# Planet Mozilla - Latest News

- Firefox Application Security Team: Firefox Security & Privacy Newsletter 2026 Q3 (2026/11/02 23:00)
  Welcome to the Q3 2026 edition of the Firefox Security & Privacy Newsletter. Security and privacy are core principles of Mozilla’s Manifesto and remain at the heart of Firefox’s development. This edition highlights key initiatives from Q3 2026 across three areas: Firefox Security: New security features, platform improvements, hardening efforts, and security integrations in Firefox. Firefox Privacy: New privacy features, enhanced protections, and privacy integrations in Firefox. Community Engagement: Highlights from our security research community and bug bounty program. Preface Note: Some of the bugs linked below might not be accessible to the general public and restricted to specific work groups. We de-restrict fixed security bugs after a grace-period, until the majority of our user population have received Firefox updates. If a link does not work for you, please accept this as a precaution for the safety of all Firefox users. Firefox Security Rapid Releases, Now Even Faster: Since September 2026, Firefox has followed a faster release cycle, bringing improvements, including security fixes, to users more quickly. We also invite you to see and shape what’s next for Firefox, including upcoming security and privacy features. Many recent releases have included more security fixes. We continue to use frontier LLMs to identify and fix security bugs in Firefox to always keep you safe online. We have also changed how we publish advisories: instead of grouping internally identified memory safety vulnerabilities under a single CVE, we now publish a separate advisory for each bug, anticipating a potential change to the CVE Numbering Rules. CVE counts alone are not a reliable measure of security, as software vendors are not required to assign CVEs to vulnerabilities they discover internally. Know when your TLS is broken: Starting with Firefox 157, you will see warnings in the site information panel (the shield icon in the address bar) and in Settings when your connection may not be completely private. For example, if an administrator configures your Firefox to record its encryption keys, you will be able to catch that. Recording these keys could allow another software to read all of your encrypted web traffic without your knowledge. Mozilla Root Store Policy v3.1: We are strengthening transparency and assurance in the WebPKI with Root Store Policy v3.1, which introduces stricter transparency, documentation, and audit requirements for public certificate authorities. Email Client Spam Prevention: Firefox now treats mailto: links as popups, requiring explicit user action before opening the email client. This prevents websites from repeatedly opening email composition windows through JavaScript and overwhelming the user. Restricting Local File Access for Extensions: Starting in Firefox 153, extensions need your explicit permission to access local files through `file://` URLs. Access is off by default for all extensions, including those already installed, so you decide which extensions can access files on your device. Firefox Privacy Fingerprinting Protections: Following a report on Alibaba’s use of audio fingerprinting, Tom Ritter published a data-backed analysis explaining how Firefox’s protections reduce the accuracy of fingerprinting through the Web Audio API. The story received coverage from Ars Technica, The Register, and others. Preventing Link Tracking: Right-clicking a link and selecting Copy Clean Link will now strip tracking parameters across arbitrary subdomains, along with 130 additional tracking parameters. Bringing Containers Directly into Firefox: Previously available as extension, Firefox 153 introduces built-in Containers, helping you keep work, personal browsing, and other activities separate. Each container maintains its own cookies and login sessions, so you could use containers for multiple accounts on the same site, or to limit online tracking by isolating certain websites in their respective containers. Firefox 155 also adds the ability to reorder containers. Tracker Blocking Information on Android: Starting in Firefox 153, the Privacy Dashboard on Android shows which types of trackers Firefox blocks,

giving you a clearer view of how Firefox protects your privacy as you browse. Built-in Ad Blocking to iOS: Introduced on September 1, Firefox's built-in Ad Blocker on iOS blocks many third-party ads and ad-related trackers, reducing clutter and distractions. It works alongside Enhanced Tracking Protection and can be enabled in the 'Browsing' settings in your Firefox on iOS. Community Engagement Advancing Web Security at Dagstuhl: Firefox security experts joined researchers and other browser vendors at the Dagstuhl Seminar on Web Application Security in July, presenting on LLM-assisted vulnerability detection, XSS protections, and web-scale security measurements. Mozilla Security Research Summit: Mozilla's Security-Engineering University Relationship Framework (SURF) is back. On November 20, 2026, we will host another invitation-only summit at TU Delft in the Netherlands, bringing together Mozilla engineers and academic researchers to explore security and privacy challenges and spark new collaborations. The summit takes place immediately after the ACM Conference on Computer and Communications Security (CCS). TLS, QUIC, HTTP/3, and Happy Eyeballs: Max Inden gave a guest lecture at TU Dresden on modern networking protocols, covering the technologies behind fast, secure web connections. Looking Ahead Firefox users receive the security and privacy improvements highlighted in this newsletter through regular updates. If you haven't tried Firefox yet, download it to experience a fast, secure browser and support Mozilla's mission to keep the web open, safe, healthy, and accessible to everyone. Thank you to everyone who helps make Firefox and the web more secure and privacy-focused, by reporting bugs, conducting research, contributing code, or sharing feedback. We look forward to sharing more updates in the Q4 2026 edition. — The Firefox Security & Privacy Teams

- Niko Matsakis: 'Trust but verify'; or, 'the correctness ratchet' (2026/10/30 09:40)
  The name of the game in agentic development is figuring out how to clearly tell agents what you want– and then how to be sure they follow through. Lately I've been tinkering with an interesting technique around this that I'd like to share.1 I call it "trust but verify"2. The basic idea is that sometimes you have things you want to "verify" but they can't really be mechanistically checked. In that case, you let your agent do its best. But you also have it create an evidence file that contains the inputs it drew on to make that decision along with its rationale. Then you gate your CI on checking that these evidence files exist and that the inputs are accurate. Those checks are deterministic and cheap. The rationale itself is natural language and can't be checked. Doing this "trust but verify" technique on its own is helpful. It creates a kind of hill-climbing towards correctness that is stronger than (e.g.) tests alone. It helps to check meta-reasoning like "is the set of tests sufficient to show this requirement is true". Documenting the input also means that if any input should change, the CI check will start to fail, and the agent will be forced to revalidate its reasoning. What I find most interesting though is that "trust but verify" can be used to create a kind of correctness ratchet over time. The idea is that, now that the inputs + rationale are documented, you can have other agents come back and double-check that reasoning, even when the inputs have not otherwise changed. As agents naturally improve, or just due to the randomness inherent in how agents work, they will naturally find things the original rationale overlooked, and you'll be able to address them. When you say "hill climbing towards correctness", what do you mean by "hill-climbing"? Agents are none too smart, but they can be awfully dogged and persistent. If you ask them to climb a hill, they are going to find a way to do it. The tricky bit of course is that they don't always climb that hill in the way you intended. Also, they may not exactly climb it so much as find a route to the top: they may be persistent, but they are also lazy f!@#kers4, and they love to take shortcuts. What if the hills are propositions about your program? The fact that agents are good at climbing hills has got a lot of people in the automated reasoning community5 very excited. After all, what better way could there be to make sure your program does what you expect than to formally prove that it is true? And we've seen that when you set a model loose with a Lean theorem, it can be quite adept at proving things6. But there are a few caveats… Myself, I'm quite excited about tools like Lean but also Verus, Aeneas, and Creusot. I think we're going to be using

them a lot more. And yet, I also think that proving propositions is not the end of the story. Who writes those propositions? Let’s start with the obvious problem. An agent may be good at proving a proposition, but who is actually writing it? Even when something can be formalized, doing that formalization is a fair bit of work. Consider this random example of a requirement from a project I am working on7: A Fleet using Functionality A must be attributed to every Functionality reachable through any number of recorded deprecation steps, while being counted only once when routes converge. This is part of a project that is analyzing fleets across Amazon and looking at how much DRAM they consume. Our goal is to make sure that every large fleet at Amazon has Rust versions of every library it would need to be rewritten into Rust. We want to be able to go to the maintainers of those libraries and say, “there are fleets accounting for X% of Amazon’s total DRAM consumption that use your library; you should make a Rust version”. The point is, while the concept of this DRAM accounting is not complex, formulating it in a precise, mathematical way is not at all trivial. You have to have some notion of transitive closure, talk about the graph of dependencies, etc. There is no way that the lazy, spoiled programmers of the future are going to write a formula like that by hand. Much more likely is that they are going to give high-level direction to the agent and ask it to connect the dots. After all, this is what agents are good for! And in this case, I would trust an agent to get it right. As I said, it’s not really complex, just… tedious. Can they say what you want? And of course this leaves aside that there are many things that are just plain hard to formalize. Consider a requirement like, “a warning must be displayed on every relevant form in a prominent fashion”. Checking this requirement requires a notion of what is “prominent”, which is unlikely to be a mathematical property one can write down. And while I would not trust an agent’s judgment over an actual graphic designer, it is also the kind of thing where the agent can likely do a “good enough” job to catch many failures. And what does it cost to do the proof? The other point is that while agents are good at authoring proofs, it still takes them tokens – and more importantly, time – to do it. And sometimes quite a bit. If developers have to wait a while for the agent to find a proof to a complex property, that better be worth it. If the property you are proving is key to your system, it probably is. But if the cost of a mistake is less than the cost of a proof, people won’t bother, so it’s worth thinking about leaner, easier ways to gain confidence. And what about when you are iterating? One final point is that applying too much rigor too early in the process can be a real problem. This one is tricky because rigor can surface problems earlier, too. But it’s often helpful to start by defining your requirements and things at a higher level and gradually get more precise as you go, tolerating a bit of “looseness” early on when you know there are details you have to nail down, but it’s not important to do so yet.8 A key point for me is that just because you haven’t formalized your requirement all the way to a mathematical proposition doesn’t mean that you’ve got nothing at all. You can often phrase a high-level version of what you want, perhaps one that is overly simplified. In fact, it’s pretty useful to do so, because when you find that your rule doesn’t quite work, that helps you figure out how to refine it into the more precise rule you actually want. Evidence and “trust but verify” The key idea of the “trust but verify” technique is that leaning on agentic reasoning is good, fine actually. But when you do so, you should have the agent explain itself and also have it lay out what inputs and evidence it used to reach that conclusion. Basically, leave yourself a trail so that you (or another agent…) can come back and audit what decisions were reached and on what basis. Following this approach helps in two ways: In the short term, it forces the agent to at least acknowledge your requirement, which generally means they will do a better job at meeting it. In the long term, it lets you go back and double-check the agent’s reasoning, leading to a kind of “correctness ratchet” in which you are generally improving the alignment of “what you have” with “what you asked for”. One example: Habeas Codicem Probably the simplest example of this “trust but verify” technique is Habeas Codicem. This is a small project I made in part as an experiment. The idea is that you write requirements into your docs and then Habeas Codicem adds a test into your system that can only be passed if an agent has written an explanation of why each of those requirements is satisfied. Those

explanations are recorded as evidence files that are committed into your repository along with the code. The interesting bit about Habeas Codicem is that the evidence files contain not only the agent's rationale and judgement, but also a list of the inputs it used to reach that conclusion. And these inputs have to have certain contents. For example, if you have a function that is labeled as part of the requirement's implementation, or a test that is tied to a requirement, then the agent must include that function or that test in its evidence. The final part of Habeas is some skills that instruct the agent to always check with you, the human, before editing requirements, and to confirm agreement on requirements before implementing. Another example: an internal data store Early in the post I mentioned an internal project I am working on at Amazon to analyze which missing dependencies might block fleets from migrating to Rust. This system involves a lot of agent-driven investigation. For example, perhaps there is a Rust version of a Java library, but it only handles some of the features (this is pretty common). Then we need to know whether the fleet's code cares about those features that aren't included. My early attempts at gathering this data were pretty crude. I basically just sat there telling Amazon Quick "try harder, make no mistakes". Not great. I didn't really trust the result. My newer attempts are much more thorough. I ask the agents to produce an evidence record of what data points they looked at and to include a written rationale for how those data points connect. I then test that the data points hold. For example, to determine if the method convertData in some library is called, the agent might register specific lines from .java files showing a call like value.convertData(). Or it might show the results from a grep that show no matches for convertData. The interesting bit is that while the rationale from the agent can't really be evaluated in an automated fashion, the evidence can be. So before we accept any agent-created conclusion, we run a deterministic checker that ensures the evidence is valid (those lines really do appear where the agent said they would) and then records a receipt that the check was done. So the agent cannot just hallucinate evidence. What we are asserting: the agent has done work Habeas Codicem does not prove your code obeys its requirements; nor does my inventory system prove that a given Java feature is used. But it does prove that the agent actually considered the question enough to write out some evidence + a justification. And assuming your agent is not actively malicious, that's actually quite a bit. One thing that we've seen is that agents often "lie" about what work they did. They'll claim to have carefully read through a document when an examination of their tool invocations shows that they've just looked at a few random bytes. Tightening and revalidating rationale One of the ideas I find most interesting in this space is the idea of revalidating rationale. Now that we have evidence files committed to git, we can absolutely have another adversarial agent that periodically pulls up those files and gives them another look. That agent can go looking for holes in the original logic and produce modified and improved proofs or counterexamples. I find this particularly compelling when you think about new and better models that might be more adept at finding holes. Applying this idea to propositions I think this technique suggests how we can begin to tackle the problem of converting "high-level requirements" into formal theorems. Instead of expecting humans to write and review logical formulae, we can ask an agent to do the conversion – and to record its thinking process (with verifiable evidence). Then we keep that evidence around and we periodically revisit it, looking for potential edge cases. We wind up with a kind of 'confidence score' – the more reviews that have approved a given "requirement->formula" translation, the more likely it is to be accurate. The risk of ping-ponging and what it means That said, I've not really put this into practice yet, and I anticipate it may be tricky to get it working right. People have told me to expect that agents may ping-pong a bit: for example, maybe one of them translates your plain English into logic, and then another says "that translation doesn't look right" and takes a new approach. But then yet another agent reverts that and goes back to the original. This seems very plausible– I suspect it can be resolved by adding enough history to recognize that it actually reflects ambiguity in your original statement, and to surface the problem up to you. I do not claim this technique is novel. I am quite sure people have been doing variants of it forever. But it's novel to me, and I don't

think I've heard people talk about some of the potential long-term implications around "the correctness ratchet". ↩ I know this phrase from Reagan, of course, but when I started reading into its background I found it had a more interesting history than I expected. Apparently it is a Russian proverb and was suggested to Reagan as a way to communicate with Gorbachev in familiar idioms. As with "less is more", it's always had that delightfully contradictory feeling ("if you have to verify, are you really trusting?") – and at least the way I interpret it, it's really more like "trust through verification". In other words, I know I can trust you because we've set up the means for me to verify your logic. ↩ I had written, "I don't usually curse at computers", but my daughter (reading over my shoulder) told me, "That is not true, you curse at computers all the time". And she's right! But I realized that it feels different. When I am using some restaurant website that seems purposefully designed not to tell me their opening hours or show me a menu, I certainly say "oh f!@#k you!", but in that case, I am actually cursing at the person who designed that website. I am not cursing at the website itself; it's just an inert vessel. In contrast, when I curse at a model, I am not cursing at the people who trained it. I am cursing at the dang model, because it just seems to be going out of its way to be annoying. Anthropomorphizing is a pain. ↩ God, agents have gotten me so frustrated lately. I'm not usually prone to cursing when sober3 but more than once lately I've found myself typing, "I have no idea what the f!#! you are talking about". This seems to happen slightly more often with Claude than GPT. Claude really likes to develop a kind of cryptophasic shorthand where it drops random words that I used 3 weeks ago as if that will mean anything to me now. "You still have to handle the first-cut case!" ↩ Automated reasoning is the same thing as formal methods, just more appealingly branded. As an aside, lately I've been hearing people talk about Neurosymbolic AI, which I believe is a shameless (and brilliant) way for your SAT solver to jump on the LLM hype train. I suppose it's also a reasonable way to push back against "AI scope creep". While I'm on the topic: it seems to me that John McCarthy's quip ("as soon as it works, no one calls it AI anymore") has been proven false. I would rather people stopped saying "AI" and just said "LLM", but nobody really cares what I want, do they? ↩ Even if they aren't actually true. ↩ I did not actually write this text, of course, an agent did. It's pretty convoluted. Sigh. ↩ This feels reminiscent of Terry Tao's classic post outlining how an experienced mathematician leans on instinct informed by rigor rather than just one or just the other. ↩

- Firefox Tooling Announcements: Happy BMO Push Day! (20261028.1) (2026/10/28 21:15)
  Github Link The following changes have been pushed to bugzilla.mozilla.org: Bug 2081856 - Migrate Bug REST resource to native Mojo API: attachments (attachments, add_attachment, update_attachment) Bug 2080901 - [BMO-ETL] When any table fails to export, the entire snapshot should be rolled back to avoid incorrect results in charts/dashboards Bug 2085028 - log_user_request logs empty request_url/user_agent and server "web" for native Mojo API routes Bug 2067852 - Migrate rst docs to markdown and host inside of bugzilla itself Bug 2085553 - Github logins are failing with JSON error after recent changes to Bugzilla Bug 2078938 - Message/Banner for people creating tickets under mozilla.org :: GitHub: Administration Bug 2086615 - Change the ability to see groups a user is a member of via /rest/user?names= to the group can_see_groups Bug 1883428 - Improve the needinfo email situation (use HTML, drop unnecessary emails) Discuss these changes in the BMO Matrix Room 1 post - 1 participant Read full topic
- Mozilla Addons Blog: Introducing enterprise add-on policies (2026/10/28 17:14)
  We are thrilled to announce that enterprise add-ons will launch on addons.mozilla.org (AMO) on December 1, 2026. In response to input from enterprise organizations that distribute add-ons strictly via Firefox enterprise policies, we are establishing a dedicated enterprise release channel. This tailored channel directly serves enterprise developers' requirements while freeing up resources to continue supporting non-enterprise developers. Why enterprise add-ons need their own policies Many organizations rely on Firefox add-ons to keep their people and data safe:

security tools, data loss prevention, identity and access management, and internal productivity tools. These add-ons work very differently from the ones most people install from AMO: They are deployed, not discovered. IT administrators install them across managed fleets using Firefox enterprise policies. Their reach is large. A single enterprise developer can serve thousands of managed browsers inside one organization. Their release cycle matters. When a security vendor ships a fix, customers expect it to reach their fleet quickly and predictably. Until now, these add-ons went through the same manual review queue as everyone else, under rules written for consumer add-ons. That slowed organizations relying on Firefox and took reviewer time away from the rest of the community without making a meaningful impact. What is changing on December 1 A new enterprise channel: Versions submitted in the enterprise channel follow the new enterprise add-on policies. Enterprise versions stay in the enterprise channel: Versions submitted through the enterprise channel are only available through that channel. They won't appear on AMO or be distributed by Mozilla in any other way. A lighter manual review queue: With a limited set of policies applying to enterprise add-ons, reviewers can spend more of their time on other add-ons. An updated agreement for everyone: To adopt the new policies, all developers must accept the updated Firefox Add-on Distribution Agreement. The same bar for safety: Enterprise add-ons are still Add-ons are subject to Mozilla's Acceptable Use Policy. The full policy text will be published on Extension Workshop. Until December 1, the current policies remain in effect. How to take advantage of it If you build add-ons for organizations: Read the enterprise add-on policies, and get ready to submit your enterprise versions through the enterprise channel from December 1. In a nutshell, you must self-declare the enterprise status via the specific manifest key "browser_specific_settings": { "gecko": { "admin_install_only": true } } If you build public add-ons: Nothing changes in how you submit or how your add-on is reviewed. The only thing you need to do is accept the updated agreement when prompted. Quick Q&A: Who is this for? Who are the enterprise add-on policies for? Developers whose add-ons are built for organizations and deployed to managed Firefox installations through enterprise policies. I'm an individual developer with a public add-on on AMO. Does this affect me? Your submission process doesn't change, but you will need to accept the updated developer agreement. My add-on is used by companies, but people also install it themselves from AMO. Is it an enterprise add-on? Only versions you submit through the enterprise channel are enterprise versions, and those are available only through that channel. To keep your add-on listed on AMO, keep submitting it the usual way. I only build an add-on for my own company's employees. Is this for me? It can be. You can use the enterprise channel, or choose self-distribution, which is often the simplest route for add-ons meant for a smaller population. I'm an IT administrator, not a developer. Do I need to do anything? No. You keep deploying add-ons with Firefox enterprise policies as you do today. Do I really need to accept the updated agreement? Yes. Every developer on AMO, enterprise or not, needs to accept it, because the new policies are part of it. When does this take effect? December 1, 2026. The post Introducing enterprise add-on policies appeared first on Mozilla Add-ons Community Blog.

- Firefox Tooling Announcements: Firefox DevTools MCP v0.10.7 released (2026/10/28 11:28)
  New release of Firefox DevTools MCP. Available on npm at https://www.npmjs.com/package/@mozilla/firefox-devtools-mcp Added New --idle-timeout option (IDLE_TIMEOUT). It sets how many seconds a --connect-existing session waits without tool calls before it releases Firefox. The default stays 1800 seconds (30 minutes). Firefox accepts one client at a time, so a shorter delay lets other clients connect sooner Fixed upload_file_by_uid works again. Since 0.10.5 it did not set any file. It now uses WebDriver BiDi input.setFiles, and no longer changes the style of a hidden file input --connect-existing now fails at once with a clear error when another WebDriver session already holds Firefox. Before, it failed after about 60 seconds with "unknown error" --connect-existing now gives up after 45 seconds when Firefox asks the user to allow the session and nobody answers. Before, it waited forever. The error points at the prompt Shutting down the server while Firefox waits for the user to allow

the session no longer leaves geckodriver running set_viewport_size now only changes the viewport of the selected tab, instead of resizing the whole window. With --connect-existing, the resized tabs are reset when the session closes evaluate_script with a timeout above 10 seconds no longer stops after 10 seconds with a generic BiDi timeout error Changed navigate_history now uses WebDriver BiDi BiDi command errors now include the message sent by Firefox The README explains how to run a --connect-existing Firefox in a dedicated profile, next to your everyday Firefox Full changelog: Release v0.10.7 · mozilla/firefox-devtools-mcp · GitHub Repository and issues: GitHub - mozilla/firefox-devtools-mcp: Model Context Protocol server for Firefox DevTools - enables AI assistants to inspect and control Firefox browser through WebDriver BiDi · GitHub Public chatroom: https://chat.mozilla.org/#/room/#firefox-devtools-mcp:mozilla.org 1 post - 1 participant Read full topic

- This Week In Rust: This Week in Rust 675 (2026/10/28 04:00)
  Hello and welcome to another issue of This Week in Rust! Rust is a programming language empowering everyone to build reliable and efficient software. This is a weekly summary of its progress and community. Want something mentioned? Tag us at @thisweekinrust.bsky.social on Bluesky or @ThisWeekinRust on mastodon.social, or send us a pull request. Want to get involved? We love contributions. This Week in Rust is openly developed on GitHub and archives can be viewed at this-week-in-rust.org. If you find any errors in this week's issue, please submit a PR. Want TWIR in your inbox? Subscribe here. Updates from Rust Community Official Variadic Generics Micro Survey results How to fund Rust contributors Newsletters The Embedded Rustacean Issue #83 Project/Tooling Updates Ring Buffers Project Summary - Redox - Your Next(Gen) OS tar-codec: a fast and secure tar parser for the Rust ecosystem Observations/Thoughts Rust Without Shortcuts: What Safety Actually Costs What stable Rust stands on Where a Flood Goes Discussing a rewrite of XArray Rust features in the future Guidelines for panics in kernel code What distributed systems and end-to-end encrypted apps have in common? CRDTs and Rust! [video] P99 CONF 2026 | Building Ultra-Performant Message Streaming by Piotr Gankiewicz Rust Walkthroughs The heat is on: handling units in Rust Finding panics with Call Graph Analyzer [video] Beyond Vibe Coding: Formal Verification for AI-Generated Python & Rust Software [video] Rust and the GPU from Native to Web: An Introduction to wgpu by Liam Rust Miscellaneous [uz] Announcing Rust Uzbekistan & The Official Uzbek Translation of The Rust Book Crate of the Week This week's crate is offline-protocol, a crate that supplies mesh-networking/peer discovery for a variety of networks, under AGPL/proprietary license. Thanks to Gokul for the self-suggestion! Please submit your suggestions and votes for next week! Calls for Testing An important step for RFC implementation is for people to experiment with the implementation and give feedback, especially before stabilization. If you are a feature implementer and would like your RFC to appear in this list, add a call-for-testing label to your RFC along with a comment providing testing instructions and/or guidance on which aspect(s) of the feature need testing. No calls for testing were issued this week by Rust, Cargo, Rustup or Rust language RFCs. Let us know if you would like your feature to be tracked as a part of this list. Call for Participation; projects and speakers CFP - Projects Always wanted to contribute to open-source projects but did not know where to start? Every week we highlight some tasks from the Rust community for you to pick and get started! Some of these tasks may also have mentors available, visit the task page for more information. sysknife - SetSshdOption reports success while an earlier sshd_config.d drop-in overrides the value it wrote sysknife - On Postgres the daemon anchors the audit chain but every reader reports the anchor unsupported and exits 2 sysknife - ci-local.test.sh stays green when the #555 exit fix, the missing-tool default or the opt-out flag is reverted Tabularis - [Feat]: Import connections from pgAdmin 4 Keepword - Say how many networks have observed a node that isn't located yet Octomind - octomind workflow docs fails with "Is a directory" when ./docs/ exists Octomind - Workflow loop with max_iterations = 0 passes validation and never runs Octomind - ACP: send tool kind, file locations and diffs with tool calls If you are a Rust project owner and are looking for contributors, please submit tasks here or through a PR to TWiR or by reaching out on Bluesky or

Mastodon! CFP - Events Are you a new or experienced speaker looking for a place to share something cool? This section highlights events that are being planned and are accepting submissions to join their event as a speaker. If you are an event organizer hoping to expand the reach of your event, please submit a link to the website through a PR to TWiR or by reaching out on Bluesky or Mastodon! Updates from the Rust Project 609 pull requests were merged in the last week Compiler account for missing turbofish in resolve errors add various #[inline] annotations for simple functions on hot paths that don't get inlined deduplicate DefId encodings in the query cache #[inline(always)] + #[target_feature(enable = "....")] #2 MIR move elimination [4/6]: MoveElimination rustc_builtin_macros cleanup, part 8 optimize Span operations, especially the hot Span::to skip redundant monomorphization checks use the span of the doctest code block fence for figuring out the line number Library const-stabilize {BinaryHeap, BTreeMap, BTreeSet, LinkedList, VecDeque)::{len, is_empty} core: implement char::is_xid_{start, continue} don't try to skip anything in Skip::advance_by(0) implement AsMut<T> and AsRef<T> for ! improve bit permutation codegen process: improve the output of Debug for ExitStatus on Unix stabilize ExitCodeExt for Windows stabilize allocators for Vec::into_iter stabilize String::into_chars stabilize ptr_cast_slice stabilize slice_split_once stabilize {Rc, Arc}::is_unique Cargo timings: peak memory as a column in unit table stop uplifting rlibs on nightly exclude abspath of path deps from -Cmetadata hash test: simplify file content assertions use build-rs for buildscripts Rustdoc include deprecated attributes in rustc_resolve intra-doc link resolution correctly handle inner deprecated attribute on modules don't panic on a nested use group with no path prefix Rustfmt prevent more comment loss in cfg_select! don't lose comments and attributes between cfg_select! predicates, outer/inner blocks, arm body and trailing commas fix cfg_if! error count leaks Clippy wildcard_imports: no longer lint in automatically derives and proc macro expansions add ifs_as_logical_ops lint add lint for map_or_else to map_or fix missing_const_for_fn false positives in const impl fix double_comparisons suggestion dropping required parentheses perf: optimise unconditional_recursion lint Rust-Analyzer automated sync of target feature implications fix NamedTempFile issues on windows complete 'main' 'lib' module in sub module compute ref-match on undotted self do not panic when inline call param usages in macro fix fetch_semantic logic bug introduced in #22272 fix status message for clients not supporting experimental/serverStatusNotification use absolute path in format_smolstr search manifest ancestors for the lockfile when fetching cargo metadata Rust Compiler Performance Triage Another week of positive changes. Few unexpected improvements from cleanups and a handful of targeted optimizations. Apart from instruction count improvements, we also merged a large reduction of query cache size in #164300. Negative changes come mostly from unexpected doc regression, which is currently under investigation. Triage done by @panstromek. Revision range: bd4c6cc4..019b59d8 Summary: (instructions:u) mean range count Regressions □ (primary) 0.8% [0.2%, 6.8%] 25 Regressions □ (secondary) 6.8% [0.0%, 189.1%] 34 Improvements □ (primary) -0.8% [-12.6%, -0.1%] 195 Improvements □ (secondary) -1.2% [-10.6%, -0.1%] 157 All □□ (primary) -0.6% [-12.6%, 6.8%] 220 3 Regressions, 6 Improvements, 3 Mixed 32 artifact comparisons made in total Full report here Approved RFCs Changes to Rust follow the Rust RFC (request for comments) process. These are the RFCs that were approved for implementation this week: 2052-epochs.md: typo 3722-explicit-extern-abis.md: typo Fix libs nominated triagebot.toml Final Comment Period Every week, the team announces the 'final comment period' for RFCs and key PRs which are reaching a decision. Express your opinions now. Tracking Issues & PRs Rust Add let_underscore_drop_guard lint. Deprecate internal function vec::from_elem core: Add assert_matches and debug_assert_matches to the next prelude clarify when Vec::into_boxed_slice() is in-place Partially stabilize ptr_alignment_type as alignment_type std: make Instant a signed Duration on all platforms Lower #[no_mangle] to #[symbol_name]/#[link_name] lang team review for new trait solver Implement Fn traits for safe functions of any ABI cmse: lint on unions crossing the secure boundary Rust RFCs RFC: Add required-targets for workspace package selection Compiler Team (MCPs only) Add support for annotate-snippets's no-graphics output format No Items entered Final Comment Period this

week for Cargo, Language Team, Language Reference or Unsafe Code Guidelines. Let us know if you would like your PRs, Tracking Issues or RFCs to be tracked as a part of this list. New and Updated RFCs No New or Updated RFCs were created this week. Upcoming Events Rusty Events between 2026-10-28 - 2026-11-25 🦀 Virtual 2026-10-28 | Virtual (Cardiff, UK) | Rust and C++ Cardiff Operating Systems Book Club: Paging Mechanisms 2026-10-29 | Virtual | Rust 🦀 Maven Creating a hexadecimal editor in Rust 2026-11-01 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: First Sunday 2026-11-03 | Virtual | Rust 🦀 Maven Solving Real-World Planning Problems in Rust with SolverForge 2026-11-03 | Virtual (London, UK) | Women in Rust 🦀 Community Catch Up 2026-11-04 | Virtual (Indianapolis, IN, US) | Indy Rust Indy.rs - with Social Distancing 2026-11-05 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-11-10 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Second Tuesday 2026-11-12 | Virtual (Nürnberg, DE) | Rust Nuremberg Rust Nürnberg online 2026-11-15 | Virtual | Rust 🦀 Maven Cursed Rust 2026-11-15 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: Third Sunday 2026-11-17 | Virtual | Rust 🦀 Maven Miscompilation in Rust 1.97 2026-11-17 | Virtual (London, UK) | Women in Rust Lunch & Learn: The Harness Wore a Trenchcoat - Building a Rust MCP Server 2026-11-17 | Virtual (Washington, DC, US) | Rust DC Mid-month Rustful 2026-11-18 | Hybrid (Vancouver, CA) | Vancouver Rust Writing a firmware for Xteink ereader in Rust 2026-11-19 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-11-24 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Fourth Tuesday Rust Bookclub Asia 2026-11-03 | Tel Aviv-Yafo, IL | Rust 🦀 TLV In person Rust November 2026 at AWS in Tel Aviv 2026-11-14 | Bengaluru, IN | Rust Bangalore November 2026 Rustacean Meetup Europe 2026-10-28 | Copenhagen, DK | Copenhagen Rust Community Rust meetup #72 sponsored by Google! 2026-10-29 | Berlin, DE | Rust Berlin Rust Berlin on location 🦀🦀 – Edition 019 2026-10-31 | Stockholm, SE | Stockholm Rust Ferris' Fika Forum #31 2026-11-01 - 2026-11-03 | Italy, IN | RustLab RustLab - The International Conference on Rust in Italy 2026-11-02 | Berlin, DE | Rust Berlin Rust Berlin Talks: The next generation 2026-11-04 | Dublin, IE | Rust Dublin Rust 263: Getting Started with the New Rust Driver for Oracle AI Database 2026-11-05 | Milano, IT | Rust Language Milan Digital Model Trains & Rust: Bare-Metal DCC Command Station on ESP32 2026-11-05 | Oxford, UK | Oxford ACCU/Rust Meetup. Epitome of synaptic excellence 2026-11-12 | Geneva, CH | Rust Geneva Rust Meetup Geneva 2026-11-12 | Milano, MI, IT | Rust Language Milan From Curiosity to Capability: Build Your Own Device Swarm 2026-11-17 | Leipzig, DE | Rust - Modern Systems Programming in Leipzig Everlock: Everything Is Git, Protocols Are Just Adapters 2026-11-19 | Aarhus, DK | Rust Aarhus Talk Night at EIVA North America 2026-10-28 | Austin, TX, US | Rust ATX Rust Lunch - Fareground 2026-10-31 | Boston, MA, US | Boston Rust Meetup North End Rust Lunch, Oct 31 2026-11-05 | Montréal, CA | Rust Montréal November Monthly Social 2026-11-05 | San Francisco, CA | Rust LA: NVIDIA NVIDIA x Lawrence Harvey | CUDA Rust 2026-11-05 | Saint Louis, MO, US | STL Rust HTMX and Rust 2026-11-12 | San Diego, CA, US | San Diego Rust San Diego Rust November Meetup - Back in person! 2026-11-14 | Boston, MA, US | Boston Rust Meetup MIT Rust Lunch, Nov 14 2026-11-17 | Los Angeles, CA, US | Rust Los Angeles Rust LA November w/ Hadrian! 2026-11-17 | San Francisco, CA, US | San Francisco Rust Study Group Rust Hacking in Person 2026-11-18 | Hybrid (Vancouver, CA) | Vancouver Rust Writing a firmware for Xteink ereader in Rust 2026-11-21 | Boston, MA, US | Boston Rust Meetup Winter Hill Rust Lunch, Nov 21 South America 2026-11-07 | Planalto Paulista, BR | Rust São Paulo Meetup Encontro do Rust-SP - Agora no Luma Oceania: 2026-11-24 | Barton, AU | Canberra Rust User Group November Meetup If you are running a Rust event please add it to the calendar to get it mentioned here. Please remember to add a link to the event too. Email the Rust Community Team for access. Jobs Please see the latest Who's Hiring thread on r/rust Quote of the Week we're at the 25 minute mark and still bikeshedding tabs vs spaces – Josh Triplett in a Library Team meeting Thanks to Theemathas for the suggestion! Please submit quotes and vote for next week! This Week in Rust is edited by: nellshamrell llogiq ericseppanen extrawurst U007D mariannegoldin bdillo opeolluwa bnchi KannanPalani57 tzilist Email list hosting is sponsored by The Rust Foundation Discuss on r/rust

- The Mozilla Blog: Five Firefox updates you might have missed, and a look ahead (2026/10/27 13:00)
  Some of our latest Firefox updates have had big moments of their own, from the launch of our new design and built-in ad blocking on iOS to Firefox Enterprise and our partnership with NVIDIA GeForce NOW. You might've even spotted a few familiar internet OGs in our latest campaign. Other improvements might've flown under the radar until you open a PDF or move your tabs around. We wanted to take a moment to highlight some of the features and improvements we've been working on, from recent additions you might have missed to a look at what's coming next. What's new in Firefox Open PDFs faster. Firefox's built-in PDF viewer now starts up to 45% faster on desktop, so there's less waiting when you open a document in your browser. Put your Android tabs where you want them. Prefer your tabs at the bottom of the screen? You now have the option to move Firefox's tab bar there. Go to Settings > Customize to choose. Find your spot in a video faster. We've made it quicker to jump to a different point in a video on desktop, whether you're skipping ahead in a tutorial or going back to catch something you missed. Keep your favorite wallpapers handy. Firefox now saves a collection of your custom wallpapers under "Your images" on New Tab on desktop, so you can switch between favorites without uploading them again. Save time filling out forms and keeping up with page changes. Smart Window can now help fill out online forms using information from your chosen tabs, so you don't have to copy details back and forth. You can also ask it to watch a page and notify you when something changes, like a price drop or a new job listing, saving you from manually checking for updates. Both capabilities are available in English in the U.S. and Canada as part of the Smart Window beta. Learn more about the new features here. What's next on the Firefox roadmap When we launched our public roadmap in June, we shared plans to make everyday browsing more useful, flexible and private. We've now updated it with more of what's ahead. We're developing these features in the open, in close partnership with our contributors, and will share more about release timing as each feature gets ready for more people to try.  Here's a look at what's in the works: Keep what you need in view. We heard your request for a way to keep a page visible while browsing. Mini Window will let you pop a tab, or part of a page, into a small floating window, so recipes or instructions stay in view while you work in another tab. Listen to pages. Give your eyes a break. Firefox will soon read articles and other web content aloud on desktop and Android, so you can listen while doing other things – even with your phone's screen off.  Take your reading offline. No connection? No problem. Offline mode on Android will let you save articles ahead of time to read or listen to without internet access, perfect for flights and commutes with spotty service. Turn speech into text on desktop. We're bringing speech recognition to Firefox on desktop, so websites that use Firefox's built-in capability can offer voice input and captions without sending your audio to a server. Those sites receive the text, while your audio stays on your device. After downloading the speech recognition model, transcription will work offline, too. Choose how VPN works for each site. Since bringing free built-in VPN to Firefox on desktop and Android (iOS coming soon), we're now adding the ability to set website rules so you can choose how VPN works on a site-by-site basis. Clear browsing data directly from your toolbar. We're bringing Private Window's familiar button to Classic Window on desktop, so you can quickly choose what browsing data to delete without digging through settings. Keep shaping Firefox with us Try the latest updates and explore the Firefox roadmap for more of what we're working on. If there's something you're looking forward to, or something you'd like us to improve, tell us on Mozilla Connect. Your feedback helps shape what comes next. The post Five Firefox updates you might have missed, and a look ahead appeared first on The Mozilla Blog.
- The Mozilla Blog: Skip the busywork with Smart Window (2026/10/27 13:00)
  The days might be getting shorter, but your to-do list is only getting longer. Whether you're getting a head start on holiday shopping or planning your next career move, Smart Window's newest features will help you keep up.  Firefox's opt-in AI browsing experience can now use information

from tabs you choose to help fill out forms. It can also watch web pages and notify you about specific changes you've asked it to track. Let your open tabs fill in the blanks Job applications ask for your work history; event registrations ask for a professional bio. If those details are already in your open tabs, Smart Window can now use pages you choose to suggest what to enter, saving you extra clicks and some Ctrl+C, Ctrl+V. For example, Smart Window could use information from your online portfolio or professional profile to fill in your work history, or draw on those details to draft an answer to a question like "Tell me about a time when…" Firefox's regular autofill still handles saved information such as your name and address, which can be entered directly into forms without relying on AI models. To get started, click into a form field. If the form has at least four text fields that Smart Window can help fill, you'll see "Fill from sources" appear in the dropdown, along with a recommended source page. Select "Fill from sources" to continue, or click the pencil icon to open "Edit sources" if you want to change or add tabs. Once you've chosen your sources, review the suggestions before filling out the form. You can edit the results before submitting; Smart Window won't submit the form for you. When you choose to use this feature, information from your selected source pages is shared with Gemini 3.1 Flash Lite, the AI model used for detecting the fields and extracting relevant information from your selected tabs. Your model choice for Smart Window chat stays the same, and saved autofill values stay on your device. Learn more about how form filling works. This feature currently supports text fields on web forms and does not save any generated form fill data. We're exploring support for more field types, including dropdown menus, as well as PDF forms and ways to save and manage generated  information for future forms. Give your refresh button a break A watched pot never boils, and refreshing a web page won't make an update appear any sooner. Smart Window can now watch a page for a specific update, whether that's an item going on sale or a new job listing that includes a job title or company name that you're following. Each task can follow up to five pages, so you can keep an eye on several stores or job boards at the same time.  Type "/watch" in the Smart Window chat to set up a task. Describe what you're looking for, add the pages to watch, and choose whether Firefox should check for updates on a daily or weekly basis. Those checks run only while the browser is open.  When there's a match, you'll see an update through the Tasks button in the toolbar. With desktop notifications enabled, Firefox can also notify you directly; click the notification to open the watched pages. You can have up to 15 active tasks at a time, and you can edit, pause or delete tasks. Smart Window will automatically pause tasks after 60 days without a matching change. If 90 days pass without a change, it will stop checking altogether.  Page monitoring uses the Qwen AI model to identify changes that match what you're looking for. Your model choice for Smart Window chat stays the same. Learn more about creating and managing tasks. Try the latest additions These two new features are available in the Smart Window beta on desktop, in English in the U.S. and Canada, and will roll out to additional markets in the coming months. Try Smart Window and share your experience on Mozilla Connect. We'd love to hear how you're using Smart Window to get things done and what you'd like it to help with next. The post Skip the busywork with Smart Window appeared first on The Mozilla Blog.

- Mozilla Privacy Blog: A Reason for Optimism on US Competition Policy (2026/10/27 00:05)
  Mozilla has long called for US antitrust reform in digital markets. After several false starts, real change has yet to materialize. In the meantime, dominant tech players have further entrenched their hold, limiting choice and degrading people's experiences. Integration of new AI services into existing ecosystems is enabling those same companies to perpetuate and accelerate that power. But, there's reason for optimism. Momentum is again building toward antitrust reform, and this time feels different. With support from start ups, middle tech challengers, civil society, and think tanks across the political spectrum – and most importantly, legislative champions in Congress – it's clear that the time is coming. This is a debate about how the largest tech companies control the products and innovations in the market, and access to the services the people use every day. At the center are questions about user choice, interoperability, privacy, security and whether independent innovators have an opportunity to

compete on merit. These topics were the focus of a recent Senate briefing hosted by Bull Moose Project and Public Knowledge, diving in on three competition bills before the Senate Judiciary Committee: The American Innovation and Choice Online Act (AICOA), the Open App Markets Act (OAMA), and the America Act. The discussion brought together policy experts and industry voices, including Mozilla, to explore how competition policy can improve people's experiences online and the ability of challenger companies to succeed. The conversation covered a range of issues, from whether existing antitrust laws are equipped to address today's digital markets, to misguided claims about what might happen if we pass antitrust reform, and the ways that dominant platforms dictate how people discover and use competing products. For Mozilla, an important part of the conversation is what meaningful user choice looks like in practice. People should be able to choose alternatives, and those choices should be respected. But structural barriers can make it harder for independent players to compete on merit, and for people to access options, including independent products that innovate in areas like privacy and security. Competition creates stronger incentives for companies to build better, and to give people the protections and products that they deserve. Antitrust reform is not about punishing companies for being successful. It's about ensuring that independent players can provide the best possible versions of their products, and that people can truly benefit from those innovations. That's why Mozilla, alongside many others, has long championed AICOA and worked to tackle longstanding anticompetitive practices. We're excited to see newly energized support across the public and private sector, and for the next steps in advancing legislation. The post A Reason for Optimism on US Competition Policy appeared first on Open Policy & Advocacy.

- Mozilla Addons Blog: Developer Spotlight: Profile Manager Pro (2026/10/26 22:24)
  If your online activities involve logging in and out of a lot of different accounts — often on the same website — Profile Manager Pro can be a time-saving game changer. The extension provides a one-click login process for multiple online accounts, so you can effortlessly jump between profiles. From social media management to web development, QA testing and more, Profile Manager Pro removes the painstaking friction of constant account switching. Simple UI makes it easy to jump between multiple accounts across various sites. "One of the most interesting aspects of developing Profile Manager Pro is realizing how many different workflows can benefit from efficient account switching," explains its creator, Islam Elsayed. Consider developers who need to save sessions for different roles, such as admins, moderators or regular users, to test functionality; freelancers working across multiple client dashboards; or support professionals investigating account-specific behavior. The potential uses for Profile Manager Pro are practically limitless. Elsayed makes clear that Profile Manager Pro isn't intended to replace Firefox's full browser profiles; rather, the aim is to make it easier to save, organize, and restore individual website sessions within a browser environment. "I think that distinction is important because it illustrates how browser extensions can complement existing browser capabilities by addressing very specific everyday frustrations." While the extension provides a fairly straightforward function, its design and development were anything but simple. Elsayed says his biggest development hurdle was creating a reliable account-switching experience across websites that handle authentication and session management differently. Modern websites often use tangled combinations of cookies, browser storage, authentication tokens and other complex security mechanisms. A technical solution that works perfectly on one website may not function at all on another. "These challenges taught me an important lesson: The best productivity tools often hide considerable technical complexity behind a simple user experience." Another challenge is balancing functionality with user privacy and security. Session-management extensions interact with sensitive browser data, so Elsayed goes to great lengths to ensure user protection — starting with a local-first approach to development. Saved session information stays within the user's browser environment by default. Users don't need to create a Profile Manager Pro account or send their browsing sessions to a third-party server. The extension also offers optional encrypted Google Drive sync, providing even more control over

stored data. “Responsible session management is essential,” says Elsayed. “This is why I believe privacy should be considered throughout the design process rather than an afterthought. Ultimately, my goal is to give users convenience without requiring them to surrender unnecessary control over their data.” As for future development plans, Elsayed frames Profile Manager Pro as an “evolving project.” He intends to focus on optimizing the profile management system and improving backup and sync experiences. “My long-term vision for Profile Manager Pro is to become an increasingly powerful yet lightweight tool for managing online identities and workflows,” says Elsayed. “I believe thoughtful improvements based on real user needs are more valuable than adding features simply for the sake of adding them. For me, the most rewarding part of development is solving those complex problems so users don’t have to think about them.” Do you have an intriguing extension development story? Do tell! Maybe your story should appear on this blog. Contact us at amo-featured [at] mozilla [dot] org and let us know a bit about your extension development journey. The post Developer Spotlight: Profile Manager Pro appeared first on Mozilla Add-ons Community Blog.

- The Rust Programming Language Blog: How to fund Rust contributors (2026/10/26 00:00)
  TLDR: To fund general maintenance, donate to the Rust Foundation Maintainers Fund or sponsor the RustNL Maintainers Team To fund specific features and improvements, sponsor fundable Rust Project Goals To fund individual contributors, donate to their GitHub Sponsors accounts There is always so much to do in the Rust Project! New features to implement, bugs to fix, performance optimizations to make, issues to triage… It takes a lot of work to keep our high bar of quality while also shipping new features and improvements. And the workload keeps increasing with the continuous growth of Rust adoption. As is unfortunately common in open source projects, a large fraction of that work is currently done by volunteers. That causes various issues, such as people burning out or complex features taking many years to ship. Unsurprisingly, one of the best ways to help improve that is to get more people who work on Rust funded! This year, several funding initiatives started ramping up, and we already experience many positive effects. They helped us improve the maintenance status of critical Rust projects and make significant progress on several long-awaited features, such as allocators, the never type, the next trait solver or the Polonius borrow checker. However, only a fraction of people working on Rust are currently funded for their work. We know that there are Rust users, both individuals and companies, who would like to help improve Rust by funding it, but it might not always be obvious how exactly to do that. This post thus provides suggestions on how you can support the development of Rust in three ways. If you're unsure which option fits, or would like to discuss funding on behalf of your company, please contact the Funding team. Fund general maintenance Maintenance is a continuous process required to keep Rust (or any software project for that matter) working. It consists of two main activities. The first is general housekeeping, such as ensuring that Continuous Integration works, dependencies are kept up-to-date and issues, bugs and security incidents are promptly triaged and resolved. The second is performing refactorings and code reviews, which ensures that the codebase will be able to evolve and continuously ship new features, without accruing extensive technical debt. Having quality maintenance is crucial for the long-term health of Rust, yet it is typically very difficult to find funding for it; funders often want to sponsor specific new features, rather than just "keeping the lights on". However, without sufficient maintenance, it becomes much more difficult, and much slower, to ship new features. Improving the maintenance of Rust thus also helps us ship features and improvements faster, though the effects are sometimes only observable over longer periods of time. Funding maintenance is a great choice if you want to let us decide how best to improve Rust. The Rust Funding team continuously looks for Rust teams and projects that require help with their maintenance efforts, and allocates available funds to Rust maintainers who could help improve the situation. To fund Rust maintainers, consider donating to the Rust Foundation Maintainers Fund or sponsoring the RustNL Maintainers Team. Fund specific features and improvements There are always many Rust improvements and new features that are in progress. We track some of those initiatives through Rust

Project Goals. Each Goal describes a specific improvement to Rust: what it is, who is going to develop and drive it forward and which Rust teams are willing to help reviewing and maintaining it. Some of those Goals are looking for funding. Either to help their owner dedicate time and focus to push the given feature forward, or to fund maintainers who will perform refactorings necessary to unblock that feature and review its implementation. Often funding is required for both. When a Goal is properly funded, the probability that it will achieve its milestones is much higher, which then translates in the given improvement being shipped sooner to Rust users. Funding Project Goals is a great choice if you want to help push forward specific Rust features or Rust toolchain improvements that will benefit you or your company. To fund contributors working on Project Goals, consider taking a look at Goals that need funding and reaching out to their funding contact. If in doubt how to proceed, reach out to the Rust Funding team. Fund individual contributors There are hundreds of people in the Rust Project who continuously work on improving Rust. Some of them have personal fundraising profiles (e.g. on GitHub Sponsors), which allows you to support them directly. There might be various reasons for supporting a specific person. Maybe they are maintaining a crate that you love to use, maybe you like reading their blog posts, or maybe you are simply a fan of the work that they do. Whatever the reason, we are sure that they would appreciate your support! Funding individual contributors is a great choice if you want to directly support Rust contributors whose work you appreciate. To fund individual Rust contributors, check out their GitHub profile or personal website to see what options there are for supporting them. You can also examine our list of Rust Project members who can be supported through GitHub Sponsors[1]. Conclusion We hope that this post is useful to anyone who would potentially like to invest funds into making Rust better. You can also find a shortened version of the "how to fund Rust guide" at the Funding page of the Rust website, or check out a RustConf 2026 talk by Niko Matsakis about how Project Goals and their funding works. If you are a company looking to fund the development of Rust, consider joining the Rust Commercial Network. Joining the network is free, and Rust Foundation membership is not required. It can help you find opportunities for funding improvements in specific areas related to Rust usage (e.g. user interfaces, language interop, supply chain security, etc.) and collaborate on funding Rust development with other companies. There are of course other ways of helping Rust other than donating funds. For example, you can check out Project Goals that are looking for contributors or more generally check our contributor onboarding guide. Companies can also help by giving employees paid time to contribute upstream, or by hiring Rust contributors to do that work. One possible model is the Engineer in Residence role, that was recently established in the Rust Foundation. Note that this list is not exhaustive; there are some Project members who accept donations through GitHub Sponsors (or another fundraising platform), but did not opt into being listed on this page. ↩

- The Mozilla Blog: Firefox reunites meme royalty to remind you what the internet used to feel like (2026/10/25 11:58)
  A group of familiar faces from the early web just got back together to ask an important question: What happened to the internet? Sure, there was a time when it was a little less polished, and a lot less speedy, but it was yours. Overly Attached Girlfriend may have wanted to know everything about you, but your browser didn’t need to. Dancing Matt boogied all over the world, without any corporations tracking him or selling his personal data. And while Annoying Orange is incredibly annoying, he’s not as annoying as forced AI features. These beloved early internet personalities remind us of a time when surfing the web felt open, organic, and most importantly, personal. And good news: With Firefox, it still can. That’s why we’re teaming up with these icons to remind people what our browser offers, and show those who wandered off why now is the time to come back. The team behind the comeback For Mozilla Chief Marketing Officer John Solomon, the aim of the campaign started with winning back people who originally loved and used Firefox. “I kept going back to Firefox’s heyday, when the internet was really fun, weird, and interesting. The challenge was: how do you lean on nostalgia and heritage, but not be stuck in the past?” The campaign was created by We Believers, one of the

world’s most awarded independent agencies, with production handled by nobodys and direction by the Sunny Sixteen collective. “We wanted to create an idea that truly belongs to Firefox — a challenger browser that helped shape the internet’s golden years and stayed independent as Big Tech took over. The goal? Remind former users that what they loved about Firefox matters more than ever now,” said Gustavo Lauria, Founder and CCO at We Believers. Solomon said the finished campaign achieves the goal of embracing Firefox’s history without living in it: “Our partners really cracked it. We wanted to remind people of what made the internet so wonderful in the first place. These are people who helped make it strange, creative, surprising, and fun. None of them were trying to go viral. They were normal, everyday people who were using the internet. That’s the way the internet should be and why we feel it’s a great moment for Firefox. Always independent and unlike big tech, not trying to extract more from everyday people.” The internet you remember is still out there Whether you used Firefox back in the day, or are simply looking for a better browsing experience, we’re still here, staying true to the values we were built on. We believe the internet should be open, accessible, and shaped by the people who use it, not just the platforms that profit from their attention and data. This means a web where you can explore without feeling tracked or used for profit, where you decide for yourself how new technology fits into your life, and you can trust that your browser protects you by default. Firefox has built-in privacy protections to help limit tracking, lets you decide if AI features appear and customize how they behave, and uses open source code that anyone can see. To learn more about what sets Firefox apart, check out five reasons to choose the browser that puts you first. We know a better internet isn’t built by finding new ways to capture your attention or information. It’s about giving you more freedom to make the web your own. The web is yours to explore The internet has changed a lot since Numa Numa Guy first danced his way into our collective memory. But the things that made it magical – creativity, curiosity, community, and the freedom to explore – are still worth protecting. Firefox has championed those values since the beginning, and we’re not giving up on them now. That’s why we’re celebrating the people who helped make the early web what it was, while reminding everyone that a more open, human internet is still an option. The early days of the internet may be behind us, but the spirit doesn’t have to be. Experience it for yourself. Now more than ever: come back to Firefox. The post Firefox reunites meme royalty to remind you what the internet used to feel like appeared first on The Mozilla Blog.

- Firefox Tooling Announcements: Firefox Profiler Deployment (October 22, 2026) (2026/10/22 17:01)
  The latest version of the Firefox Profiler is now live! Check out the full changelog below to see what’s changed: Highlights: [fatadel] Make CompositorScreenshot markers schema-based (#6261) [fatadel] Expose and clear URL-imported call tree search in the CLI (#6387) [Markus Stange] Make the scrollable editor view in the source view cover the full height of the panel, even if the source code is short. (#6405) [Nazım Can Altınova] Add `–js-sources-from-func-names` to `profiler-edit` (#6394) [fatadel] Use fractional seconds with adaptive precision for CLI marker timestamps (#6384) [Markus Stange] Switch to TypeScript 7 (tsgo) and replace ESLint with oxlint (#6354) Other Changes: [Nazım Can Altınova] Fix the upload progress bar not showing in light mode (#6348) [fatadel] Bump profiler-cli version to 0.12.0 (#6416) [fatadel] Sync: l10n → main (October 22, 2026) (#6415) [Markus Stange] Some preparations for a function list panel (#6401) [Nazım Can Altınova] Look for a source file’s address proof in all of its frames (#6407) [Nazım Can Altınova] Silence profile loader logs in profile-querier-url tests (#6409) [Markus Stange] Use inverted transforms in the tests for inverted call node paths. (#6393) [Markus Stange] Accept more fields from symbolication. (#6373) [Markus Stange] Support index source maps. (#6399) Big thanks to our amazing localizers for making this release possible: el: Jim Spentzos uk: Lobodzets, uya2521, Ihor Hordiichuk Find out more about the Firefox Profiler on profiler.firefox.com! If you have any questions, join the discussion on our Matrix channel! 1 post - 1 participant Read full topic
- Firefox Nightly: Three Cheers for Volunteers! – These Weeks in Firefox: Issue 211 (2026/10/22 16:56)

Highlights Here are some big-ticket volunteer contributions this week: Thank you to volunteer contributor Cornelius, who made console evaluation able to re-declare let and const variables in our DevTools! Thank you to F3tch, who landed several patches to allow setting of camera and microphone permissions, reject zero-sized screenshots, and add a vendor-specific WebDriver BiDi command for listing installed extensions, as well as update cookie test expirations. Thank you to SMRehan, who landed a (disabled by default) mode that lets Picture-in-Picture open when the browsing window is occluded (as opposed to when changing tabs). This can be tested by setting media.videocontrols.picture-in-picture.enable-when-occluded.enabled to true In Nightly 160, we have enabled the playback speed feature for Picture-in-Picture by default: Thank you to volunteer contributor Simon Rosen for the bulk of the initial work on this! The custom wallpaper library for newtab is now available, starting in Firefox 158, so you can now store multiple custom wallpapers: Friends of the Firefox team Resolved bugs (excluding employees) Script to find new contributors from bug list Volunteers that fixed more than one bug :Vincent Amadi any1here Duncan McIntosh [:dmcintosh] f3tch [:f3tchcodes] ianlewisuk1 Ivan Smagin Khalid AlHaddad Logan Rosen [:Logan] Mayank Bansal Vir Patel New contributors ( = first patch) Ahmed ElGendy: Add a `(inline-)grid-lanes` badge in the markup view for element with display: (inline-)grid-lanes Mubashshir: Extension icon missing in old sidebar switcher Anton Bershanskyi: Introduce browser.runtime.PlatformArch.ARM64 Amadi: Clicking the New Tab button does not show pressed appearance with vertical tabs Not localized search matches in Devtools Remove unused WebsiteFilter module getter from browser.js Caleb Pickard: New sidebar revision does not have “By Most Visited” as a sort option for History Daniel mungai chege: Add autocomplete for CSS random() dchen705: syntax and handling of async runtime.onMessage listeners is confusing Diego: Pressing ctrl+shift+numpad1 causes tab to be moved to end of tab list regardless of NumLock f3tch [:f3tchcodes]: “WebDriver:TakeScreenshot” doens’t throw “unable to capture screen” error if height or width of captured image is 0 Increase expiry date range for network cookie tests [wdspec] ValueError: “list.remove(x): x not in list” for “self.event_listeners[name].remove(fn)” [firefox-devtools-mcp] Support webextensions.moz:listExtensions command in WebDriver BiDi Allow “permissions.setPermission” and “WebDriver:SetPermission” to set camera and microphone permissions Ole Baumann [:darkintaqt]:  Drop the redundant level from UrlbarUtils.getBackspaceBlock’s return value ianlewisuk1: Add Nova icons for Firefox :: Messaging System The corrupt_file telemetry metric counts session files with an incompatible format version as healthy files Jakub Stasiak [:jstasiak]: Early WebSocket messages can be missing in the Network tab Jaron: Show the remote control stripe on the address bar only  Lavindeep: Subprocess fcntl binding is not variadic, causing incorrect pipe flags and possible native-host launch failures on Apple silicon  Sho Sakuma: Fix WP_WIN and WP_MACOSX typos in LaunchChild’s #endif comments Raúl: Remove dead PanelUI-fxa-menu-setup-sync-container from the account menu  Jason Salerno: Remove the unused tooltip attribute from the urlbar view markup Ivan Smagin: Improve pref-handling in xpcshell test ‘test_ext_contentscript_module_import.js’ Add test for unknown content dialog for downloads with a mimetype that doesn’t have an associated file extension Nazeer Ahmed Shaikh: Instrumentation tests on artifact builds do not check `object` metric schema Vir Patel: Make DisplayMenuBar pass a string to runOncePerModification Handle the “cleared” perm-changed notification in the permission exceptions dialogs Valerie Racine (:v-racine): The corrupt_file telemetry metric counts unrecognised session file read errors as healthy files  Yeet: Saved tab groups not actually saved if browser crashes Project Updates Add-ons / Web Extensions Addon Manager & about:addons Removed the colorway migration code and the about:addons UI informing users about colorways on AMO – Bug 1990883 Thanks to Jason Birchall (from AMO engineering) for collaborating with us on this. As part of about:addons Nova restyling followups Parameterized the built-in theme preview images, and fixed the default theme previews rendering with the wrong colors when about:addons uses the opposite color scheme (regression introduced and fixed within Firefox 158) – Bug 2070289 / Bug 2074428 Thanks to Emilio Cobos Álvarez for driving the theme preview parameterization.

Fixed the Nova theme preview gradients being mirrored in RTL locales – Bug 2073245 Hid the (non-localized) theme card descriptions in the about:addons themes list view, fixed in Firefox 158 and uplifted to Firefox 157 – Bug 2071337 Fixed the per-add-on-card message bar styling in about:addons to better match the Nova designs – Bug 2079672 Avoided redundant AMO metadata update checks when the app locale is unchanged – Bug 2052032 Added options to AddonManager.getInstallForURL to validate the add-on type, id and/or version before installing – Bug 1973071 Dictionary add-ons are now required to be signed, starting in Firefox 160 – Bug 1753276 WebExtensions Framework As part of work to refactor about:newtab built-in widgets in their own privileged extensions, allowed about:newtab pages to load extension pages as subframes without listing them in web_accessible_resources – Bug 2070235 Improved pref handling in the test_ext_contentscript_module_import.js xpcshell test by using pref_set – Bug 1986064 Thanks to Ivan Smagin for the cleanup in the content script module import test. WebExtension APIs Fixed publicSuffix.isKnownSuffix to throw for invalid host names, in line with the API proposal, MDN and Chromium – Bug 2066620 Added runtime.getVersion() – Bug 1992418 Thanks to Carlos for implementing the new runtime API method. Introduced runtime.PlatformArch.ARM64 (replacing AARCH64) to match Chromium, landing in Firefox 160 – Bug 2080310 Thanks to Anton Bershanskyi for the runtime.PlatformArch rename. Stopped exposing non-browser windows in windows.onFocusChanged, fixing a “gBrowserInit is undefined” error hit through tabs.query – Bug 1634240 Fixed sessions.forgetClosedTab() forgetting the wrong tab when the window has a closed tab group – Bug 2076684 Thanks to Dão Gottwald for investigating and fixing this bug hit by the sessions API. Fixed extension menu items being unreachable by accesskey, including items with Chinese, Japanese or Korean labels – Bug 2074182 / Bug 2075142 Thanks to Dão Gottwald for driving these menus accesskey fixes. Added a warning when sendResponse is called by an async runtime.onMessage listener, landing in Firefox 160 – Bug 1877223 Thanks to dchen705 for the runtime.onMessage warning. DevTools External contributions Jakub Stasiak fixed an issue in the Netmonitor where we could miss early WebSocket messages (#1719394) Amadi localized string in Debugger search panel (#2035151) Daniel mungai chege added autocomplete in the Rules view for the random() function (#2072513) Ahmed ElGendy added (inline-)grid-lanes badges in the markup view for element with display: (inline-)grid-lanes (#2076779) Khalid AlHaddad removed an unused argument in the KeyShortcuts constructor (#1979386) Team/Project Update Florian Quèze [:florian] made DevTools splitters keyboard accessible (#1860733) and fixed numerous other accessibility issues across the toolbox (#2071228, #2071230, #2071232, #2071233, #2071236, #2075809, #2075820) Leo McArdle [:leo] started modernizing the DevTools Settings panel (#2072146, #2072216, #2074393, #2074399) Nicolas Chevobbe [:nchevobbe] Handle percentage in CSS explainers tooltip (#2041621) and show the tooltip for single non-pixels values (#2074405, #2064400) (pref: devtools.inspector.css-explainers) Hubert Boma Manilla (:bomsy) is working on adding support for speculation rules in DevTools. Prefetched requests can now be identified in the Netmonitor (#2077519) and he plans to add more information about speculations rules in a dedicated tab in the Application panel (#2077995, #2078853) WebDriver Beatriz Rizental improved the Marionette harness to accept multiple –tag arguments similar to other in-tree harnesses. Khalid AlHaddad improved the PollPromise to throw an error by default and not only log a warning when it times out. Alexandra Borovova added support for the imageSize argument to the browsingContext.captureScreenshot command, which allows clients to specify the maximum size of the resulting screenshot image. Alexandra Borovova added support for the destinationFolder argument to the browsingContext.startScreencast command, which allows clients to choose the folder where the recorded video file is saved. Alexandra Borovova implemented the support for the prefers-color-scheme parameter in the emulation.setMediaFeaturesOverride command, which lets clients emulate dark or light mode on a website. Henrik Skupin improved the handling of recommended preferences for automation to use the default branch, preventing them from leaking into subsequent Firefox sessions. Henrik Skupin fixed a bug where accepted server sockets still had

Nagle’s algorithm enabled, causing up to 40ms delays for back-to-back writes on Linux. Henrik Skupin improved geckodriver’s shutdown handling by waiting for Firefox’s hang monitor to terminate hung processes before force-killing them. Henrik Skupin added a temporary workaround to prevent Glean from causing shutdown hangs for short-lived Firefox processes. Holger Benl added machine-readable CDDL for all Firefox WebDriver BiDi vendor extensions (“moz:” commands, events, parameters, capabilities…). Privileged context support was added to browsingContext.locateNodes and input.performActions by Holger Benl, and to browsingContext.captureScreenshot by Julian Descottes. Julian Descottes fixed the network.getData command to handle 304 – Not Modified responses with no timeouts. Fluent Shout out to flod who’s started to chip away again at migrating .properties files to Fluent! For the first time since we started tracking, there are fewer than 4000 .properties strings in the Firefox source code! Migration Improvements spohl is working on a stack of patches that’ll update the migration workflow for macOS 27 to reduce the number of data access permission warnings that users see New Tab Page We’re testing a few different layout variations in the wild – one of them is a few different takes on “spaces”: The MCAB search input is currently being deployed to a small population in the wild to see how it works / is received A train-hop is scheduled to be deployed this week which should make widgets available for all users (though for most users, these will be disabled by default – they’ll be available to enable in the customize panel). Reem Hamoui removed the Enzyme/Karma test infrastructure from the New Tab Page, which is a CI/dev tooling change with no direct runtime user impact but should reduce test maintenance and flakiness for future user-facing fixes. Jack Brown switched New Tab message visibility checks to an IntersectionObserver so messages below the fold are no longer marked as activeNotifications, preventing offscreen messages from being targeted and avoiding unnecessary JS evaluation when not visible. Jack Brown cleared activeNotification targeting when navigating away from about:newtab so message targeting no longer persists after leaving the page, preventing carry-over interactions and incorrect metrics attribution. akulyk [a11y] fixed dialog focus containment on the New Tab Page by enforcing a focus trap and proper Tab/Shift-Tab handling, restoring expected keyboard and screenreader workflows for modal dialogs. Scott fixed pocket.click and topsite.click to trigger context menus correctly, restoring right-click/long-press context menus on Pocket tiles and Top Sites so users can access actions like open-in-new-window and remove. Maxx Crawford updated per-widget Jest fixtures for Stocks, Privacy and Picture of the Day to omit the widgets.enabled flag in test setups Per-widget Jest fixtures for Stocks, Privacy and Picture of the Day omit widgets.enabled, which aligns test fixtures with runtime enablement behavior and reduces CI flakes and regression risk for those widgets; there is no runtime UI change for end users but widget rendering tests should be more reliable. Maxx Crawford removed the Sports widget from the New Tab Page and widget picker, which directly removes sports headline/cards from users’ NTP experience (users who had Sports selected will no longer see it) and slightly reduces client-side DOM/paint work and widget surface area for future NTP changes. It also removed about 1MB of image / video assets from the omni.ja / newtab XPI. Maxx Crawford added region/locales dynamic control for the Weather widget, allowing users to change the Weather card’s region/locale on New Tab so forecasts and localized labels update immediately. Herraj Luhano updated the POTD API contract to return the previous day’s POTD, enabling the New Tab image module to fetch and display yesterday’s Picture of the Day to avoid empty or stale image slots. Nina Pypchenko [:nina-py] re-ordered New Tab settings in about:preferences, improving discoverability by placing New Tab controls in a more consistent, prominent location for users managing Start Page options. Dre standardized context menu button UI across all widgets, unifying markup, CSS variables and ARIA to make context-menu hit areas and keyboard focus consistent across widget surfaces. Dre standardized header typography across all widgets, aligning font sizes, line-heights and tokens so widget headers render consistently on New Tab pages. Maxx Crawford updated WallpaperFeatureHighlight to allow for alternate strings to be injected so the New Tab wallpaper feature highlight UI can show injected copy variants (for localization or experiments), which

directly changes the text users see in the wallpaper picker/highlight flow and should be monitored for clarification/regression in impressions and CTA click rates. David Durfee landed a meta change to enable score-based selection for sponsored tiles in the New Tab experiment, which replaces the previous selection heuristic with a ranking/scoring pipeline so experiment cohorts may see different sponsored tiles on New Tab (affects which ads/content are shown, sponsored impressions/CTR, and A/B metrics for desktop and mobile New Tab surfaces). Nina Pypchenko renamed the Stocks widget to “Finance” on the New Tab Page, updating the activity-stream UI/i18n label users see in the widget header. Brendan Birdsong adjusted the newtab score-based selection to sum day counts from remote-settings config, which can change which cards/tiles are selected and their ordering for users depending on remote config values. Scott Downe removed the discoverystream.optIn-region-weather-config pref from newtab, a config cleanup that does not change runtime New Tab UI or user-visible weather opt-in behavior. Irene Ni normalized the layout_name value in New Tab DSCard impression telemetry, ensuring consistent impression records for analytics and A/B experiments without altering the UI seen by users. Scott Downe removed the telemetry.structuredIngestion.endpoint pref from newtab, removing a pref-based override for structured-ingestion endpoints (affects developer/testing overrides, not regular user behavior). Irene Ni removed unused ImpressionStats dispatches from New Tab DSCard clicks, eliminating redundant telemetry dispatches on DSCard click paths with no UI change. Irene Ni normalized Top Sites telemetry keys to their Glean extra names, stabilizing telemetry naming for Top Sites without changing user-visible behavior. Dre fixed a slight flash when reusing a previously uploaded wallpaper in the New Tab Page, removing a visible flicker for users who set custom uploaded wallpapers. Maxx Crawford added a debug pref/setting to revert region controls, enabling targeted rollbacks/enabled testing for affected-region UI behaviors when the pref is toggled. Jared Wein fixed TypeScript errors in newtab/lib blocking inclusion in newtab’s tsconfig.json, resolving a build/tooling blocker (developer-only change; no direct user impact). Irene Ni made the Spaces arrow-buttons variant background panel occupy the full page height on the New Tab Page, removing a visual gap and ensuring arrow controls stay aligned with page scroll. Alexandre Hanot fixed Ads client initialization on Firefox 157 and older so the Ads client now initializes instead of throwing errors on legacy builds, restoring ad content on affected users’ New Tab Pages. Dre prevented the Clocks widget menu button from overlapping clocks on hover by adjusting z-index and hover layout, restoring visibility and correct hit targets on the New Tab Page. Irene Ni enabled a widget in the first content position of New Tab sections, updating New Tab rendering logic so a prioritized widget slot is populated in the first content position and becomes visible by default to users. Jared Wein enabled type checking for newtab/lib with exception for failing files, adding incremental static-analysis/CI checks to catch regressions in newtab code without changing UI behavior today. Maxx Crawford set default enabled value to false for all existing widgets, changing widget metadata/prefs so existing widgets are opt-in and will no longer render for users unless explicitly enabled. David Durfee added differential privacy noise to scores in newtab score-based selection, anonymizing per-user signals which will alter ranking/discovery behavior on the New Tab page while underscoring privacy guarantees. Brendan Birdsong moved newtab score storage to PersistentCache, persisting scores across sessions and reducing recompute/lookup latency so personalized selections are more stable and faster to load. Jared Wein fixed more newtab/lib TypeScript errors so 11 more files can be type-checked, enabling type-checking across newtab/lib (11 files) and lowering regression risk for New Tab Page behavior. Dre used useWidgetTelemetry in the Weather widget to standardize telemetry for the Weather widget (New Tab Page), improving diagnostic signals that help us find and fix user-facing widget issues faster. Maxx Crawford turned off all New Tab widgets in automation with one pref, allowing automation to run with widgets disabled to reduce flakiness in tests that interact with New Tab UI. Dre updated the Focus Timer widget to use useWidgetTelemetry, so Focus Timer telemetry now respects widget-level prefs and will avoid sending unguarded pings—privacy impact only, no UI change. Nina Pypchenko [:nina-py] fixed transparent New Tab dialogs when the

train-hop is applied to Firefox 158 and 159, restoring dialog opacity/CSS so newtab overlays are readable again when train-hop is enabled on those releases. Picture-in-Picture kpatenio fixed a bug on Windows and Linux where clicking a previous control overrides the Space play/pause shortcut. kpatenio landed a few patches to make keyboard navigation better when the playback speed control is enabled (Bug 2055070, Bug 2074319). As of Nightly 159, the PiP player UI has been updated for Nova. Aside from being visually consistent with Nova, scrubbers and settings panels now have better contrast on High Contrast Mode. Search and Urlbar Urlbar Dao landed a patch to show the “Search tabs” keyboard shortcut in the search mode switcher menu: 2078113 Logan Rosen changed the Accel+Shift+A shortcut from the Add-ons Manager to Search Tabs: 1666654. Dao followed up by notifying people who used the old shortcut that it moved and revised the wording of that message: 2077823, 2079208 James added an option to skip the menu when tabbing through the address bar, in both the address bar and the preferences page: 2064708, 2068609 Dao fixed a bug so that entering an address in search mode now navigates to that URL: 2036347 Dao fixed two urlbar.bounce telemetry bugs: 2069403, 2076827 Caleb fixed a bug where the address bar results dropdown covered the calculator tail150 game: 2077358 Daisuke fixed a bug where search terms in the address bar moved when pressing Enter or when edited: 2074921 Dao fixed the search engine button being cut off in the Search dropdown with Nova enabled: 2074871 Daisuke fixed the search switcher’s active button color being too strong with low text contrast in Nova: 2068504 Emilio fixed the Quick Actions background button missing in the Light theme on Ubuntu in Nova: 2076191 Moritz continued with MCAB (multi-context address bar) follow-on work: 2078231, 2078658, 2078672. Moritz also enabled TypeScript type checking for UrlbarView and UrlbarBaseInput: 2077545, 2077543 Dao significantly improved the address bar docs, including how to add a new search surface end to end, how results and providers work across the process boundary, and how to debug and log the address bar:  Search Caleb fixed a bug where the address bar swallowed the first character typed into it if you typed too quickly: 1953361 Mark continued working on the new search config, search-config-v3: 2063279, 2063264 Places Dao fixed a bug where the bookmark creation/editing panel had no keyboard shortcut mnemonic for the Save button: 1877510 Storybook/Reusable Components/Acorn Design System Dialogs in about: pages now match Nova designs more closely Moz- widgets should now have types found when querying the element with document.querySelector(“moz-button”) and TS is configured A11y fixes for profiles dialog, page nav keyboard handling (no longer steals Ctrl+arrows)

- Mozilla Localization (L10N): L10n Report: October Edition 2026 (2026/10/22 16:41)
  Please note some of the information provided in this report may be subject to change as we are sometimes sharing information about projects that are still in early stages and are not final yet.  Welcome! New localizers Welcome Samin Yasar, a new contributor who took the initiative to restart localization and is bringing a renewed energy to the Bengali community. Are you a locale leader and want us to include new members in our upcoming reports? Contact us! New community/locales added Persian (fa) is now enabled on addons.mozilla.org (AMO). Congratulations and thank you to the Persian localization community for reaching 100% on both AMO projects in Pontoon. What’s new or coming up in Firefox desktop Felt privacy and protection In upcoming versions of Firefox, there will be a number of updates focused making the privacy protection features of Firefox felt by the user, in addition to communicating those features in an easy to understand way. Translation has been ongoing and some of these will be released from Firefox 159. Private Window redesign A redesigned Private Window experience when opening a new Private Window will be rolled out in an experiment to users starting in Firefox 159, but can be viewed by opening about:config and setting the browser.privateWindowRedesign.enabled flag to true, then opening a new Private Window. Fire button in Classic Mode A button that allows users to start fresh in a browsing session, they can clear browsing data in a single click. Strings for this will mostly involve updated wording for existing strings around what data to clear. This is one of many features mentioned in the Firefox roadmap. Privacy Panel The privacy panel that shows

your protection status for a website (accessible by clicking the shield icon in your address bar) will be getting a complete redesign to more prominently display the number of trackers blocked, important security warnings, and critical site issues. This is still in progress but expect to see strings soon. Mini Window An early version of this feature is coming up in Firefox 160 and strings are landing now. The Mini Window gives you the ability to detach a tab or custom section into an always-on-top, scaled-down frame. It expands on the familiar Picture-in-Picture functionality for general web content, making it effortless to keep documentation, messaging apps, or references visible as you work. You can find additional details and a demo of Mini Window on Mozilla Connect. What's new or coming up in mobile With our recent switch to releases in 2-week cadence, mobile will see a slew of small but mighty features pop-up in short succession. Some of the recently launched or soon-to-be revealed features include: Firefox for Android Read aloud [AI feature]: Catch up on your reading list on the go by having Firefox read your articles out loud. In reader mode, you will be able to activate the listening feature, select your preferred reading voice and adjust the playback speed, as well as scrub backwards and forwards, so you can skip to the juicy bits quicker. Data consent for extensions: We always want to make sure users know exactly what privacy and data collection settings are required when updating their installed extensions. We are making updates to give you even more visibility and control when reviewing which privacy settings are being modified in an update. That way you can decide whether or not to grant permissions and proceed with confidence. Firefox for iOS Quick Answers [AI feature]: use your voice to ask a question straight from your Firefox homepage and get a quick answer backed by sources in just a few seconds! Your voice, your questions and the generated answer stay totally private. Transfer Your Data between Browsers: move to Firefox in a snap by importing your bookmarks, history and reading lists from another browser, all in a few simple clicks! You can also export your Firefox data to other browsers. Built-in VPN: After making its grand debut in desktop and Android, built-in VPN is coming to Firefox for iOS! Navigate with extra protection on known networks and public wifi. Select from a large selection of locations, or let Firefox choose the fastest one for you. It's always free and signing up with a Mozilla Account gets you extra bandwidth. What's new or coming up in web projects Firefox Enterprise is here! It launched on October 13 alongside Firefox 158: Firefox with enterprise controls and a management console that organizations run on their own infrastructure. Read the announcement and explore the website. The launch starts in France, Germany, Austria and Switzerland, so the website is live in English, French and German. Our translation agency handled these pages, so you won't find them in Pontoon. Thank you to everyone who has helped with firefox.com so far. If you're curious about how localization works for firefox.com and Firefox Enterprise, or have thoughts on it, you're always welcome to get in touch. Strings for the main firefox.com website continue to land in the firefox.com project in Pontoon, for the locales that have it enabled. What's new or coming up in SUMO Semantic search is now live! It was officially launched with Kitsune 1.8.0 on October 6, 2026. This initial rollout is limited to Knowledge Base articles, with broader search improvements still to come. A big thank-you to everyone who participated in the testing phase and helped us improve the experience! Dark mode is coming to SUMO! You can get an early look by watching the demo in the previous Monthly Community Call or you can also try it yourself on our staging site. What's new or coming up in Pontoon Monthly health reports. Following the Community Health Score introduced in the August report, teams can now opt into a monthly health report, delivered as both an email and an in-app notification, with a new toggle in contributor settings. The feature was developed by our intern Jamie (functionzz). Editor and Machinery. The editor can autofill from composed Machinery matches, the 100% match indicator moved into the editor menu, and leaving an autofilled 100% TM match no longer triggers an unsaved-changes warning. New contributors. We're happy to welcome another group of contributors who made their first patches to Pontoon in the past few months: baha-bouali (6 patches), scottcampbelldata (4), tahakotil (2), nyomanjyotisa (2), kwy404 (2), annki, Goudarz, Hanabi9248, HarshRajSinghania, and MaxFreedomPollard. Their work spans terminology, search and dropdown behavior, the API, plural

handling, theming, insights, right-to-left support, and documentation — thank you all! Friends of the Lion Image by Elio Qoshi Know someone in your l10n community who's been doing a great job and should appear here? Contact us and we'll make sure they get a shout-out! Useful Links #l10n-community channel on Element (chat.mozilla.org) Localization category on Discourse Mastodon L10n blog Questions? Want to get involved? If you want to get involved, or have any question about l10n, reach out to: Bryan – Project Manager (Firefox, Mozilla Accounts) Camila – Project Manager (Firefox for Android, Firefox for iOS) Eda – Project Manager (Web projects) Francesco (flod) – Project Manager, L10n Team Manager Francis – Project Manager (Common Voice, Mozilla Foundation) Théo – Project Manager (Mozilla Foundation) Kiki – Project Manager (SUMO) Eemeli – Developer (Pontoon, Infrastructure) Eric – Developer (Pontoon, Infrastructure) Matjaž (mathjazz) – Developer (Pontoon) Did you enjoy reading this report? Let us know how we can improve it.

- Firefox Tooling Announcements: Happy BMO Push Day! (20261021.1) (2026/10/21 22:23)
  Github Link The following changes have been pushed to bugzilla.mozilla.org: Bug 2064523 - [meta] tracking bug for BMO dependency upgrades Bug 2081849 - Migrate Bug REST resource to native Mojo API: field metadata (fields, legal_values, flag_types) Bug 2082654 - 2_test_flags2.t picks a component before the bug page has loaded the new product's components Bug 2083156 - Add an alert for excessive lag in the phab-bot feed processing queue Bug 2083532 - Webhook bug.modify payload has empty bug.flags when flags are set at bug creation Bug 2020752 - [PHABBUGZ] Improve testing of phabricator and bmo integration Bug 2079840 - allow @username in the textarea like github Bug 2081859 - Migrate Bug REST resource to native Mojo API: comments (comments, add_comment, render_comment, reactions, tags) Bug 2078770 - Migrate User REST resource to native Mojo API: MFA (enroll, verify_totp_code) Discuss these changes in the BMO Matrix Room 1 post - 1 participant Read full topic
- The Mozilla Blog: Still here, still yours: 5 reasons to choose Firefox (2026/10/21 16:00)
  The internet used to feel like a community, not a commodity. Browsers were tools for exploration, and you could wander the web without feeling tracked everywhere.Today, as the web becomes increasingly surveilled and commercialized, Firefox preserves the magic, freedom, and fun of the early internet. Still open source, still independent, still privacy-focused. No billionaire owner, no hidden agenda.So, whether you're new here, back after a break, or just curious, here are five reasons to choose the browser that puts you first. Backed by a non-profit, not billionaires Firefox is backed by the Mozilla Foundation, a non-profit committed to an open internet. Unlike other big tech companies, we're built with privacy as the default. Our open source code is visible, reviewable, and continuously stress-tested by a global community, truly putting power in the hands of the people it's built for. Private and fast by design While other browsers allow third-party trackers to profile you, Firefox's Enhanced Tracking Protection (ETP) helps stop cross-site tracking before it starts, letting your browser work faster. Plus, when you sync information like history and bookmarks across devices, it's encrypted locally, meaning only you can access it. Learn about some of our other privacy-focused features like password manager, built-in VPN, and Relay in our guide to strengthening your online security for free with Firefox. AI on your terms We believe the only person who should decide how you interact with AI features in your browser is you. Firefox offers a clear, easily accessible AI controls section where you can turn off AI features entirely — or decide which ones stay on depending on your workflow and preferences. For those wanting a little more help getting things done, Smart Window is Firefox's most integrated AI experience. With your permission, its built-in, AI-powered assistant can work directly with your open tabs and browsing history to connect the dots. This means it can compare information, summarize pages, and help plan projects without requiring you to provide context from your past and current browsing activity every time you start a new prompt. Newer feature additions to Smart Window include the ability to group related tabs together and visual history previews that

help you find a page you visited earlier when you can’t remember the exact URL or phrase searched. Also, our partnership with Exa enables Smart Window to find current web information to include in responses while also displaying the sources behind these answers, so you can verify what you’re seeing without interrupting your workflow. Smart Window remains in beta as we continue to learn and improve from our community’s feedback. Try it yourself here. Not just Chrome in disguise Firefox is not built on Chromium, the open source browser project that supports Google Chrome and many other browsers. As a matter of fact, we are one of the last major browsers that isn’t. Instead, Firefox runs on Mozilla’s independent Gecko browser engine, which was developed specifically to uphold our strong privacy standards and give people more control over their data and browsing experience. This independence also impacts how Firefox handles browser add-ons, particularly ad blockers. Unlike Chromium-based browsers that now limit how powerful ad-blocking tools can be, Firefox still allows ad blockers like uBlock Origin to work at full strength. This means people who use Firefox can enjoy stronger, more reliable ad and tracker blocking without compromise, something increasingly difficult on Chromium-based browsers due to imposed limitations. Plus, we have a really cute Firefox Firefox is constantly building features and solutions to help you navigate the web on your own terms, and we wanted to make this support visible. Enter Kit, Firefox’s companion to the internet.  Kit isn’t an AI assistant or chatbot, just a friendly face that you might see during moments that are meant to feel friendly and supportive, like when you’re just getting set up, exploring a new feature, or celebrating a win such as successfully adjusting a setting. Rooted in values, built for you While the internet is ever-changing, Firefox remains rooted in the values that defined its beginning. Our browser was created on the principles of privacy, independence, and choice,and we continue to uphold them as we innovate and evolve. In a web landscape dominated by profit, surveillance, and sameness, our browser preserves the open and human spirit of the early internet. New, returning, or loyal for years, there’s always a place for you here. Download Firefox and make the web yours again. The post Still here, still yours: 5 reasons to choose Firefox appeared first on The Mozilla Blog.

- Firefox Tooling Announcements: PerfCompare deployed (Oct 21st, 2026) (2026/10/21 15:20)
  The latest version of PerfCompare is now live! Check out the change-log below to see the updates: [kala-moz]: Bug 2058236: Show a notice when column filters hide result rows Bug: 2075044 Significance filter and sort selections don’t persist [gopar]: Bug-2039589 - Add caching to ttest/mwu views Bug 1846013 Add ability to display multiple framework results Bug-1941337 - Add new formatter for delta values [moijes] Bug-1982588: Align breadcrumbs and replicates button in Subtests View Thank you for the contributions! Bugs or feature requests can be filed on Bugzilla. The team can also be found on the #perfcompare channel on Matrix. Come and chat! 1 post - 1 participant Read full topic
- This Week In Rust: This Week in Rust 674 (2026/10/21 04:00)
  Hello and welcome to another issue of This Week in Rust! Rust is a programming language empowering everyone to build reliable and efficient software. This is a weekly summary of its progress and community. Want something mentioned? Tag us at @thisweekinrust.bsky.social on Bluesky or @ThisWeekinRust on mastodon.social, or send us a pull request. Want to get involved? We love contributions. This Week in Rust is openly developed on GitHub and archives can be viewed at this-week-in-rust.org. If you find any errors in this week's issue, please submit a PR. Want TWIR in your inbox? Subscribe here. Updates from Rust Community Official Announcing Rust 1.99.1 Announcing the GPU notification group and Linker Working Area Newsletters Rust Trends Issue 85 - If It Compiles, the Agent Got It Right Project/Tooling Updates Introducing tsrs: a Port of TypeScript to Rust Observations/Thoughts Cloudflare is rewriting everything in Rust What agents mean for Rust (and vice versa) Black-Box HashDoS: Attacking Hash Tables Without Knowing the Hash Function 2026 is an exciting year for the Rust compiler (and me) What the crates.io download count really measures: 90 days of daily data from 69 crates Safe self-referential data structures in kernel Rust Cross-crate forward

declarations in Rust Why tellus message handlers are synchronous Upstreaming Tyr: Rust, DRM, and the Linux Kernel Rust Walkthroughs I forced a speech model onto the GPU for months. The CPU was faster. Superoptimizing an FPU emulator Draw and Animate Loading Spinner with GPUI [series] Encoding TCP in Rust's Type System, Part 1: A Sequence Number Is Not a u32 [series] Running 10 million Codex CLIs on one server - LFI processes Research An Incremental Approach to JIT Construction [video] [Scheme'26] An Incremental Approach to JIT Construction Miscellaneous Crate of the Week This week's crate is eilmeldung, a TUI RSS reader. llogiq is pretty pleased with his choice, despite a lamentable lack of suggestions. Please submit your suggestions and votes for next week! Calls for Testing An important step for RFC implementation is for people to experiment with the implementation and give feedback, especially before stabilization. If you are a feature implementer and would like your RFC to appear in this list, add a call-for-testing label to your RFC along with a comment providing testing instructions and/or guidance on which aspect(s) of the feature need testing. No calls for testing were issued this week by Rust, Cargo, Rustup or Rust language RFCs. Let us know if you would like your feature to be tracked as a part of this list. Call for Participation; projects and speakers CFP - Projects Always wanted to contribute to open-source projects but did not know where to start? Every week we highlight some tasks from the Rust community for you to pick and get started! Some of these tasks may also have mentors available, visit the task page for more information. ScreenCue ScreenCue - Add a reproducible desktop bug-report issue form sysknife sysknife - README says the CLI's approval gate is sysknife approve; the CLI approves in-process after its own prompt sysknife - --dry-run exits 0 for a plan that breaks --max-risk or that this distro cannot run sysknife - Re-running sysknife-setup while the daemon runs fails with ETXTBSY and can leave mismatched binaries Manguebyte Manguebyte - Put ParamSmoother under the allocation guard Manguebyte - Treat a NaN delay as the shortest delay in DelayLine::read and Delay::set_time Manguebyte - Check security advisories in the cargo deny job Manguebyte - Let a plugin report its tail to hosts Manguebyte - Add audio widgets, starting with an on-screen keyboard krate krate - Fix two warnings on the wasm32 and Windows targets (easy) krate - Linux: follow the person's dark mode and text size (medium) krate - krate pack: add a --keep-debug flag (medium) If you are a Rust project owner and are looking for contributors, please submit tasks here or through a PR to TWiR or by reaching out on Bluesky or Mastodon! CFP - Events Are you a new or experienced speaker looking for a place to share something cool? This section highlights events that are being planned and are accepting submissions to join their event as a speaker. If you are an event organizer hoping to expand the reach of your event, please submit a link to the website through a PR to TWiR or by reaching out on Bluesky or Mastodon! Updates from the Rust Project 601 pull requests were merged in the last week Compiler improve diagnostic for literals in lint attributes improve diagnostics for non-well-formed associated items fix diagnostic::on_unknown format placeholder for paths in expressions and type paths move fully_perform early return earlier overhaul span compression perf: drop error in unwrap_failed instead of unwrap port lint attributes provide context span pointing at trait/item when talking about assoc items restore the old behavior for derive(Default) on fieldless types skip move checks when the size limit is zero stabilize -Zembed-metadata support function args const generics Library add codeview_annotation intrinsic add core::process::resumable_breakpoint make std::fs::{File, ReadDir, DirEntry} always needs_drop even when unsupported make tuple PartialOrd chaining easier on mir-opt windows: stabilise inherit_handles Cargo build-std: Add build.build-std.when and crates config keys prerelease: support [patch] with -Zprerelease config: Apply dotted tuple support for rustflags config: Don't load registry config from credentials.toml fingerprint: choose a stable path when source mtimes tie util-schemas: compile out the file-path hint where url can't build it shell completion show all the packages having the feature check crate type when pruning -L paths Rustfmt config: fix lookup of global config files don't remove empty generic args when formatting in a macro context enforce comment width in item lists fix cfg_select! parser recovery handling and parser session error count leak fix statement comment wrapping prevent rustfmt from removing

expression attrs when removing nested parens prevent rustfmt from removing necessary blocks in cfg_select! arms Rustdoc add disambiguators for identically named method in sidebar improvements for rustc_must_implement_one_of Clippy allow allow_attributes from inside macros bail all doc lints when attributes come from external macro expansion clear collection reassignment extend from_str_radix_10 to NonZero types fix try_from_instead_of_from_str lint suggestion causes error when TryFrom::try_from parameter is not named value fix wildcard_imports infinite loop on macro generated glob imports fix trailing semicolon with nonstandard_macro_braces new lint: bufreader_stdin useless_attribute: no longer lint clippy::allow_attributes Rust-Analyzer viewMir panicking in code containing field projections bail out of struct_tail_erasing_lifetimes on infinite structs build non-ambiguous monikers in crates with multiple targets emit E0029 for range patterns with inferred invalid types escape keyword module name completions improve attribute completions without snippet ops: resolve ref-version-mismatch, stale-action-refs, and unpinned-uses Rust Compiler Performance Triage A week dominated by changes of small magnitude. However, a number of improvements landed that results in an overall average improvement of 1.4%, which is a quite nice to see. Triage done by @simulacrum. Revision range: 7211ea59..bd4c6cc4 6 Regressions, 9 Improvements, 8 Mixed; 0 of them in rollups 43 artifact comparisons made in total Full report here Approved RFCs Changes to Rust follow the Rust RFC (request for comments) process. These are the RFCs that were approved for implementation this week: Fix libs nominated triagebot.toml Final Comment Period Every week, the team announces the 'final comment period' for RFCs and key PRs which are reaching a decision. Express your opinions now. Tracking Issues & PRs Rust core: Add assert_matches and debug_assert_matches to the next prelude Lower #[no_mangle] to #[symbol_name]/#[link_name] stabilize String::into_chars stabilize {Rc, Arc}::is_unique vec: stabilise allocators for into_iter Don't require unsafe for struct and array patterns against union fields Add a future incompat warning for #[no_link] raise ambiguity error on attribute macro that could be tool attribute stabilize -Znext-solver=globally Stabilize --remap-path-scope in rustdoc and the documentation scope Compiler Team (MCPs only) Add support for annotate-snippets's no-graphics output format No Items entered Final Comment Period this week for Rust RFCs, Cargo, Language Team, Language Reference, Leadership Council or Unsafe Code Guidelines. Let us know if you would like your PRs, Tracking Issues or RFCs to be tracked as a part of this list. New and Updated RFCs Deprecate std::env::{vars, args} RFC: document a strict Future contract that forbids snoozing RFC: extern "Rust_manyregs" ABI RFC: add poll_progress to AsyncIterator Upcoming Events Rusty Events between 2026-10-21 - 2026-11-18 □ Virtual 2026-10-21 | Hybrid (Vancouver, CA) | Vancouver Rust Rust Study/Hack/Hang-out 2026-10-21 | Virtual (Cardiff, UK) | Rust and C++ Cardiff Operating Systems Book Club: Paging Mechanisms 2026-10-22 | Virtual | Rust □ Maven Rust and the GPU from Native to Web: An Introduction to wgpu 2026-10-22 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-10-26 | Virtual | Rust □ Maven No Python was harmed: teaching a tiny MCU to learn as it goes 2026-10-27 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Fourth Tuesday Rust Bookclub 2026-10-27 | Virtual (London, UK) | Women in Rust Lunch & Learn: Reasoning with Async Rust 2026-10-29 | Virtual | Rust □ Maven Creating a hexadecimal editor in Rust 2026-11-01 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: First Sunday 2026-11-03 | Virtual | Rust □ Maven Solving Real-World Planning Problems in Rust with SolverForge 2026-11-03 | Virtual (London, UK) | Women in Rust □ Community Catch Up 2026-11-04 | Virtual (Indianapolis, IN, US) | Indy Rust Indy.rs - with Social Distancing 2026-11-05 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-11-10 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Second Tuesday 2026-11-12 | Virtual (Nürnberg, DE) | Rust Nuremberg Rust Nürnberg online 2026-11-15 | Virtual | Rust □ Maven Cursed Rust 2026-11-15 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: Third Sunday 2026-11-17 | Virtual | Rust □ Maven Miscompilation in Rust 1.97 2026-11-17 | Virtual (London, UK) | Women in Rust Lunch & Learn: The Harness Wore a Trenchcoat - Building a Rust MCP Server 2026-11-17 | Virtual (Washington, DC, US) | Rust DC Mid-month Rustful 2026-11-18 | Hybrid (Vancouver, CA) | Vancouver Rust

Writing a firmware for Xteink ereader in Rust Asia 2026-11-03 | Tel Aviv-Yafo, IL | Rust □ TLV In person Rust November 2026 at AWS in Tel Aviv 2026-11-14 | Bengaluru, IN | Rust Bangalore November 2026 Rustacean Meetup Europe 2026-10-22 | Karlsruhe, DE | Rust Hack & Learn Karlsruhe Karlsruhe Rust Hack and Learn Meetup bei BlueYonder 2026-10-22 | Toulouse, FR | Rust Toulouse Rust Toulouse Meetup - Rust & Python interoperability 2026-10-27 | Aarhus, DK | Rust Aarhus Hack Night: Rust meets AI 2026-10-27 | Paris, FR | Rust Paris Rust meetup #89 2026-10-28 | Copenhagen, DK | Copenhagen Rust Community Rust meetup #72 sponsored by Google! 2026-10-31 | Stockholm, SE | Stockholm Rust Ferris' Fika Forum #31 2026-11-01 - 2026-11-03 | Italy, IN | RustLab RustLab - The International Conference on Rust in Italy 2026-11-04 | Dublin, IE | Rust Dublin Rust 263: Getting Started with the New Rust Driver for Oracle AI Database 2026-11-05 | Milano, IT | Rust Language Milan Digital Model Trains & Rust: Bare-Metal DCC Command Station on ESP32 2026-11-17 | Leipzig, DE | Rust - Modern Systems Programming in Leipzig Everlock: Everything Is Git, Protocols Are Just Adapters North America 2026-10-21 | Hybrid (Vancouver, CA) | Vancouver Rust Rust Study/Hack/Hang-out 2026-10-21 | San Francisco, CA, US | Bay Area Rust Bay Area Rust - Embedded Meetup 2026-10-22 | New York, NY, US | Rust NYC Rust NYC w/Bryan Cantrill - Code Matters: Reflections on Rust in the LLM Age 2026-10-24 | Boston, MA, US | Boston Rust Meetup Davis Square Rust Lunch, Oct 24 2026-10-28 | Austin, TX, US | Rust ATX Rust Lunch - Fareground 2026-10-31 | Boston, MA, US | Boston Rust Meetup North End Rust Lunch, Oct 31 2026-11-05 | Montréal, CA | Rust Montréal November Monthly Social 2026-11-05 | San Francisco, CA | Rust LA: NVIDIA! NVIDIA x Lawrence Harvey | CUDA Rust 2026-11-05 | Saint Louis, MO, US | STL Rust HTMX and Rust 2026-11-12 | San Diego, CA, US | San Diego Rust San Diego Rust November Meetup - Back in person! 2026-11-14 | Boston, MA, US | Boston Rust Meetup MIT Rust Lunch, Nov 14 2026-11-17 | Los Angeles, CA, US | Rust Los Angeles Rust LA November w/ Hadrian! 2026-11-17 | San Francisco, CA, US | San Francisco Rust Study Group Rust Hacking in Person 2026-11-18 | Hybrid (Vancouver, CA) | Vancouver Rust Writing a firmware for Xteink ereader in Rust South America 2026-11-07 | Planalto Paulista, BR | Rust São Paulo Meetup Encontro do Rust-SP - Agora no Luma If you are running a Rust event please add it to the calendar to get it mentioned here. Please remember to add a link to the event too. Email the Rust Community Team for access. Jobs Please see the latest Who's Hiring thread on r/rust Quote of the Week To me it seems the same things that make Rust a desirable target for LLM generated code are the same things that make it desirable for me as a human writing code, that being the fantastic compiler feedback and the first class tooling. – Christian Haynes on rust-users Thanks to Moy2010 for the suggestion! Please submit quotes and vote for next week! This Week in Rust is edited by: nellshamrell llogiq ericseppanen extrawurst U007D mariannegoldin bdillo opeolluwa bnchi KannanPalani57 tzilist Email list hosting is sponsored by The Rust Foundation Discuss on r/rust

- Mozilla Security Blog: Protecting Firefox users following the compromise of eMudhra intermediate certificates (2026/10/20 12:54)
  On Saturday, October 17, at 20:06 UTC, Mozilla received a report that private keys associated with eight eMudhra intermediate certificates had been compromised. These keys could allow attackers to issue fraudulent HTTPS certificates for websites they do not control. An attacker who could intercept a user’s connection could use those certificates to impersonate websites and potentially read or modify traffic. Mozilla engineers responded immediately, blocking the affected certificates in Firefox using OneCRL, Mozilla’s centralized list of revoked certificates. OneCRL delivers this protection without requiring users to update or restart Firefox. By 22:59 UTC that same day, in less than 3 hours after we received the report, the updated OneCRL distribution was complete for all Firefox users with standard configurations. Today, we are also releasing Firefox 158.0.1 for Desktop and Android, along with Firefox ESR 115.43.1 and Firefox ESR 153.5.1. These releases will distribute this protection for a minority of users whose custom settings may have prevented them from receiving the OneCRL update. We are also updating the trusted certificate list in NSS (NSS 3.131.1) to remove the root certificate that signed the affected intermediates. Firefox is already protected through

OneCRL, but this additional update helps protect other applications that use Mozilla’s trust store or versions derived from it and do not receive OneCRL updates. Mozilla believes everyone has a fundamental right to security and privacy online. We remain committed to defending those rights and acting quickly to protect users. Transparency is essential to trust. We will share more information as soon as circumstances allow. Update added October 21: On Wednesday, October 21, at 08:29 UTC, we released an additional update for Firefox for Android (158.0.2) to fix an issue where some clients were not applying changes to the bundled OneCRL blocklist immediately after startup. The post Protecting Firefox users following the compromise of eMudhra intermediate certificates appeared first on Mozilla Security Blog.

- The Mozilla Blog: Why is the CMA asking us to choose the furniture before the house? The UK’s search choice screens need browser choice screens alongside them (2026/10/20 07:00)
  Mobile browsers are a key gateway to the internet for 56 million people in the UK. Smartphone users spend around half an hour a day in these apps, and opening a browser is the most common way they look up information online. Yet most people are using a smartphone browser that was preinstalled and set as the default for them, without ever being asked which browser they would actually prefer. Mozilla believes people deserve an active choice in the tools they depend on every day, which is why the latest proposals from the UK Competition and Markets Authority (CMA) matter. We welcome the CMA’s plans to give people a choice of search service and are asking it to consult on browser choice now, so both screens can launch together. On September 23rd, the CMA published updated plans that would require Google to show people in the UK a choice of search service on new Android devices, covering the default search app and home-screen search widget as well as Chrome. Search choice screens would also appear in Chrome on other platforms, including through a major update for existing users. Some AI assistants would also be eligible to appear alongside search engines. The proposals are well designed, but solve only half the problem The CMA’s mobile browser investigation found that preinstallation, prominent placement and default settings restrict browser competition. Mozilla’s innovative 2023 experiment showed that properly designed browser choice screens can help people choose a browser they expect to keep using, and can be an effective tool for giving people agency over their online life. It also showed that an overwhelming majority of people (98%!) want to see them. The EU Digital Markets Act has since demonstrated their impact in practice. We welcome the CMA’s work on search choice, but people in the UK deserve browser choice too. Asking people to pick a search service inside a browser that most of them never actively chose in the first place leaves some of the biggest issues identified in the CMA’s own mobile browser investigation unaddressed. The CMA commissioned user research on search choice screens to refine and improve the proposals it first consulted on in January. Providers’ positions on the choice screen would be fully randomised. Users would have to scroll to the bottom before confirming, and could “test drive” a search provider for at least a week. The CMA has also recognized that search is changing. Its research found that many people were already using AI assistants to search alongside traditional search services. Under the revised, more technology-neutral eligibility criteria, an AI assistant could also qualify to appear on the search choice screen if it meets the requirements, including being able to search the web and attribute its sources. But browser choice should not be left behind. Why the two choices belong together Less confusion. The CMA’s new research found that participants often confused browsers with search services and were unsure what their search choice would change. Mozilla’s research has revealed similar confusion. These findings support designing and testing the two screens together, with clear explanations of each decision. Less duplication. The CMA’s work on search choice can also inform browser choice. Developing the two together would allow the design, order and timing to be tested as one journey, helping to avoid separate rounds of changes and interruptions for users. Order matters. Nobody chooses the furniture before the house. Browsers have their own search defaults, which is why we recommend showing the browser choice screen first, followed by the search screen, with

wording that makes clear these are two separate decisions.  The EU and Japan already show both screens The EU's Digital Markets Act introduced browser choice screens on iOS for the first time and brought browser and search choice together during Android set-up. Browser first, then search, so the relationship between the two makes sense. In Japan, search and browser choice screens are presented on both Android and iOS. And people are making use of the browser screen to select alternatives. In May, we reported Firefox is selected through a DMA choice screen once every ten seconds, with retention five times higher than for organic downloads. Other independent browsers have also reported substantial growth.  Choice matters more as AI moves into the browser The CMA's proposals recognize that AI is changing how people search. AI is also changing what browsers offer, adding to the ways they compete on features, privacy and user control. Firefox already lets people on desktop pick which AI assistant sits in the sidebar, and our Smart Window beta extends that choice to the model behind the browser's own assistant. There are meaningful differences between browsers. AI is the latest example of why giving people an active choice matters. When people aren't offered an active browser choice, they may be choosing search and AI services within a browser someone else picked. One decision taken at the factory can end up shaping the ones that follow. A search choice screen gives people a say over their search provider; offering browser choice alongside it would give them a say over that first decision too. Our ask of the CMA We're asking the CMA to consult now on browser choice screens for Apple's and Google's mobile platforms, so they can launch alongside the search screens. Apple and Google already offer browser choice screens in the EU and Japan. The CMA can draw on that experience and its own research to give people in the UK an active choice of both browser and search service. Mozilla's response to the consultation is available here. The post Why is the CMA asking us to choose the furniture before the house? The UK's search choice screens need browser choice screens alongside them appeared first on The Mozilla Blog.

- The Rust Programming Language Blog: Announcing rustup 1.29.2 (2026/10/20 00:00)
  The rustup team is happy to announce the release of rustup version 1.29.2. Rustup is the recommended tool to install Rust, a programming language that empowers everyone to build reliable and efficient software. What's new in rustup 1.29.2 This is a small release which only updates the rustls-webpki and webpki-root-certs dependencies to address the problematic eMudhra intermediate certificates issue described in CVE-2026-111943. See also RUSTSEC-2026-0382. Further details are available in the changelog! How to update If you have a previous version of rustup installed, getting the new one is as easy as stopping any programs which may be using rustup (e.g. closing your IDE) and running: $ rustup self update Rustup will also automatically update itself at the end of a normal toolchain update: $ rustup update If you don't have it already, you can get rustup from the appropriate page on our website. Rustup's documentation is also available in the rustup book. Caveats Rustup releases can come with problems not caused by rustup itself but just due to having a new release. In particular, anti-malware scanners might block rustup or stop it from creating or copying files, especially when installing rust-docs which contains many small files. Issues like this should be automatically resolved in a few weeks when the anti-malware scanners are updated to be aware of the new rustup release. Thanks Thanks again to all the contributors who made this rustup release possible!
- Chris H-C: Eleven-Year Moziversary (2026/10/19 13:43)
  I've been working at Mozilla for eleven years, my golly. I'm still on my third hardware refresh (a Dell XPS I continue to dislike), I'm up to my fourth CEO, and 76 reorgs have happened (well, I know that quite a few things that would normally have counted haven't been counted there, so it's probably a higher number than that). I'm still getting Firefox Desktop to use Glean instead of Legacy Telemetry. We're more than two-thirds of the way done with removing the legacy system, with concrete plans about what to do with the rest. I'm still not blogging as much as I'd like, but I have some honest hope that I might change that in no small part thanks to someone who mentioned in passing that they liked my posts. Looking

back at last year's predictions, I did indeed requisition a second work monitor, which is working out well. I was right that FOG Migration hasn't completely removed all of Legacy Telemetry, though we're a lot closer than I thought we'd be. I do have a little time to think about doing other things, but not all that much, so the part of the prediction where I'd do something else is a "maybe". My desire to push for a sensible user population measurement did not have a chance to be acted upon. We might be shifting what the business counts, but not to anything I've had a say in. At least one of the reorgs that happened changed the organizational distance between my team and the other parts of Data that would be involved in such a measurement, other non-work things took way too much time and resulted in slower overall work velocity, and FOG Migration kept taking priority. I didn't think there'd be an All Hands, and in the off-change there were, it'd be a multi-city "Summit style" affair. I was half wrong twice: there wasn't an All Hands but there was indeed an all-Engineering Summit; it wasn't in multiple cities but just one. I was right that Toronto finally got its chance. I enjoyed the heck out of it as it really reminded me of some of the things I really missed about the All Hands of yore. It could've been a day longer, it could've been a little more of a polished experience like Brianna-era events were… but for not having done anything like this for two years, what an accomplishment. I was initially skeptical about spending hours of the first day making small talk deliberately with people you don't know on the subject of the work you do and how you do it, but once it was framed as an effort to scale-up and operationalize The Hallway Track I suddenly understood it. I had wishful, hopeful thinking that the genAI bubble would have burst or a new shiny late-stage capitalist exploitation mechanism would be draining focus away from it by now. Alas neither. Though maybe things are beginning? I can dream. I was also half-wrong about there being a new CEO but not yet sworn in. I didn't account for the candidate coming from within the org, meaning it could be quick. We've had Anthony hired and active for nearly a year now and he's… fine? So far the changes he's made to MoCo haven't been to any of the things I feel we need to change, so I can't really evaluate him. (Assuming that, since he (well, his reports) are able to evaluate my performance, it's only fair that I get to evaluate his in turn. That's how this works, right?) Predictions for the coming year: We're going to stick with Anthony as CEO, keeping my count stable Firefox Enterprise is going to be a runaway success, finally giving MoCo that revenue diversification we've been aiming for Maybe the genAI bubble will have burst? (Yes, this is a duplicate from last year) I'll read The Psychology of Software Teams. It will frustrate me to finally be able to name and cite sources for how badly this industry treats its workers. FOG Migration will be "complete" (Legacy Telemetry probes will be gone, code will have been removed) and I'll move on to working on… maybe data consent management? Developing a way to find out which metrics are unused, and removing them? I have a few ideas. I'll still be complaining about this Dell XPS laptop, and considering what I'll be refreshing to the following February. Maybe I'll finally leave Windows behind? I will have blogged at least three times (setting my prediction deliberately and achievably low), probably mostly video game musings in my ongoing So I've Finished series. But I do have a draft of a work post about the history of public data tooling at Mozilla that I've been meaning to finish… To another year of supporting The Mission!

- Firefox Tooling Announcements: MozPhab 2.23.0 Released (2026/10/16 17:32)
  Bugs resolved in Moz-Phab 2.23.0: bug 2068797 Local test suite runs read and write the real user_info.json bug 2076869 Record a profile of each moz-phab run, and open it in the Firefox Profiler Discuss these changes in #engineering-workflow on Slack or #Conduit Matrix. 1 post - 1 participant Read full topic
- Firefox Tooling Announcements: Firefox DevTools MCP 0.10.6 released (2026/10/16 16:08)
  New release of Firefox DevTools MCP. Available on npm at https://www.npmjs.com/package/@mozilla/firefox-devtools-mcp Added New print_page tool. It saves the current page as a PDF file and has optional landscape and background options. It uses print CSS, so you can use it to check print

styles Fixed --profile-path now reuses a profile that the server created, instead of creating a new profile inside it. The server marks its profiles with a .firefox-devtools-mcp file. Paths without this marker still use the firefox_devtools_mcp_profile subfolder list_pages, select_page, close_page, navigate_page and new_page no longer bring every tab to the front. They now read the tab list with WebDriver BiDi select_page and close_page now act on the tab from the last tab list, even if tabs were opened or closed since then Changed list_pages no longer lists tabs that are not loaded yet, for example restored tabs that were never opened If the title of a tab cannot be read within 1 second, the tab is listed with the title (unknown) Full changelog: Release v0.10.6 · mozilla/firefox-devtools-mcp · GitHub Repository and issues: GitHub - mozilla/firefox-devtools-mcp: Model Context Protocol server for Firefox DevTools - enables AI assistants to inspect and control Firefox browser through WebDriver BiDi · GitHub Public chatroom: https://chat.mozilla.org/#/room/#firefox-devtools-mcp:mozilla.org 1 post - 1 participant Read full topic

- The Mozilla Blog: This map shows the most important asset that VC-backed companies need right now (2026/10/16 13:00)
  What's the most vital asset for an AI company today?  For venture backed AI business, it might be "funding" or "revenue". But, there's another asset that's just as critical: trust.  At Mozilla, building technology that people and businesses can genuinely trust has been a long-standing, guiding principle. That's why we're supporting the launch of the Trustworthy AI Map, a market map of VC-backed companies building the infrastructure for trustworthy AI.  Supported by seven organizations, including Mozilla, the Trustworthy AI Map features 79 companies across 15 countries, from early to later-stage venture. It maps companies building products for AI deployment across seven layers of the AI stack, including identity, data control, interoperability and evaluation.  The map connects the products with the companies and capital behind them, helping investors, founders and businesses explore commercial opportunities. To be featured, trust must be baked into its product or business model through human agency, trust by design or aligned business models.  This initiative aligns with much of our work over the past year. We've invested in companies through Mozilla Ventures, launched Otari, an open source enterprise tool which allows you to control your LLM infrastructure from a single layer, and published landmark research in our State of Open Source AI report that's demystifying the open source AI landscape and shows that the capability gap between open and closed models is closing.  We've also partnered with Mila: the Quebec AI Institute, to build an open source AI foundation layer that enables organizations and institutions to own and operate their own advanced AI systems locally, ensuring full control over their technology and data. The project has the backing of the Canadian government, and industry leaders like Hypertec.  At a time when fear and uncertainty about AI are on the rise, trust is not just a virtue; it's a path towards profitability. Explore the Trustworthy AI Map to see how.  Learn more at trustworthyaimap.org. The post This map shows the most important asset that VC-backed companies need right now  appeared first on The Mozilla Blog.
- The Mozilla Blog: What does the (Fire)fox say? (2026/10/15 16:00)
  It all started with a question: What does Firefox sound like to you?Over the past year, Mozilla developed a small but meaningful set of sounds to add a sonic dimension to the Firefox experience. When we started this process, we knew we wanted something warm, human, and representative of our open source identity. We also knew we didn't want to invent these sounds in a vacuum. So we did what we've always done and turned to our community.With the help of creative audio company Sounds Like These, we reached out to Firefox users, contributors, and team members across the globe asking them to send us audio and written descriptions of what Firefox sounds like to them. This feedback became our compass.From there, we began building three new sounds that bring to life our logo, mascot Kit, and Firefox notifications. The logo and Kit sound have already begun rolling out across the Firefox brand (check them out in our new design launch announcement) – and we're excited to bring the notification sound to the browser in a future update.Not only did we want these new sounds to reflect Mozilla's mission and

values, we also wanted to ensure they felt connected to one another. This is the story of how they came to be. The logo sound: A human chorus rooted in nature Because Mozilla uses a community-based approach to create world-class open source software, we wanted Firefox's audio logo to share this same spirit. This meant creating something built by many and rooted in real human experience.Inspired by early community feedback that associated Firefox with nature, we started with delicate organic textures, including bird song and the thump of wood being struck to evoke the feeling of Kit moving through different environments. We also knew we wanted to utilize the voices of real Mozilla employees, or "Mozillians." As Sounds Like These Co-founder Sam Heath reflected, "What better way to represent the brand than the sound of the people behind it?"At first, we considered setting up a professional recording session in a large city where multiple Mozillians could drop in to be recorded. However, we quickly realized this would limit the voices we would be able to include to one geographic location. It was then suggested: Why not just let people record wherever they are? After all, Firefox is used in family living rooms, shared offices, and really most places one can access the internet.The final result includes the voices of 23 real Mozillians saying "Firefox!", in which you can hear their subtle differences in accent, tone, and timing. Sam shared that some of these employee submissions included the echoes of distant chatter, families, and all the other subtle sounds of Mozillians' real life playing out as they recorded. He explained that Sounds Like These did not deliberately edit out the entirety of this ambient sound to preserve the texture and authenticity that comes from real people living their real lives."That was something that was very clear from Mozilla from the start. We don't want to be this shiny, distracting tech brand. We wanted this to feel real, natural, and human," said Sam. The Kit sound: Friendly, familiar, and uniquely Firefox Kit holds the prestigious title of VP of Mascoting at Firefox, and we worked hard to ensure their sound properly reflects such an important position. Kit's not an AI assistant, nor a chatbot. They're not an actual fox, or a red panda either. They're a Firefox, its own creature not of this world. Kit may appear in your Firefox browser during moments that are meant to feel welcoming and encouraging, like when you're getting started, trying out a new feature, or successfully making a settings change. Our goal when creating the Kit sound was to convey Kit's friendly, familiar, and trustworthy presence without mimicking any real animal. The sound had to be flexible enough to express movement and emotion across animations and videos, while also remaining instantly recognizable.Early experiments ranged from pure animal sounds to abstract musical expressions. Foxes were considered, but as Sam noted, "Foxes are incredibly difficult to record, and they do make cute sounds, but never around people. You only really seem to hear them literally screaming – and we definitely didn't want anyone screaming."We also explored a more feline route, specifically a cat purr. As Oli Slack, Co-Founder and Head of Audio at Sounds Like These, explains, "The cat purr was almost kind of perfect…other than the fact that it sounded just like a cat purring. So we asked, 'How do we recreate the feeling of what the purr sounds like, without it being an actual cat purr?'"This is where the process really got interesting. While brainstorming cat purr alternatives, the Sounds Like These team researched the psychology behind what makes certain sounds more comforting and familiar. Soft tones and warmth from other humans instinctively register as more reassuring and honest, and this also crosses over into animals, particularly animals we keep as companions. This validated our initial draw toward the cat purr, and confirmed we were heading in the right direction sonically.The breakthrough came when Sounds Like These stopped looking at animal sounds as a starting point, instead using a human voice as the basis of a custom purr. This wasn't someone imitating a cat or "purring" themselves, it served as a foundation vocalization that was then uniquely shaped and layered to convey Kit's friendly and encouraging presence. And thus, the Kit sound was born. The notification sound: A gentle whoosh of helpfulness, coming soon to Firefox Notifications don't have to be jarring or intrusive, and that's exactly what we wanted to avoid when creating one for Firefox.Our new notification sound – which we look forward to rolling out in a future Firefox update – was carefully crafted to gently alert, not interrupt. It draws inspiration from community feedback that described the feeling of using Firefox like a door

opening and referenced moments of quiet discovery filled with gentle wonderment and joy.This sense of seamless exploration is reflected in the sound’s design: a soft whoosh, like a door swinging open, followed by a subtle click, all in an airy, rounded tone to feel helpful, not demanding.The result is a sound that softly notifies rather than disrupts or controls, staying true to the feeling of joyful curiosity that many users associate with browsing the web through Firefox. A family of sounds shaped by community From the very beginning of this process, we leaned into Mozilla’s values of openness and community.We consulted our fans and team members. We used real voices. We embraced the imperfections of being human.These sounds are now a part of Firefox’s story, and we hope every time you hear them, they remind you that you’re using a browser built by the people, for the people. The post What does the (Fire)fox say? appeared first on The Mozilla Blog.

- Firefox Tooling Announcements: MozPhab 2.22.1 Released (2026/10/15 03:01)
  Bugs resolved in Moz-Phab 2.22.1: bug 2078299 Allow overriding base url and repo call sign for development purposes Discuss these changes in #engineering-workflow on Slack or #Conduit Matrix. 1 post - 1 participant Read full topic
- Firefox Tooling Announcements: MozPhab 2.22.0 Released (2026/10/15 02:13)
  Bugs resolved in Moz-Phab 2.22.0: bug 1501077 moz-phab does not update the reviewers list in Phabricator to match local changes bug 2074122 moz-phab --apply-to here --no-branch changed my tree and branch without prompting bug 2077318 moz-phab should block submitting patches that still contain conflict markers bug 2078249 use threading.Event().wait() rather than time.sleep() in moz-phab spinner/wait message bug 2078366 AttributeError fetching revision with Jujutsu bug 2079200 Moz-phab patch doesn’t commit new files when not automatically tracked by jujutsu bug 2082222 Fix some formatting issues after Bug 2074122 Discuss these changes in #engineering-workflow on Slack or #Conduit Matrix. 1 post - 1 participant Read full topic
- The Rust Programming Language Blog: Announcing Rust 1.99.1 (2026/10/15 00:00)
  The Rust team has published a new point release of Rust, 1.99.1. Rust is a programming language that is empowering everyone to build reliable and efficient software. If you have a previous version of Rust installed via rustup, getting Rust 1.99.1 is as easy as: rustup update stable If you don't have it already, you can get rustup from the appropriate page on our website. What's in 1.99.1 Rust 1.99.1 fixes the following regressions from prior releases: Missing symbol wcslen: Some Rust targets were getting link errors for missing wcslen, and that function is now present in Rust's compiler-builtins library as a fallback. The LLVM loop-idiom-wcslen pass may introduce calls to wcslen as an optimization, and some Rust targets did not have that function readily available in libc or elsewhere. The UEFI targets in particular were broken for any project using std, as some of these calls were introduced in the precompiled standard library, but the issue was also reproducible for similar user code patterns. Miscompilation of FFI bool return values on x86_64: Rust functions returning bool through FFI could leave bits 1-7 of the return register unspecified, causing callers to observe raw values other than 0 or 1. This return type is now marked for zero-extension again, as required by the ABI. rust-lld failed to load libLLVM on macOS: This linker could fail to run if the compiler was invoked without the rustup proxy, as there was no direct setting for the appropriate library search path. Now rust-lld is built with an rpath entry for the directory containing libLLVM.dylib. If you'd like to help us out by testing future releases, you might consider using the beta (rustup default beta) and nightly (rustup default nightly) channels locally and in your CI. Please report any bugs you might come across! Contributors to 1.99.1 Many people came together to create Rust 1.99.1. We couldn't have done it without all of you. Thanks!
- Niko Matsakis: What agents mean for Rust (and vice versa) (2026/10/14 04:19)
  Me, calmly assessing the implications of LLMs for programming language design(ers). As someone who has made a career of programming

language design, I definitely have moments of existential dread about the rise of LLMs. I started coding in 5th grade on an x861 that ran at 5MHz, had something like a 5MB hard-disk and 512K of RAM2. Computers were really, really different back then. But honestly, programming wasn't. For all the advances in IDEs and programming languages and so forth, the basic act of entering code into a file has remained surprisingly constant3 – right up until a few years ago, when everything kinda turned on its head. That's been exciting and empowering at times4 but also scary and disorienting. Will my skills in programming matter? What about my skills in programming language design (such as they are)? This blog post is my attempt to answer those questions. So as not to keep you in suspense, let me just give you the conclusion. In short, yes, I believe there will still be a role for humans in programming. And yes, I think that programming language design is still relevant. And, naturally, I also happen to think Rust is a really good fit for agentic development – but that's not to say we can't make it a better fit. And that's what this post is ultimately about: what makes Rust and agentic development work well, and where we could double down on those strengths. This is a kind of "blog post companion version" of a talk I gave this week at REBASE, part of SPLASH. You might also enjoy the slides from the talk I gave at IWACO, IWACO and Rust: hidden links revealed, which was a bit more of a retrospective (no soundtrack for those, though). In any case, this turned out to be a great conference! Jack Huey presented on the Rust Vision Doc and, quite to my surprise, Steve Klabnik and Dorian Scheidt presented "LLMs as Collaborators in Language Specification and Design", talking about their process in building out Rue. It was fun to see Steve again. On top of that, there was a ton of great research presented at IWACO in particular. I was particularly taken by the work on Reachability Types and Information Borrowing. I also really liked learning about Hylo Lang, particularly their approach to projections. Clever and cool. Rust is popular in the agentic era From the most important Slack channel at Amazon, #aws-memes. Rust is popular right now. Very popular: Stratospheric among the breakout languages is TypeScript, cropped from the chart and sitting currently at +1200% growth from the prior year. [..] The second most rapid recent growth is in Rust. Meta engineers have written twice as much first-party Rust code so far this year than in the previous 9 years combined. — David Tolnay (emphasis mine) Some of that is because agents make it very easy to try out new things. And some of that is because Rust has a reputation for being performant and, if it's just as easy to pick any language, why not pick one that will yield faster execution times? But I think Rust is popular for another reason: because Rust happens to be a really good fit for agentic coding6. Of course, I would, wouldn't I? The point of this post is (1) to argue for why that is and then (2) to discuss what we might change about Rust to make it work even better. Foundational Software construction with agents It seems to me that an effective "software factory" will have three pillars: Design: figure out and specify clearly what the code is supposed to do. Construction: author code that does what it is supposed to do. Optimization: optimize the code so that it runs efficiently. None of these pillars are solved, but some have made more progress than others. Rust and optimization: fast by default, knobs when you need 'em Rust is designed to give "good performance by default", making the idiomatic thing typically also be the fastest thing. But another key part of Rust is that you have the knobs to eke out better performance when you need it. You control when something is stack- or heap-allocated and where pointer indirection occurs. If you want to, you can force the layout of your structs to be done in a particular way, or decide where to introduce boxing into your async stack frames. In the limit, you can put in inline assembly. In the past, learning how to "tune the performance knobs" could be a bit of a black art: but agents are making that much easier.7 It turns out that agents are really good at fine-tuning the performance of existing code. For example, people who work on super high-performance cryptographic algorithms used to do so by hand-tuning assembly code. But those people tell me that they are now turning to agents for that same job. More prosaicly, a lot of people building network services are finding that, using a profiler like dial9, agents can quite effectively diagnose async performance pitfalls, like long polls or blocking. Therefore, I think the task for Rust is clear. We keep on with our goal of "low-level enough for a kernel, high level enough for a GUI". If

anything, I think the shift around agents means we can tolerate a bit more “noise” in the form of explicit calls to clone() or share() – but not too much, since we still want the “happy path” of the code to be clear and evident8, even as we surface details that are important in specific scenarios (like tracking precisely how many clones you have of a channel endpoint). Combined with a profiler, you get an effective feedback loop to get your code running fast: flowchart LR Code -- Benchmark --> Profile Profile -- Analyzed by --> Agent Agent -- Propose --> PR PR -- Improve --> Code Reliability is king, as always As usual though, while a reputation for good performance is what brings people to Rust, what makes them want to stay is a sense that using Rust “just seems to work better”. The basic theory is that Rust’s type system and guardrails help keep agents on track: rust is a perfect language for agents, given that if it compiles it's ~correct— Greg Brockman (@gdb) January 2, 2026 I think there’s truth in that: languages like Rust, Ocaml, or Haskell all share this “if it compiles, it works” feeling, and I think that agents will benefit from that. Testing is not enough, type systems help Testing can show the presence of bugs, but not their absence. — Edsger W. Dijkstra, 1970 Agents presenting their work. Writing tests has never been enough to show that your program is correct – but, with agents, it’s not even close. It happens so often that my first few tests seem great and then the next one goes completely bonkers. Usually bonkers in a way where I cannot even imagine how that bug could have come about. Then I dig in and I realize the agent completely missed the boat on the logic I was trying to express.9 This is where a type systems really shines – particularly an advanced one like Rust. Type systems give you some guardrails that keep the agent on track. In Rust, those guardrails extend beyond simple stuff like “do all fields have the correct names” and to stuff like never having to close a socket. Type systems are not enough either Type systems can show the absence of bugs, but not the presence of correctness. — Me, failing to coin a phrase of equal eloquence10, 2026 So yeah, type systems are helpful! But they have inherent limits. A type system is a simple set of rules that, if you follow them, make sure that you avoid (certain kinds of) mistakes. But they don’t tell you that you do the right thing – only that you don’t do (certain kinds of) wrong things. If you want to go further, you need a proof system – something that will let you show that richer properties are actively true, not just that you’re avoiding bugs. As it happens, there are a lot of interesting proof tools for Rust, like Verus, Creusot, Aeneas, and Kani. As a starting point, you can prove that your code is panic-free, which will eliminate a lot of bugs, though it is still fundamentally a “type-checker-like” property (i.e., you avoid doing things wrong). But you can also go further and add assertions that show your code is computing the results you expect. Strong guardrails give better performance Deplying proofs is not just good for getting code right – it’s good for getting code fast. This is somewhat counterintuitive, but it’s true. Earlier I mentioned how optimized cryptographic algorithms were now being written by agents. Part of what makes that possible is that cryptographic algorithms have a very clear mathematical definition. This means that you can basically tell the agent “do whatever you want, as long as you can prove that the code still computes the correct result”. These proofs can be done with theorem provers like Lean or Rocq. The blog post “Why Lean is faster than Rust” makes the point quite well. For Rust, a good example of this would be around unsafe code. Right now, people are sometimes reluctant to deploy unsafe code even if it might make their code faster because it’s not worth the risk of getting it wrong. And I applaud that effort. But what if you could use a prover to show that your unsafe code is, in fact, safe (i.e., incapable of causing undefined behavior)? Then you’d be free to optimize to your heart’s content. If we are going to prove things, do we need type systems? You might wonder: if we’re deploying a proof system on our code, do we still need type systems? I think the answer is definitely yes. Type systems are infinitely simpler to work with, which is why they are much more popular. Using a type system lets you rule out all kinds of annoying bugs so that you can focus your “proof energy” only on the important stuff. As a simple example of this, consider the following Java code: void setFoo(Bar bar) { this.foo = bar; assert this.foo == bar; } Without seeing more code, it is impossible to prove whether this assertion is true or not! This is because there could be any number of other threads running around mutating

this. In contrast, the following Rust code must be true (at least, so long as the Eq operator is reflexive for Bar): fn set_foo(&mut self, bar: Bar) { self.foo = bar; assert_eq!(self.foo, bar); } We know this because the type system of Rust guarantees that &mut self implies a unique reference. This is in fact exactly how Verus works. By trusting the Rust borrow checker to track uniqueness etc, it simplifies the job of the SMT solver, making proof checking must faster and more reliable than it would be otherwise. Where I think Rust should go on reliability So my take is that Rust should double down on if it compiles, it works. As expressive as Rust's type system is, there are plenty of things that it cannot capture. This includes fairly common things, like this method only reads one particular field or internal references, but also fancy things like "no logging of personally identifiable information". We should work to lift those limits, giving agents room to prove more complex properties. We also need to improve the experience of integrating a proof tool by working towards an extensible compiler. As cool as the existing Rust-based proof tools are, using them is a janky experience involing annoying macros and pinned compiler versions. We can do better. A north star: No more unsafe code I'd like to propose a north star for Rust: no more unsafe code. By that, I don't mean that we never use the unsafe keyword. Rather, I mean that we have the power to prove that all the unsafe code we write is correct and cannot cause undefined behavior. Don't get me wrong: knowing exactly what code needs to be vetted is a huge step forward, as is normalizing the idea that libraries need to be responsible for protecting their users from undefined behavior via safe API boundaries11. But given agents' ability to find vulnerabilities, I think we need to up-our-game if we want to stop the steady stream of CVEs.12 Design is an open problem But before you go about proving things or optimizing them, you need to have a clear idea of what you are building. As far as I can tell the state of the art here is basically variations on "markdown-driven-development", developing docs and hoping that agents pay attention. I don't know how much of a role for Rust there is in this space. I think there are some steps we might wind up taking that could be helpful here – for example, having ways to separate out interface from implementation so that it's easier to review the interface in isolation or to see when the agent made changes there.13 Or perhaps just providing betters tools to review that. I also think there's space for dedicated "modeling languages", something that let you iterate quickly with agents on the high-level designs and invariants without having to get into the weeds and read the code. I have more thoughts on this but it merits many blog posts all of its own. For the moment, I'll start by pointing you all at this post by Kari Wilhelm, "Principles of Anchor Engineering", which I think does a nice job of describing what it means to pull out the key elements of a design and drive the coding from there. Conclusion We covered a lot of ground here, so let me put this all together. What makes Rust a good choice for developing with agents? It offers good default perf (why pay more?) and, in conjunction with a profiler, can be tuned as much as you want. The type system gives stronger guardrails, and if you setup your agents right, you can strengthen those guardrails through techniques like new types. And what could make it better, from a language design perspective? Overall, I think it's very similar to what we were doing, except that the way we weight the tradeoffs is tweaked to account for agents doing the code authoring, and humans focused more on code comprehension: If it compiles, it works – our type system is our big advantage, we need to make it more expressive and more functional so that agents don't get stuck trying to work around it! And as an adjunct, establish a North Star: Enable Unsafe code without unsafety by supporting proof, and see the most widely used Rust libraries proven sound. Low-level enough for a kernel, high-level enough for a GUI – make sure we expose the knobs needed to tune for performance, while also making the code's intent clear (but don't worry about golfing the code down to an absolute minimum number of characters). Extensible and versatile – one of Rust's great strengths is that, with macros + our type system, we support DSLs and a wide variety of targets. We need to bring that spirit of extensibility into the compiler so we can target more styles of development (e.g., GPU, deep language interop, etc) and also integrate theorem provers and other tools that require more than just tokens. As a final close, let me point you at this interesting article by Marc Brooker, "What's easy now? What's hard

now?”. He posits that the more it is possible to clearly specify success criteria for a task, the easier it is, and that we may find that tasks we traditionally think of as ‘hard’ (e.g., building out an efficient implementation of a cryptographic algorithm) may soon become automated, whereas tasks that we have thoght of as ’easy’ (e.g., making a unique, compelling website) may become more valued. Looking at it from a PL point-of-view, I think Marc’s point reinforces my sense that the biggest action is going to come from design languages, things that let you capture the architecture and intended behavior at a very high-level. The goal would be to let humans intuitively specify a design that gets translated into effective guardrails, whether that’s type system checks, lints, or theorems to be proven. Agents would then take on the task of translating the design into code written in a traditional PL like Rust, with humans looking at it only rarely. This doesn’t make the design of traditional PLs less interesting, but it definitely makes the tradeoffs very different than they were before. Secretly I have always been jealous of people that coded using punchcards and stored things on gigantic rolls of tape or whatever. I just want to be world weary so badly! I am kinda delighted that I’m finally old enough that I can say things like “I remember when we had to edit CONFIG.SYS before we could play a computer game!” and younger programmers will gasp in horror.5 But I still feel like I must sound like a silly kid to folks who were around “back in the day”. ↩ I still remember when my dad told me he had bought me a 10MB disk and secretly he had bought a 40MB disk to surprise me. It seemed absurdly large. “I’ll never need a new disk again!” ↩ I mean my god, we’re even using vi again! My first editor! ↩ I built an electron app! And it was so easy to do! And reading weird papers and things is so much more accessible now that I can ask an LLM to translate all the math jargon for me. ↩ I remember once trying to tell Alex Crichton how I used to “order shareware from a paper catalog by putting a check in the mail” and he was like “I do not understand any of the nouns in that sentence”. Highlight of my life, right after Rust 1.0. Oh, and maybe the birth of my child and getting married. I guess. ↩ I’m not saying Rust is the only language that’s a good fit. I’m not even saying that the way that Rust is a good fit is the only way to be a good fit. I have heard for example that Go is a really good fit for agentic coding and I believe it: I think it hits a different sweet spot, one that helps agents iterate quickly (although I do suspect agents will let data races and subtle bugs slip through that Rust would have prevented). ↩ The fact that agentic development makes more room for knobs is good for Rust performance, but it is quite likely an even better fit for performance in other languages. Tuning a GC has traditionally been a job that really requires a GC expert, but agents can likely fill that role pretty well these days. ↩ On this pont, I’m even more convinced of the importance of ergonomic RC now that I work with agents. Agents are good at introducing the extra clones and things you need to deal with capturing around async tasks, but the resulting code is so annoying to read, it really hides what’s going on. ↩ “Claude, this is not what we talked about!” – me, all too often. ↩ So close though, right? I just .. couldn’t land the plane. ↩ All too often, C++ libraries consider it adequate to tell users “you’re holding it wrong” – i.e., that they are responsible for causing undefined behavior by misusing the API. Rust takes a clear position that a safe API has to protect its users, no matter how much they abuse it. ↩ To be fair, if Rust is encountering a stream of CVEs, other languages like C++ and Java are encountering a veritable flood, since they lack the protection of Rust’s type system. st takes a clear position that a safe API has to protect its users, no matter how much they abuse it. ↩ The revenge of .h files? ↩

- This Week In Rust: This Week in Rust 673 (2026/10/14 04:00)
  Hello and welcome to another issue of This Week in Rust! Rust is a programming language empowering everyone to build reliable and efficient software. This is a weekly summary of its progress and community. Want something mentioned? Tag us at @thisweekinrust.bsky.social on Bluesky or @ThisWeekinRust on mastodon.social, or send us a pull request. Want to get involved? We love contributions. This Week in Rust is openly developed on GitHub and archives can be viewed at this-week-in-rust.org. If you find any errors in this week's issue, please submit a PR.

Want TWIR in your inbox? Subscribe here. Updates from Rust Community Official Infrastructure Team 2026 Q3 Recap and Q4 Plan Foundation Rust Foundation Board Updates | October 2026 Newsletters The Embedded Rustacean Issue #82 Project/Tooling Updates Bevy 0.20 Creusot 0.14.0 Yagi Signal Lab Observations/Thoughts All code will converge on Rust Why 'externalized' proofs of cyclic trait impls does not work Visualizing crossbeam Epoch Based Reclamation How Signal and WhatsApp deploy Rust to billions of Android and iOS devices An update on Rust's project goals Analyzing Rust programs with Charon Owning C types from Rust code [Video] RustConf 2025 Interview: Manish Goregaokar, Rust Devtools Rust Walkthroughs [JA] 同じ質問に2回お金を払うのをやめた: LLM の前に置くだけの重複排除プロキシを Rust (Tokio) で作った Using TrustZone-M in Rust on the Raspberry Pi Pico 2 Writing a hypermedia app in Rust Custom structs as Tree Nodes How to Profile Rust Performance using Grafana and Prometheus [Video] RustCurious lesson 11: Reborrowing and Rust's Invisible Lifetimes [Video] BAML’s Rust-Powered Telemetry: 1000× Faster & Lossless [Video] Fuzz Testing a Discrete Event Simulator Crate of the Week This week's crate is hotcode, a crate for hot-reloading Rust functions at runtime. Thanks to alordash for the self-suggestion! Please submit your suggestions and votes for next week! Calls for Testing An important step for RFC implementation is for people to experiment with the implementation and give feedback, especially before stabilization. If you are a feature implementer and would like your RFC to appear in this list, add a call-for-testing label to your RFC along with a comment providing testing instructions and/or guidance on which aspect(s) of the feature need testing. No calls for testing were issued this week by Rust, Cargo, Rustup or Rust language RFCs. Let us know if you would like your feature to be tracked as a part of this list. Call for Participation; projects and speakers CFP - Projects Always wanted to contribute to open-source projects but did not know where to start? Every week we highlight some tasks from the Rust community for you to pick and get started! Some of these tasks may also have mentors available, visit the task page for more information. No Calls for participation were submitted this week. If you are a Rust project owner and are looking for contributors, please submit tasks here or through a PR to TWiR or by reaching out on Bluesky or Mastodon! CFP - Events Are you a new or experienced speaker looking for a place to share something cool? This section highlights events that are being planned and are accepting submissions to join their event as a speaker. No Calls for papers or presentations were submitted this week. If you are an event organizer hoping to expand the reach of your event, please submit a link to the website through a PR to TWiR or by reaching out on Bluesky or Mastodon! Updates from the Rust Project 603 pull requests were merged in the last week Compiler allow unary operand types to be inferred later avoid cloning impl trait AST nodes in late resolution avoid double-checking self.bound_vars' len in ValidateBoundVars' TypeVisitor impl check is_indirect_first_projection when replacing in RefProp implement clone_from for BorrowckDomain MIR move elimination (3/6): PreciseLiveness perf: cache FFI safety results in improper_ctypes lint reserve more capacity for Symbol interner skip runtime fma detection when enabled at compile time on x86_64 stabilize the WebAssembly wide-arithmetic feature stabilize optimize attribute treat union fields as aliasing each other in value_analysis Library add fs::rename_noreplace directly deallocates the nodes instead of traversing every element when BTreemap dropping fast path for BTreeMap::Iter::next make NonNull::{read,write}_volatile unstably const stabilize ptr::try_cast_aligned vectorized vec::dedup pre-scan Cargo avoid absolute paths in fingerprints if they are relative to the workspace root fix: cargo installing from a directory source fails for binary packages with no dependencies config: Don't process includes for credentials.toml files git: Add future-incompat warning in prep for removing auto-detection hints: support minimum optimization levels job: Allow explicit child process breakaway on Windows layout: exclude a build-dir created as a parent from backups mock-std: Set stable crate versions to 0.0.0 pkgid-spec: Builtin package spec parsing and formatting refactor: modernize some API usages for 1.99 restrict workspace-relative dep-info to workspace packages rustdoc: suppress edition unspecified warnings test: mark requiring ssh for tests shelling out to it Rustdoc fix more incoherent_impl linking prefer local paths over remote ones when

foreign item is locally reexported check reexport chain if inline before hidden Clippy configuration: add too_long_first_doc_paragraph lint to the check-private-items list do not trigger manual_is_multiple_of on literal divisors ensure a deprecation warning is emitted for lint-inconsistent-struct-field-initializers new lints deprecated_attributes_without_{since, note} fix {panic, todo, unimplemented, unreachable} fix module_name_repetitions false negative on grouped re-exports fix needless_lifetimes linting where clauses lifetimes manual_filter: don't lint when the condition needs &mut a… multiple_unsafe_ops_per_block: fix false positive in with write to union field new lint: unnecessary_box_pin skip try_from_instead_of_from_str in macro expansions support trailing ::self for unnecessary_self_imports Rust-Analyzer support inline call in macro call add downcast handling in enum closure capture assist do not add lifetime only turbofish check call no-paren after path colons complete cfg key-value in string inside cfg!() complete in top-level after inner-doc don't source-map #[doc] strings that contain escapes enum niche decoding should match rustc fix handling of generic params in anon consts pass rustc path to unstable-book-gen when the initializer in let diverges, allow non-diverging else mini refactors in closure capture inference perf: querify some things that are called by the solver and are fairly expensive to compute stabilize optimize attribute sync target feature implications from rustc_target typeMismatch quickfix position error: to_owned() Rust Compiler Performance Triage No significant improvements or regressions this week. Most of the PRs marked as improvements had a similar perf. result, which was caused by codegen noise in the measurements. There were some nice wins for memory usage though, thanks to #163915 and especially #163887. Triage done by @Kobzol. Revision range: cc9a14f7..7211ea59 Summary: (instructions:u) mean range count Regressions  (primary) 0.2% [0.1%, 0.6%] 73 Regressions  (secondary) 0.3% [0.1%, 1.4%] 57 Improvements  (primary) -0.3% [-1.0%, -0.1%] 21 Improvements  (secondary) -0.3% [-1.7%, -0.1%] 28 All  (primary) 0.1% [-1.0%, 0.6%] 94 3 Regressions, 21 Improvements, 2 Mixed; 1 of them in rollups 27 artifact comparisons made in total Full report here. Approved RFCs Changes to Rust follow the Rust RFC (request for comments) process. These are the RFCs that were approved for implementation this week: No RFCs were approved this week. Final Comment Period Every week, the team announces the 'final comment period' for RFCs and key PRs which are reaching a decision. Express your opinions now. Tracking Issues & PRs Rust tier 3 target policy: target CPU and features must be supported by upstream backend Implement AsMut<T> and AsRef<T> for !. borrowck dataflow: respect borrow shallowness when killing borrows that conflict with assignments move implied bounds computation out of borrowck Allow : after pat_param 1.100-beta regression: box patterns have been removed (feature box_patterns) Stabilize -Zembed-metadata Make std::fs::{File, ReadDir, DirEntry} always needs_drop even when unsupported. Rust RFCs RFC: Allow (outer) cfg-attributes on elements of tuple types and tuple (struct) patterns Language Reference nail down the final validity rules: references and unions Leadership Council All Hands 2027 hotel pre-booking No Items entered Final Comment Period this week for Cargo, Compiler Team, Language Team or Unsafe Code Guidelines. Let us know if you would like your PRs, Tracking Issues or RFCs to be tracked as a part of this list. New and Updated RFCs RFC: Support registry withholding RFC: unsafe global_asm RFC for limited crates.io self-service version deletion Upcoming Events Rusty Events between 2026-10-14 - 2026-11-11  Virtual 2026-10-14 - 2026-10-17 | Hybrid (Barcelona, ES) | EuroRust EuroRust 2026 2026-10-15 | Hybrid (Seattle, WA, US) | Seattle Rust User Group October, 2026 SRUG (Seattle Rust User Group) Meetup 2026-10-18 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: Third Sunday 2026-10-20 | Virtual (Washington, DC, US) | Rust DC Mid-month Rustful 2026-10-21 | Hybrid (Vancouver, CA) | Vancouver Rust Disposable Agent Sandboxes in Rust 2026-10-21 | Virtual (Cardiff, UK) | Rust and C++ Cardiff Operating Systems Book Club: Paging Mechanisms 2026-10-22 | Virtual | Rust  Maven Rust and the GPU from Native to Web: An Introduction to wgpu 2026-10-22 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-10-26 | Virtual | Rust  Maven No Python was harmed: teaching a tiny MCU to learn as it goes 2026-10-27 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Fourth Tuesday Rust Bookclub 2026-10-27 | Virtual

(London, UK) | Women in Rust Lunch & Learn: Reasoning with Async Rust 2026-10-29 | Virtual | Rust  Maven Creating a hexadecimal editor in Rust 2026-11-01 | Virtual | Rust  Maven Cursed Rust 2026-11-01 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: First Sunday 2026-11-03 | Virtual | Rust  Maven Solving Real-World Planning Problems in Rust with SolverForge 2026-11-03 | Virtual (London, UK) | Women in Rust  Community Catch Up 2026-11-04 | Virtual (Indianapolis, IN, US) | Indy Rust Indy.rs - with Social Distancing 2026-11-05 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-11-10 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Second Tuesday Asia 2026-11-03 | Tel Aviv-Yafo, IL | Rust  TLV In person Rust November 2026 at AWS in Tel Aviv Europe 2026-10-14 | Barcelona, ES | BcnRust 22nd bcnrust session 2026-10-14 - 2026-10-17 | Hybrid (Barcelona, ES) | EuroRust EuroRust 2026 2026-10-14 | Köln, DE | Rust Cologne Rust in October: Macro-magic 2026-10-20 | Leipzig, DE | Rust - Modern Systems Programming in Leipzig Creating a realtime web Multiuser Dungeon game with Dioxus 2026-10-22 | Karlsruhe, DE | Rust Hack & Learn Karlsruhe Karlsruhe Rust Hack and Learn Meetup bei BlueYonder 2026-10-22 | Toulouse, FR | Rust Toulouse Rust Toulouse Meetup - Rust & Python interoperability 2026-10-27 | Aarhus, DK | Rust Aarhus Hack Night: Rust meets AI 2026-10-27 | Paris, FR | Rust Paris Rust meetup #89 2026-10-31 | Stockholm, SE | Stockholm Rust Ferris' Fika Forum #31 2026-11-01 - 2026-11-03 | Italy, IN | RustLab RustLab - The International Conference on Rust in Italy 2026-11-04 | Dublin, IE | Rust Dublin Rust 263: Getting Started with the New Rust Driver for Oracle AI Database 2026-11-05 | Milano, IT | Rust Language Milan Digital Model Trains & Rust: Bare-Metal DCC Command Station on ESP32 North America 2026-10-14 | Los Angeles, CA, US | Rust Los Angeles Rust LA October: AI & Rust w/ Oxen.AI & Origin Lab! 2026-10-15 | Hybrid (Seattle, WA, US) | Seattle Rust User Group October, 2026 SRUG (Seattle Rust User Group) Meetup 2026-10-17 | Boston, MA, US | Boston Rust Meetup Central Cambridge Rust Lunch, Oct 17 2026-10-20 | San Francisco, CA, US | San Francisco Rust Study Group Rust Hacking in Person 2026-10-21 | Hybrid (Vancouver, CA) | Vancouver Rust Disposable Agent Sandboxes in Rust 2026-10-21 | San Francisco, CA, US | Bay Area Rust Bay Area Rust - Embedded Meetup 2026-10-22 | New York, NY, US | Rust NYC Rust NYC w/Bryan Cantrill - Code Matters: Reflections on Rust in the LLM Age 2026-10-24 | Boston, MA, US | Boston Rust Meetup Davis Square Rust Lunch, Oct 24 2026-10-28 | Austin, TX, US | Rust ATX Rust Lunch - Fareground 2026-10-31 | Boston, MA, US | Boston Rust Meetup North End Rust Lunch, Oct 31 2026-11-05 | Saint Louis, MO, US | STL Rust HTMX and Rust South America 2026-11-07 | Planalto Paulista, BR | Rust São Paulo Meetup Encontro do Rust-SP - Agora no Luma If you are running a Rust event please add it to the calendar to get it mentioned here. Please remember to add a link to the event too. Email the Rust Community Team for access. Jobs Please see the latest Who's Hiring thread on r/rust Quote of the Week I think that there are a nonzero number of people who are experts in the field who would genuinely respond to "how does C work?" with "it doesn't, next question" – Clar Fon at the rust-lang zulip Thanks to kleines Filmröllchen for the suggestion! Please submit quotes and vote for next week! This Week in Rust is edited by: nellshamrell llogiq ericseppanen extrawurst U007D mariannegoldin bdillo opeolluwa bnchi KannanPalani57 tzilist Email list hosting is sponsored by The Rust Foundation Discuss on r/rust

- Firefox Tooling Announcements: Happy BMO Push Day! (20261013.2) (2026/10/13 22:21)
  Github Link The following changes have been pushed to bugzilla.mozilla.org: Bug 2079152 - Unable to select content-type when creating attachment Bug 2079453 - Upgrade devbridge-autocomplete to 2.0.1 and fix autocomplete dropdown reopening after the field loses focus Bug 2065173 - Migrate Group REST resource to native Mojo API Bug 2078690 - Migrate User REST resource to native Mojo API: user objects (get, create, update, suggest, whoami, offer_account_by_email) Bug 2079991 - When one update requests needinfo from several users , the webhook’s event.changes has a single flag.needinfo entry for the last requestee only; the others are dropped. Discuss these changes in the BMO Matrix Room 1 post - 1 participant Read full topic

- Firefox Developer Experience: Firefox WebDriver Newsletter 158 (2026/10/13 18:02)
  With the WebDriver related release notes part of the MDN developer release notes and having a 2-week cycle for new Firefox releases now we are going to stop publishing this newsletter. This one will be the last one of the series of blog posts that we did since Firefox 114.Thank you to everyone who followed these posts and provided feedback over the past releases. WebDriver is a remote control interface that enables introspection and control of user agents. As such, it can help developers to verify that their websites are working and performing well with all major browsers. The protocol is standardized by the W3C and consists of two separate specifications: WebDriver classic (HTTP) and the new WebDriver BiDi (Bi-Directional). This newsletter gives an overview of the work we've done as part of the Firefox 158 release cycle. Contributions Firefox is an open source project, and we are always happy to receive external code contributions to our WebDriver implementation. We want to give special thanks to everyone who filed issues, bugs, and submitted patches. In Firefox 158, multiple WebDriver bugs were fixed by contributors: F3tch landed several patches to allow setting of camera and microphone permissions, reject zero-sized screenshots, add a vendor-specific WebDriver BiDi command for listing installed extensions, and update cookie test expirations. Khalid AlHaddad improved the PollPromise to throw an error by default and not only log a warning when it times out. WebDriver code is written in JavaScript, Python, and Rust, so any web developer can contribute! Read how to set up the work environment and check the list of mentored issues for Marionette or the list of mentored JavaScript bugs for WebDriver BiDi. Join our chatroom if you need any help to get started! All Changes A complete list of developer-facing changes included in this Firefox release is available in the MDN Firefox 158 Release Notes.
- Firefox Tooling Announcements: Firefox Profiler Deployment (October 13, 2026) (2026/10/13 15:10)
  The latest version of the Firefox Profiler is now live! Check out the full changelog below to see what's changed: Highlights: [Markus Stange] Speed up activity graph drawing (#5888) [Florian Quèze] profiler-cli: accept several marker handles and ranges in marker info (#6277) [Nazım Can Altınova] Add "profiler-cli profile save " (#6340) [Nazım Can Altınova] Add global `–permalink` and `–short-permalink` flags to profiler-cli (#6338) [Markus Stange] Serve production build by default during local development (#6353) [Nazım Can Altınova] Add global `–permalink` and `–short-permalink` flags to profiler-cli (#6338) [Nazım Can Altınova] Add the ability to apply source maps to the profiles using `profiler-edit` (#6382) Other Changes: [Nazım Can Altınova] Fix the upload progress bar not showing in light mode (#6348) [Nazım Can Altınova] Fix the variable name passed to the malformed response error strings (#6351) [Markus Stange] Add missing await when calling Chrome importer (#6346) [Markus Stange] Use "falls through" for all fallthrough comments (#6344) [Markus Stange] Move source map related profile logic files into profile-logic/source-maps/ (#6349) [Markus Stange] Make the source view actually read-only. (#6313) [Markus Stange] Run CI on merge queue branches (#6352) [Nazım Can Altınova] Fix "… was not wrapped in act" warnings in tests (#6362) [Nazım Can Altınova] Add a scope tree test for shorthand destructuring defaults (#6367) [fatadel] Accept --session before subcommands in profiler-cli (#6358) [Nazım Can Altınova] Replace the deprecated loose option with Babel assumptions (#6366) [fatadel] Clarify how to save profiles for later access (#6355) [Nazım Can Altınova] Make `yarn test-cli` and `yarn test-all` faster (#6369) [Nazım Can Altınova] Use a profile with tab information in the tab selector URL tests (#6370) [fatadel] Move FileIO table labels into the Gecko profile upgrader (#6375) [fatadel] Clear preview selections when zooming out in the CLI (#6380) [fatadel] Sync: l10n → main (October 13, 2026) (#6389) Big thanks to our amazing localizers for making this release possible: de: Ralf Duehnfahr el: Jim Spentzos en-GB: Ian Neal es-CL: ravmn fr: Théo Chevalier fy-NL: Fjoerfoks ia: Melo46 it: Francesco Lodolo nl: Fjoerfoks ro: IulianArian ro: robbp ru: Valery Ledovskoy sv-SE: Andreas Pettersson tr: Selim Şumlu tr: Sena nur zh-TW: Pin-guang Chen Find out more about the Firefox Profiler on profiler.firefox.com! If you have any questions, join the discussion on our Matrix channel! 1 post - 1 participant Read full

topic

- The Mozilla Blog: Mozilla introduces Firefox Enterprise (2026/10/13 04:01)
  Institutions are recalculating who controls the technology they rely on. Mozilla has an independent option to offer. For years, organizations consolidated technology around a small number of providers to make things simpler. Fewer systems. Tighter integrations. More economies of scale. But that consolidation went well beyond technology. It consolidated dependency, too. The resulting risks have been debated, studied, and challenged along the way, but concentration continued. More work, data and critical infrastructure kept moving to the same few providers, while credible alternatives narrowed.The deeper that dependency runs, the harder and more costly it becomes to change course. When something fails, changes or falls outside an institution’s control, that dependency becomes a risk for CIOs and CISOs to manage. Nowhere are these stakes more visible than in Europe. Now institutions are re-assessing. Governments are pushing for greater digital autonomy, while regulated industries face growing pressure to find and manage their dependencies.  There’s a name for much of this debate about technological control: “sovereignty.”  What digital sovereignty means in practice varies. Across Europe, governments are investing in the security and resilience of critical IT infrastructure, reducing strategic dependencies and putting technological autonomy higher on national agendas. Different approaches, same underlying concern. Strip away the regional policy definitions and industry distinctions and a pretty practical, universal question emerges: how much technological dependency should we risk? And a very specific, very important layer has been largely missing from that conversation: the browser. The overlooked layer of modern business The browser. Applications run there. Employees authenticate in it. Sensitive data transits through it. Files are uploaded and downloaded and AI tools are increasingly used through it. That raises the same questions of dependency and control: whose browser technology are you relying on, and who controls the infrastructure used to manage it? Introducing Firefox Enterprise Firefox Enterprise is built for organizations with strict security, governance, and compliance requirements. It pairs the Firefox browser, built on Mozilla’s independent Gecko engine, with a management console organizations can run on their own infrastructure. From that console, IT teams can centrally manage browser policy and access, govern how data moves through the browser and set controls around AI use. IT and security teams get more control over their data, IT infrastructure and browser environment — helping strengthen technological independence and operational resilience. “If all your options are built with the same technology underneath, ‘choice’ is a lie. So is control,” said Anthony Enzor-DeMeo, CEO of Mozilla. “We built Firefox differently. And we’re bringing its independent architecture to institutions that are ready to rethink their dependencies and risk.” Bringing the browser under control We’re starting with the browser because we see it as a key, but under-governed, layer. Firefox Enterprise gives institutions more control over the browser and the infrastructure managing it. And because Firefox runs on Gecko, the only major browser engine developed outside Google and Apple, institutions have another path towards reducing dependence on a single architecture. Finally, Firefox Enterprise is built to live alongside the systems that IT operations already use. Keep your identity provider. Keep your SIEM. Keep your MDM. Just add a browser whose management infrastructure you operate, so you decide where data lives and have greater visibility into how it moves. For us, supporting the goal of technological sovereignty comes down to four things: More control. More transparency. More choice. More resilience. Learn more about Firefox Enterprise. A big bet on familiar ground We know this territory. Independence isn’t a new direction for Mozilla. We’ve spent more than 25 years building and securing browser technology that intentionally works outside of the dominant architecture of the day. Firefox is used by roughly 200 million people every month and is already deployed across thousands of organizations.That independence has always mattered to the web. Now it has practical value for institutions looking to reduce technological concentration — and dependency — in their own environments.  This is the beginning Our ambition is big but simple: make technology

independence practical. That means putting institutions in real control of the infrastructure and data they depend on, without asking them to rebuild or start over. Closing blind spots. Defining where policies, access and data live. Keeping core management within the institution's environment. And bringing an overlooked layer — the browser — under institutional control. Firefox Enterprise is unveiling across Europe with an initial sales focus on France, Germany, Austria and Switzerland, with more markets to follow. Thunderbird Enterprise is also in development, extending this customer-controlled approach to email and communications. We'll have more to share as that work progresses. If your organization wants greater control over its browser infrastructure, we'd like to work with you. The post Mozilla introduces Firefox Enterprise appeared first on The Mozilla Blog.

- Niko Matsakis: Why 'externalized' proofs of cyclic trait impls does not work (2026/10/10 15:23)
  For this post, I wanted to talk about two different approaches to handling supertraits. I'm calling them modular proofs vs external proofs. The key idea of this post is that, if we want to have cyclic trait impls, we really need to use a modular proof strategy, where the impl establishes all supertraits hold. Previously we had considered an external strategy, where the piece of code using the impl has the obligation to prove the supertraits hold. Modular proofs always seemed better but I did not think they were workable in the past. But I have become convinced that external proofs are incompatible with Rust as designed, and hence modular proofs are really the only option1. This post dives into that reasoning, and also gives a bit of explanation of what I mean by proofs in the first place. Traits and supertraits So what do I mean by modular vs external proofs? Well, it all comes down to who is responsible for proving that supertrait obligations hold. Consider a trait like Magic: trait Magic: Copy { } The supertrait declaration means that, whenever X: Magic for some type X, it should be true that X: Copy. We make use of this in generic functions: fn is_copy<T: Copy>() { } fn is_magic<T: Magic>() { // Legal, because `T: Magic` implies `T: Copy` is_copy::<T>(); } The trick is that the compiler has to make sure that this implication holds – i.e., for every type X that implements Magic, X also implements Copy. So how does it do it? Modular proofs: the impl must show supertraits hold The obvious answer is to make proving supertraits part of deciding whether an impl is valid. For any impl of Magic, we can require that the Copy supertrait holds. So an impl like this would be illegal: // In a modular system, this impl is *illegal* impl Magic for String { } This impl is illegal because it would require that String: Copy, and that does not hold. Seems good. Modular proofs are a bit tricky I am calling these proofs modular because the idea is that we can prove an entire program is valid by proving each part of it separately. In "programming language" theory, this is typically called a "modular" check, as it works by breaking up the entire program into modules that can be independently checked. The idea with a modular proof is that we can trust impls to show that the supertrait relationships hold, we don't have to go and re-prove them over and over. If the impl is wrong, the impl will be invalid, but our code is fine. So if we have impl Magic for String, that implies the rest of the program can prove that String: Magic: fn string_is_magic() { // Legal, because there is an impl for `String: Magic`: is_magic::<String>(); } In fact, since we know that Magic implies Copy, the rest of the program can even rely on impl Magic for String to conclude that String: Copy: fn string_is_copy() { // Legal, because there is an impl for `String: Magic`, // and `Magic` implies `Copy`: is_copy::<String>(); } So long as impl Magic for String is invalid, none of this poses a problem to soundness, since the program overall doesn't type-check. Comparison with functions An easy way to understand the idea of modular checks is to think of functions. Imagine you have a function like this one: fn compute_sum(a: i32, b: i32) -> i32 { format!("{a} + {b}") // <-- Error } Clearly, this function is not legal. It takes two integers and promises to return a third integer, but in fact it returns a String. So the function is illegal. But if you have a call to that function from elsewhere, we consider that other call to be legal: fn use_sum() { let c: i32 = compute_sum(2, 20); // OK } Here, use_sum is relying on compute_sum to obey its contract. It's not the job of use_sum to check that, it can just assume it is true. The catch: how do we decide the impl is

invalid There is a bit of a catch though. How do we decide if the impl is invalid? The basic idea was that impl Magic for String would have to prove that String: Copy. But we just saw that it could, in fact, do that by using itself. In other words, if we aren't careful, we can provide a proof that String: Copy like… String: Copy because Magic implies Copy and String: Magic because impl Magic for String exists and then we would (incorrectly) conclude that the impl is valid. So clearly we need to do something to rule that out. We need a rule that says, when we are proving that an impl is valid, that proof cannot recursively rely on the impl itself.[^termination] I'll come back in a future post to ways we might do that, but for now, I want to explore another alternative. External proofs: the user of the impl must show supertraits hold When we first looked at this problem, way back in 2018 or so, we thought of another approach. What if we said that an impl is not responsible for proving supertraits. Instead, the idea would be that impl Magic for String is not enough to say that String: Magic. It only says that Shallow(String: Magic) – i.e., String implements Magic in a shallow way, but not in a deep way that includes the full supertraits. To prove that String: Magic, we have to show that Shallow(String: Magic) and Shallow(String: Copy):2 Shallow(String: Magic) Shallow(String: Copy) ---------- Magic fully implemented String: Magic This has the somewhat counterintuitive implication that impl Magic for String is actually legal in an "external proof" approach: // In an external system, this impl is LEGAL // (but unusable) impl Magic for String { } The saving grace is that, while this impl is legal, you can't actually use it. This function for example does not compile: fn string_is_magic() { // NOT legal in an external system: // * We can prove that `Shallow(String: Magic)` // * We CANNOT prove that `Shallow(String: Copy)`. is_magic::<String>(); } Here, String: Magic doesn't hold even though there is an impl of Magic for String, because the caller also has to check that String: Copy is implemented, and it is not. Huh, interesting. Comparison to functions: external is awkward the "external proof" approach for impls is clearly a bit awkward. If we make the comparison to functions, it's as if the caller has to double check that the callee's body matches its return type, it can't actually trust the declared signature. But, awkward or not, it does resolve our problem: given impl Magic for String, we cannot prove String: Copy, and hence we cannot prove that String: Magic. We can only prove that Shallow(Magic: String), which doesn't imply that the supertraits hold. But external doesn't work with unsafe traits Based on the above, for a long time, I was working with the assumption that, weird as they are, we would go with the "external proof" approach. However, as Ralf Jung and lcnr pointed out to me recently, this is very challenging to reconcile with unsafe traits. Consider an unsafe trait like Nullable: // A type that can be safely transmuted from `0_usize`. unsafe trait NullWord { } The way that Rust works, when we write an unsafe impl, it is the job of that impl to prove that the unsafe conditions hold. Other parts of the program get to trust the impl. So if I write a function like this one, it should be considered safe:3 fn foo<T: NullWord>() -> T { std::mem::transmute(0_usize) } Now imagine that I wrote an invalid impl like this one: // INVALID: We are asserting that `Box` can be null, // which is not true! unsafe impl<T> Nullable for Box<T> { } Given this program I could clearly call foo::<Box<u32>>(), but that would "go wrong" (cause "undefined behavior"). I think we would all agree that the fault lies in the impl. And yet, that is inconsistent: we say that the impl alone cannot be trusted to figure out if the supertraits are implemented, but it can be trusted to figure out if the unsafe impl is valid? Conclusion I definitely believe that we want to treat the "extra conditions indicated by unsafe" as a more general version of the other obligations that an impl has to establish to show that the trait holds– and therefore that we must have modular proofs. That's kind of a relief, because something always felt wrong about external proofs, but it was hard to put my finger on a concrete problem. In the next post in this series (whenever that may be…), I expect to cover the approach to coinductive modular proofs that I landed on. Then I expect to talk about an alternative that was proposed to me that I find quite appealing. I think this was obvious to Ralf Jung from the start. But it took me a bit. ↩ This notation is called an inference rule. The conditions above the line are the premises and the bottom line is the conclusion. It says that, if you know the premises are true, you can infer that the conclusion holds. ↩ In point of fact, I believe this will not

compile because of special rules about unsafe, but that’s not relevant to the point I’m trying to make. ↩

- Firefox Tooling Announcements: Firefox DevTools MCP 0.10.5 released (2026/10/09 14:03)
  New release of Firefox DevTools MCP. Available on npm at https://www.npmjs.com/package/@mozilla/firefox-devtools-mcp Fixed set_firefox_prefs and get_firefox_prefs work again. They now use WebDriver BiDi to run in the privileged context list_pages no longer fails when the URL or title of a tab cannot be read. That tab is listed with (unknown) values instead Changed take_snapshot and the UID-based input tools now use WebDriver BiDi instead of WebDriver Classic The snapshot script now runs in a sandbox, so page scripts cannot interfere with it and it does not add globals to the page The Cursor plugin manifest is replaced by the root plugin.json and mcp.json files from the Agent Plugins 1.0 standard, which cursor.directory reads Full changelog: Release v0.10.5 · mozilla/firefox-devtools-mcp · GitHub Repository and issues: GitHub - mozilla/firefox-devtools-mcp: Model Context Protocol server for Firefox DevTools - enables AI assistants to inspect and control Firefox browser through WebDriver BiDi · GitHub Public chatroom: https://chat.mozilla.org/#/room/#firefox-devtools-mcp:mozilla.org 1 post - 1 participant Read full topic
- The Servo Blog: August in Servo: CSS Grid, text selection, and more! (2026/10/09 00:00)
  Servo 0.6.0 contains all of the changes we landed in August, which came out to 417 commits! The big news is that we’ve turned on support for CSS Grid by default (@nicoburns, #45621). This has had a huge positive impact on the visual appearance of many websites! We’ve shipped several new web platform features: touch-action (@yezhizhen, #47043) animation-composition (@nicoburns, #47405) color-scheme (@TimvdLippe, #47042) full-width, math-auto, full-size-kana values for text-transform (@SimonSapin, #47160, #47160, #47225) Plus a bunch of new DOM APIs: requestAnimationFrame() on DedicatedWorkerGlobalScope (@Taym95, #46232) ancestorOrigins on Location (@TimvdLippe, @mrobinson, #46056) imagesrcset and imagesizes on <link rel=preload as=image> (@lumiscosity, @TimvdLippe, #47103) style on CSSFontFaceRule (@simonwuelker, #47388) isReloadNavigation and isHistoryNavigation on Request (@Taym95, @yezhizhen, #46917) name=color-scheme on <meta> (@TimvdLippe, @jdm, #46993) focusin and focusout events (@RichardTjokroutomo, @yezhizhen, #46980) new TextEvent() (@moamenhesham0, @mrobinson, #47391) PeriodicWave and createPeriodicWave on BaseAudioContext (@kevlu93, #47492) This release also contains significant improvements to navigation, document loading, and session history, which are some of the gnarliest parts of the HTML specification. Who knew that loading about:blank was so complicated? The upshot to all this work is that pages that open additional windows or include iframes work much more reliably. This is another big update, so here’s an outline: Security Real world compat Work in progress Embedding API For users and developers More on the web platform Performance and stability New contributors Security We upgraded to SpiderMonkey v153, which brings many new security and crash fixes to Servo’s JavaScript engine integration (@sagudev, @Gae24, #47489). Navigating away from a document that activated fullscreen mode now exits fullscreen mode (@shubhamg13, @jdm, #47123). Real world compat CSS Grid enabled by default means that many websites with traditional grid layouts now look correct, like rustweek.org. v0.5.0 v0.6.0 2026.rustweek.org It also makes a lot of websites look better that you wouldn’t assume rely on CSS Grid: v0.5.0 v0.6.0 labs.jensimmons.com Panning gestures initiated by touch inputs now lock to the dominant axis (@yezhizhen, #46983). This results in more predictable panning when interacting with pages in Servo. We fixed a crash affecting gumroad.com.au (@lumiscosity, #47266). We’re interested to hear how well your favourite websites run in Servo! Report successes in this Zulip thread or this Github discussion, and failures in our GitHub issues. Work in progress Accessibility support in Servo continues to make great strides; it can be enabled with the --pref accessibility_enabled command line argument. Servo now reports positions and sizes for items in the content accessibility tree (@philocalyst, @alice, @mrobinson, #46996), allowing macOS VoiceOver to work more reliably. We also report many ARIA roles for web content elements to accessibility toolkits (@mukilan, #47318).

Full list of supported roles alert banner blockquote caption code complementary contentinfo definition deletion directory document emphasis feed figure generic insertion list log main math navigation none note paragraph presentation rowgroup search status strong term time timer Servo’s support for Document.execCommand() is rapidly improving, and can be enabled with --pref dom_exec_command_enabled. We fixed issues with the inserttext command (@Psychpsyo, #47224, #47306), editing uneditable content (@Psychpsyo, #47362), computing contained nodes (@TimvdLippe, #47429), adjusting selections in contenteditable (@mrobinson, #47393, #47581, #47581), and implemented the indent and insertLineBreak commands (@Psychpsyo, #47416, #47445). We made a lot of progress on the LargestContentfulPaint performance metric, implemented behind the --pref largest_contentful_paint_enabled argument. Servo now supports the element and loadTime attributes, (@shubhamg13, #46553, #47475), URLs for video element posters (@shubhamg13, #43376), accurate size information (@shubhamg13, #43378), and painting text nodes (@shubhamg13, #47212). There was preparatory work for a future AudioWorklet implementation (@niyabits, #46856, #47016, #47199). See Niya’s blog post about her Outreachy project for more details! We’re continuing work on the ServiceWorker feature, which can be enabled with --pref dom_serviceworker_enabled. This month we implemented the CookieStore interface behind the --pref dom_cookiestore_enabled command line argument (@Taym95, #47044) as well as the Cache interface (@gterzian, #47220). Servo now supports a disk-backed HTTP cache that can be enabled with --pref network_http_disk_cache (@Narfinger, #43819). This cache persists between runs, but there is not yet any way to clear it. Embedding API Breaking change: WebView::toggle_sampling_profiler has been removed (@lumiscosity, #47014). To perform sampling profiling on Servo, follow the instructions in the book. Breaking change: The variants of MouseButton have been renamed to Primary, Secondary, and Auxiliary (@mrobinson, @SimonSapin, #47327). We’ve added several new Cargo features reflecting common usage patterns: default_web_features, to include optional web platform features in the build (e.g. WebXR and WebGPU) (@jschwe, #44032) bundled, to include required resources in the binary (@jschwe, #44032) brotli-compression-stream, enabling support for brotli content in the CompressionStream/DecompressionStream web APIs (@Narfinger, @jdm, #47197) webcrypto, to enable support for the Crypto and WebCrypto web APIs (@Narfinger, #47198) webgl, to enable support for WebGL (@janeoa, #47200) For users and developers One visible change is that interactive text selection now works! This is the result of lots of work across our script and layout code to ensure it respects the actual geometry of the page content (@mrobinson, @SimonSapin, #47022, #47025, #47329, #47358, #46982, #47425, #47438, #47437, #47581, #47070, #47048, #46963). Building Servo now requires at least Clang 19 on all platforms (@jschwe, #47421). Additionally, the cargo-ohos tool is now required for building for OpenHarmony (@jschwe, #47334). Debug builds on Windows can now complete successfully (@jschwe, #47372). Linux arm64 builds have been customized to work on Raspberry Pi 5 (@mukilan, #47113). Any libservo embedders targeting the same platform will require a similar change if they are using jemalloc. servoshell Under Wayland and OpenHarmony, we now integrate with the system clipboard (@TheZoq2, @yezhizhen, @jschwe, #46938, #47177). The macOS command key and Windows logo key is now reported as Meta instead of Super (@eachra-bawn, #47330). The Android port has had many rough edges filed off as part of modernizing it for the Compose UI (@veyndan, #47656, #47491, #47494, #47697, #46955, #47644, #47652, #47276, #47277, #47299, #47439) and converting its implementation to Kotlin (@veyndan, #46926, #46925, #46951, #46965). Library consumers are no longer responsible for implementing pause and resume support (@veyndan, #47555). Meanwhile, the OpenHarmony port now supports external keyboards with US keyboard layouts (@jschwe, #47359). Additionally, there are now better fallback font selections for CJK characters (@RichardTjokroutomo, #47116, #47664). Setting the device_pixel_ratio_override preference is no longer ignored on Android and OpenHarmony (@jschwe, #47081). Performance and stability There were many small changes this month that removed unnecessary allocations (@Narfinger, @jschwe, @yezhizhen, #47264, #47147, #47066,

#47249, #47249, #47285, #47307, #47343, #47361, #47395, #47427, #47454, #47474, #47506, #47479, #47530, #47541, #47566, #47509, #47622). We also reduced GC interactions leading to better JS performance (@jdm, @Narfinger, @webbeef, @TimvdLippe, @mrobinson, #47513, #47250, #47158, #47165, #47049, #47251, #47510, #46981). On platforms that memory map font data (such as OpenHarmony and macOS), we can now save time that would otherwise be spent loading font files from disk (@jschwe, #47029). Similarly, reading system fonts is no longer in the critical path of starting up the engine (@Narfinger, @jdm, #46476). There were were also a number of savings in the final binary size (@jschwe, @kkoyung, @Narfinger, @jschwe, #47211, #47214, #47210, #46950). We fixed crashes related to media playback (@calvaris, #46891), the 2d canvas renderer (@yezhizhen, #46915), fonts with negative line gaps (@jschwe, #47374), and resolving fonts variants (@simonwuelker, #47356). Canvas renderer worker threads can now be adjusted by the thread_pool_canvas_workers preference, allowing more performance granularity according to embedder needs. (@jschwe, #46691, #47018) We've continued our long-running effort to use the Rust type system to make Servo's integration with SpiderMonkey safer and more reliable (@jdm, @treetmitterglad, @Narfinger, @Gae24, #47398, #47296, #47051, #47005, #47323, #47297, #47322, #47368, #47446, #47317). We're also working on statically preventing a footgun that can lead to uncollectible GC cycles and leaking memory (@Gae24, #47355, #47650). Historically, about:memory has reported all memory usage by Rust implementations of JS APIs as one undifferentiated js/malloc-heap number. We now break it down by memory usage of nodes in document trees (@pylbrecht, #46931) and interfaces that are not part of the DOM (@jdm, #47531). We also report memory usage for the network stack's cookie database (@Narfinger, #47248) and the canvas renderer backend (@janeoa, @jdm, #46958). We've added an experimental threadboost mechanism (@jschwe, @mrobinson, #46843, #47027) that has been observed to improve Speedometer results by 50% and allow canvas rendering to reach 60 FPS. This is initially enabled for OpenHarmony, but future expansion to other Linux-based platforms is expected. More on the web platform We upgraded to resvg 0.48, fixing many issues with SVG images with missing width/height attributes that specified a viewBox (@nicoburns, #46937). Upgrading to Taffy 0.14 fixed a number of CSS grid-related crashes, support for grid names, better intrinsic sizing, and improved performance (@nicoburns, #47031, #47493). We also fixed some issues in the Taffy grid integration for more accurate baselines (@nicoburns, #47518). Session history traversals (i.e. navigating backwards or forwards in history) now wait for any outstanding traversal to complete (@mrobinson, #47293, #47485). Meanwhile, navigations to about:blank are skipped if the initial about:blank document is present (@jdm, #47074). about:blank documents no longer use the asynchronous HTML parser (@jdm, #47050), and no longer report a readyState of LOADING (@mrobinson, #47450). Opening auxiliary windows will wait to dispatch the load event until the initial about:blank document has been replaced (@jdm, @mrobinson, #47442, #46975). For non-about:blank content, we also made the document loading phases more reliable (@TimvdLippe, @mrobinson, #47426), improved conformance of javascript: URL evaluation and navigation target selection (@mrobinson, #47486, #47516), and addressed longstanding compatibility issues with cross-origin Window and Location objects (@mrobinson, #47549). Our media playback has improved; we fixed several issues around canPlayType() on HTMLMediaElement that allow Servo to report that MP4 content is playable (@jdm, #47612, #47641, #47670). Input elements with the autofocus property inside of a <dialog> now receive focus when that dialog is opened (@skyz1, @mrobinson, #45826). We improved the exception error text for numerous web APIs (@jdm, @eachra-bawn, #46960, #47124, #47168, #47580, #47430, #47127). We also fixed issues related to many other APIs, including: matching other browsers' treatment of Access-Control-Allow-Headers (@yezhizhen, #47600) WebCrypto CryptoKey usage checks (@kkoyung, #47112, #47141, #47178) and key lengths (@kkoyung, #47452, #47453) determining referrers when navigating (@mrobinson, #47487) reading name on Window inside of iframes (@TimvdLippe, @jdm, #47270) many edge cases around determining dirty roots for incremental layout

(@mrobinson, #46906) edge cases around compareDocumentPosition on Node (@SimonSapin, #46886) resolving ties in web font matching (@simonwuelker, #47449) Range.getClientBoundingRect()/getClientRects() for empty ranges (@yezhizhen, #47143) JS module fetching failures can now be retried (@Gae24, #46919) KeyEvent.keyCode values for many nonprintable keys now match other browsers (@eachra-bawn, #47461) making timing on Performance return a PerformanceTiming object (@jdm, @aniebietafia, #46990) synthesizing font variations from font-style (@simonwuelker, @mrobinson, #46918, #47364) the Get Element Tag Name WebDriver command (@yezhizhen, #47216) removed nonstandard WebDriver-related properties on the global object (@mrobinson, #47542) sniffing RAR responses correctly (@wood-ghost, #47366) Attribute nodes created during parsing now have the expected namespace prefix (@Ms2ger, #47030) improving text-transform: capitalize (@mrobinson, #46910) zero-sized margins inside of inline boxes (@Loirooriol, #47020) incorrect buttons values in pointer capture transitions (@steigeo, #45628) empty lists returned by load() on FontFaceSet (@simonwuelker, #47564) cancelling more ongoing network requests when performing a navigation (@mrobinson, #47576) shaping text across inline boxes (@mrobinson, #46829) scrollable overflow for position: sticky (@rayguo17, #43766) fixing hyperlinks for blob: URLs (@TimvdLippe, #47226) firing the error callback for decodeAudioData on BaseAudioContext (@amittenak47, @jdm, #46870) New contributors A special thanks to the following people for landing their first patch in Servo: Frans Skarman (@TheZoq2, #46938) Moamen Hesham (@moamenhesham0, #47391) Miles Wirht (@philocalyst, #46996) Vaibhav Srivastava (@vaibhav8a, #47626) wood-ghost (@wood-ghost, #47366) 禾可 (@Lfan-ke, #47514) Interested in helping build a web browser? Take a look at our curated list of issues that are good for new contributors! Donations Thanks again for your generous support! We are now receiving 8128 USD/month (+3% from July) in recurring donations. This helps us cover the cost of our speedy CI and benchmarking servers, one of our latest Outreachy interns, and funding maintainer work that helps more people contribute to Servo. Servo is also on thanks.dev, and already 36 GitHub users (+1 from July) that depend on Servo are sponsoring us there. If you use Servo libraries like url, html5ever, selectors, or cssparser, signing up for thanks.dev could be a good way for you (or your employer) to give back to the community. We now have sponsorship tiers that allow you or your organisation to donate to the Servo project with public acknowlegement of your support. If you’re interested in this kind of sponsorship, please contact us at join@servo.org. 8128 USD/month 10000 Use of donations is decided transparently via the Technical Steering Committee’s public funding request process, and active proposals are tracked in servo/project#187. For more details, head to our Sponsorship page.

- Mozilla Privacy Blog: Canada’s bill C-22 threatens encryption, users’ privacy and the security of digital economies (2026/10/08 09:35) Encryption is woven into everyday digital life. It protects our messages and passwords, but also banking and payments, health data, government services and the digital infrastructure societies rely on every day. Yet across jurisdictions, governments are increasingly considering laws that expand lawful access to data for law enforcement and national security purposes. In practice, this can mean requiring companies to create new ways to access encrypted data, introduce technical capabilities that bypass existing protections, or make otherwise secure systems insecure. These measures can weaken the very security encryption is designed to provide, with consequences for cybersecurity, privacy, fundamental rights and the wider economy. Canada’s Bill C-22 is the latest example of this trend, raising serious concerns about the security and privacy consequences of expanding government access to data. For Mozilla, protecting that security is fundamental to both the products we build and the principles we advocate for. One of the foundational principles that guide Mozilla’s mission and work holds that individuals’ security and privacy on the internet are fundamental and must not be treated as optional. Protecting people’s privacy and security is not an aspiration for us, but shapes the products we build every day: Firefox blocks trackers, protects you from profiling via cookies and fingerprinting, comes with

malware protection and a built-in VPN, offers HTTPs-only mode and protects your passwords and credit card information by encrypting them. Some of these protections are being threatened by Canada's Lawful Access Act, also known as bill C-22. Part two of the bill, the "Supporting Authorized Access to Information Act" (SAAIA) would introduce sweeping new powers to require electronic service providers to build and maintain capabilities that facilitate government access to information. If passed in its current form, companies could be asked to introduce vulnerabilities, build backdoors, bypass, weaken or otherwise defeat encryption, access data before encryption or decrypt encrypted data to provide access to law enforcement. Service providers could also be compelled to retain and access data they have purposefully chosen not to collect. Let us be clear: there is no safe way to create exceptional access to encrypted data that only the intended actor can use. An access mechanism created for law enforcement can also be discovered, exploited or abused by potentially bad actors. Such vulnerabilities and backdoors do not only undermine people's fundamental rights to privacy and data protection, but also the trust and security premises societies everywhere depend on. The same encryption that protects a private conversation also protects financial transactions, sensitive health information, business systems and critical digital services. Expanding capabilities of AI systems are only exacerbating these risks – governments should encourage the disclosure and patching of vulnerabilities, not compel companies to introduce insecurities deliberately. Only trustworthy and transparently governed digital infrastructures can be the basis for digitally sovereign societies. We are also concerned by C-22's scope, which does not stop at Canadian services or users. C-22 expansive surveillance capabilities and data retention obligations would undermine people's privacy and security everywhere, and the bill's confidentiality requirements would make it impossible for services to inform their users about security breaches or backdoors introduced. Guided by our Surveillance Principles for a Secure, Trusted Internet, we call on Canadian policymakers not to rush the legislative process to take experts' feedback into account in amending C-22 to protect everyone's security and privacy. Canada's interests are best served by regulation that protects encryption, strengthens cybersecurity and emphasizes transparency, checks and balances. The post Canada's bill C-22 threatens encryption, users' privacy and the security of digital economies  appeared first on Open Policy & Advocacy.

- The Mozilla Blog: Firefox partners with Anschutz Entertainment Group at the Uber Arena and Uber Eats Music Hall in Berlin (2026/10/08 07:00)
  Firefox is teaming up with AEG and taking to the main stage at Berlin's biggest and best indoor arenas. Starting this October, as the indoor music scene heats up and the basketball and ice hockey seasons are in full swing, fans flocking to the Uber Arena, Uber Eats Music Hall, and Uber Platz will see Firefox highly visible around the venue. They may even bump into our very own mascot, Kit, who will be on hand to help them skip the main queues courtesy of Firefox.  We know that the experiences people care about often begin online. The internet is the gateway to how they discover, plan, and navigate their way to concerts, sports events, and nights out before they happen in real life.  Firefox's presence will help make every step of that journey simple and smoother, starting at the door. People with Firefox downloaded on their phone will have access to a dedicated Firefox Fast Lane that allows them to skip the main line at the Uber Arena. To ensure the smoothest possible entry, a QR code for downloading Firefox will also be available at the Fast Lane. "People who use Firefox tend to choose things deliberately, their browser, their music, their nights out. This partnership is for them. A first for Firefox. Most browsers want more of your time online. We want you at the show, faster. Berlin is where we are starting, right across the river from our office. See you there," said John Solomon, Mozilla-Firefox Chief Marketing Officer. We are excited to bring what we stand for to Berlin's premier entertainment venue — choice, control, and a better experience to the moments that people want to enjoy and truly care about. The post Firefox partners with Anschutz Entertainment Group at the Uber Arena and Uber Eats Music Hall in Berlin appeared first on The Mozilla Blog.
- The Mozilla Blog: Jambu Palaniappan and David Parry-Jones join the Mozilla Corporation Board of Directors (2026/10/07 18:37)

Today, we are very pleased to announce two new additions to the Mozilla Corporation Board of Directors: Jambu Palaniappan and David Parry-Jones. As we build the next chapter of Mozilla, we've been focused on bringing people onto our Board who combine deep operating experience with a real commitment to the values Mozilla stands for. Jambu and David bring deep experience building and leading technology businesses, and each brings a unique perspective to our Board. Jambu Palaniappan is the Chief Executive Officer of Checkatrade, where he drives the company's mission of Powering Great Work. The home improvement platform connects consumers across the UK with quality tradespeople — every year, Checkatrade improves millions of homes and delivers billions of pounds of work to small and medium-sized businesses. Before joining Checkatrade, Jambu was Managing Partner and Head of Europe at OMERS Ventures, where he invested in and supported high-growth technology companies. Earlier in his career, he spent several years at Uber and Uber Eats, joining as one of the first 100 employees and leading expansion across Europe, the Middle East, and Africa. Jambu holds a degree in Public Policy and Economics from Vanderbilt University. David Parry-Jones is a board director and former Chief Revenue Officer (CRO) with more than 30 years of experience scaling technology businesses internationally. Most recently, he served as CRO at DeepL and previously held senior leadership roles at Microsoft, VMware, and Twilio. He has led large teams across diverse markets and cultures, working with both early-stage companies and global enterprises.  As a board member and adviser, he brings a practical perspective on growth, international expansion, and building strong teams that turn strategy into results. Please join me in welcoming Jambu and David to the Mozilla Corporation Board of Directors. The post Jambu Palaniappan and David Parry-Jones join the Mozilla Corporation Board of Directors appeared first on The Mozilla Blog.

- The Mozilla Blog: 100 days later: Microsoft still steers Windows and Copilot users to Edge, everywhere the law lets it (2026/10/07 10:00)
  One hundred days ago, Mozilla released Over The Edge 2.0, the second independent report from leading experts on deceptive design Dr. Harry Brignull and Cennydd Bowles, on how Windows, Edge, Bing, and Copilot are designed to steer people away from the browser they chose.  We published the report, and the researchers published the evidence for anyone to analyze.  Browsers are a powerful tool. So it's no surprise that the history of the web has many examples of powerful platforms trying to deny people browser choice. Now, as AI reshapes our online experiences, the stakes couldn't be higher. Browsers are a key distribution layer for AI tools and services, while still determining many of the privacy and security functionality people rely on every day. Steering people toward one browser can therefore shape which AI services reach users and the protections that come with it.  We also shared the report with Microsoft and talked to them about the issues raised in the report. We knew that an immediate change was unlikely, but we hoped that, when faced with independent evidence of harmful design practices impacting Windows users, they would also want better for people. Instead, it became clear that Microsoft will respect user choice only when it is forced to. It is important to think about what that means for a company whose operating system runs on more than a billion machines. Microsoft's standard for respecting people's choices is legal compliance. Not what's best for its users. Not its own published principles. Not what's technically feasible. Not even what it already does for users in Europe. That's the most shocking part. Not that Microsoft engages in such tactics – we've seen that before. It's the fact that a better version of Windows already exists, and it is already available today – but only if you live in the European Economic Area (EEA), where the Digital Markets Act (DMA) forced the issue. If you live in the United States, India, the United Kingdom, or most other locations around the world, your experience and your freedom to choose for yourself are worse. One hundred days later, here is what people outside of the EEA still get:  The Windows 10 message "You're almost done setting up your PC" uses tricky wording to pull people off the default they picked.  Windows Search and Widgets open links in Edge instead of the browser set as default.  Windows keeps Edge as the default for common browser file types like PDFs and SVG images, even after another browser has been set as default. And people in the EEA also

deserve better: Edge remains pre-pinned to the taskbar; this seems to directly violate the “principled approach to app pinning and app defaults.” Copilot, Microsoft’s AI tool opens web links in a side-panel instead of the browser people chose, in every region tested, Europe included. Windows Backup, the tool Microsoft uses to guide people through the migration to Windows 11, silently tries to set Edge as the default when it restores a PC and fails to properly bring the user’s own browser along.  Most of the 2024 patterns are still there. The 2026 patterns are worse. Microsoft is using new surfaces like Copilot and the migration to Windows 11 to further undermine user choice. Our ask has always been simple: First, give all Windows users the fairer designs that are live in the EEA – regardless of where they are located. Second, stop the remaining harmful design practices that persist in the EEA. That means: Stopping the remaining harmful design tactics in the EEA and ship that standard worldwide. Honoring people’s browser choice everywhere it is currently ignored, including Windows Search, Widgets, Copilot, and Windows Backup restoration.  Applying Microsoft’s own 2023 app pinning principles to Edge itself.  The Mozilla-commissioned report has never been about intent. It’s about accountability and Microsoft’s responsibility to stop undermining people’s choices. The practices we documented need to change. Statements about intent are not a substitute for action. In the past 100 days, we’ve stopped waiting for Microsoft to act. Instead, we’ve stepped up public pressure on the company to respect people’s choices. We are also taking the report’s findings to regulators and policymakers. In Europe, the DMA is the only thing that has moved Microsoft. Everywhere else, the lesson is the same: Microsoft ships only when a regulator requires fairer design.  We hope that regulators in the US, the UK, India, and Brazil are paying attention.  Microsoft can do better. It already has, for people in the EEA. Until it does the same in all jurisdictions, we will keep pointing out the company’s choices and the impact they have on hundreds of millions of users who want something different. The post 100 days later: Microsoft still steers Windows and Copilot users to Edge, everywhere the law lets it appeared first on The Mozilla Blog.

- This Week In Rust: This Week in Rust 672 (2026/10/07 04:00)
  Hello and welcome to another issue of This Week in Rust! Rust is a programming language empowering everyone to build reliable and efficient software. This is a weekly summary of its progress and community. Want something mentioned? Tag us at @thisweekinrust.bsky.social on Bluesky or @ThisWeekinRust on mastodon.social, or send us a pull request. Want to get involved? We love contributions. This Week in Rust is openly developed on GitHub and archives can be viewed at this-week-in-rust.org. If you find any errors in this week's issue, please submit a PR. Want TWIR in your inbox? Subscribe here. Updates from Rust Community Official Announcing Rust 1.99.0 Generic Const Args and You [video] Rust Release Changelog - 1.99.0 Foundation A Fond Farewell To Three Rust Foundation Colleagues Newsletters This Month in Rust OSDev: September 2026 Rust Trends Issue 83 - NVIDIA Brings Rust to the GPU Kernel Rust Trends Issue 84 - Google Puts Agents on the Rust Rewrite Project/Tooling Updates Generate PDFs from Rust with HTML and CSS Catharsis for Noisy Audio: A Pure-Rust Restoration Toolkit with No ffmpeg and No Black Boxes Release mold 3.0.0 · rui314/mold Observations/Thoughts Rust for CPython (Python Language Summit 2026) Lies, damned lies, and Rust in the TechEmpower Web Framework Benchmarks Beyond the & The Performance Cost of RwLock in Our Read-Heavy Workload Shipping JPEG XL in Chrome Proving Rust Web Application Correctness with Lean 4 Hardware-Aware Programming in Rust Rust Walkthroughs The Missing Piece in Rust Error Handling Compiling the kernel with gccrs Declarative Macros in Rust: A Simple and Practical Introduction A dynamic drone fail-safe system that adapts as the situation changes. Build a Burglar Alarm with ESP32-C5 That Sends Telegram Alerts Crate of the Week This week's crate is karatepe, a statically typed localisation language and library. Thanks to miro for the self-suggestion! Please submit your suggestions and votes for next week! Calls for Testing An important step for RFC implementation is for people to experiment with the implementation and give feedback, especially before stabilization. If you are a feature implementer and would like your RFC to appear in this list,

add a call-for-testing label to your RFC along with a comment providing testing instructions and/or guidance on which aspect(s) of the feature need testing. No calls for testing were issued this week by Rust, Cargo, Rustup or Rust language RFCs. Let us know if you would like your feature to be tracked as a part of this list. Call for Participation; projects and speakers CFP - Projects Always wanted to contribute to open-source projects but did not know where to start? Every week we highlight some tasks from the Rust community for you to pick and get started! Some of these tasks may also have mentors available, visit the task page for more information. issuerd - Add a French (fr) message bundle for login pages and emails issuerd - Add proptest suites for issuerd-protocol parsers issuerd - Add an additional client installation provider (adapter config download format) ruxen - Expand globs in include ruxen - Use nginx's status reason phrases everywhere ruxen - Implement proxy_method If you are a Rust project owner and are looking for contributors, please submit tasks here or through a PR to TWiR or by reaching out on Bluesky or Mastodon! CFP - Events Are you a new or experienced speaker looking for a place to share something cool? This section highlights events that are being planned and are accepting submissions to join their event as a speaker. RustWeek 2027 | CFP closes 2027-01-10 | Utrecht, The Netherlands | Event date: 2027-05-24 TokioConf 2027 | CFP closes 2026-11-30 | Portland, Oregon, USA | 2027-04-26 - 2027-04-27 If you are an event organizer hoping to expand the reach of your event, please submit a link to the website through a PR to TWiR or by reaching out on Bluesky or Mastodon! Updates from the Rust Project 653 pull requests were merged in the last week Compiler add a single-entry parent SpanData cache add fast path to generalization optimize Cranelift with PGO Library add mul_add_relaxed methods for floating-point types add std::fs::{Home|Media}Dirs expose Rc::is_unique stabilize CStr::display stabilize debug_closure_helpers Cargo add new peak memory table to cargo timings enabled via -Zmem-stats config: Proper dotted tuple support with legacy fallback git: default to net.git-fetch-with-cli if git is present improved testsuite file permissions cleanup lint: Making the lint name a terminal hyperlink to docs trim-paths: stabilize profile.trim-paths use trusted publishing for Cargo crates Rustdoc Correctly handle rustc_allow_incoherent_impl on primitive methods Correctly link to (imported) enum variants with "jump to def" Fix how Deref items are handled Rustfmt items: format comments after where using clause_shape budget use saturating arithmetics for adjust_max_width Clippy manual_range_patterns: support char and byte literal let_unit_value bail out if initializer is cfg-dependent extend needless_borrowed_reference to lint mutable ref patterns fix exponential-time performance bug in has_non_owning_mutable_access_inner improve items_after_test_module: don't let derive expansions hide trailing items new lint: unnecessary_as_slice optimize msrv calls (again) Rust-Analyzer complete 'let' 'letm' in arm expr and closure expr complete turbofish when fn can't infer param do not suggest arg-list in expected callable arg fix unicode-ident, take 2 add missing HIR database when running unresolved-references complete let in macro when expand at macro stmts don't panic on malformed let-pattern with mismatched or-arm arities generate variant for self in impl improve in-block heuristic check in nested ambiguous name-match ignore leading tailing underscore transform usage path when extract trait to module fixed Implement opaques_with_sub_unified_hidden_type for the next-sol… Rust Compiler Performance Triage A relatively quiet week, but a very positive one nonetheless. Highlights are a 3.1% improvement in rustdoc speed from not using the metadata based crate_hash for rustdoc runs, a 0.5% improvement from a new fast path in the trait solver, and a 0.4% improvement from a cache for the parents of SpanData. Triage done by @JonathanBrouwer. Revision range: c1070d69..cc9a14f7 Summary: (instructions:u) mean range count Regressions □ (primary) 0.6% [0.4%, 1.0%] 12 Regressions □ (secondary) 0.4% [0.1%, 0.9%] 26 Improvements □ (primary) -1.1% [-6.2%, -0.2%] 212 Improvements □ (secondary) -1.9% [-15.7%, -0.1%] 208 All □□ (primary) -1.0% [-6.2%, 1.0%] 224 4 Regressions, 3 Improvements, 1 Mixed; 5 of them in rollups 33 artifact comparisons made in total Full report here Approved RFCs Changes to Rust follow the Rust RFC (request for comments) process. These are the RFCs that were approved for implementation this week: Change default branch to main f16b type Final Comment Period Every week, the team

announces the 'final comment period' for RFCs and key PRs which are reaching a decision. Express your opinions now. Tracking Issues & PRs Rust Make std::fs::{File, ReadDir, DirEntry} always needs_drop even when unsupported. [rustdoc] Add tabs to settings popover rustc: Stabilize the WebAssembly wide-arithmetic feature Error on non-literal expressions in doc attributes on macro calls stabilize ptr_cast_slice Add extra types to VaArgSafe Reject cfg on expressions that cannot be safely removed fn_addr_eq: we actually can guarantee basically nothing Stop using dlltool for generating import libraries on MinGW Stabilize ptr::try_cast_aligned [disposition: close] 1.99 beta crater regression: overflow evaluating the requirement Compiler Team (MCPs only) Move hir::Params from Body of functions to FnDecl. MCP: Add -Zasync-panic for binary size Upstreaming BorrowSanitizer Experimentally in Nightly Rust Language Reference Guarantee that the never type ! is zero-sized and 1-aligned. No Items entered Final Comment Period this week for Rust RFCs, Cargo, Language Team, Leadership Council or Unsafe Code Guidelines. Let us know if you would like your PRs, Tracking Issues or RFCs to be tracked as a part of this list. New and Updated RFCs RFC: Add required-targets for workspace package selection RFC: unsafe global_asm Cromulent Copy closure captures RFC for limited crates.io self-service version deletion This RFC will appear in the Call for Testing section of the next issue (#) of This Week in Rust (TWiR). You may remove the call-for-testing label. Please feel free to leave the call-for-testing label in place if you would like this RFC to appear again in another issue of TWiR. Upcoming Events Rusty Events between 2026-10-07 - 2026-11-04 □ Virtual 2026-10-07 | Virtual (Indianapolis, IN, US) | Indy Rust Indy.rs - with Social Distancing 2026-10-08 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-10-08 | Virtual (Nürnberg, DE) | Rust Nuremberg Rust Nürnberg online 2026-10-10 | Virtual (Gdansk, PL) | Stacja IT Trójmiasto [BEZPŁATNIE] Programowanie w języku Rust 2026-10-10 | Hybrid (Kuala Lumpur, Malaysia) | Rust Malaysia Meetup Rust Meetup October 2026 2026-10-11 | Virtual (Bengaluru, India) | Embedded Rust Discord Silicon Sundays 4 2026-10-13 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Second Tuesday 2026-10-14 - 2026-10-17 | Hybrid (Barcelona, ES) | EuroRust EuroRust 2026 2026-10-18 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: Third Sunday 2026-10-20 | Virtual (Washington, DC, US) | Rust DC Mid-month Rustful 2026-10-21 | Hybrid (Vancouver, CA) | Vancouver Rust Disposable Agent Sandboxes in Rust 2026-10-22 | Virtual (Berlin, DE) | Rust Berlin Rust Hack and Learn 2026-10-22 | Virtual | Rust □ Maven Rust and the GPU from Native to Web: An Introduction to wgpu 2026-10-26 | Virtual | Rust □ Maven No Python was harmed: teaching a tiny MCU to learn as it goes 2026-10-27 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Fourth Tuesday Rust Bookclub 2026-10-27 | Virtual (London, UK) | Women in Rust Lunch & Learn: Reasoning with Async Rust 2026-10-29 | Virtual | Rust □ Maven Creating a hexadecimal editor in Rust 2026-11-01 | Virtual (Dallas, TX, US) | Dallas Rust User Meetup Rust Deep Learning: First Sunday 2026-11-03 | Virtual (London, UK) | Women in Rust □ Community Catch Up 2026-11-04 | Virtual (Indianapolis, IN, US) | Indy Rust Indy.rs - with Social Distancing Asia 2026-10-09 | Hybrid (Kuala Lumpur, MY) | Rust Malaysia Meetup Rust Meetup August 2026 2026-11-03 | Tel Aviv-yafo, IL | Rust □ TLV In person Rust November 2026 at AWS in Tel Aviv Europe 2026-10-08 | Oslo, NO | Rust Oslo Rust Hack'n'Learn at Kampen Bistro 2026-10-08 | Geneva, CH | Rust Geneva Rust Meetup Geneva 2026-10-14 | Barcelona, ES | BcnRust 22nd bcnrust session 2026-10-14 - 2026-10-17 | Hybrid (Barcelona, ES) | EuroRust EuroRust 2026 2026-10-20 | Leipzig, SN, DE | Rust - Modern Systems Programming in Leipzig Creating a realtime web Multiuser Dungeon game with Dioxus 2026-10-22 | Karlsruhe, DE | Rust Hack & Learn Karlsruhe Karlsruhe Rust Hack and Learn Meetup bei BlueYonder 2026-10-22 | Toulouse, FR | Rust Toulouse Rust Toulouse Meetup - Rust & Python interoperability 2026-10-27 | Aarhus, DK | Rust Aarhus Hack Night: Rust meets AI 2026-10-31 | Stockholm, SE | Stockholm Rust Ferris' Fika Forum #31 2026-11-01 - 2026-11-03 | Italy, IN | RustLab RustLab - The International Conference on Rust in Italy North America 2026-10-08 | Lehi, UT, US | Utah Rust Lightning Talks N' Chill 2026-10-08 | New York, NY, US | Rust NYC Rust NYC: Zero Knowledge Proofs & GPUs in Rendering 2026-10-08 | San Diego, CA, US | San Diego Rust San Diego Rust October Meetup - Back in person! 2026-10-10 | Boston, MA, US | Boston Rust Meetup Back

Bay Rust Lunch, Oct 10 2026-10-14 | Los Angeles, CA, US | Rust Los Angeles Rust LA October: AI & Rust w/ Oxen.AI & Origin Lab! 2026-10-20 | San Francisco, CA, US | San Francisco Rust Study Group Rust Hacking in Person 2026-10-21 | Hybrid (Vancouver, CA) | Vancouver Rust Disposable Agent Sandboxes in Rust 2026-10-21 | San Francisco, CA, US | Bay Area Rust Bay Area Rust - Embedded Meetup 2026-10-28 | Austin, TX, US | Rust ATX Rust Lunch - Fareground South America 2026-10-08 | Buenos Aires, AR | Rust en Español WebApp Ergonomics y Secretos Distribuidos. If you are running a Rust event please add it to the calendar to get it mentioned here. Please remember to add a link to the event too. Email the Rust Community Team for access. Jobs Please see the latest Who's Hiring thread on r/rust Quote of the Week There ain't no rules here in Quote of the Week - it's survival of the wittest – Simon Buchan on rust-users Thanks to Jonas Fassbender for the suggestion! Please submit quotes and vote for next week! This Week in Rust is edited by: nellshamrell llogiq ericseppanen extrawurst U007D mariannegoldin bdillo opeolluwa bnchi KannanPalani57 tzilist Email list hosting is sponsored by The Rust Foundation Discuss on r/rust

- Firefox Tooling Announcements: Happy BMO Push Day! (20261006.1) (2026/10/06 21:43)
  Github Link The following changes have been pushed to bugzilla.mozilla.org: Bug 2075358 - Native Mojo REST endpoints serve anonymous requests when requirelogin is enabled Bug 2074690 - BMO REST API cannot set any DATE-type custom field Bug 2059957 - Reject Mojolicious-truncated request bodies before native routes dispatch Bug 1273193 - Get Comments webservice docs don’t mention author Bug 2078352 - Selenium test 4_test_votes.t fails with Firefox ESR 153 because go_to_bug doesn’t wait for the module expand animation Discuss these changes in the BMO Matrix Room 1 post - 1 participant Read full topic
- The Mozilla Blog: Strengthen your online security for free with Firefox (2026/10/06 16:00)
  Every October, Cybersecurity Awareness Month reminds us the importance of protecting our digital lives.The theme for 2026 is ‘Don’t Make It Easy for Them,’ with the National Cybersecurity Alliance (NCA) and the Cybersecurity and Infrastructure Security Agency (CISA) providing simple, actionable steps users can take to make it harder for cybercriminals to attack.Everyone wants to be safe online, but one of the most common excuses we hear is “I don’t have the time or the money to invest in it right now.” Well, we have good news: you can improve your online security easily, for the low price of $0.Firefox provides several free features designed to make it easier to protect your online life and harder for cybercriminals to target you.  Password manager: Use a unique password for every account Reused passwords are one of the easiest ways attackers break into multiple accounts. If one site is breached, your login details could be tested on many others.Firefox’s free password manager can generate and securely store a strong, unique password per site, which is the single highest-leverage fix against password stuffing. For convenience, Firefox can automatically fill in your saved login information the next time you access a previously visited site. However, you still have the option to turn off password saving for specific websites, or completely if you’re using a shared device or simply prefer to enter it each time.You can manage your stored logins on a singular device at any time. To get a head start, you also have the option to import passwords saved on several other popular browsers. In addition, you can create a free Mozilla account to safely sync them across your signed-in devices. VPN: Privacy built into the browser Imagine you’re settling into a corner table at a bustling airport cafe, checking some emails on your phone before you board your flight. Or perhaps you took your laptop to the campus library to finally start that research paper you’ve been putting off. Maybe you’re polishing off that last presentation slide between sessions at an out-of-state conference.While these moments are routine, connecting to public Wi-Fi anywhere can open a small window of risk – as your IP address, approximate location, and browsing activity can be more exposed on networks you don’t control.That’s where Firefox’s free built-in, browser-based VPN steps in. While Firefox already encrypts your traffic with HTTPS, the built-in VPN feature routes your browsing traffic through a proxy network, hiding your IP address before it reaches a

website. This means the sites you visit will see a proxy IP address instead of your own. It offers an industry-leading 50 gigabytes of free VPN-browsing each month, with no extension or extra app to install. Simply sign in, turn it on, and start browsing.Firefox's free built-in VPN first launched on desktop in March, later expanding to Android in July. Unlike a standalone or full-service VPN, it's designed specifically to protect your browsing traffic within Firefox, not your entire device. This means apps, downloads, and other system traffic are not routed through the browser-based VPN. For people who need broader, device-wide VPN protection, Mozilla also offers a separate paid VPN service.You can check the full list of regions where built-in VPN in Firefox is currently available here, and learn more about the differences between Firefox's free built-in VPN and Mozilla VPN here. Relay: Contain the damage of any one breach Think about how often you give out your personal email address – for an online purchase, RSVPing to an event, or signing up to a new app. Each time, you trust the site owner will protect your information, but, spam piles up and breaches can still happen – even when you think you're being careful.Firefox Relay gives you a smarter way to share. Instead of handing out your real email address, this feature provides secure, random email masks to use any time a website, store, or app asks you for your information. It forwards messages to your real email, while keeping your identity hidden to senders. This means if one service gets compromised, your other accounts stay safe.You can also use Relay in the U.S. and Canada to shield your phone number, so you can receive calls and texts through masks and reply without exposing your real number.The best part? To make protecting your privacy even easier, Firefox increased its limit to 50 free email masks earlier this year, allowing you to shield dozens of logins at no cost. Browse freely: Firefox protects you by default Even if you sometimes forget to use Firefox protection tools like password manager, VPN, and Relay, there are multiple always-on features operating within the browser to help keep your session secure and private. These include enhanced tracking protection to block hidden scripts and known trackers that follow you across the web, browser fingerprinting protection to prevent sites from building a unique profile of your device, and warnings about deceptive or malicious websites.Together, Firefox's individual and always-on security features work to help keep your browsing safe and your data private. So, this Cybersecurity Awareness Month, take a moment to build new habits, try out our free cybersecurity tools, and make life more difficult for cybercriminals. Small steps add up, and we are here to help. Stay safe, stay smart, and keep exploring the web with confidence on Firefox.  The post Strengthen your online security for free with Firefox appeared first on The Mozilla Blog.

- Ryan Hunt: Wasm Stack Switching in SpiderMonkey (2026/10/06 15:00)
  WebAssembly stack switching in SpiderMonkey is the most complex feature I've ever implemented. It's one of those projects where you get lost and start rethinking the basics, like "what is the stack even really?" It was fun to give a talk on it. Check out the recording below, or on YouTube.
- Mozilla Data YouTube Channel: Towards a Telemetry Taxonomy (2026/10/05 18:47)
  Leif Oines talks about an effort to define a more complete taxonomy for Mozilla's data.
- Mozilla Data YouTube Channel: Data Club Talk: Jan-Erik Rediger - The Glean UniFFI migration and how no one noticed (2026/10/05 15:53)
  Given at the Mozilla Data Club on August 12th, 2022.
- Firefox Developer Experience: Firefox WebDriver Newsletter 157 (2026/10/05 14:21)
  WebDriver is a remote control interface that enables introspection and control of user agents. As such, it can help developers to verify that their websites are working and performing well with all major browsers. The protocol is standardized by the W3C and consists of two separate specifications: WebDriver classic (HTTP) and the new WebDriver BiDi (Bi-Directional). This newsletter gives an overview of the work we've done as part of the Firefox 157 release cycle. Contributions Firefox is an open source project, and we are always happy to receive external code contributions to our WebDriver implementation. We want to give special thanks to everyone who filed issues, bugs, and submitted patches. In

Firefox 157, multiple WebDriver bugs were fixed by contributors: Sameem merged two internal helpers (remote/shared/Sync.sys.mjs and remote/marionette/sync.sys.mjs), which removed some code duplication. Khalid AlHaddad unified the navigation commands in Marionette and WebDriver BiDi to always initiate navigations via browsingContext.loadURI(). Khalid AlHaddad updated the WebDriver BiDi client used by WebDriver tests so that browsing_context.locate_nodes returns the nodes array directly instead of the enclosing result object, matching the updated command response. WebDriver code is written in JavaScript, Python, and Rust, so any web developer can contribute! Read how to set up the work environment and check the list of mentored issues for Marionette or the list of mentored JavaScript bugs for WebDriver BiDi. Join our chatroom if you need any help to get started! All Changes A complete list of developer-facing changes included in this Firefox release is available in the MDN Firefox 157 Release Notes.

- Mozilla Data YouTube Channel: Outreachy Mentorship: A Retrospective (2026/10/05 10:59)
  Will Lachance does a retrospective on the Glean Dictionary outreachy internship. See also "Linh's Outreachy Internship Highlights" https://www.youtube.com/watch?v=UJdlkHDPgGQ To learn more about Outreachy, see https://www.outreachy.org/
- Mozilla Data YouTube Channel: Responsible Data Collection is Good, Actually (Ubisoft Data Summit 2021) (2026/10/04 23:55)
  Firefox Telemetry Engineer and Data Steward Chris H-C (:chutten) gives a talk at Ubisoft's Data Summit 2021 about how Responsible Data Collection as practised at Mozilla makes cataloguing easy, stops instrumentation mistakes before they ship, and allows you to build self-serve analysis tooling that gets everyone invested in data quality. Oh, and it's cheaper, too.
- Mozilla Data YouTube Channel: Data Club: Jan-Erik Rediger - Little Bobby Tables - from metrics.yaml to data-filled columns (2026/10/04 19:17)
  A short story about Little Bobby Tables and how we know what data to fill in where.
- Mozilla Data YouTube Channel: Data Club Lightning Talk: Jan-Erik Rediger - Your personal Glean data pipeline (2026/10/04 16:52)
  This talk was given as part of the Data Club Lightning Talk Session on February 11th, 2022. More on https://blog.mozilla.org/data/2022/02/25/this-week-in-glean-your-personal-glean-data-pipeline Information about Glean: https://mozilla.github.io/glean/book/index.html
- Mozilla Data YouTube Channel: GLAM Datasets (2026/10/04 07:56)
  Marina Samuel and Anthony Miyaguchi talk about the ETL pipeline created for the GLAM project (https://github.com/mozilla/glam).
- Mozilla Data YouTube Channel: Data Club: Jeff Silverman - Data Science & Astronomy: AAS 243 & ATDS 6 (2026/10/04 05:05)
- Mozilla Data YouTube Channel: Introducing Glean Annotations (2026/10/03 19:51)
  Leif Oines and Will Lachance introduce Glean Annotations: a process and technology for curating and communicating knowledge about the data we collect in Mozilla's products.
- Mozilla Data YouTube Channel: Mozillians sharing the 2021 SciPy Conference experience (2026/10/02 04:24)
  The 2021 SciPy conference (https://www.scipy2021.scipy.org/) involved the showcasing of the latest open source Python projects for advancement in scientific computing. Mozilla was a diversity sponsor and a few Mozillians attended and shared their experience of the event.
- Mozilla Data YouTube Channel: Last Lecture: Writing the Data Docs (2026/10/02 03:38)
  Will Lachance gives a last lecture on writing data documentation at Mozilla.
- The Rust Programming Language Blog: Demoting i686 Windows targets to std-only (2026/10/02 00:00)
  With Rust 1.100.0, the following changes to 32-bit Windows targets will happen: i686-pc-windows-msvc Tier 1 with host tools target will be

demoted to Tier 1 without host tools. i686-pc-windows-gnu Tier 2 with host tools target will be demoted to Tier 2 without host tools. Builds of the standard library will continue to be distributed, but host tools such as the compiler will be no longer available. i686-pc-windows-msvc as a Tier 1 target still undergoes CI testing. To build 32-bit Windows binaries, cross-compiling from a still-supported host toolchain (such as a 64-bit Windows ones) will be required from now on. Background Desktop and Server 32-bit only x86 CPUs are no longer sold for over 15 years, and general 32-bit Windows support has ended in October 2025. This means that the development platforms these targets are meant for hardly exist these days, and even if they do exist they typically aren't capable enough for development. Even on the modern x86_64 hardware, building i686 Windows toolchains has proven to be problematic. We have encountered compiler binaries crashing when built with the i686 MSVC target, and the GNU C++ toolchain failing with OOMs during LLVM build. Considering all these things, cross-compiling these targets from a better supported one is what we have found to be the best solution forward. As part of that, we stopped producing host tools for these targets. For the time being, the prebuilt standard library is still available, and in case of i686-pc-windows-msvc still tested on CI. What Changes? After Rust 1.100, it will no longer be possible to install toolchains on 32-bit Windows hosts. We recommend cross-compiling from a still-supported host (such as a 64-bit Windows toolchain) instead. Other 32-bit platforms are not impacted by this change. For more details about these demotions, see RFC 3999 for i686-pc-windows-msvc demotion, and MCP 1020 for i686-pc-windows-gnu demotion.

- Mozilla Data YouTube Channel: Monitoring Sensitive Data: How do we monitor data we don't store? (2026/10/01 23:23)
  We try to be responsible with data. For example, we: - store as little sensitive data as possible - monitor changes in incoming data on which we've built models But what happens when those two approaches conflict? How do we monitor changes in incoming data that we don't want to store? This talk explains the schema we use to monitor changes in what people are searching for in Firefox...even when we deliberately don't store some of what people are searching for.
- Thunderbird Blog: Thunderbird for iOS: A First Look at the Native iPhone Email App in Development (2026/10/01 22:14)
  The team has been working hard behind the scenes laying a strong foundation for the Thunderbird iOS app. Rather than leveraging existing code, we are building natively from scratch. This fresh start allows us to take everything we've learned from Thunderbird Desktop and Android to ensure a fast, reliable, and modern experience on your iPhone right from the start. Our ultimate goal is seamless cross-device experience. Whether you're sitting at your desktop computer, checking messages on the go, your workflow should feel effortless and unified.  Right now, we are focused heavily on getting the core foundations right. We are excited about the momentum because having clean architecture today means we can layer on exciting new features much faster in the future. Below we've captured some of our work in progress. Nothing is final, but we wanted to share what we are cooking up so far. If you have feedback, keep reading on. At the bottom of the post we have  a link to a user research study where we want to hear from you and what you want to see from Thunderbird for iOS. Hassle-free setup Adding account user flow with options for email services like Google and Microsoft We know people are really busy, often doing more than one thing at a time. We're making login fast, secure, and fully automatic, so whether you're signing in with a single click or setting up multiple accounts, you can get to your inbox quickly.  Solid reliable connections Thunderbird inbox An email app is only as good as its connection. Behind the scenes, we're building a background sync that maintains a steady connection with your provider that keeps your inbox up to date while respecting your battery life. Core daily tools Formatting toolbar (highlighted to show the details) Your daily email routine should feel effortless. We're polishing the essentials: a clutter-free, easy-to-read inbox layout paired with a clean, modern editor that lets you format messages smoothly without getting in your way. Designed for growth Thunderbird has always belonged to its community. We're structuring the app with clean code and an experimental feature

flag system so it’s easy for community developers to contribute. This means power users will be able to test out new experimental features early, and external contributors can easily help us extend what the app can do. When can you try it? Our goal is to have a first version ready for community testing in the next few months. Building with our community, not just for people, is core to what we are trying to do. To help us decide what to build next after our first release, we’d love your input. If you had to pick the top three things that matter most to your daily email routine, what would they be? Survey linkUpdate: the survey has now closed. Thanks to all that responded! A note to the community  Lastly, we know how long many of you have been asking for and waiting for Thunderbird on iOS. Whether you’ve dropped a comment on our forums, tagged us on social media, or simply kept an eye on our progress, thank you for your patience! Bringing Thunderbird to iPhone and iPad isn’t just about building an app; it’s about doing it right so it serves you reliably for years to come. Today, we want to pop open the hood, share where we are, and work with everyone to design the best possible experience. The post Thunderbird for iOS: A First Look at the Native iPhone Email App in Development appeared first on The Thunderbird Blog.

- Mozilla Data YouTube Channel: Glean Dictionary Looker Demo (2026/10/01 17:48)
  A quick demonstration of the Glean Dictionary's new integration with Mozilla's instance of Looker.
- The Mozilla Blog: Under the Hood: Prompt tuning Shake to Summarize for recipes (2026/10/01 16:00)
  Following the rollout of Firefox’s Shake to Summarize feature to Android devices this May, we wanted to take a closer look at the modeling work behind the feature. In a previous blog post, we discussed our model selection process. Now that Shake to Summarize has been available on both iOS and Android for a few months, we wanted to take it a step further and outline our approach to prompt development. This is the story of a Shake to Summarize use case that required a little extra prompt tuning: online recipes. Framing the Problem The first step in developing a useful prompt is clearly describing what you want the LLM to do. LLMs thrive on specificity, so the more sharply you can define your task, the better your results are likely to be.  This is especially important when using smaller LLMs (like we are here), since these little models are not as good at reading between the lines and intuiting unstated intentions as their more powerful cousins are.  For this application, we were looking to create summaries. On the face of it, this seems pretty straightforward. However, as we iterated on prompts, we quickly discovered that what constitutes a “good” summary depends largely on what one is summarizing.  For example, a useful summary of a novel should provide us with a quick overview of the plot without getting into too many specific details; we wouldn’t expect the summary to contain anything from the text verbatim. In contrast, a summary of a recipe website should include the recipe essentially as written. If the recipe says “cook lovingly” we might be OK with shortening it to “cook,” but if it calls for 4 cups of vegetable broth, a Tbsp of oregano and a tsp of thyme, we want these details relayed to us exactly. Merely stating, “this recipe calls for some broth and some spices” would not be adequate.  Forming the Prompt From here, it became clear that we wanted not just one “summarize” instruction, but a whole set of instructions — one for each category of webpage that we wished to summarize. We worked with the product team to compile a list of article types that we were targeting for this feature. For each article type, we then gave a brief description of what a good summary would look like:  Recipe – Ingredients as written, along with the key steps, time required, and any tips given by the author or commenters News – Only the important details: what happened, when, and what would be the likely consequences to the reader How-to – Start with the required materials, skills, tools, etc.,  along with the main steps and any specific warnings called out by the author Review – Highlight the bottom line rating. If it is a product review, include the pros and cons, the price, and who the target audience is. Research – Key finding and level of confidence the researchers have in their results as well as the projected real-world impact. Opinion – The main argument, along with any key evidence cited in support We then wrapped this list in some general instructions to ground the

model in the task and came up with our initial prompt (formatted for readability) [source]: You are a Content Summarizer. You create mobile-optimized summaries by first understanding what users actually need from each type of content. Process: Step 1: Identify and Adapt. Step 2: Based on content type, prioritize: Recipe – Servings, Total time, Ingredients list, Key steps, Tips. News – What happened, when, where. How-to – Total time, Materials, Key steps, Warnings. Review – Bottom line rating, price. Opinion – Main arguments, Key evidence. Personal Blog – Author, main points. Fiction – Author, summary of plot. All other content types – Provide a brief summary of no more than 6 sentences. Step 3: Format for mobile using concise language and paragraphs with 3 sentences maximum. Bold critical details (numbers, warnings, key terms). We Foxfooded the above prompt and found that it mostly worked pretty well. The summaries the model produced were concise, accurate, and informative. The only exception was recipes. We found that, when the prompt above was used for recipes, the model tended to summarize too much — often leaving out key ingredients, or in some cases, leaving out the recipe entirely. For example, when asked to summarize a lentil soup recipe the model returned: The webpage provides a detailed recipe for a vegan lentil soup, which the author claims is the best they've ever made. The soup is made with pantry staples like lentils, diced tomatoes, vegetable broth, and spices (cumin, curry powder, thyme), along with fresh vegetables (onion, carrots, garlic) and greens (kale or collard greens). A key step is blending a portion of the soup to create a creamy texture without dairy. The summary is accurate, so far as it goes, but it isn't exactly useful. If our goal is to have the summary be a quick way to access the recipe itself, without having to read through the narrative preamble, this wouldn't cut it. Focusing the Prompt To address this problem, we needed to be more explicit with the instructions we gave the model when summarizing recipes: we needed to tell it which information we were interested in and which it could safely discard. Although we could have added this extra guidance directly to the prompt above, doing so risked causing the model to over-index on the recipe formatting and ignore the other article types. Instead, we developed a separate prompt which contained only the recipe instructions and directed recipe summarization requests to use it, rather than the general one [source]. To perform this routing, we used the structured data embedded in each webpage. While this categorization could have been handled by another model pass, metadata from the webpage itself allowed us to quickly and deterministically identify the page categories without additional inference overhead. To capture cases where this data is either missing or inaccurate, we also maintain some lightweight recipe guidance in the general prompt. After this change, the summary returned by the model for the above recipe became much more usable (shortened for brevity): Servings: 6 Total Time: 50 minutes Prep Time: 10 minutes Cook Time: 40 minutes Ingredients ¼ cup extra virgin olive oil 1 medium yellow or white onion, chopped 2 carrots, peeled and chopped … Instructions Warm the olive oil in a large Dutch oven or pot over medium heat. Add the chopped onion and carrot, cooking until the onion softens and turns translucent, about 5 minutes. … Tips Use an immersion blender for easier blending without transferring soup. … Nutrition Calories: 320 … With this change in place, we ran a quick test over a curated set of recipe sites and found that the new system was more than twice as likely to return a complete and accurate summary than our previous one. Success! The system was now working as expected: summaries were useful and recipes were complete. Reflections From this experience we learned that the model produced the best results when it was told explicitly what we wanted it to do. When the instructions were vague or left too much up to the model, performance suffered. To this end, we found that framing this problem as a routing problem — where the specific kinds of articles are routed to specific prompts — worked well. Since the task of summarization is not monolithic, our summarization pipeline should not be either. Even though our current approach has only a single category-specific prompt, we hypothesize that the system would see further gains by using dedicated prompts for other page types as well. More broadly, this experience reinforced an important lesson for us: improving AI systems is not solely about building larger or more capable models. Some of the biggest gains come from reducing ambiguity, narrowing the task, and

designing systems that help the model succeed. Within the right harness, smaller, open source models can deliver great value. While building more capable models continues to advance the field, our experience shows that thoughtful system design and solid engineering still matter. The post Under the Hood: Prompt tuning Shake to Summarize for recipes appeared first on The Mozilla Blog.

- Mozilla Data YouTube Channel: An opinionated intro to NLP (text analytics) (2026/10/01 13:06)
  Rebecca BurWei from Mozilla Data Science gives an introduction to Natural Language Processing.
- Mozilla Data YouTube Channel: Data Incident Process (2026/10/01 10:49)
  Mike Droettboom talks about Data @ Mozilla's process for handling incidents.
- The Rust Programming Language Blog: Announcing Rust 1.99.0 (2026/10/01 00:00)
  The Rust team is happy to announce a new version of Rust, 1.99.0. Rust is a programming language empowering everyone to build reliable and efficient software. If you have a previous version of Rust installed via rustup, you can get 1.99.0 with: $ rustup update stable If you don't have it already, you can get rustup from the appropriate page on our website, and check out the detailed release notes for 1.99.0. If you'd like to help us out by testing future releases, you might consider updating locally to use the beta channel (rustup default beta) or the nightly channel (rustup default nightly). Please report any bugs you might come across! What's in 1.99.0 stable extern "C" variadics Rust 1.99.0 stabilizes defining C-ABI variadic functions with "C" and "C-unwind" ABIs. Variadic functions defined this way use a variable argument list (...) and accept an arbitrary number of arguments. Rust could already call externally-defined variadic functions (e.g., libc::printf). With Rust 1.99, these functions can now be written in Rust itself: /// SAFETY: must be called with (at least) 2 i32 arguments. unsafe extern "C" fn sum(mut args: ...) -> i32 { // SAFETY: guaranteed by the caller. let a = unsafe { args.next_arg::<i32>() }; let b = unsafe { args.next_arg::<i32>() }; a + b } fn foo() -> i32 { unsafe { sum(0i32, 2i32) } } The type of ... is VaList, which is ABI-compatible with the C va_list type across targets. What types can be read from a VaList is guarded by the VaArgSafe trait. For more details on c-variadic functions, see the Reference. This release also stabilizes support for defining naked variadic functions with non-"C" ABIs, which must be written via inline assembly. Layout information from raw pointers This release settles the safety requirements for retrieving the size and alignment on raw pointers to both Sized (trivially safe, already possible on stable) and non-Sized types. This is done by stabilizing three functions: Layout::for_value_raw mem::size_of_val_raw mem::align_of_val_raw Recommend against round-trip unleaking after Box::leak While there are no changes to the language semantics in Rust 1.99, we have updated the documentation on Box::leak to recommend against patterns that later deallocate that memory. This was done because such code was found to have problematic interactions with current and future potential compiler optimizations, and is especially problematic with the upcoming stabilization of custom allocators. Instead, Box::into_non_null or Box::into_raw should be preferred. This guidance also applies to other leak functions in the standard library. Stabilized APIs IntoIterator for Box<[T; N]> IntoIterator for &Box<[T; N]> IntoIterator for &mut Box<[T; N]> VecDeque::retain_back core::ffi::VaList Box::into_non_null Box::from_non_null Vec::into_parts Vec::from_parts core::mem::size_of_val_raw core::mem::align_of_val_raw core::alloc::Layout::for_value_raw String::from_utf8_lossy_owned string::FromUtf8Error::into_utf8_lossy FusedIterator for StepBy<I> std::fs::set_times std::fs::set_times_nofollow Other changes Check out everything that changed in Rust, Cargo, and Clippy. Contributors to 1.99.0 Many people came together to create Rust 1.99.0. We couldn't have done it without all of you. Thanks!
- About:Community: A fresh Firefox and MozFest count down (2026/09/30 17:01)
  Hi Mozillian, In this edition, we’re exploring the new Firefox design, Mozilla’s recent partnership with Mistral to expand choice in AI-powered browsing, and several ways to get involved with MozFest. You can join a livestream about the value of bringing open-source communities

together in person, volunteer at MozFest in Barcelona, or take part in the latest privacy discussion from the Firefox community on Reddit. Read on for the latest updates and opportunities to participate! More modern. More flexible. Still Firefox. Firefox 157 introduces a new Firefox design with updated colors, icons and themes that make the browser feel more modern while preserving the familiar Firefox experience. The release also brings back Compact Mode and adds easier theme selection, new wallpapers and more ways to customize the New Tab page. Underneath the new look, Firefox remains independent and open source, with built-in privacy protections and controls that let you decide how your browser works, including how AI features appear. Learn more Mozilla and Mistral partner to expand AI choice Mozilla and Mistral have announced a partnership aimed at increasing competition and preserving user choice in AI-powered browsing. As part of the collaboration, the open-source Mistral Small 4 model is coming to Firefox Smart Window Beta as a new option for users in the US and Canada. Smart Window Beta is also expanding to France with official French-language support, with additional European markets planned. Users can continue choosing from multiple AI models, reinforcing Firefox's commitment to avoiding lock-in and keeping the web open to different technologies and providers. Learn more Does open source need IRL? The case for MozFest Is showing up in person important for open source? Join the Owners Not Renters livestream on Tuesday, September 29, to hear the story behind MozFest and explore why gathering in person still matters for open-source communities ahead of MozFest, taking place October 28–30 in Barcelona. Kali Villarosa (MoFo), Seher Shafiq (MoFo) and Brian Behlendorf (co-founded Apache) will join host Marcus Rein for an open conversation, followed by a live Q&A. Bring your questions and your hot takes! Watch and participate MozFest call for Volunteers From 28–30 October 2026, the Mozilla Foundation will host the Mozilla Festival in Barcelona, bringing together people from diverse fields to explore how we can work collectively to "re-wild" the web. To help bring the Festival to life, we're building a team of volunteers. By joining, you'll help deliver a major event in support of Mozilla's mission while also having opportunities to participate in sessions throughout the three-day Festival. Sign up to volunteer From the Reddit Community Privacy conversations are going mainstream, and Firefox has been at this for 20+ years. In light of a recent video making the rounds about the importance of privacy, the Firefox Team shared a rundown of built-in privacy features (ad blocking, tracking protection, encrypted sync, a free VPN, and more). Jump in at r/firefox and share your own privacy tips and configs! Join the discussion P.S. Enjoyed these updates? Subscribe to the Mozilla Community Newsletter and get the latest updates delivered straight to your inbox.

- Mozilla Addons Blog: Nova is here: what changes for your Firefox theme (2026/09/29 15:34)
  On September 29, Firefox 157 shipped Nova, the biggest refresh of the Firefox interface since Proton back in 2021. The browser frame is cleaner, the sidebar plays a bigger role, and vertical tabs are a first-class layout. Your theme continues to load, and most color and image properties work as they did before. What does change is where those colors and images display, how some of them look next to each other, and a handful of properties that no longer have a visible effect. Most themes will look fine. Some, especially themes built around a background image, look different enough that you will want to take a look. This post walks through what changed compared with the Proton interface, what to check in your theme, and what is still being worked on. If you only have five minutes, jump to the checklist. What changed in the browser frame The toolbar and sidebar now sit flush with the window edges, with no gaps between them. For your theme, this means your frame color and background image have less empty space around them, and the edges where the toolbar, sidebar, and page meet are more visible. Area Proton Nova (Firefox 157) Toolbar and sidebar Flush with the window, no gaps or floating borders Same as before Horizontal tabs Theme background applied to the tab strip Same as before Vertical tabs Tab bar color on the tab strip Tab bar color also shows on the toolbar Sidebar Separate panel Shares the theme background with the toolbar area colors, accents and tabs Accent color. The default Firefox theme uses Nova's purple accent.

Every other theme, including yours, uses the operating system’s accent color for buttons and other accented controls. Themes cannot set the accent color through the manifest, and is why your theme cannot show the purple you see in the default theme. Focus and link colors. Firefox still applies its own blue and cyan to focus rings and some links when a theme other than the default is active. The selected tab. Proton always drew a shadow under the selected tab, which helped it stand out even when a theme did not style it. Nova removes that shadow. If your theme relies on it, the selected tab can blend into the tab strip. Set the selected tab colors, with tab_selected and tab_line, so the selected tab stands out. Button hover colors. Your button_background_hover and button_background_active colors also apply to buttons on the tab strip, and for buttons that sit on your frame background with vertical tabs. A solid color picked to match the toolbar can almost disappear there. Semi-transparent colors, for example rgba(0, 0, 0, 0.15) for hover and rgba(0, 0, 0, 0.3) for pressed, work on both backgrounds. Sidebar and vertical tabs The sidebar shares your background. The theme background now continues under the sidebar, so the sidebar and the toolbar area appear as one surface. Set sidebar colors explicitly. Use sidebar and sidebar_text for sidebar panels. If you leave them out, Firefox picks colors for you, relevant to your theme’s color. sidebar_border is now a separator. The sidebar splitter is gone; the property now colors the line between the sidebar and the page. Sidebar buttons follow your text color. Buttons and icons in the sidebar are derived from your text color rather than your toolbar button colors, so they can look slightly different from the toolbar. Vertical tabs and the toolbar. With vertical tabs, your tab bar color is also applied to the toolbar. If those two colors were chosen to contrast, check how they look together. Expand on hover. In Firefox 157, a sidebar that expands on hover may not show your theme’s background. This is fixed in Firefox 158. Properties that changed or no longer work Some theme properties behave differently in Nova. Your theme still loads if it uses them; they just have no visible effect, or style something different. Property In Nova ntp_text No longer styles the new tab page search box popup_text No longer styles the address bar dropdown; other popups still use it sidebar_highlight No visible effect; the default highlight color is used sidebar_highlight_text No visible effect toolbar_bottom_separator Acts as a separator next to or below the toolbar and around the page The theme manifest reference on MDN is being updated with these changes. Themes with background images Themes using images are where you are most likely to see a difference. Your image can now show behind the sidebar and vertical tabs, and an image designed for a wide, short toolbar rarely looks good stretched or tiled down a vertical panel. In our testing of popular themes, the image themes that look off in the sidebar mostly did before Nova too. We did test a number of themes but we invite you to do your own testing. Nova makes the sidebar more visible, so the issues are easier to notice. You can now choose where your images go with the backgrounds_area property, available from Firefox 156: “auto” (the default): Firefox decides from additional_backgrounds_alignment. If any image is aligned to the vertical center or bottom, images stay in the top toolbars; otherwise they cover the whole window. “window”: images cover the whole window, including behind the sidebar and vertical tabs. “top_toolbars”: images stay in the toolbars at the top of the window, and the sidebar uses your frame color. If your image was made for the toolbar only, “top_toolbars” with a frame color that suits the image is usually the quickest fix. Also set a toolbar color, even a semi-transparent one. Without it, notification bars that appear above the page sit directly on your image and can be hard to read. New Gradient Support Nova’s color themes use gradients, and you can too. The Nova theme manifests are the best working example; look at how they set colors for each surface, including the sidebar. backgrounds_area applies to gradients the same way it applies to images. Older versions of Firefox ignore color properties they don’t recognize, so your theme still loads there; if you use a feature older versions reject, set strict_min_version in your manifest. Checklist: what to check in your theme Install your theme in Firefox 157 and look at it in each of these states: Horizontal tabs and vertical tabs. Sidebar closed, open, and expanded on hover. Also toggle “Customize sidebar” while the sidebar is open or expanded. Compact density, if your users are likely to use it (under settings -> appearance density(. A private window. Light

and dark system settings, since the accent follows the operating system. Hover over the buttons on the tab strip. Then, if something looks off: Set tab_selected and tab_line so the selected tab stands out without Firefox's old shadow. Set sidebar and sidebar_text to colors that match your theme. Use semi-transparent button_background_hover and button_background_active colors. Check that your tab bar and toolbar colors work together with vertical tabs enabled. For image themes, choose a backgrounds_area and set a toolbar color. Remove or ignore properties that no longer have an effect. Publish an update on AMO so your users get the changes. What we are still working on Nova will evolve after launch. Firefox 158 ships on October 13 with the expand-on-hover sidebar fix. We are also working on these, with no release date yet: Focus and link colors that fit your theme instead of the fixed blue and cyan. A popup_icons color for icons in menus and panels, separate from toolbar icons. Theme colors for hover and selected states inside sidebar panels. Better hover states for tab strip buttons. We will post again when these land. For users who prefer the old look Some of your users may miss Proton. From September 29, Mozilla is featuring a Firefox Proton theme in the Add-ons Manager for anyone who wants the previous look. It is a normal theme, so users can switch between it and yours at any time. Resources Theme manifest reference on MDN: every color, image, and property, including backgrounds_area. Static themes on Extension Workshop : updated for Nova. Nova theme manifests: worked examples for gradients and sidebar colors; the themes themselves are on AMO, for example Nova Sun. Questions? We want your theme to look its best in Nova. If something looks wrong and the checklist does not fix it, tell us on Discourse and include a link to your theme and a screenshot. Those reports help us find the cases that need a Firefox fix rather than a theme change. Thank you for making Firefox look like you and like your users.  The post Nova is here: what changes for your Firefox theme appeared first on Mozilla Add-ons Community Blog.

- The Mozilla Blog: More modern. More flexible. Still Firefox. (2026/09/29 13:00)
  What comes to mind when you think of Firefox? Is it our best-in-class privacy features and ad blocking capabilities? Our partnerships with Wrexham, NVIDIA or Mistral? Or maybe our adorable mascot Kit? If you've been following Firefox this year, you've seen us add new ways to get things done while giving you more choice over how you browse. That includes browsing two pages side by side, adding an extra layer of privacy with built-in VPN, blocking trackers for smoother scrolling and faster load times, and deciding how AI shows up in Firefox.  For the past few months, many of you have also been trying out our new Firefox design in Nightly and sharing your thoughts with us as we continued to refine the experience. Today, we're rolling out that design with FX 157 to everyone using Firefox across desktop and mobile devices. Firefox has done a lot of growing over the last year; now it looks the part.  A more modern Firefox When we first shared our plans to design Firefox for the future, we said we wanted Firefox to feel "current, but not generic. Warm, but still precise."  That design ethos now carries across all of Firefox. We've refreshed the colors, icons and themes throughout the browser, from the tabs and toolbars you use every day to your New Tab page, Private Browsing experience and more, with no cost to your browser performance. This shared design language across desktop and mobile gives Firefox a consistent foundation for new features and experiences, while keeping the personality that makes Firefox unique.  Make Firefox yours We've added more choices for how Firefox looks and works, including the return of a feature many of you asked for: Compact Mode is back. As people started trying the new design, we heard from many of you who wanted to fit more into your browser window. Compact Mode reduces the size of your tabs and toolbar to give more of your screen to the web, with an auto-compact option for smaller screens. Find your look. A new theme picker makes it easier to explore different looks for Firefox, with new themes and wallpapers to choose from. Set up New Tab your way. Choose what you want to see on your homepage. Now, you have a dedicated space to pin and organize shortcuts so sites you use all the time stay put. Refined, not reinvented Themes and wallpapers may change over time, but the things that made people choose Firefox in the first place are still

the same. We’re still independent and open source, with privacy protections built into the browser and controls that put you in charge of how Firefox works for you. This carries through everything we build, from the height of your toolbars and what belongs on your New Tab page to which AI features show up in your browser, which model you choose and what context you want it to have. Today’s Firefox is a renewal, grounded in the same principles that have shaped it from the beginning, with a new design built to carry Firefox forward.  For those who have come with us on this journey, thank you for helping us shape the future of Firefox.  If you’ve been away on hiatus, welcome back. Try the latest Firefox and let us know what you think. The post More modern. More flexible. Still Firefox. appeared first on The Mozilla Blog.

- Firefox Nightly: Shoutout to More Layouts – These Weeks in Firefox: Issue 210 (2026/09/28 19:31)
  Highlights Rév O’Conner added a settings popup to control the layout of the Inspector. “Auto” automatically switches the layout at a given toolbox width. The new sidebar (sidebar.revamp) and switcher setting will be enabled by default for users along with Nova in DevEdition, Beta and Release channels. To ease the transition for old sidebar users, the “Open tools from sidebar” option will be unchecked for those users to more closely resemble the old sidebar experience. The migration won’t apply to users who had already tried the new sidebar and reverted back. The custom wallpaper library for the New Tab page is coming along! This allows you to save multiple images to your wallpaper folder, rather than replacing one image at a time. While it’s still in development, you can test it by setting browser.newtabpage.activity-stream.newtabWallpapers.customWallpaper.library.enabled to true, starting in Firefox 157. Mark added an MDN search engine for Nightly and Developer Edition. Select it from the menus, or type @mdn <search term>. Friends of the Firefox team Resolved bugs (excluding employees) Script to find new contributors from bug list Volunteers that fixed more than one bug :Benjamin Peterson Amadi any1here Caleb Pickard Chris Vander Linden Gopalarathnam Venkatesan Igor [:ExplodingJoysticks] Khalid AlHaddad Lukáš Lipinský Mayank Bansal Nazeer Ahmed Shaikh Noah Varghese Sameem [:sameembaba] Sebastian Zartner [:sebo] sfshah Xiaosen Zhuang New contributors (□ = first patch) Amadi: Network monitor opens files in last tab Remove the unused _discarded field in UrlbarParentController.sys.mjs Update comments that point at the removed openUILinkIn function Screenshot default filename contains trailing whitespace on pages without title Caleb Pickard □ Remove BROWSER_SEARCH_PREF constant The phrase “The count URI loads…” appears repeatedly in the search telemetry code. It should read “The count *of* URI loads…” dchen705: Remove unnecessary module import in browser/components/urlbar/tests/browser/head.js Ekerin Agboola: Remove old documentation related to the removed SEARCH_SUGGESTIONS_LATENCY_MS histogram Igor [:ExplodingJoysticks]: Cleanup dead code in PageThumbs (preserveAspectRatio) Dead code in services/crypto/modules/utils.sys.mjs# Gopalarathnam Venkatesan: “Add class” button (.cls) should be disabled when a class can’t be added to the selected node Remove unused lightweightThemes.getMoreURL pref Remove orphaned selectHistoryMode JSDoc from browser/components/preferences/tests/privacy/head.js gBrowser.selectedTabs does not return tabs in tab order □ Jieke Jiang: Remove unused AboutReaderParent._getArticle □ Andrew: White flash on startup caused by abouthome_cache storing incompatible startup frame □ Rév O’Conner: Inspector does not respect panel layout □ Kana: browser_privatebrowsing_resetPBM.js awaits an array instead of Promise.all, so the tab-close wait does nothing Raúl: L10nCache.add() throws on an unknown Fluent ID instead of reporting it □ Samuel Mbabhazi: Use new color syntax throughout the DevTools K: “Go to line” sets a cursor line which does not exist in footer location indicator sfshah: Turn devtools/client/jsonview/converter-child.js into an ES class □ Fix misspellings in SearchService.shouldShowInstallPrompt Nazeer Ahmed Shaikh Remove deprecated logs from tt(c) harness □ test_dont_suggest_default_engine task is duplicated in browser_contextualsearch.js mochitest Noah Varghese: Make browser/components/syncedtabs/sidebar.xhtml an HTML file □ Remove usages of “six” from toolkit/components/telemetry/tests/marionette/ Valerie Racine (:v-racine): Fix incorrect assertion message in

browser_adoptTab_failure.js  Xiaosen Zhuang: Fix the undefined –border-color-dead-row custom property in aboutMemory.css Use documentGlobal instead of the removed ownerGlobal in FormAutofillPreferences Project Updates Add-ons / Web Extensions Addon Manager & about:addons As part of Project Nova related work: Fixed responsive layout issues that caused horizontal scrolling in about:addons at high zoom levels – Bug 2059864 Fixed the “Discover extensions” button appearing multiple times in about:addons – Bug 2070281 Added telemetry for theme-picker shown events and for appearance/native-theme changes in about:addons – Bug 2069417 / Bug 2070341 Added an accessible name to the themes mode control in about:addons, fixed in Firefox 158 and uplifted to the Firefox 157 beta channel – Bug 2064563 Thanks to Mark Kennedy for the help on fixing this accessibility gap Updated the built-in dark and light theme previews to use the nova style – Bug 2070274 Thanks to Emilio Cobos Álvarez for the fix to the default light and dark theme previews. Fixed the Nova default and AMO curated theme previews to follow the OS light/dark mode while ignoring the Website appearance setting, fixed in Firefox 158 and uplifted to the Firefox 157 beta channel – Bug 2070562 Stopped setting unused taarId session cookies for AMO on startup – Bug 1871561 Thanks to Manuel Bucher for helping us to clean up these unused internals. WebExtension APIs Fixed webRequest requestBody parsing so POST form fields named __proto__ are no longer dropped – Bug 2061470 Changed alarms.clearAll() to resolve with undefined instead of a boolean, starting in Firefox 157 – Bug 2067229 Thanks to Ollie Hensman-Crook for the fix to the alarms API. Implemented native messaging support via xdg-native-messaging-proxy for Flatpak and Snap builds – Bug 1955255 Thanks to Jan Horak for the native messaging work on Linux packaging. As part of Florian Quèze effort to investigate and fix intermittent failures: Made windows.update() wait for the window to actually resize or move before resolving, fixing an intermittent Linux test failure – Bug 1307759 Fixed a pageAction popup incorrectly opening via a commands API shortcut while the extension was already shutting down – Bug 2039637 Fixed action.openPopup() rejecting when a tab hover preview was showing, a regression from Bug 2022281 in Firefox 150, with the fix shipping in Firefox 157 and no uplift planned – Bug 2062229 DevTools The team has a few sizable ongoing projects. Bomsy is working on moving Stylesheets to the Debugger so we can retire the (XUL-based) Style Editor, and turn the Debugger into a “Sources” panel (keeping all the existing functionality of course). (pref: devtools.debugger.features.stylesheets-in-debugger) Alex is investigating ways to better handle CSS authored text and CSSOM changes in the Inspector, building information directly from the Rust CSS engine, with the hope it will bring performance improvement editing styles in the Rules view. Nicolas is implementing a way to show to the user how the engine goes from a CSS declaration value to the computed value so it’s easier to follow what’s the result of the different call sites, or how the different units (em, %, vmin, …) are computed to their final (e.g. px) values. (pref: devtools.inspector.css-explainers) External contributions Chris Vander Linden finalized a multi-months project to add a search toolbar in the Netmonitor Raw Response panel, adapting and reusing the component we are using in the Debugger (#1941575) Amadi made the “Open in new tab” context menu entry to open tab next to the selected tab, not at the very end of the tab list (#2059979) Mayank Bansal optimized the performance of console.clear() (#2068156, #2068157, #2067000) Benoit added JSON Lines (JSONL/NDJSON) support in the Netmonitor Response panel (#2059673) Gopalarathnam Venkatesan made the “Add class” button (.cls) to be disabled when a class can’t be added to the selected node (e.g. it’s a text node, or a comment, or the doctype node) (#2000150) K fixed an issue with the “Go to line” action in the Debugger (#2064575) Samuel Mbabhazi refactored our CSS codebase to use new color syntax (e.g. rgb(255 0 0) instead of rgb(255, 0, 0)) (#2054228) sfshah turned devtools/client/jsonview/converter-child.js into an ES class (#2004273) Team/Project Update Florian Quèze [:florian] is working on fixing test flakiness in devtools/client/debugger and already landed an impressive stack of patches (#2001947, #2070917, #2070920, #2070932, #2070970, #2027953, #1932082, #2070923, #2070933, #1814093, #1868969, #2071238) that also includes fixing various accessibility issues that were spotted by a11y_checks #2070911, #2070955, #2071227,

#2071231) Note that Julian Descottes [:jdescottes] also has an ongoing quest to reduce intermittent failures in our codebase and fixed a few those last 2 weeks (#1767707, 2069100, #2009386) [florian] 24 patches landed on Friday, you can see the impact on https://tests.firefox.dev/tests.html?path=devtools%2Fclient%2Fdebugger Nicolas Chevobbe [:nchevobbe] fixed a few issues spotted by Jake in the Inspector for the new-ish attr()CSS function (#2068479, #2068480, #2068483) Alexandre Poirot [:ochameau] fixed an OOM crash when viewing source mapped file for a large wasm module with many workers (#2058286) Julian Descottes [:jdescottes] improved performance of the Inspector when Devtools was opened on a page with many grid elements (#1988868) Hubert Boma Manilla (:bomsy) fixed an infinite loop in the Debugger search if you were searching for $ on a big JS file (#2068517) WebDriver Sameem merged two internal helpers (remote/shared/Sync.sys.mjs and remote/marionette/sync.sys.mjs) which removed some code duplication. Khalid AlHaddad unified the navigation commands in Marionette and WebDriver BiDi to always initiate navigations via browsingContext.loadURI(). Khalid AlHaddad updated the WebDriver BiDi client used by WebDriver tests so that browsing_context.locate_nodes returns the nodes array directly instead of the enclosing result object, matching the updated command response. Julian Descottes made the destinationFolder mandatory when calling the browser.setDownloadBehavior command with type=”allowed” – which aligns us with the specification. In order to restore the default behavior without having to specify a folder, clients should call setDownloadBehavior with null instead. Henrik Skupin fixed a bug in geckodriver where specifying androidIntentArguments in the capabilities caused the default intent arguments to be omitted. Fluent According to arewefluentyet.com, ~74.77% of our strings are now using Fluent! Lint, Docs and Workflow According to arewemozsrcyet.com, ~59% of our modules are now using moz-src. eslint-plugin-mozilla has switched from using Mocha to Jest for tests. Planning on future updates to investigate getting devtools & newtab to use the jest installed at the top-level, rather than separately installed copies. Information Management/Sidebar We’re working away on bug fixes for sidebar, vertical tabs, split-view, and Nova Thanks Florian (:fqueze) for landing a set of patches that have markedly reduced test flakiness in the Sidebar component New Tab Page Custom wallpaper library Dre added a “Your images” folder to the wallpaper picker on the New Tab Page, letting users save and switch between multiple custom wallpapers directly from the picker UI (desktop channels), which reduces friction for users who frequently change backgrounds. Dre implemented storage and migration for multiple custom wallpapers so users keep their existing wallpaper during upgrade while gaining the ability to save and switch multiple wallpapers — migration moves old single-wallpaper prefs into the new multi-wallpaper storage and the NTP now reads the wallpaper array at load/sync points. Dre updated New Tab Page strings to support multiple custom wallpapers so users now see accurate, localized labels and accessibility text when managing or switching between multiple custom backgrounds, reducing confusion in the UI during wallpaper selection. Dre added a delete affordance for saved custom wallpapers inside the New Tab Page picker so users can remove unwanted images from their saved set without going to profile files or prefs, improving manageability of user-provided wallpapers. We’ve completed a mass migration of our unit tests from Mocha (not mochi) / Karma to Jest/RTL, which is more modern and actively maintained. Sections is now enabled globally for all regions that show the content feed! Reem Hamoui fixed Lists widget so that keyboard users can now tab to completed items on the New Tab Page and see a visible selection/focus state Reem Hamoui fixed the Timer widget so that keyboard users can toggle focus vs break modes with standard keys (Enter/Space/arrow keys) thanks to restored keyboard handlers and tabindex semantics. Reem Hamoui made it so that screenreaders now announce wallpaper names via added aria-labels/alt text and role improvements in the Customize UI. Maxx Crawford made it so that the Customize UI now filters wallpapers by their metadata and no longer surfaces images marked hidden or in specific visibility groups. Irene Ni added Section Layout Name (String) telemetry to newtab content impression & click events so impressions/clicks now carry layout identifiers for

more granular UX and A/B analysis Reem Hamoui added accessible names (aria-label/aria-labelledby) to the HNT Customize panel back buttons to fix screen-reader discoverability and keyboard/AT announcement when navigating the New Tab Customize flow. Nina Pypchenko [:nina-py] preserved iframe rendering for New Tab Page widgets during drag operations to prevent iframe-backed widgets from going blank while being moved. Nina Pypchenko [:nina-py] restored the Stocks widget dropdown and menu visibility under High Contrast Mode by adjusting contrast-aware CSS and ARIA roles so users relying on HCM can access stock controls again. Nina Pypchenko [:nina-py] fixed a results text overflow for long search queries in the Stocks widget by adding word-wrap/clamping to result lines, preventing layout breakage and truncated controls when users enter long symbols or queries. Nina Pypchenko [:nina-py] added telemetry for theme selection in the New Tab customization panel by emitting a selection ping on change, enabling measurement of user theme choices without changing UX. Irene Ni restored the Lists widget switcher checkmark for the selected list by fixing menu item selection-state rendering so users can again see which list is active in the switcher. Reem Hamoui updated the Privacy widget CTA and spacing on the New Tab Page, adjusting CSS/layout and CTA copy so the action is more discoverable and has a clearer tap/click target across responsive breakpoints. Reem Hamoui updated the zero-state copy for the New Tab Privacy widget to clarify what the widget does and the next steps for users when no privacy events or trackers are present, reducing confusion for first-time or cleared-state users. Maxx Crawford landed Fluent strings for the Nova launch New Tab customization callout, enabling localized text in the New Tab customization callout so users in supported locales see translated UI instead of missing-string fallbacks. Nina Pypchenko made the customization panel reset when closed, ensuring about:newtab customization state (temporary tile/layout previews) is cleared on close so users no longer see stale previews when reopening the panel. Dão Gottwald fixed New Tab to pick up system High Contrast Mode on Linux and macOS, restoring correct high-contrast colors/contrast on those platforms for users who rely on OS-level accessibility settings. Irene Ni restored theme matching for New Tab context menus by reapplying theme-aware CSS variables in the New Tab Page contextmenu styles, fixing mismatched colors for users with custom/light/dark themes. Dustin Whisman fixed New Tab section context menu button color regression by adjusting the section-contextmenu button style rules (button color variables and state selectors) so section actions now respect theme contrast. Irene Ni restored transparency to New Tab card backgrounds by removing opaque background rules so cards correctly display background images/themes and improve visual consistency across DPI/compositing setups. Picture-in-Picture Thanks to Sebastian Zartner [:sebo] for fixing broken Dailymotion captions on the PiP player. kpatenio fixed YouTube’s displayed playback speed settings being out of sync with PiP. Reminder that the playback speed settings can be enabled with media.videocontrols.picture-in-picture.playback-speed.enabled. Thanks to Sylvestre Ledru [:Sylvestre] for helping fix a broken link in our PiP docs as a part of Bug 2071683. Search and Urlbar Address Bar Dao and Moritz are working on bringing the full urlbar experience into the newtab page, now enabled on nightly and going through polish and improvements.. (Bug 2068104, Bug 2068633, Bug 2069260, Bug 2064747) Drew enabled AMP and Wikipedia by default in AT, BE, CH, ES, IE, LU, NL, PL, PT and SE. (Bug 2066294) James gated adaptive autofill page results behind a minimum score threshold. (Bug 2066171) any1here stopped disabled Manage AI and Labs quick actions being displayed when disabled. (Bug 2070378) Search Dale landed the Recent Searches widget which will be be used launched as an experiment (Bug 2063718) Caleb fixed the search box eating the first character typed into it. (Bug 1953361) Mark configured Wikipedia search for Sardinian (sc) to use the Sardinian Wikipedia rather than the Italian one. (Bug 2057690) Places Marco fixed bookmarked Google Maps using Google’s default favicon. (Bug 1575809) Marco fixed the Downloads and History windows going full-screen instead of floating on macOS. (Bug 2058062) Tests Dashboards: Most frequent oranges: https://tests.firefox.dev/intermittent.html#date=21days Folders with the most flaky tests: https://tests.firefox.dev/flaky.html?kind=mochitest#date=21days&view=list Issues of tests in a given folder:

https://tests.firefox.dev/tests.html Eg devtools debugger tests shows progress of the recent effort to fix these tests recently Flakiness burn down Multi-agent Claude sessions running for hours investigating and fixing tests of a given set provided in the prompt. Still iterating on the process. Did this 3 times so far: Bug 2062142 – [meta] Fix the flakiness in browser/components/extensions/test/browser Top 20 intermittent mochitest bugs annotated by sheriffs. Probably what made the mochitest line go down for the first time on https://tests.firefox.dev/ And what helped with the sidebar component, even though sidebar wasn't my target. Bug 2070797 – [meta] Fix the flakiness in devtools/client/debugger/test/mochitest https://tests.firefox.dev/tests.html?path=devtools%2Fclient%2Fdebugger%2Ftest%2Fmochitest Profile your claude session for cost and context size: https://github.com/fqueze/claude-profiler To monitor cost during your sessions, try https://github.com/padenot/foxtail

- Firefox Tooling Announcements: MozPhab 2.21.0 Released (2026/09/25 18:18)
  Bugs resolved in Moz-Phab 2.21.0: bug 2068637 code review bot fails to apply patches when the parent is not known bug 2073291 moz-phab submit could load reviewers and review groups in parallel with revisions and diffs bug 2075401 "Phabricator Error: Validation errors" from moz-phab submit should display the revision Discuss these changes in #engineering-workflow on Slack or #Conduit Matrix. 1 post - 1 participant Read full topic
- The Mozilla Blog: Classic, Private, or Smart? Choose the right Firefox window for every task (2026/09/24 17:00)
  We spend a lot of time browsing online. Whether you are researching for work, planning a trip, paying your bills, or scrolling through social media, your browser is there for it all.While all these tasks are very different in nature, a lot of people still tend to stick to the same window type for all their use cases. But what if there was a different way to browse?You may have noticed Firefox now has three window options for desktop. On top of the Classic and Private windows you've known for years, a third choice has been rolling out in beta: Smart Window. This optional AI-powered browsing experience is built to move your work forward and help finish what you started online.Following the recent addition of several new Smart Window capabilities, we thought now would be the perfect time for a breakdown of what each of our three window options actually does, and when to use them. Classic: Timeless and tailored to you Classic Window is still the most customizable option of the three. While Private and Smart windows share many of the same features and capabilities as Classic, a Classic Window offers the widest range of Firefox's standard personalization options. It gives you the broadest canvas for tailoring Firefox to the way you browse.What makes it Classic: Full oversight over controls, buttons, and toolbars. Access to the entire Firefox add-ons library. Advanced customization and configuration options.  The ability to sync your history, bookmarks, and open tabs across your signed-in devices. When to use it:Classic is your home base. It's the best place to start customizing around your browsing needs, whether that means an extension setup specifically tailored to your workflow, a special theme that matches your style, or just the familiar Firefox experience you created over years of tweaking to your personal taste. While many customizations are also available on Private and Smart window, Classic offers the most extensive line-up of optional browser features, extensions, and aesthetics to personalize your setup.Classic is the right move for general browsing like scrolling through social media, conducting a quick Google search, or reading online articles. It's also the best place to start if you are looking to build your browser around you. Private Window: Browse with greater peace of mind Private Window stays true to its name and number one priority: it's designed to limit the information associated with your browser session. When a Private Window is closed, Firefox doesn't retain session information, such as what you searched for, the sites you visited, or the cookies they set.What makes it Private: Pages visited from a Private Window will not be added to the list of sites in Firefox's History menu, the Library window's history list, nor the address bar drop-down list. Nothing entered into text boxes on web pages and Search bar will be saved for Form autocomplete. Cookies set in a Private Window are kept only temporarily in memory, separate from your regular cookies, and are discarded

after your last Private Window is closed. Extensions are off by default in a Private Window, unless you explicitly grant them permission. When to use it:A Private Window is recommended when you don’t want activity from a browsing session sitting in your history, such as shopping for a gift. It’s also helpful when browsing on a shared device, ensuring your history, logins, and cookies will not be saved for the next person.You may not need a Private Window every time you open your browser, but it’s always there for you when you do. Smart Window: Firefox’s privacy-first, AI-powered browsing experience to help get stuff done Smart Window is Firefox’s newest window type, currently rolling out in Beta to users in the U.S., Canada, and France. Think of it as a Classic Firefox window with a built-in AI assistant layered on top – one that you can choose to have utilize your open tabs, recent browsing, and the page in front of you, so it can help you do more without switching contexts – all with the privacy and security you expect from Firefox.What makes it Smart: A built-in assistant that can summarize pages, generate recommendations, compare information, and plan tasks using your open tabs. The ability to retrieve current web information and display the sources behind its response without taking you to a separate search results page, supported by Firefox’s new partnership with Exa. Suggested groups for related tabs.  Visual previews of pages from your browsing history, so you can surface the page you’re looking for more easily among similar results without opening each one. The choice of three AI models to power the assistant, as well as the option to bring your own model. Optional “memories,” built from either your Smart Window chats or your browsing activity in both Smart and Classic windows, or both. Memories help the assistant give you more helpful, personalized answers over time. They are stored locally on your device and can be deleted at any time. When to use it:Smart Window shines when you need a little extra help finishing what you started. Its ability to work with the context you choose to share can help you make progress on ongoing tasks that may require more than one session, like trip or event planning, comparison price shopping, or conducting research for work or personal reasons.And with Smart Window’s optional memories, which learn your patterns and preferences over time (with your permission), Smart Window gets more useful the more you use it, keeping you from having to spell everything out from scratch each time. Selecting the right window for the job None of these windows is “better” than the others, they’re built for different moments.The good news is you are never locked into one. Firefox lets you move between all three depending on what you’re doing, so you can choose the window that fits each task.You can access Classic and Private windows at any time from the Firefox menu. As for Smart Window, it remains in beta, with current availability to people in the U.S., Canada, and France.To try Smart Window, visit https://www.firefox.com/smart-window. Or, if it’s not available in your region yet, you can sign up to be notified when it is. The post Classic, Private, or Smart? Choose the right Firefox window for every task appeared first on The Mozilla Blog.

- Thunderbird Blog: Thunderbird Monthly Development Digest: September 2026 (2026/09/23 22:35)
  Greetings once more from the Thunderbird development team! As the Northern hemisphere summer comes to an end and a slew of PTO, company holidays and projects have wrapped up, what better time to share progress and next steps on the work the Thunderbird Desktop Engineering team has on their plates!? Exchange support moving well The powerhouse Exchange engineering team had some great momentum over the summer and completed their work on implementation of the Graph protocol to support email accounts – well in advance of their deadline. They have since moved on to start work to support Exchange calendars using the Graph protocol which is a significant milestone given that our calendar code hasn’t had significant changes for some time. While some team members are now juggling multiple projects, good progress has been made to wrestle the foundations into place, with calendar discovery and persistence in place, along with display of calendar items. Following a brief hiatus to focus on other priorities, the team is once again back into gear. Keep track of our Graph API Calendar implementation here.  Account Setup & Authentication Since I last wrote, many of the account setup and account authentication improvements

have landed (mostly in versions 156 & 157), unlocking some important security use cases which are becoming more frequently requested and mandated by organizations and legislation around the world.  While diving into the code is an enjoyable adventure, the newly-minted articles devoted to the topic of Oauth customization might be a better starting point for most:Knowledge Base – Custom OAuth for Thunderbird UI overhauls – why just one?! The front end engineering team has completed the rebuild of our account setup manual configuration flows to include new protocol options and security customizations, so it is time to pick up where we left off on our Calendar UI rebuild. The majority of the front end team is now either focused on Calendar Event data write operations stemming from the Event Read dialog (delete event, RSVPs, Reminders) – or beginning the largest part of the journey, which is the implementation of a new Calendar Create/Edit dialog, along with REDUX-based state management, a new WYSIWYG editor and more reusable components such as date pickers. Follow the Calendar Event Create/Edit work here While that work is underway, why not weave in some more improvements?! After some extensive user research and testing, our design team has begun work to improve the Settings UI, starting with some updates to humanize and homogenize the various options, headings and guidance provided in that area of the application. Read more about that journey here Contributions are on the rise! Some AI guidance… In the past few months, we've seen a wonderful increase in contributions from community members – partially thanks to our move away from Mercurial to a more familiar Git-based workflow – and partially thanks to many LLMs doing a great job of interpreting our public codebase and making it far more accessible. Or it could be that our team is just such a friendly bunch that we've attracted a great group of people willing to support. Whatever the reason, we're happy to see new names. faces and ideas in our chats and review queue!! Having said that…We've reached the point where the velocity of work is beginning to pile up and clog our review queue – so I felt it might be time to share some guidance to make the process easier for both external contributors and the internal review team. Bear in mind that while we use AI workflows in several areas, our goal is to preserve a codebase which can be maintained by humans. We're working on a policy that may word some of these points better, but from the past few months of my experience here's what I see as being helpful: Keep patches to a digestible size and understandable complexity – Contributions should save more development time than it takes to review. If the patch contribution creates heavy work for maintainers, it will be deprioritized – so the best approach is to break the work up into pieces that can be understood more easily and discussed together. Bear in mind that many cloud-based models aren't primarily designed to be frugal with token usage so it's easy for patches to grow (much) larger than they actually need to be to solve the problem. Be specific with prompts to minimize the surface area of changes so they are reviewable. Humanize or write comments (within code and throughout the code review process) by hand – While we've become accustomed to interacting with a variety of automated systems in our daily lives, a good chunk of the joy involved in our work is to share knowledge with each other and refine the code so that it can be reused and understood by humans. Be wary of license infringement – much of our work is under the MPL license and we must all take care not to introduce refactored code that originally came from licensed software which can be rewritten and translated using LLM, then relabelled under our licence. Be transparent – it will often help to share prompts and your model when submitting code for review. Refining all aspects of our workflow is important, and we can all benefit from a collective evolution of skills and prompts that improve over time with input from the team and community. We encourage contact with a member of the team to discuss approach and architecture in advance of spending time and tokens on a patch. Some areas of the codebase are already slated for redevelopment and it's not always obvious what is underway and planned. Having a quick conversation to say "Hey, I'm thinking about working on this" is a great start to landing something that spends less time in review and uses less tokens, energy & the world's natural resources. Of course if none of these guidelines are followed, we'll still be very happy to receive help from the wider community – the process may just take longer and be a bit less pleasant. Enterprise As parts of the world

turn their infrastructure priorities toward digital sovereignty, we've seen an increase in interest from governments, institutions and organizations around the world who are looking to either improve their existing FOSS-based infrastructure or deploy for the first time. That has resulted in some focused efforts throughout Q3 to improve our Enterprise security and policy framework to bring it in line with the strides that the Firefox engineering team has made, and also roll out more Thunderbird-specific administrative mechanisms that allow granular control over end user configurations. Read more about your options and follow the work below: Thunderbird Enterprise Policies and Relevant Preferences Thunderbird Enterprise PoC Work Items Maintenance, Upstream adaptations, Recent Features and Fixes Waves of changes from upstream Firefox have continually kept the team on their toes throughout July & August, but the small but mighty gang have kept Daily builds largely intact each day. In addition, the team and contributor community have continued landing a series of reliability, stability, and usability improvements across the application. Recent highlights include: Nico has a series of calendar improvements in the works with several now cleared for landing: D319397, D320418, D320419  Hurtmut has again come to the rescue with a number of patches but most recently solved a widespread filtering problem introduced upstream which will need to be uplifted as a matter of urgency. Richard pushed many patches since my last update, many related to upstream changes required after the Firefox project "Nova" was unleashed. He's also been on the lookout for other breaking changes such as XUL attribute modifications. Max was on fire and in constant comms with the team to deliver a large number of important changes but also helped to refine some of our processes relating to AI-driven workflows. Recently, he led an investigation into growing concerns over instability with connections with Gmail accounts in Bug 2028760. and many more which are listed in release notes for beta. If you would like to see new features as they land, and help us find some early bugs, you can try running daily and check the pushlog to see what has recently landed. This assistance is immensely helpful for catching problems early. — Toby Pilling Senior Manager, Desktop Engineering The post Thunderbird Monthly Development Digest: September 2026 appeared first on The Thunderbird Blog.

- Firefox Tooling Announcements: MozPhab 2.20.0 Released (2026/09/23 16:45)
  Bugs resolved in Moz-Phab 2.20.0: bug 2066178 Drop support for Python 3.9 bug 2068605 preserve file logging from moz-phab list --format json and moz-phab patch --raw bug 2071586 Review queue reminders should ignore group reviews (at least optionally) bug 2073088 moz-phab submit is getting the file size from hg/git even if it could infer it from the blob itself bug 2074036 List index out of range Discuss these changes in #engineering-workflow on Slack or #Conduit Matrix. 1 post - 1 participant Read full topic

From:
https://wiki.tromjaro.alexio.tf/ - **TROMjaro wiki**

Permanent link:
**https://wiki.tromjaro.alexio.tf/doku.php?id=news:planet:mozilla**

Last update: **2021/10/30 11:41**