- ▶ Using (and contributing to) Bevy since the 0.2
- ▶ Maintainer and SME on several topics
- ▶ Come talk to me if you have ideas to improve testing the engine!





HOW TO TEST A GAME ENGINE

UNIT TEST

- ▶ Test a specific function
- ▶ Coverage in Bevy is very low
 - ▶ We're trying to increase it
 - ▶ Aiming for 100% coverage is not a good thing, it would mean useless tests and harder to maintain code

DOC TEST

- ▶ Show how to use a function
- ▶ UX test, typically doesn't have assertion

INTEGRATION TEST

- ▶ Test the public API, and how different modules work together
- ▶ There are a few in Bevy!
- ▶ They're actually documentation on how to test a game
 - ▶ [tests/how_to_test_systems.rs](#)
- ▶ Bevy doesn't have integration tests

BENCH TEST

- ▶ Check performances of a function, compared to the last run
- ▶ Mostly focused on the ECS
- ▶ A few on reflection and on async tasks

EXAMPLES

- ▶ Big focus of Bevy
 - ▶ How to use features
 - ▶ Demo of the engine
- ▶ Run the examples in CI, catch crashes
- ▶ Encourage contributors to run examples related to changes

**WHAT ABOUT
RENDERING?**

TESTS

- ▶ Check that the rendering world has the expected entities / components
- ▶ Check the rendering commands generated
- ▶ Hard to write, hard to maintain

VISUAL TESTS

- ▶ Snapshot testing
 - ▶ Take a screenshot before the change
 - ▶ Take a screenshot after
 - ▶ Compare them
- ▶ Comparison?
 - ▶ Pixel by pixel
 - ▶ By structural similarity
 - ▶ FLIP from NVIDIA
 - ▶ By histogram

REUSING THE EXAMPLES

- ▶ Run every examples
- ▶ Take a screenshot
- ▶ Compare the screenshot with last execution on main

DRIVING EXAMPLE EXECUTION

- ▶ Feature bevy_ci_testing

(

```
setup: (
    fixed_frame_time: Some(0.03),
),
events: [
    (200, Screenshot),
    (900, AppExit),
]
)
```

MAKING THE EXAMPLES DETERMINISTS

- ▶ Examples need to render exactly the same at a given frame
 - ▶ Fixed timestep
 - ▶ Only seeded random

INTRODUCING PIXEL EAGLE

- ▶ <https://pixel-eagle.vleue.com/>
- ▶ Image comparison as a service
- ▶ Made with Bevy in mind
 - ▶ Hash comparison of screenshots first
 - ▶ Pixel by pixel if needed
- ▶ Still basic for now, still evolving



BENCH TEST FOR THE RENDERER

- ▶ Coming soon...
- ▶ Blocked on having actual hardware
- ▶ Start by documenting and standardising how to bench test the renderer



BUT... ON ALL
PLATFORMS?

ON ALL PLATFORMS

CONSTRAINTS

- ▶ In the cloud
- ▶ Headless
- ▶ Reliable

LINUX

- ▶ Use llvmpipe software renderer
- ▶ Use xvfb to get a virtual X11 desktop
- ▶ SystemInfo { os: "Linux 22.04 Ubuntu", kernel: "6.5.0-1021-azure", cpu: "AMD EPYC 7763 64-Core Processor", core_count: "2", memory: "15.6 GiB" }
- ▶ AdapterInfo { name: "llvmpipe (LLVM 15.0.7, 256 bits)", vendor: 65541, device: 0, device_type: Cpu, driver: "llvmpipe", driver_info: "Mesa 23.3.6 - kisak-mesa PPA (LLVM 15.0.7)", backend: Vulkan }
- ▶ TODO for Wayland

MACOS

- ▶ Just works on the m1 GitHub runners
- ▶

```
SystemInfo { os: "MacOS 14.4.1 ", kernel: "23.4.0", cpu: "Apple M1 (Virtual)", core_count: "3", memory: "7.0 GiB" }
```
- ▶

```
AdapterInfo { name: "Apple Paravirtual device", vendor: 0, device: 0, device_type: IntegratedGpu, driver: "", driver_info: "", backend: Metal }
```

WINDOWS

- ▶ “Just” works on the windows GitHub runners
- ▶

```
SystemInfo { os: "Windows Server 2022 Datacenter", kernel: "20348", cpu: "AMD EPYC 7763 64-Core Processor", core_count: "2", memory: "16.0 GiB" }
```
- ▶

```
AdapterInfo { name: "Microsoft Basic Render Driver", vendor: 5140, device: 140, device_type: Cpu, driver: "", driver_info: "", backend: Dx12 }
```
- ▶ Actually very slow and very unstable
 - ▶ Most 3d examples don't work
 - ▶ Disabled since 2024/05/15

WASM / WEBGL2

- ▶ Ubuntu / Firefox combination works (with xvfb and llvmpipe)
- ▶ Limited scope for now, only 6 examples
- ▶ Driving a browser with Playwright
- ▶ Uses the screenshot feature of Playwright, not the built in from Bevy

ON ALL PLATFORMS

WASM / WEBGPU

- ▶ macOS / Chromium combination works
- ▶ Same status as WebGL2

IOS

- ▶ Tested on iPhone 13, 14 and 15, with iOS 15, 16, 17
- ▶ Thanks to BrowserStack Open Source support
- ▶ Run the mobile example
- ▶ Test done with Appium
- ▶ Uses the screenshot feature of Appium, not the built in from Bevy

ANDROID

- ▶ Tested on Xiaomi Redmi Note 11, Google Pixel 6, Samsung Galaxy S23, Google Pixel 8, with Android 11, 12, 13 and 14
- ▶ Thanks to BrowserStack Open Source support
- ▶ Same as iOS
- ▶ Android has more limitations
 - ▶ Some GPUs fail with MSAA
 - ▶ Some GPUs fail with shadows
 - ▶ Some GPUs fail with GPU preprocessing



HOW DOES IT LOOK

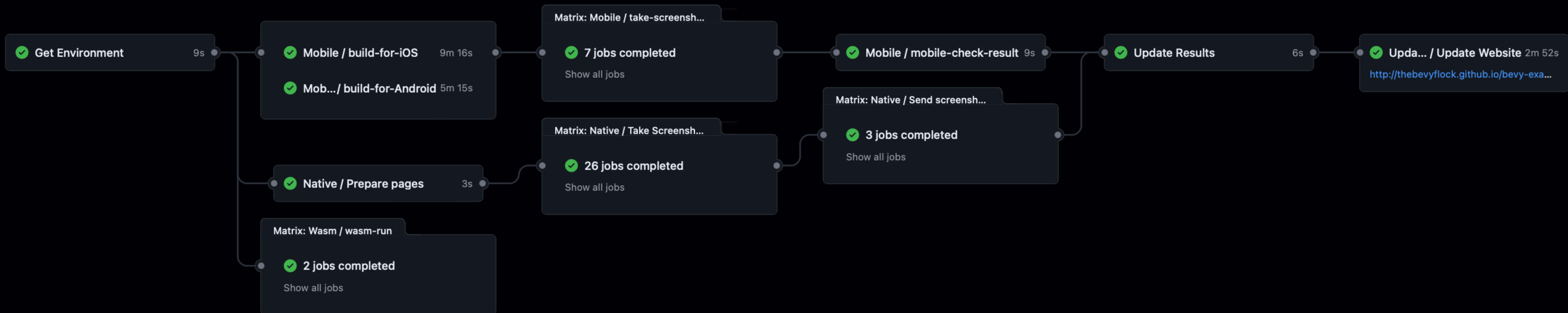
HOW DOES IT LOOK

GITHUB ACTIONS

- ▶ <https://github.com/TheBevyFlock/bevy-example-runner/actions/workflows/report-main.yml>

report-main.yml

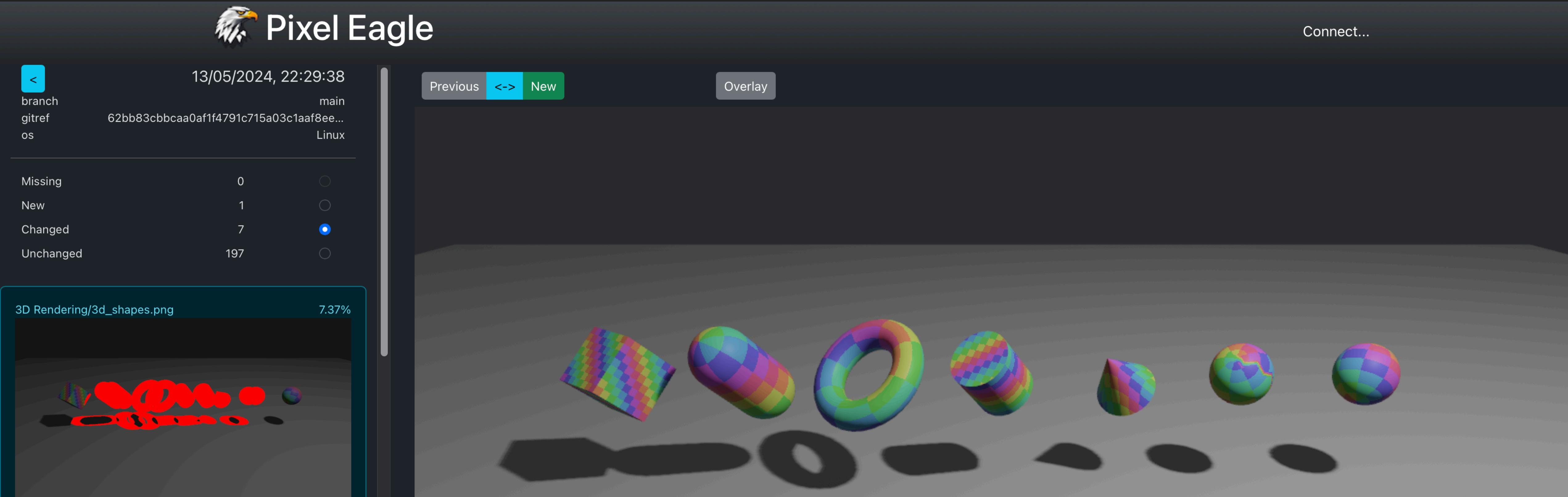
on: schedule



HOW DOES IT LOOK

PIXEL EAGLE

- ▶ <https://pixel-eagle.vleue.com/project/B25A040A-A980-4602-B90CD480AB84076D>



HOW DOES IT LOOK

REPORT PAGE

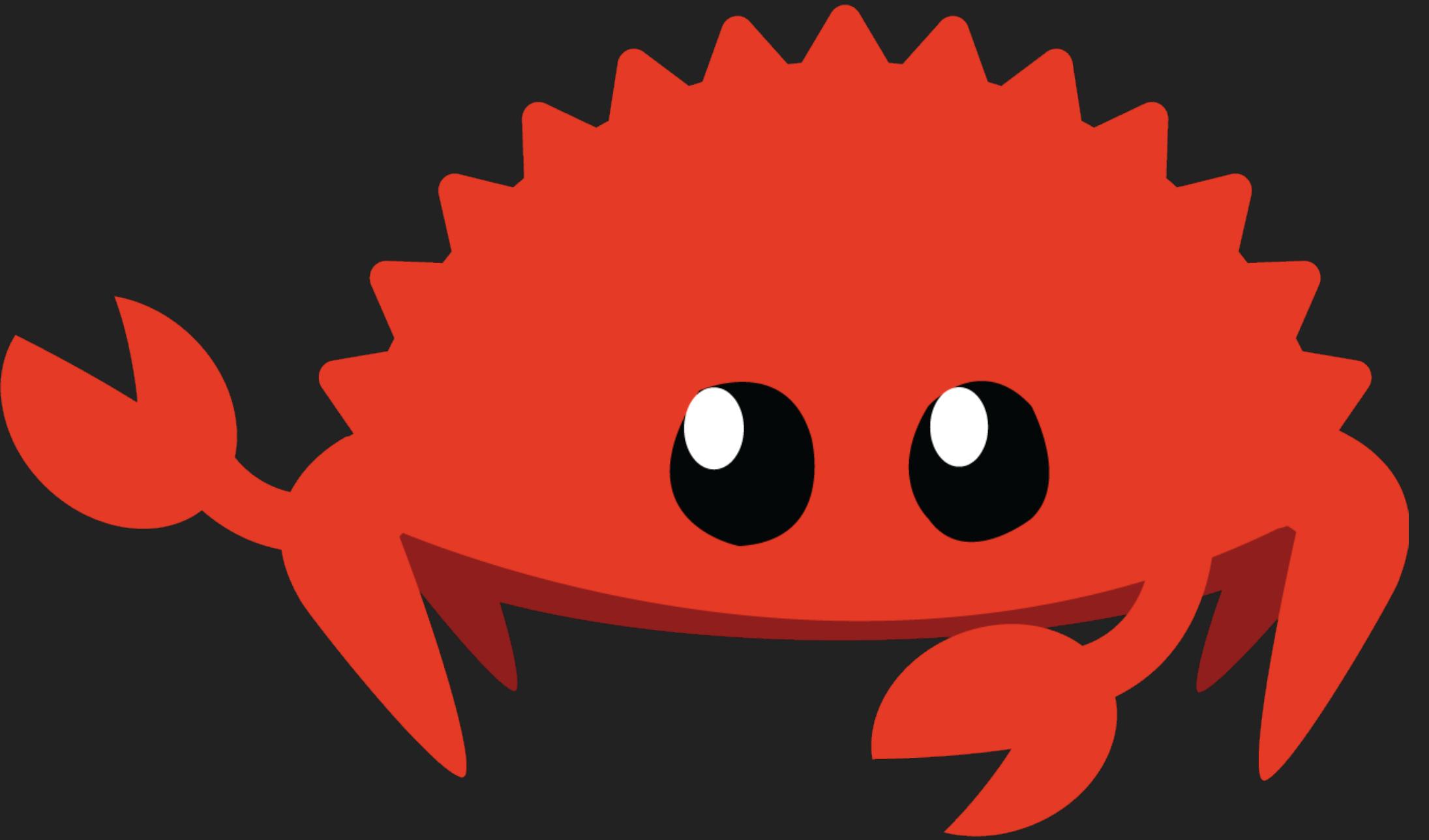
► <https://thebevyflock.github.io/bevy-example-runner/index.html>

✓ Example ran successfully, screenshot didn't change
⌚ Example ran successfully, but couldn't capture a screenshot
★ Example ran successfully, screenshot changed
✗ Error running the example

Only show flaky examples

	2024-05-22 06:10 60afec2	2024-05-22 00:34 a785e3c	2024-05-21 00:34 53f4c38	2024-05-19 00:37 2aed777	2024-05-18 18:09 a55e0e3	2024-05-17 23:26 ee6dfdf3	2024-05-17 20:25 ee6dfdf3	2024-05-17 18:09 ee6dfdf3	2024-05-13 19:20 62bb83c	2024-05-13 18:08 1a3549a	2024-05-13 00:35 bfc1338	2024-05-12 18:09 dc0fdd6	2024-05-12 02:21 443ce9a	2024-05-12 443ce9a	About	
3D Rendering / depth_of_field	⌚	✓	★	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	-	
3D Rendering / fog	⌚	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3D Rendering / generate_custom_mesh	⌚	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3D Rendering / irradiance_volumes	⌚	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3D Rendering / lighting	⌚	✓	✓	✓	✓	✓	✓	✓	★	✓	✓	✓	✓	✓	✓	✓
3D Rendering / lightmaps	⌚	✓	✓	✓	✓	✓	✓	✓	★	✓	✓	✓	✓	✓	✓	✓

Focal distance: 7 m (Press Up/Down to change)
Aperture F-stops: f/0.125 (Press Left/Right to change)
Sensor height: 18.66mm
Focal length: 22.524612mm
Mode: Bokeh (Press Space to change)



NEXT STEPS

MORE MOBILE DEVICES

- ▶ Waiting for some bugs to be fixed on main (in milestone 0.14)
- ▶ https://www.browserstack.com/list-of-browsers-and-platforms/app_automate
- ▶ Need good coverage of
 - ▶ GPU
 - ▶ OS version
 - ▶ Device manufacturer

NEXT STEPS

IMPROVE WASM TESTS

- ▶ Include them in the report page
- ▶ Run more than 6 tests
- ▶ Take screenshots through Bevy

ERROR REPORTING

- ▶ Getting error log from native build is OK, but hard from wasm or mobile
- ▶ Integration with Sentry https://github.com/vleue/vleue_sentry
- ▶ <https://vleue.sentry.io/share/issue/e2578283652d4c449218ccfbdba37c3b/>

The screenshot shows a Sentry error report interface. At the top, there's a header with "Event Highlights" and several metadata fields: release (0.14.0-dev), executable (shader_prepass), Rust: Name (rustc), Rust: Version (1.78.0), Os: Name (macOS), and a timestamp (2023-07-10T14:23:11Z). On the right side of the header are buttons for "Feedback", "View All", and "Edit". Below the header, there's a "Stack Trace" section with the word "panic" in bold. Under "panic", it says "wgpu error: Validation Error". In the "Caused by:" section, it lists several nested notes: "In a RenderPass", "note: encoder = `prepass_command_encoder`", "In a draw command, indexed:true indirect:false", "note: render pipeline = `pbr_prepass_pipeline`", "Incompatible bind group at index 0 in the current render pipeline", "note: Should be compatible an with an explicit bind group layout with label = `prepass_view_layout_motion_vectors`", "note: Assigned explicit bind group layout with label = `mesh2d_view_layout`", and "note: Entry 2 not found in assigned bind group layout". At the bottom right of the report area, there are buttons for "Most Relevant", "Full Stack Trace", "↑ Newest", and an ellipsis (...).

INTERACTIONS

- ▶ New `bevy_ci_testing` command to send keyboard / mouse input to an example
- ▶ Change rendering settings in some examples, take a screenshot for each
- ▶ Render different scenes on mobile to test different pipelines

DEBUG BY PRINTLN / RUN ONE SHOT SYSTEM

- ▶ New `bevy_ci_testing` command to print diagnostics in logs
 - ▶ Can be retrieved from outside execution, to help debug from CI
- ▶ New `bevy_ci_testing` command to run a one shot system
 - ▶ Can run complex checks on entities / components, crash if not valid
 - ▶ Can trigger complex changes in the game to switch state or scene

PR VALIDATION

- ▶ Currently, only run after merge, every 6 hours
- ▶ Run a subset of examples on PR validation before merging
- ▶ Comment on PR with result
 - ▶ Do not block merging on difference, sometimes they are expected
- ▶ PR opened: <https://github.com/bevyengine/bevy/pull/13248>

NEXT STEPS

IMPROVE PIXEL EAGLE

- ▶ Better listing page
- ▶ More comparison modes than pixel by pixel
- ▶ Ignored zones
- ▶ Find more users

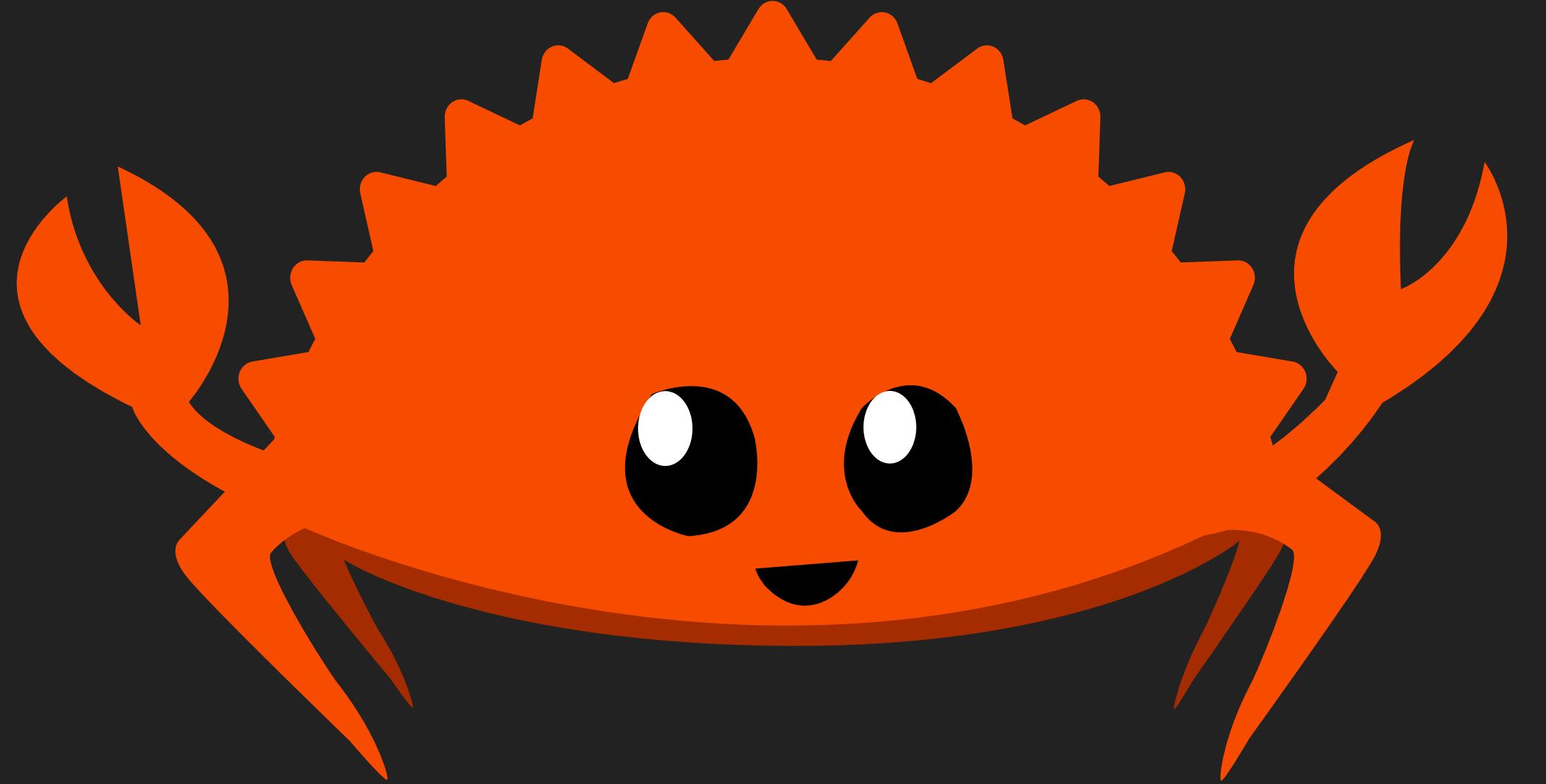


HOW CAN YOU HELP

HOW CAN YOU HELP

HOW CAN YOU HELP

- ▶ Some of the next steps just need contributors
 - ▶ Open PRs!
- ▶ Some of these things would cost money to go further
 - ▶ Sponsor the Bevy foundation!
- ▶ Would visual tests be useful for your project / organisation / company?
 - ▶ Come talk to me!



CLOSING THOUGHTS

NEAR THE END OF WHAT'S POSSIBLE FOR FREE

- ▶ Running every example is slow... when there are 241 examples, and regularly increasing
- ▶ Running faster would need more / bigger runners
- ▶ Enabling Windows again probably need bigger Windows runners
- ▶ Testing against realistic user configurations would need actual hardware

BACK OF NAPKIN COST ESTIMATIONS

- ▶ 20 examples by matrix job, split into 13 jobs per OS
 - ▶ Linux: 250 minutes
 - ▶ macOS: 137 minutes
 - ▶ Windows: 387 minutes
- ▶ Would cost \$13 per run (\$19 with Windows) on GitHub standard runners
- ▶ Around 40 PRs merged per week, \$2080 / month of CI
 - ▶ Larger runners start at 2x price, wouldn't be 2x performances
- ▶ Hardware colocated in a datacenter
 - ▶ Part time admin (on-call, on-site for hardware work): \$2000 / month
 - ▶ Datacenter colocation: \$150 / month

NOT JUST FOR THE GAME ENGINE

- ▶ Most of the setup used to test Bevy is built in and free
- ▶ You can use it in your game!
 - ▶ Test all targets
 - ▶ Test your shaders
 - ▶ Test UI positioning visually
 - ▶ Test transitions